```
finder methods / findby Methods __ in spring data [pa
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                m ways of writing pensistrence logics
                                                                                                                                                                                                                                                                                                                                                                                                                                                            in spring data [pe
a] using pre-definedRepository methods
  **These are abstract method declarations done in corresponding by following correlations converties:
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            ()Creditopository ii) PagingAndSortingHopository
II) JosEspository

    Using these we can perform only Select Operations and Non-Select Operations
are not possible

                                                                                                                                                                                                                                                                                                                                                                                                                                                                 II) Jasiapository
b) finder methods in surreguestory (select operations)
c) @Covey methods (To use POL. SO/Covering for select operations)
d) @Covey + @Modifying methods (for tion -elect operations)
are not possible within implantation in the inflations phase possible within finder methods within implantation in the inflations; they class of Appaidery Interface general/PES and SQL quarter internally ... with exhibit operations which are not possible with finds only methods of different positions. Department on the developed using using those custom finder anniholds.
                                                                                                                                                                                                                                                                                                                                                                                                                                                                   The finder nestods of greedefined Repositories are such to perform either built select operations or suggestions which operation may high length who an incretionis value, increase perform finder related perform by taking other Wissians as the orbital values we need to conclude the finder residuels.
                      There finder methods can be used for

    d) Entity operations (selecting all the cell values of the table)
    eg: celect ** from Movie (or)

                                                                                                                                                                                                                                                                                                                          ddc) all pole of the oblitable
(or) select raid, came, undergover, from Movie where year in [7,2,2).
                          b) Scalar operations/Projections (wheeting specific single color mustiple colorates of Chinatis)
eg: volum midgle anne from hinde where mids 2 and mids 2
eg: volum mids midses/ midses/ midses/
eg: volum midses/ midses/ midses/ midses/
eg: volum midses/ midses/ midses/ midses/
eg: volum midses/ midses/ midses/
eg: volum midses/ midses/ midses/
eg: volum midses/ midses/ midses/ midses/
eg: volum midses/ midses/ midses/ midses/ midses/
eg: volum midses/ midses/ midses/ midses/ midses/
eg: volum midses/ midses/ midses/ midses/ midses/ midses/
eg: volum midses/ m
                 Example on Brilly Queries (select queries giving 0 or more records by selecting all collectors).
                      In Repsolary interface
                          public interbunt MassesSeptember in New Republic of Wome, Integers (
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        Why do we need findByXxx() methods as the custom methods in Repository interfaces?
                                                   (Juded mid, managen, juding) has Markinshar manager 
sold of the North Andry Interesting and the Interesting area of 
Juded and programs, and allow from Markinshare manager 
sold of the North Andry Interesting and and 
Juded and programs, and allow from Markinshare manager 
public to the North Andry Interesting and 
Juded and programs and allow from Markinshare manager 
public to the North Andry Interest 
Jude to Andry 
Ju
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          Ani) The pre-defined Mederil is methods of pre-defined Repository Interfaces like 
flexibles(*) inclusifiespository and perioderenosity(*) intibility production are captable 
performing packet operations only by Ishing (#8) operative wise the critical value 
flax we want to relect the records (relects based on other preparity/properties using 
site critical values. For that we need these custom index (fundational) methods 
are the critical values. For that we need these custom index (fundational) methods.
                                                                                                                                                                                                                                                                                                                                                                                           All the 3 parthods are
                                                                                                                                                                                                                     none: If condition is not specified in findily methods
