LombokAPI,Eclipse Debugging => Session on 17th April,timing :: 7.30PM IST
link will be shared to both the batches

Plz watch navinreddy sir youtube video of ====> SpringMongoDB
link::
https://www.youtube.com/watch?v=kYiLzIiHVY8

Morning session[Enterprise java batch]
==========
Tuesday,Wednesday,Thursday ===> Timings :: 6.30AM IST to 9.00AM IST
topics pending: SpringBootMongoDB, Spring AOP,Spring Security,Spring Mail

JpaRepository(I)
==========
findAll(Example<S> example,Sort sort)

Example Object is a Container object holding Entity Object,It is just like Optional<T>object.

application.properties
==========
spring.jpa.properties.hibernate.enable_lazy_load_no_trans=true //required while working with getById(),getReferenceById(,)

When we use JpaRepository, we need to enable lazy loading through a special property of hibernate as shown above.

What is the difference b/w deleteAllByIdInBatch(Iterable<ID> ids) of JpaRepository and deleteAllById(Iterable<ID> ids) of CRUD Repository?
deleteAllByIdInBatch(Iterable<ID> ids) => generates single sql delete query having in clause to delete the records,if id's not avaialble it
will not throw any Exception.

this is JPA Repository

deleteAllById(Iterable<ID> ids) => generates multiple sql delete query to delete multiple records of the given ids,if any one of the
given id is not available then it would throw Exception.

What is the difference b/w findAll() methods of different repositories?
findAll()=> JpaRepository => sorting available, no pagination,passing of Example object,return type :: List<T>
findAll()=> CrudRepository => sorting not avalable,no pagination, no passing of Example Object, return type :: Iterable<T>
findAll()=> PagingAndSortingRepository => sorting available, pagination available, no passing of Example object ,return type :: Iterable<T>

Note:
save() => It comes from CrudRepository, we can perform both insert and update operation,it this process to perform commit and rollback
operation, it takes the support of TransactionManger(tx.commit(),tx.rollback()).
saveAndFlush() => It comess from JpaRepository, we can perform both insert and

update operation,in this process it uses flush() to write the
                        changes to the database without any TransactionManger
support.

=> Prefer using CrudRepository and PagingAndSortingRepository becoz these
repositories are common repositories while working with
            SpringData-JDBC,SpringData-JPA and SpringData-Mongodb,.....

Custom Persitence Operations in SpringDataJPA
===
 1. To Perform peristence operation with our choice conditions
 2. To execute HQL,SQLqueries,NativeSQL Queries
 3. To call StoredProcedures and To perform insertion of BLOB/CLOB

Mechanisms
==========
a. finder methods(only for select operation)
b. @Query methods(To execute HQL/JPQL,native sql select queries)
c. @Query + @Modifying Methods(To execute HQL/JPQL,native sql non-select queries)

finder methods
==============
 => These are custom abstract methods placed in our repository interface which will
be converted into select sql query.
 => It support both Entity select operation(all col values) and scalar select
operation(specific col values)[Projection]
 => We can prepare finder methods having one or more conditions with different
clauses like and,or,in,....
            syntax:: public <Return Type>
findBy<propertyNames><conditions>(params...)
 => Implementation of finder methods takes place in the spring data jpa generated
InMemoryProxy class.
 => we can take finder method without any condition,then by default condition that
will be applied is "equals(=)" on the given property/column.

```
@Entity
public class CoronaVaccine implements Serializable {
      private static final long serialVersionUID = 1L;
      @Id
      @GeneratedValue(strategy = GenerationType.IDENTITY)
      private Long regNo;
      private String name;
      private String company;
      private String country;
      private Double price;
      private Integer requriedDoseCount;
}

public interface ICoronaVaccineRepo extends JpaRepository<CoronaVaccine, Long> {
      public List<CornoVaccine> findByCompany(String company)
}
```

            eg::DAO-SpringDataJPA-CustomQueryApp

Static Projection
=================
 In case of Static Projection,we have 2 Proxy classes

```
      a. ResultView(I) =======> To hold the Result[column names] returned by the
Query
      b. ICoronaVaccineRepo(I) ===> To Represent a DAO Repository(take the help of
JpaRepository)

In case of Dynamic Projection
   => here we can get varying specific single column or mulitple columns from
dbtable using the support of finder methods.
   => For this support we take multiple types of interface having hierarchy as show
below

interface View{

}
interface ResultView1 extends View{
      public String getName();
      public String getCompany();
}

interface ResultView2 extends View{
      public Long getRegNo();
      public Double getPrice();
      public String getCountry();
}
interface ResultView3 extends ResultView1{
      public String getPrice();
}


DaoLayer
========
public interface ICoronaVaccineRepo extends JpaRepository<CoronaVaccine, Long> {
      public <T extends View> List<T> findByCompanyOrderByCompanyDesc(String
company, Class<T> clazz);
}                 |
|
            Controlling Type of 'T' which is returned as List<T>
RunTime supplied class


Executing Storedprocedure
=========================

USE `enterprisejavabatch`$$

DROP PROCEDURE IF EXISTS `P_GET_PRODUCT_BY_NAME`$$

CREATE DEFINER=`root`@`localhost` PROCEDURE `P_GET_PRODUCT_BY_NAME`(IN name1
VARCHAR(20), IN name2 VARCHAR(20))
BEGIN
            SELECT pid,pname,price,qty FROM products WHERE pname IN (name1,name2);
      END$$

DELIMITER ;

                        refer:: DAO-SpringDataJPA-StoredProcedureApp

Working with Date and Time Operation
====================================
```

JDK8 features(Jodha API)
LocalDateTime,LocalDate,LocalTime

refer:: DAO-SpringDataJPA-DateTimeInsertionAPI