**Assignment No:6**

**Title:** Design and Develop MongoDB Queries using CRUD operations. (Use CRUD operations, SAVE method, logical operators etc.).

**Problem Statement:** Development of five tailored queries leveraging MongoDB's capabilities, focusing on the meticulous design and meticulous implementation to address the ongoing demands and intricacies of your data operations.

**Objective:** To gain knowledge of NoSQL databases for processing unstructured data.

**Outcome:** Use NoSQL databases for processing unstructured data.

**Tools Required:** Ubuntu OS, MongoDB.

**Theory:**

**1. Introduction:**

MongoDB is a free and open-source NoSQL document database used commonly in modern web applications. MongoDB works on concept of collection and document.

**1.1 Advantages of MongoDB over RDBMS**

* Schema less: MongoDB is a document database in which one collection holds different
* documents. Number of fields, content and size of the document can differ from one
* document to another.
* Structure of a single object is clear.
* No complex joins.
* Deep query-ability. MongoDB supports dynamic queries on documents using a document-based query language that's nearly as powerful as SQL.
* Tuning.
* Ease of scale-out: MongoDB is easy to scale.
* Conversion/mapping of application objects to database objects not needed.
* Uses internal memory for storing the (windowed) working set, enabling faster access
* of data.

**2. MngoDB Command**

**2.1 use Command**

MongoDB use DATABASE\_NAME is used to create database. The command will create a

new database if it doesn't exist, otherwise it will return the existing database.

|  |
| --- |
| **Syntax:**  use DatabaseName |

**2.2 createCollection() Method** MongoDB db.createCollection(name, options) is used to create collection.

|  |
| --- |
| **Syntax:**  db.createCollection("collection\_name") |

**2.3 drop() Method:** MongoDB's db.collection.drop() is used to drop a collection from the database.

|  |
| --- |
| **Syntax:**  db.Collection\_Name.drop() |

**2.4 insert() Method :** To insert data into MongoDB collection, you need to use MongoDB's insert() or save() method.

**insertMany() method:** You can insert multiple documents using the insertMany() method.

|  |
| --- |
| **Syntax:**  db.Collection\_Name.insert(document)  db.empDetails.insertMany([{document 1},{ document 2}..{ document n}])  **Example:**  db.users.insert({title: "DBMS", description: “Database Management System", Credits: 4}) |

**2.5 Update() Method:** The update() method updates the values in the existing document.

|  |
| --- |
| **Syntax:**  db.Collection\_Name.update(Selection\_Criteria, Updated\_Data)  **Example:**  db.mycol.update({'title': DBMS },{$set:{'title':'Advanced DBMS '}}) |

**2.6 find() Method:** To query data from MongoDB collection, you need to use MongoDB's find() method.

|  |
| --- |
| **Syntax:**  db.Collection\_Name.find() |

**2.7 remove() Method:** MongoDB's remove() method is used to remove a document from the collection. remove() method accepts two parameters. One is deletion criteria and second is justOne flag.

|  |
| --- |
| **Syntax:**  db.Collection\_Name.remove(Delletion\_Critteria)  **Example:**  db.mycol.remove({'title':'DBMS’}) |

**2.8 MongoDB Query Operators:**

The following operators can be used in queries to compare values:

* $eq: Values are equal
* $ne: Values are not equal
* $gt: Value is greater than another value
* $gte: Value is greater than or equal to another value
* $lt: Value is less than another value
* $lte: Value is less than or equal to another value
* $in: Value is matched within an array

The following operators can logically compare multiple queries.

* $and: Returns documents where both queries match
* $or: Returns documents where either query matches
* $nor: Returns documents where both queries fail to match
* $not: Returns documents where the query does not match

|  |
| --- |
| **Example:**  db.mycol.find({"likes":{$lt:50}}).pretty()  db.mycol.find({"likes":{$gt:50}}).pretty()  db.mycol.find({"name":{$in:["Raj", "Ram", "Raghu"]}}).pretty()  db.mycol.find({$and:[{"by":"NileshKorade"},{"title": "DataScience"}]}).pretty()  db.mycol.find({$or:[{"by":"NileshKorade"},{"title": " DataScience"}]}).pretty() |

**2.9 sort() Method**

To sort documents in MongoDB, you need to use sort() method. The method accepts a document containing a list of fields along with their sorting order. To specify sorting order 1 and -1 are used. 1 is used for ascending order while -1 is used for descending order.

|  |
| --- |
| **Syntax:**  db.Collection\_Name.find().sort({Key:1}) |

**2.10 Limit() Method**

To limit the records in MongoDB, you need to use limit() method. The method accepts one number type argument, which is the number of documents that you want to be displayed.

|  |
| --- |
| db.Collection\_Name.find().limit(Number) |

**Conclusion:**

We have successfully implemented different operation using MongoDB.