then e-[equal to] condition on the given property name will be applied externationly
                      to Hunner taks
                               #Component public Nothink Test Rouges implements Communition Rouges (
                                                            \#Autoralind private Movie Report (in Memory on x_i and x_i and x_i in Report (and the injected private X).
                                                                                                                                                                                                                                                                                                                                                                                                                               In MySUL, the finder method generated SQL Query
does not apply one mentionly by default, where as
the Oracle generated SQL Query applies non-mentionly
by default, by incoming in many layers care Semiliarity
and the meth. "Applichtpure/Gen?"
                                                          public tist-Artists-
findityCategorySquaddymenCate(String category);
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            Imity data name
                                                                 nose: In our repository interface if the findity () method mixim type Burd's or increased stated intermedity generated States State (see year) perform on the past for given condition on given property(vol.)
                                                                                       public test Mexico. (miles Mexico quick transport production and participation of the second condition generate density (where manuary) relect a new
                                                                                                                                                                                                                                                                                                                                                                                                                                                              "PLIANCE or transpless generic type
Living or transpless generic type
Living or over-defined
insertice/class than in internally generates
sodar select SUL query (selecting specific
points values)
                                                                                                                relect query
(select * from «Table» or
velect col1.col2, ./all cols) from «Table»)
                                                                                                                                note: The rb table names, colinames are not case sensitive .. but colidata/values are case sensitive
                                                                                                                                                                                                                                                                    none: from upday Ås we can develop flatfolded) methods also as the 
<u>goldpited</u> for modifylated methods, but other words like stonelysted), footbylated, as no sistened
      Reference Image for finder methods
                                                        Service Address Addres
                                                                                                                                    Gelfelbest entrol (Add provided provide
                                                            Deliverable
                                                                                                                                    Settle Control of the settle o
                                                              Repository Interface
                      public interface (Massic Representation), Qui Reposition y Winnie, Integra 2 (
                                               (find ed mid, manny, year, midney from Minde where gramme?)
subtlet are started fractionaries and plant of many.

(find ed mid, manny, year, sidns, from Minde where gramme?)

(find ed mid, manny, year, sidns, from Minde where gramme?)

(find ed mid, midney, year, sidns, from Minde where gramme?)

(find ed mid, midney, year, sidns, from Minde where gramme?)

(add all and de Minde where find Myenners (Manny, see all).
                                                 //select mid.gramm.your, ratings from Movie where gramm. The "Ref
public transfer-through finding-frames terting/Whi(string init/frame):
                                                   //webst and, annuary, year, attags, from Moves where area are like for
public iterable: Novie: - findDyNisameCodingWithString is others):
                                                   //select mid.gng.ng.year.ratings from Movie where gng.ng like "K<u>dha</u>W public literable shire in the Explanance Containing (String share).
                                                 //select mid.govers.year.rethus from Movie where govers. The Yighe W public nearbles storage find by smarre-quality constant-jump name);
                                                                              //white it makes many processings from Whom where no one his "Sallest" public hembles Movies Proble Whome Constraining procedure String than it
                                                                            Halling this grants particular from their when grants like Neil III works training with R. Bucket this grants securellars from Neik when grants like Neil III works training with R. Bucket and proving security from their works and the security with the security particular their security with the security of the securi
```

