Date : 06/04/2021
Spring Boot 9AM
Mr. RAGHU

Spring Data MongoDB Operations

1. save(obj) : This method is used to execute either
   INSERT or UPDATE (based on ID/PK)

 Given Object is converted into JSON Format and stored
  in Collection (inside MongoDB Database)

2. saveAll(Iterable<T> list) : This called as bulk
   insert/update.

 We can provide list of objects for insert/update.

3. findById(id):Optional<T>
   This method is used to fetch one row data if exist
   else null. Read data using get() method after doing
   null check using isEmpty(), isPresent() methods

4. findAll(): List<T>
   This method is used to fetch all rows data from DB
   Collection

5. existsById(ID):boolean
    This method is find , given id exist in collection or not?

6. findAllById(Iterable<T> list) :List<T>
    This method is used to fetch selected rows by id.

7. deleteById(id):void
    this method is used to remove one JSON document by id

8. deleteAll(): This method is used to remove all JSON
   Documents from Collection.

==========

9. findAll(Pageable<T>):Page
   This method is used to implement pagination concept
   by providing input like Pagenumber and Page size.

10. findAll(Sort) :List<T>
    This method is used to fetch data by sorting operations

----------Full Code----------

#1. Create Starter Project
Name : SpringBoot2DataMongoDbCrudApp
Dep : Spring Data MongoDB, Lombok

