Many To Many Association mapping using spring data jpa

At Db s/w level, we need 3 tables for building many to many Association

table1---> for holding parent table records

table2 ---> for holding child table records

table3 ---> for holding join table records (Actual relationship will be build here using multiple FK columns)

Many to Many Example scenarios

Faculty Doctor Project

Student (Each faculty can teach for multiple students and each student can get training multiple faculties)
Patient

programmer

Many to Many Assocation =

1 to many association from parent

+

fid(pk)

JPA_MTOM_Faculty (table1 --parent table)

fname faddrs fqlfy

1 to many association from child

JPA_MTOM_STUDENT (table2

child table)

sid(pk)

sname

saddrs

college

101 102

raja

hyd

B.E

1001

anil

rajesh

hyd

B.Tech

1002

sagar

hyd hyd

CBIET JBIET

JPA_MTOM_FACULTIES_STUDENTS

(table3 - Association table)

faculty_id(FK)

student_id(FK)

Composite PK both FK columns

101

1001

101

1002

102

1002

=> if PK constraint is applied on single column of db table then is called singular PK => if PK constraint is applied on multiple columns of db table then is called Composite PK (Here combination values of multiple cols should not be duplicated)

Required Annotations ::

note::

a) @ManyToMany ---> To build the Association

b) @JoinColumn ---->To specify the FK column c) @JoinTable

------> To specify about third table and its FK columns

(Here @JoinColumn will be used inside @JoinTable annotation)

In spring boot data jpa, we need to take two entity classes mapped with 3 tables while building many to many association mapping example apps

HAS-A

HAS-A

note:: To build this association, the Parent class should have Collection type property to hold bunch of child class objs and the child class should have collection type property to hold bunch of parent class objs note::: Many To Many Association is always Bi-Directional Association

Example app

=========

BootJpaProj13-ManyToManyAssociation [boot]

src/main/java

//Faculty.java

package com.nt.entity;

>

>

>

#com.nt

>

BootJpaProj13ManyToManyAssociationApplication.java

com.nt.entity

> Faculty.java

> Student.java

#com.nt.repository

>

IFacultyRepository.java

> IStudentRepository.java

com.nt.runners

> ManyToManyAssociation Mapping TestRunner.java

com.nt.service

> CollegeMgmtServiceImpl.java

>

ICollegeMgmtService.java

src/main/resources

application.properties

src/test/java

JRE System Library [JavaSE-17]

Maven Dependencies

target/generated-sources/annotations

target/generated-test-sources/test-annotations

>

>

src

>

target

HELP.md

mvnw

mvnw.cmd

Mpom.xml

```
import java.util.HashSet;
import java.util.Set;
import jakarta.persistence.CascadeType;
import jakarta.persistence.Column;
import jakarta.persistence.Entity;
import jakarta.persistence.FetchType; import jakarta.persistence.GeneratedValue; import
jakarta.persistence.GenerationType; import jakarta.persistence.Id;
import jakarta.persistence.JoinColumn;
```

```
import jakarta.persistence.JoinTable;
import jakarta.persistence.ManyToMany;
import jakarta.persistence.SequenceGenerator;
import jakarta.persistence.Table;
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.NonNull;
import lombok.RequiredArgsConstructor; import lombok.Setter;
@Entity
@Table(name="JPA_MTOM_FACULTY")
@Setter
@Getter
@NoArgsConstructor
@RequiredArgsConstructor
public class Faculty {
@SequenceGenerator(name="gen1",sequenceName = "FID_SEQ",initialValue = 1, allocationSize = 1)
@GeneratedValue(generator = "gen1",strategy = GenerationType.SEQUENCE)
@Id
private Integer fid;
@NonNull
@Column(length = 30)
private String fname;
@NonNull
@Column(length = 30)
private String faddrs;
@ManyToMany(targetEntity = Student.class, cascade = CascadeType.ALL, fetch = FetchType.EAGER)
@JoinTable(name = "JPA_MTOM_FACULTIES_STUDENTS", //third table
joinColumns = @JoinColumn(name="faculty_id", referencedColumnName = "fid"), // owning side FK column
inverseJoinColumns = @JoinColumn(name="student_id", referencedColumnName = "sid")) //non owning FK columr
private Set<Student> studentsInfo=new HashSet();
//Student.java
package com.nt.entity;
import java.util.HashSet;
import java.util.Set;
import jakarta.persistence.CascadeType;
import jakarta.persistence.Column;
import jakarta.persistence.Entity;
}
```

```
//toString() (alt+shift+s,s)
@Override
public String toString() {
return "Faculty [fid=" + fid + ", fname=" + fname + ", faddrs=" + faddrs + "]";
}
import jakarta.persistence.FetchType;
import jakarta.persistence.GeneratedValue; import jakarta.persistence.GenerationType; import jakarta.persistence.Id;
import jakarta.persistence.ManyToMany; import jakarta.persistence.SequenceGenerator; import
jakarta.persistence.Table;
import lombok.AllArgsConstructor;
import lombok.Getter;
import lombok.NoArgsConstructor;
import lombok.NonNull;
import lombok.RequiredArgsConstructor;
import lombok.Setter;
@Entity
@Table(name="JPA_MTOM_STUDENT")
@Setter
@Getter
@NoArgsConstructor
@RequiredArgsConstructor
public class Student {
@SequenceGenerator(name="gen1",sequenceName = "SID_SEQ", initialValue = 100,allocationSize = 1)
@GeneratedValue(generator = "gen1",strategy = GenerationType.SEQUENCE)
@Id
private Integer sid;
@Column(length = 30)
@NonNull
private String sname;
@Column(length = 30)
@NonNull
private String saddrs;
@NonNull
@Column(length = 30)
private String college;
@ManyToMany(targetEntity = Faculty.class, cascade = CascadeType.ALL, fetch = FetchType.EAGER,
)
@JoinTable(name = "JPA_MTOM_FACULTIES_STUDENTS", //third table
```