Incrementation

republish (Manus Pyude "Anthor") ka hakiliya kana adaprishigi System asa prishigi """ k sesa ka da William da Kilon Handardi System asa sakishi at System asa prishigi "" k

1

finder methods / findBy Methods

in spring data jpa interface =>These are abstract method declarations done in our repository by following certain naming convetions

- =>Using these we can perform only Select Operations and Non-Select Operations are not possible
- => These finder methods will be implemented in the InMemory Proxy class of Repository Interface generating Select SQL queries internally ...
- => The select operations which are not possible with findXxx() methods of differnt prefined Repositories can be developed using using these custom finder methods..

syntax::

fixed

<public> <RT> findByypropertyName(s)><Condition(s)>(<params>)

getBy readBy

- =>These finder methods can be used for
- a) Entity operations (selecting all the col values of db table) eg: select * from Movie

Different ways of writing persisstence logics

in spring data jpa

- a) using pre-defined Repository methods
- i) CrudRepository ii) PagingAndSortingRepository iii) JpaRepository
- b) finder methods in our repository (select operations)
- c) @Query methods (To use JPQL, SQLQueries for select operations)
- d) @Query + @Modifying methods (for Non-select operations)
- =>The finder methods of pre-defined Repositories are useful to perform either bulk select operations or single record select operation only by taking id value as the criteria value.. In order to perform similar select operations by taking other id values as the criteria values we need to use these finder methods

all cols of the db table

(or) select mid,name,ratings,year from Movie where year in(?,?,?)

b) Scalar opeations/Projections (selecting specific single col or multiple col values of Db table)

eg:

select mid,name from Movie where mid>=? and mid<=?

eg:

eg:

select mid, year from Movie where mid>=? and mid<=? select name, year from Movie where mid>=? and mid<=?