#2. Model class

```
package in.nareshit.raghu.model;

import org.springframework.data.annotation.Id;
```

```
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor;

@Data
@NoArgsConstructor
@AllArgsConstructor
public class Book {
        @Id
        private Integer bookId;
        private String bookName;
        private Double bookCost;

}
#3. Repository Interface
package in.nareshit.raghu.repo;

import org.springframework.data.mongodb.repository.MongoRepository;
import in.nareshit.raghu.model.Book;
public interface BookRepository
        extends MongoRepository<Book, Integer> {

}

#4 Generator class
package in.nareshit.raghu;
import java.util.Random;
public interface MyIdGen {

        public static int getId() {
                return new Random().nextInt(999999);
        }
}

#5. Runner class
package in.nareshit.raghu.runner;

import java.util.Arrays;
import java.util.List;
import java.util.Optional;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.data.domain.Page;
import org.springframework.data.domain.PageRequest;
import org.springframework.data.domain.Sort;
import org.springframework.stereotype.Component;

import in.nareshit.raghu.MyIdGen;
import in.nareshit.raghu.model.Book;
import in.nareshit.raghu.repo.BookRepository;

@Component
public class BookTestRunner implements CommandLineRunner {
        @Autowired
        private BookRepository repo;
```

```
	public void run(String... args) throws Exception {
		//for ddl-auto=create
		repo.deleteAll();

		//1. save() - insert/update
		repo.save(new Book(MyIdGen.getId(), "Core Java",
500.0));
		repo.save(new Book(1506, "Adv Java", 600.0));
		Book b = repo.save(new Book(1507, "SpringBoot",
800.0));
		System.out.println(b);


		repo.saveAll(
				Arrays.asList(
						new Book(2508,
"Angular", 600.0),
						new Book(2509, "MS",
800.0),
						new Book(2510, "HTML",
900.0)
						)
				);

		Optional<Book> opt = repo.findById(2508);
		if(opt.isEmpty()) {
			System.out.println("Data not exist");
		} else {
			System.out.println("Data is " + opt.get());
		}

		boolean exist = repo.existsById(1100);
		System.out.println("Is exist ?" + exist);

		System.out.println("------------------");
		repo.findAll()
		.forEach(System.out::println);
		System.out.println("------------------");
		repo.findAllById(
				Arrays.asList(1506,2508,2510)
				).forEach(System.out::println);
		System.out.println("------------------");
		System.out.println("Total rows " + repo.count());

		//repo.deleteById(1506);
		//repo.deleteAll();

		System.out.println("-----------DONE-------");

		Page<Book> page = repo.findAll(PageRequest.of(0, 3));
		System.out.println("Is Empty page " + page.isEmpty());
		List<Book> list =  page.getContent();
		list.forEach(System.out::println);

		System.out.println("Is First Page " + page.isFirst());
		System.out.println("Is Last Page " + page.isLast());
```

```
                System.out.println("Has Next Page " + page.hasNext());
                System.out.println("HAs previous Page " +
page.hasPrevious());

                System.out.println("Page size " + page.getSize());
                System.out.println("Total Docs " +
page.getTotalElements());
                System.out.println("Total Pages " +
page.getTotalPages());

                //================================================
                System.out.println("**********************");
                //Sort sort = Sort.by(Direction.DESC,"bookName");
                //Sort sort =
Sort.by(Order.asc("bookName"),Order.desc("bookCost"));
                Sort sort = Sort.by("bookCost");
                repo.findAll(sort)
                .forEach(System.out::println);



        }

}
============================================================
Q) What are defaults for MongoDB Connection using
   Spring Boot?
A) host default localhost
   port default 27017
   database default test
   Comes with no Security concept
       ie username and password are null.

----------MongoDB Client Commands---------------
#1. Start MongoDb Server (non-secure mode)
cmd> mongod

#2. Start MongoDB Client
cmd> mongo

#3 Goto Database
> use nit

#4.*** create user with name,pwd and roles
> db.createUser(
   {
        user : "NIT",
        pwd  : "RAGHU",
        roles: ["readWrite","dbAdmin"]
   }
  )

#5 Stop both MongoDB Client and server
   (ctrl+C)

#6.*** Start MongoDb Server (in secure mode)
cmd> mongod --auth
```

```
#7. Start MongoDB Client
cmd> mongo

#8  Goto Database
> use nit

#9 View Collections
> show collections

#10.*** Login
> db.auth("NIT","RAGHU")

.. now execute cmds...

#11.*** Logout
> db.logout()

*) Incase of login failed/did not login for cmds
  "errmsg" : "there are no users authenticated",
  "code" : 13,
  "codeName" : "Unauthorized"

Q) What is the diff b/w cmds :
    a. mongod
    b. mongod --auth

A)
  cmd : mongod => This commands is used to start
         Mongo Database server without security
         [Non-Secure Mode]
  ie any application/client/Program can communicate
  with MongoDB server without any username/pwd.

  cmd: mongod --auth
       => This commands is used to start
         Mongo Database server in security mode
  ie any application/client/Program can communicate
  with MongoDB server they must provide username/pwd.
=========================================================
*) At Our Application we need to provide username and
   password when MongoDB server started in securty
   mode.
   Else  MongoCommandException: Command failed with error
   13 (Unauthorized): 'not authorized on nit to execute
                       command'

 in Spring Boot application add below keys:
   spring.data.mongodb.username=NIT
   spring.data.mongodb.password=RAGHU

*) if we provide invalid user/pwd
   spring.data.mongodb.username=SAMPLE
   spring.data.mongodb.password=NODATA

   MongoCommandException: Command failed with error
   18 (AuthenticationFailed)
```

```
=======================================================
a. findBy : same as like as JPA [SELECT]

--Repository---
package in.nareshit.raghu.repo;
public interface BookRepository
        extends MongoRepository<Book, Integer> {

        List<Book> findByBookCostLessThan(Double cost);
        List<Book> findByBookName(String name);
}
--Runner---
package in.nareshit.raghu.runner;
@Component
public class FindByTestRunner implements CommandLineRunner {

        @Autowired
        private BookRepository repo;

        public void run(String... args) throws Exception {
                //repo.findByBookCostLessThan(900.0)
                repo.findByBookName("SpringBoot")
                .forEach(System.out::println);
        }

}
-----------------------------------------
b. @Qurey : it is different.
```