Question 7

Install an Open Source NoSQL Data base MongoDB & perform basic CRUD(Create, Read, Update & Delete) operations. Execute MongoDB basic Queries using CRUD operations.

Solution

1. Installing Open Source NoSQL Data base MongoDB

Please refer to the blog below which contains detailed procedure of installing Open Source NoSQL Data base MongoDB.

2. Perform basic CRUD(Create, Read, Update & Delete) operations.

1. Start MongoDB.

Launch the MongoDB daemon using the following command:

sudo systemctl start mongod

2. Start the MongoDB Shell

Launch the MongoDB shell to perform basic CRUD operations.

mongosh

3. Switch to a Database (Optional):

If you want to use a specific database, switch to that database using the use command. If the database doesn’t exist, MongoDB will create it implicitly when you insert data into it:

test> use bookDB

switched to db bookDB

bookDB>

4. Create the ProgrammingBooks Collection:

To create the ProgrammingBooks collection, use the createCollection() method. This step is optional because MongoDB will automatically create the collection when you insert data into it, but you can explicitly create it if needed:

bookDB> db.createCollection("ProgrammingBooks")

This command will create an empty ProgrammingBooks collection in the current database (bookDB).

5. INSERT operations

a. Insert 5 Documents into the ProgrammingBooks Collection :

Now, insert 5 documents representing programming books into the ProgrammingBooks collection using the insertMany() method:

bookDB> db.ProgrammingBooks.insertMany([

{

title: "Clean Code: A Handbook of Agile Software Craftsmanship",

author: "Robert C. Martin",

category: "Software Development",

year: 2008

},

{

title: "JavaScript: The Good Parts",

author: "Douglas Crockford",

category: "JavaScript",

year: 2008

},

{

title: "Design Patterns: Elements of Reusable Object-Oriented Software",

author: "Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides",

category: "Software Design",

year: 1994

},

{

title: "Introduction to Algorithms",

author: "Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein",

category: "Algorithms",

year: 1990

},

{

title: "Python Crash Course: A Hands-On, Project-Based Introduction to Programming",

author: "Eric Matthes",

category: "Python",

year: 2015

}

])

b. Insert a Single Document into ProgrammingBooks:

Use the insertOne() method to insert a new document into the ProgrammingBooks collection:

bookDB> db.ProgrammingBooks.insertOne({

title: "The Pragmatic Programmer: Your Journey to Mastery",

author: "David Thomas, Andrew Hunt",

category: "Software Development",

year: 1999

})

6. Read (Query) Operations

a. Find All Documents

To retrieve all documents from the ProgrammingBooks collection:

bookDB> db.ProgrammingBooks.find().pretty()

[

{

\_id: ObjectId('663eaaebae582498972202df'),

title: 'Clean Code: A Handbook of Agile Software Craftsmanship',

author: 'Robert C. Martin',

category: 'Software Development',

year: 2008

},

{

\_id: ObjectId('663eaaebae582498972202e0'),

title: 'JavaScript: The Good Parts',

author: 'Douglas Crockford',

category: 'JavaScript',

year: 2008

},

{

\_id: ObjectId('663eaaebae582498972202e1'),

title: 'Design Patterns: Elements of Reusable Object-Oriented Software',

author: 'Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides',

category: 'Software Design',

year: 1994

},

{

\_id: ObjectId('663eaaebae582498972202e2'),

title: 'Introduction to Algorithms',

author: 'Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein',

category: 'Algorithms',

year: 1990

},

{

\_id: ObjectId('663eaaebae582498972202e3'),

title: 'Python Crash Course: A Hands-On, Project-Based Introduction to Programming',

author: 'Eric Matthes',

category: 'Python',

year: 2015

},

{

\_id: ObjectId('663eab05ae582498972202e4'),

title: 'The Pragmatic Programmer: Your Journey to Mastery',

author: 'David Thomas, Andrew Hunt',

category: 'Software Development',

year: 1999

}

]

b. Find Documents Matching a Condition

To find books published after the year 2000:

bookDB> db.ProgrammingBooks.find({ year: { $gt: 2000 } }).pretty()

