Experiment No: Group_C_02 **Date:** 24/07/2024

Name: Kushal Kishor Shankhapal **Subject:** ADBMS LAB

Roll No: 61

Aim: Execute at least 10 queries on any suitable MongoDB database that demonstrates following querying techniques:

1. find and findOne (specific values)
2. Query criteria (Query conditionals, OR queries, $not, Conditional semantics)
3. Type-specific queries (Null, Regular expression, Querying arrays).

Objectives: To learn various MongoDB operations to access data from mongo Database.

Commands:

1. mongosh: Launch MongoDB and Access MongoDB Shell

```
it@IT-LL-12:~$ mongosh
Current Mongosh Log ID:        66a089ef8a57c39e8e149f47
Connecting to:           mongodb://127.0.0.1:27017/?
directConnection=true&serverSelectionTimeoutMS=2000&appName=mongosh+2.2.10
Using MongoDB:                 7.0.12
Using Mongosh:           2.2.10
mongosh 2.2.12 is available for download: https://www.mongodb.com/try/download/shell

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


------
   The server generated these startup warnings when booting
   2024-07-24T10:12:52.743+05:30: Using the XFS filesystem is strongly recommended with the
WiredTiger storage engine. See http://dochub.mongodb.org/core/prodnotes-filesystem
   2024-07-24T10:13:01.206+05:30: Access control is not enabled for the database. Read and write access
to data and configuration is unrestricted
   2024-07-24T10:13:01.206+05:30: vm.max_map_count is too low
```

2. use: Create a Database

```
test> use KushalDB_C_02
switched to db KushalDB_C_02
```

3. createCollection(): Create a Collection

```
KushalDB_C_02> db.createCollection("users")
{ ok: 1 }
```

4. insertMany: Insert Sample Data

```
KushalDB_C_02> db.users.insertMany([
...   { name: "Alice", age: 30, city: "New York" },
...   { name: "Bob", age: 25, city: "Los Angeles" },
...   { name: "Charlie", age: 35, city: "Chicago" },
...   { name: "David", age: 28, city: "Houston" },
...   { name: "Eve", age: 32, city: "Miami" }
... ])
{
  acknowledged: true,
  insertedIds: {
    '0': ObjectId('66a08a468a57c39e8e149f48'),
    '1': ObjectId('66a08a468a57c39e8e149f49'),
    '2': ObjectId('66a08a468a57c39e8e149f4a'),
    '3': ObjectId('66a08a468a57c39e8e149f4b'),
    '4': ObjectId('66a08a468a57c39e8e149f4c')
  }
}
```

5. find(): Retrieves all documents from the `users` collection.

```
KushalDB_C_02> db.users.find()
[
  {
    _id: ObjectId('66a08a468a57c39e8e149f48'),
    name: 'Alice',
    age: 30,
    city: 'New York'
  },
  {
    _id: ObjectId('66a08a468a57c39e8e149f49'),
    name: 'Bob',
    age: 25,
    city: 'Los Angeles'
  },
  {
    _id: ObjectId('66a08a468a57c39e8e149f4a'),
    name: 'Charlie',
    age: 35,
    city: 'Chicago'
  },
  {
    _id: ObjectId('66a08a468a57c39e8e149f4b'),
    name: 'David',
    age: 28,
    city: 'Houston'
  },
  {
    _id: ObjectId('66a08a468a57c39e8e149f4c'),
    name: 'Eve',
    age: 32,
    city: 'Miami'
  }
]
```

6. findOne(): Finds the first document in the `users` collection where the `name` field is "Alice".

```
KushalDB_C_02> db.users.findOne({ name: "Alice" })
{
  _id: ObjectId('66a08a468a57c39e8e149f48'),
  name: 'Alice',
  age: 30,
  city: 'New York'
}
```

7. $or: Retrieves documents where the `age` is greater than 30 or the `city` is "Houston".

```
KushalDB_C_02> db.users.find({ $or: [ { age: { $gt: 30 } }, { city: "Houston" } ] })
[
  {
    _id: ObjectId('66a08a468a57c39e8e149f4a'),
    name: 'Charlie',
    age: 35,
    city: 'Chicago'
  },
  {
    _id: ObjectId('66a08a468a57c39e8e149f4b'),
    name: 'David',
    age: 28,
    city: 'Houston'
  },
  {
    _id: ObjectId('66a08a468a57c39e8e149f4c'),
    name: 'Eve',
    age: 32,
    city: 'Miami'
  }
]
```