Example on Entity Queries (select queries giving 0 or more records by selecting all col values) In Repsoitory interface

public interface IMovieRepo extends JpaRepository<Movie, Integer> {

//select mid,mname,year,ratings from Movie where mname=? public List<Movie> findByMnameEquals(String name); //select mid,mname, year, ratings from Movie where mname=? public List<Movie>

```
findByMnamels(String name); //select mid,mname,year,ratings from Movie where mname=? public
List<Movie> findBymname (String name);
}
In Runner calss
All the 3 methods are same
note:: if condition is not specified in findBy____ methods
Why do we need findByXxx() methods as the custom methods in Repository interfaces? Ans) The
pre-defined finder() methods of pre-defined Repository interfaces like findByld(-) inCrud Repository and
getReferenceByld(-) inJPARepository are captable performing select operations only by taking @ld property
value the criteria value But we want to select the records /objects based on other prpperty/properties values
as the criteria values.. for that we need these custom finder /findByXxx() methods
then =(equal to) condition on the given property name will be applied automatically
@Component
public class FinderMethodsTestRunner implements CommandLineRunner { @Autowired
private IMovieRepo repo; // InMememory proxy class obj of our Repository(I) will be injected
@Override
public void run(String... args) throws Exception {
//=====finder methods ========
repo.findByMnameEquals("Anthim").forEach(System.out::println);
System.out.println("
repo.findByMnamels("Don").forEach(System.out::println);
-");
System.out.println(".
-");
repo.findBymname("RRR").forEach(System.out::println);
In MySQL, the finder method generated SQL Query does not apply case sensitivity by default.. where as the
Oracle generated SQL Query applies case sensitivity by default. So in oracle if want ignore case Sensitivity
add the work "EqualsIgnoreCase"
public List<Artist>
findByCategoryEqualsIgnoreCase(String category);
}
Entity class name
note:: In our repository interface if the findBy_____() method return type List<T> or Iterable<T> then
internally generated SELECT SQL Query will perform db table for given condition on given property/col)
entity Operation (select all col values of
public List<Movie> findByMnameEquals(String name);
ग् generate entity
select query
```

```
generates the where clause condtion (where mname=?)
(select * from <Table> or
select col1,col2,..(all cols) from <Table> )
=>n List<-> or Iterrable<-> generic type having class
is other than entity like String or user-defined interface/class then it internally generates scalar select SQL
query (selecting specific colum values)
note:: The db table names, col names are not case-sensitive.. but col data/values are case-sensitive.
Reference image for finder methods
note:: from spring 3.x we can develop findByXxx() methods also as the
getByXxx() or readByXxx() methods. but other words like showByXxx(), fetchByXxx() are not allowed
Keyword
And
Or
Is, Equals
Between
LessThan
LessThanEqual
Greater Than
Greater ThanEqual
After
Before
IsNull
findByAgeGreater ThanEqual
findByStartDateAfter
findByStartDateBefore
findByAgeIsNull
IsNotNull, NotNull findByAge (Is) NotNull
Like
NotLike
findByFirstnameLike
findByFirstnameNotLike
Sample
{\tt findByLastnameAndFirstname}
findByLastnameOrFirstname
JPQL snippet
...where x.lastname = ?1 and x.firstname = ?2 ...where x.lastname = ?1 or x.firstname
= ?2
```

```
findByFirstname, findByFirstnameIs, findByFirstname Equals ... where x.firstname = 1?
findByStartDateBetween
findByAgeLessThan
findByAgeLess Than Equal
{\tt findByAgeGreaterThan}
... where x.startDate between 1? and ?2 ...where x.age < ?1
\dotswhere x.age <= ?1
\dotswhere x.age > ?1
...where x.age >= ?1
StartingWith
{\tt findByFirstnameStartingWith}
Endingwith
findByFirstnameEndingWith
Containing
findByFirstnameContaining
OrderBy
findByAgeOrderByLastnameDesc
Not
Ιn
findByLastnameNot
findByAgeIn(Collection<Age> ages)
...where x.startDate > ?1
...where x.startDate < ?1 ...where x.age is null where x.age not null
...where x.firstname like ?1
where x.firstname not like ?1
... where x.firstname like ?1 (parameter bound with appended %)
where x.firstname like ?1 (parameter bound with prepended %)
... where x.firstname like ?1 (parameter bound wrapped in %)
where x.age = ?1 order by x.lastname desc
...where x.lastname <> ?1
...where x.age in ?1
NotIn
findByAgeNot In (Collection<Age> age)
...where x.age not in ?1
True
False
IgnoreCase
```

```
findByActiveTrue()
findByActiveFalse()
findByFirstnameIgnoreCase
...where x.active= true
...where x.active= false
...where UPPER (x.firstame) = UPPER(?1)
More examples
Repository Interface
public interface IMovieRepo extends JpaRepository<Movie, Integer> {
}
//select mid,mname,year,ratings from Movie where mname=? public List<Movie> findByMnameEquals(String
name); //select mid,mname,year, ratings from Movie where mname=? public List<Movie>
findByMnamels(String name); //select mid,mname, year, ratings from Movie where mname=? public
Iterable<Movie> findBymname (String name); //select mid,mname,year, ratings from Movie where mname like
'R%' public Iterable<Movie> findByMnameStartingWith(String initChars); //select mid,mname,year, ratings
from Movie where mname like '%n' public Iterable<Movie> findByMnameEndingWith(String lastChars);
//select mid,mname,year, ratings from Movie where mname like '%dhe%' public Iterable<Movie>
findByMnameContaining(String chars); //select mid,mname,year, ratings from Movie where mname like
'%dhe%' public Iterable<Movie> findByMnameEqualsIgnoreCase(String name); //select
mid,mname,year,ratings from Movie where mname like '%dhe%' public Iterable<Movie>
findByMnameContainingIgnoreCase(String chars); //select mid,mname, year, ratings from Movie where
mname like 'R%' // movies starting with R //select mid,mname, year, ratings from Movie where mname like
      '// 3 letter movies //select mid,mname, year, ratings from Movie where mname like '%R%' //Containing
letter R //select mid,mname, year, ratings from Movie where mname like '%R' //ending letter R public
Iterable<Movie> findByMnameLike(String chars); // pass wild chars while calling method
In runner class
//=====finder methods ======
repo.findByMnameEquals("Anthim").forEach(System.out::println);
System.out.println("
-");
repo.findByMnamels("Don").forEach(System.out::println);
System.out.println("
repo.findBymname("RRR").forEach(System.out::println);
```