[

{

\_id: ObjectId('663eaaebae582498972202df'),

title: 'Clean Code: A Handbook of Agile Software Craftsmanship',

author: 'Robert C. Martin',

category: 'Software Development',

year: 2008

},

{

\_id: ObjectId('663eaaebae582498972202e0'),

title: 'JavaScript: The Good Parts',

author: 'Douglas Crockford',

category: 'JavaScript',

year: 2008

},

{

\_id: ObjectId('663eaaebae582498972202e3'),

title: 'Python Crash Course: A Hands-On, Project-Based Introduction to Programming',

author: 'Eric Matthes',

category: 'Python',

year: 2015

}

]

7. Update Operations

a. Update a Single Document

To update a specific book (e.g., change the author of a book):

bookDB>db.ProgrammingBooks.updateOne(

{ title: "Clean Code: A Handbook of Agile Software Craftsmanship" },

{ $set: { author: "Robert C. Martin (Uncle Bob)" } }

)

//verify by displaying books published in year 2008

bookDB> db.ProgrammingBooks.find({ year: { $eq: 2008 } }).pretty()

[

{

\_id: ObjectId('663eaaebae582498972202df'),

title: 'Clean Code: A Handbook of Agile Software Craftsmanship',

author: 'Robert C. Martin (Uncle Bob)',

category: 'Software Development',

year: 2008

},

{

\_id: ObjectId('663eaaebae582498972202e0'),

title: 'JavaScript: The Good Parts',

author: 'Douglas Crockford',

category: 'JavaScript',

year: 2008

}

]

b. Update Multiple Documents

To update multiple books (e.g., update the category of books published before 2010):

bookDB> db.ProgrammingBooks.updateMany(

{ year: { $lt: 2010 } },

{ $set: { category: "Classic Programming Books" } }

)

//verify the update operation by displaying books published before year 2010

bookDB> db.ProgrammingBooks.find({ year: { $lt: 2010 } }).pretty()

[

{

\_id: ObjectId('663eaaebae582498972202df'),

title: 'Clean Code: A Handbook of Agile Software Craftsmanship',

author: 'Robert C. Martin (Uncle Bob)',

category: 'Classic Programming Books',

year: 2008

},

{

\_id: ObjectId('663eaaebae582498972202e0'),

title: 'JavaScript: The Good Parts',

author: 'Douglas Crockford',

category: 'Classic Programming Books',

year: 2008

},

{

\_id: ObjectId('663eaaebae582498972202e1'),

title: 'Design Patterns: Elements of Reusable Object-Oriented Software',

author: 'Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides',

category: 'Classic Programming Books',

year: 1994

},

{

\_id: ObjectId('663eaaebae582498972202e2'),

title: 'Introduction to Algorithms',

author: 'Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein',

category: 'Classic Programming Books',

year: 1990

},

{

\_id: ObjectId('663eab05ae582498972202e4'),

title: 'The Pragmatic Programmer: Your Journey to Mastery',

author: 'David Thomas, Andrew Hunt',

category: 'Classic Programming Books',

year: 1999

}

]

8. Delete Operations

a. Delete a Single Document

To delete a specific book from the collection (e.g., delete a book by title):

bookDB> db.ProgrammingBooks.deleteOne({ title: "JavaScript: The Good Parts" })

{ acknowledged: true, deletedCount: 1 }

You can check whether the specified document is deleted by displaying the contents of the collection.

b. Delete Multiple Documents

To delete multiple books based on a condition (e.g., delete all books published before 1995):

bookDB> db.ProgrammingBooks.deleteMany({ year: { $lt: 1995 } })

{ acknowledged: true, deletedCount: 2 }

You can check whether the specified documents were deleted by displaying the contents of the collection.

c. Delete All Documents in the Collection:

To delete all documents in a collection (e.g., ProgrammingBooks), use the deleteMany() method with an empty filter {}:

//delete all documents in a collection

bookDB> db.ProgrammingBooks.deleteMany({})

{ acknowledged: true, deletedCount: 3 }

//verify by displaying the collection

bookDB> db.ProgrammingBooks.find().pretty()

9. Delete the Collection Using drop():

To delete a collection named ProgrammingBooks, use the drop() method with the name of the collection:

bookDB> show collections

ProgrammingBooks

bookDB> db.ProgrammingBooks.drop()

true

bookDB> show collections

bookDB>

The command db.ProgrammingBooks.drop( ) will permanently delete the ProgrammingBooks collection from the current database (bookDB).

After deleting the collection, you can verify that it no longer exists by listing all collections in the database using the command show collections.