8. $and: Retrieves documents where the `age` is between 25 and 35 inclusive.

```
KushalDB_C_02> db.users.find({ $and: [ { age: { $gte: 25 } }, { age: { $lte: 35 } } ] })
[
  {
    _id: ObjectId('66a08a468a57c39e8e149f48'),
    name: 'Alice',
    age: 30,
    city: 'New York'
  },
  {
    _id: ObjectId('66a08a468a57c39e8e149f49'),
    name: 'Bob',
    age: 25,
    city: 'Los Angeles'
  },
  {
    _id: ObjectId('66a08a468a57c39e8e149f4a'),
    name: 'Charlie',
    age: 35,
    city: 'Chicago'
  },
  {
    _id: ObjectId('66a08a468a57c39e8e149f4b'),
    name: 'David',
    age: 28,
    city: 'Houston'
  },
  {
    _id: ObjectId('66a08a468a57c39e8e149f4c'),
    name: 'Eve',
    age: 32,
    city: 'Miami'
  }
]
```

9. $not: Retrieves documents where the `age` is not greater than 30.

```
KushalDB_C_02> db.users.find({ age: { $not: { $gt: 30 } } })
[
  {
    _id: ObjectId('66a08a468a57c39e8e149f48'),
    name: 'Alice',
    age: 30,
    city: 'New York'
  },
  {
    _id: ObjectId('66a08a468a57c39e8e149f49'),
    name: 'Bob',
    age: 25,
    city: 'Los Angeles'
  },
  {
    _id: ObjectId('66a08a468a57c39e8e149f4b'),
    name: 'David',
    age: 28,
    city: 'Houston'
  }
]
```

10. $gt: Retrieves documents where the `age` is greater than 30.

```
KushalDB_C_02> db.users.find({ age: { $gt: 30 } })
[
  {
    _id: ObjectId('66a08a468a57c39e8e149f4a'),
    name: 'Charlie',
    age: 35,
    city: 'Chicago'
  },
  {
    _id: ObjectId('66a08a468a57c39e8e149f4c'),
    name: 'Eve',
    age: 32,
    city: 'Miami'
  }
]
```

11. null: Retrieves documents where the `city` field is null.

```
KushalDB_C_02> db.users.find({ city: null })
```

12. $regex: Retrieves documents where the `name` starts with the letter "A".

```
KushalDB_C_02> db.users.find({ name: { $regex: /^A/ } })
[
  {
    _id: ObjectId('66a08a468a57c39e8e149f48'),
    name: 'Alice',
    age: 30,
    city: 'New York'
  }
]
```

13. pretty(): Displays query results in a formatted, easy-to-read manner.

```
KushalDB_C_02> db.users.find().pretty()
[
  {
    _id: ObjectId('66a08a468a57c39e8e149f48'),
    name: 'Alice',
    age: 30,
    city: 'New York'
  },
  {
    _id: ObjectId('66a08a468a57c39e8e149f49'),
    name: 'Bob',
    age: 25,
    city: 'Los Angeles'
  },
  {
    _id: ObjectId('66a08a468a57c39e8e149f4a'),
    name: 'Charlie',
    age: 35,
    city: 'Chicago'
  },
  {
    _id: ObjectId('66a08a468a57c39e8e149f4b'),
    name: 'David',
    age: 28,
    city: 'Houston'
  },
  {
    _id: ObjectId('66a08a468a57c39e8e149f4c'),
    name: 'Eve',
    age: 32,
    city: 'Miami'
  }
]
```

14. updateOne(): Updates the `age` field of the document where `name` is "Bob" to 26.

```
KushalDB_C_02> db.users.updateOne({ name: "Bob" }, { $set: { age: 26 } })
{
  acknowledged: true,
  insertedId: null,
  matchedCount: 1,
  modifiedCount: 1,
  upsertedCount: 0
}
```

15. insertOne(): Inserts a new document into the `users` collection.

```
KushalDB_C_02> db.users.insertOne({ name: "Frank", age: 40, city: "Seattle" })
{
  acknowledged: true,
  insertedId: ObjectId('66a08b4e8a57c39e8e149f4d')
}
```

16: quit(): Exit the shell.

```
KushalDB_C_02> quit()
```