System.out.println("

System.out.println("--

System.out.println("

repo.findByMnameStartingWith("Ra").forEach(System.out::println);

repo.findByMnameEndingWith("n").forEach(System.out::println);

-");

```
-");
repo.findByMnameContaining("m").forEach(System.out::println);
System.out.println("
-");
repo.findByMnameEqualsIgnoreCase("rrR").forEach(System.out::println);
System.out.println("
-");
repo.findByMnameContainingIgnoreCase("r").forEach(System.out::println);
System.out.println(".
-");
//repo.findByMnameLike("R%").forEach(System.out::println); //or
//repo.findByMnameLike("_____").forEach(System.out::println); //or
//repo.findByMnameLike("%R").forEach(System.out::println);//or
repo.findByMnameLike("%R%").forEach(System.out::println); //or
BootJpaProj04-JpaRepository-findByMethods [boot]
src/main/java
#com.nt
BootJpaProj04 JpaRepositoryApplication.java
com.nt.entity
> JobSeeker.java
#com.nt.repository
> JobSeekerRepository.java
com.nt.runner
> FinderMethods TestRunner.java
src/main/resources
src/test/java
//Entity class package com.nt.entity;
import jakarta.persistence.Column;
import jakarta.persistence.Entity; import jakarta.persistence.GeneratedValue; import
jakarta.persistence.GenerationType; import jakarta.persistence.ld;
import jakarta.persistence.SequenceGenerator;
import jakarta.persistence.Table;
import jakarta.persistence.Transient;
import lombok.AllArgsConstructor;
import lombok.Data;
import lombok.NoArgsConstructor; import lombok.NonNull;
import lombok.RequiredArgsConstructor;
```

```
@Entity
@Table(name="JOB_SEEKER_INFO")
@Data
@AllArgsConstructor
JRE System Library [JavaSE-17]
> Maven Dependencies
//@NoArgsConstructor
@RequiredArgsConstructor
target/generated-sources/annotations
public class JobSeeker {
target/generated-test-sources/test-annotations
@ld
> src
target
@Column(name="JS_ID")
HELP.md
mvnw
mvnw.cmd
M pom.xml
//Repository Interface
package com.nt.repository;
import java.util.List;
@SequenceGenerator(name="gen1",sequenceName = "jsld_seq",initialValue = 1000, allocationSize = 1)
@GeneratedValue(generator="gen1",strategy = GenerationType.SEQUENCE)
//@GeneratedValue(strategy = GenerationType.AUTO)
private Integer jsld;
@Column(name="JS_NAME",length =20)
@NonNull
private String jsName;
@Column(name="JS_QLFY",length =20)
@NonNull
```

```
private String qlfy;
@Column(name="JS_PERCENTAGE")
@NonNull
private Double percentage;
//@Transient
@Column(name="JS_CONTACT_INFO")
@NonNull
private Long mobileNo;
public JobSeeker() {
System.out.println("JobSeeker:: 0-param
constructor::"+this.getClass()+"....."+this.getClass().getSuperclass());
Runner class
@Component
public class Finder Methods TestRunner implements CommandLineRunner {
@Autowired
private JobSeekerRepository jsRepo;
import org.springframework.data.jpa.repository.JpaRepository;
import com.nt.entity.JobSeeker;
public interface JobSeekerRepository extends JpaRepository<JobSeeker, Integer> {
public List<JobSeeker> findByJsNameEquals(String name);
public List<JobSeeker> getByJsNamels(String name);
public List<JobSeeker> readByJsName(String name);
public List<JobSeeker> findByPercentageBetween (double start,double end); public List<JobSeeker>
findByJsNameStarting With(String nameInitialChars);
public List<JobSeeker> findByJsNameEnding WithIgnoreCase(String nameLastChars);
public List<JobSeeker> findByJsNameContainingIgnoreCase(String chars);
public List<JobSeeker> findByJsNameLikelgnoreCase(String pattern);
public List<JobSeeker> findByQlfyIn(List<String> qualifications);
public List<JobSeeker> readByMobileNolsNull();
public List<JobSeeker> findByQlfyInOrderByQlfyAsc(List<String> qualifications);
@Override
public void run(String... args) throws Exception {
/*List<JobSeeker> list=jsRepo.findByJsNameEquals("mahesh");
list.forEach(System.out::println);
```

```
/* jsRepo.readByJsName("mahesh").forEach(System.out::println);
System.out.println("----
jsRepo.getByJsNamels("mahesh").forEach(System.out::println);*/
//isRepo.findByPercentage Between (45.0, 89.0).forEach(System.out::println);
//jsRepo.findByJsNameStarting With("M").forEach(System.out::println);
//jsRepo.findByJsNameEndingWithIgnoreCase("H").forEach(System.out::println);
//jsRepo.findByJsNameContainingIgnoreCase("ah").forEach(System.out::println);
/* isRepo.findBvJsNameLikeIgnoreCase("m%").forEach(Svstem.out::println):
System.out.println(".
jsRepo.findByJsNameLikelgnoreCase("%h").forEach(System.out::println);
System.out.println("_
_"); ");
jsRepo.findByJsNameLikelgnoreCase("%sh").forEach (System.out::println);*/
// isRepo.findByQlfyIn(List.of("B.E","B.Sc")).forEach(System.out::println);
//jsRepo.readByMobile NolsNull().forEach(System.out::println);
jsRepo.findByQlfyInOrderByQlfyAsc(List.of("B.E","B.sc", "B.Tech")).forEach (System.out::println);
}
```

note:: finder methods or findBy methods are good as custom methods in JpaRepository as along with we are designing the methods using single condition on single property data otherwise its better to use @Query methods basele custom methods