Date : 08/04/2021
Spring Boot 9AM
Mr. RAGHU

MongoDB:
https://docs.mongodb.com/v3.6/crud/

Projections using MongoDb Query

*) By default custom query is fetching all variables/keys data
   from collection.

*) But this time using Projections to fetch only selected variables/keys
   from the collection.

```
   @Query(value="{ ... condition...}", fields =" {...variable : 1.. }" )
```

=> Here Fields section is used to select variables for display
   one (:1) means display variable, zeor(:0) means do not display variables

```
=> Default values for display
   PrimaryKey Variable ------- default is 1
   other variables     ------- default is 0
```

*)Output (ReturnType of method) Custom Query for Project is only
  two Types a. String b. ClassType

  ex: List<String>, String, List<Book>, Book. No Object[], Double..etc
        allowed.

=========Code==============

1. Model

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

import org.springframework.data.annotation.Id;
import org.springframework.data.mongodb.core.mapping.Document;

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

@Data
@NoArgsConstructor
@AllArgsConstructor
@Document
public class Book {
        @Id
        private Integer bookId;
        private String bookName;
        private String bookAuthor;
        private Double bookCost;
        private String bookType;
}
```

2. Repository

```
package in.nareshit.raghu.repo;

import java.util.List;

import org.springframework.data.mongodb.repository.MongoRepository;
import org.springframework.data.mongodb.repository.Query;

import in.nareshit.raghu.model.Book;

public interface BookRepository
        extends MongoRepository<Book, String> {


        //SQL: select bookName,bookAuthor from book where bid>?
        @Query(value = "{ bookId : { $gt: ?0 } }",fields = "{
bookName:1, bookAuthor:1 }")
        List<String> getAllBooksById(Integer bookId);
        //List<Book> getAllBooksById(Integer bookId);

        //SQL: select bookAuthor from Book where bookType=?
        @Query(value = "{ bookType : ?0 }", fields = "{ bookAuthor:1 ,
_id:0 }")
        List<String> getBooksDataA(String bookType);

        //SELECT * FROM BOOK WHERE bookAuthor like ?
        @Query("{ bookAuthor : { $regex : ?0 } }")
        List<Book> getBooksByAuthor(String input);

        //SELECT * FROM BOOK WHERE bookId In (....)
        @Query("{ bookId : { $in : ?0 } }")
        List<Book> getBooksByIds(List<Integer> ids);

        //SELECT * FROM BOOK WHERE bookId between ?1 and ?2
        @Query("{ bookId : { $gt : ?0 , $lt: ?1 } }")
        List<Book> getBooksByIdsBetween(Integer id1,Integer id2);

        //SELECT count(*) FROM BOOK WHERE bookId >?
        @Query(value = "{ bookId : { $gt : ?0} }", count = true)
        Integer getBooksCount(Integer id);


        //SELECT * FROM BOOK WHERE bookId >? order by bookType
        @Query(value = "{ bookId : { $gt : ?0} }", sort = "{ bookType:
1}")  // ASC
        //@Query(value = "{ bookId : { $gt : ?0} }", sort = "{
bookType: -1}")  // DESC
        List<Book> getBooksInOrder(Integer id);

        @Query(value = "{ bookId : { $gt : ?0} }", delete = true)
        Long deleteBooksById(Integer id);

}
```

3. Data Insert Runner

```
package in.nareshit.raghu.runner;
```

```
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.stereotype.Component;

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

//@Component
public class MyDataRunner implements CommandLineRunner {

        @Autowired
        private BookRepository repo;

        public void run(String... args) throws Exception {
                repo.deleteAll();

                repo.save(new Book(2505, "Spring Boot", "SAM",
500.0,"BackEnd"));
                repo.save(new Book(2506, "Microservices", "SYED",
600.0,"BackEnd"));
                repo.save(new Book(2507, "Angular 11", "RAM",
800.0,"FrontEnd"));

        }

}
```

4. test Runner

```
package in.nareshit.raghu.runner;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.stereotype.Component;

import in.nareshit.raghu.repo.BookRepository;

@Component
public class DataTestRunner implements CommandLineRunner {

        @Autowired
        private BookRepository repo;

        public void run(String... args) throws Exception {
                //repo.getAllBooksById(100)
                //repo.getBooksDataA("BackEnd")
                //repo.getBooksByAuthor("^S")  //starting with
^<input>
                //repo.getBooksByAuthor("M$")  //ending with <input>$
                //repo.getBooksByAuthor("A")  //Containing  <input>
                //repo.getBooksByIds(Arrays.asList(2505,
2507,33,44,55)) //in operator
                //repo.getBooksByIdsBetween(2500, 2900)
                //repo.getBooksInOrder(2500)
                //.forEach(System.out::println);


                //Integer count = repo.getBooksCount(2500);
                Long count = repo.deleteBooksById(2500);
                System.out.println(count);

        }

}
```

5. application.properties

```
spring.data.mongodb.host=localhost
spring.data.mongodb.port=27017
spring.data.mongodb.database=nit
```