joinColumns = @JoinColumn(name=" faculty_id', referencedColumnName = "fid"), //owning side FK column inverseJoinColumns = @JoinColumn(name="student_id', referencedColumnName = "sid')) non-owning side FK column

```
private Set<Faculty> facultiesInfo=new HashSet<>();
//toString() (alt+shift+s)
return "Student [sid=" + sid + ", sname=" + sname + ", saddrs=" + saddrs + ", college=" + college + "]";
@Override
public String toString() {
}
}
```

Repositories

=========

```
package com.nt.repository;
import org.springframework.data.jpa.repository.JpaRepository;
import com.nt.entity.Faculty;
public interface IFacultyRepository extends JpaRepository<Faculty, Integer> {
}
//repository2
package com.nt.repository;
import org.springframework.data.jpa.repository.JpaRepository;
import com.nt.entity.Student;
public interface IStudentRepository extends JpaRepository<Student, Integer> {
}
```

Service Interface

```
//Service Interface
package com.nt.service;
public interface ICollegeMgmtService {
public void save Data UsingParent();
public void load Data UsingParent();
public void delete Data UsingParent();
}
```

Service Impl class

```
package com.nt.service;
import java.util.Optional;
import java.util.Set;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.nt.entity.Faculty;
```

```
import com.nt.entity.Student;
import com.nt.repository.IFacultyRepository;
import com.nt.repository.IStudentRepository;
@Service
public class CollegeMgmtServiceImpl implements ICollegeMgmtService {
@Autowired
private IFacultyRepository facultyRepo;
@Autowired
private IStudentRepository studentRepo;
@Override
public void saveDataUsingParent() {
//prepare parent objs
Faculty faculty1=new Faculty("raja", "hyd"); Faculty faculty2=new Faculty("rajesh", "delhi"); //prepare childs objs
Student stud1=new Student("anil","hyd","CBIET"); Student stud2=new Student("suresh","hyd","JBIET"); Student stud3=new Student("sagar", "hyd","NBIET"); //assign students faculties
faculty1.getStudentsInfo().add(stud1);
faculty1.getStudentsInfo().add(stud2);
faculty1.getStudentsInfo().add(stud3);
faculty2.getStudentsInfo().add(stud1);
faculty2.getStudentsInfo().add(stud2);
//assign faculties to students
stud1.getFacultiesInfo().add(faculty1);
stud1.getFacultiesInfo().add(faculty2);
stud2.getFacultiesInfo().add(faculty1);
stud2.getFacultiesInfo().add(faculty2);
stud3.getFacultiesInfo().add(faculty1);
//save data using parent
facultyRepo.save(faculty1);
facultyRepo.save(faculty2);
System.out.println("Faculties and the associated students are saved");
}//method
@Override
public void loadDataUsingParent() {
Iterable<Faculty> itFaculites=facultyRepo.findAll();
itFaculites.forEach(faculty->{
System.out.println("Parent::"+faculty);
Set<Student> childs=faculty.getStudentsInfo();
```

```
childs.forEach(stud->{
System.out.println("child::"+stud);
});
});
}
@Override
public void deleteDataUsingParent() {
//Load Parent
Optional<Faculty> opt facultyRepo.findById(3);
if(opt.isPresent()) {
Faculty factulty=opt.get();
Set<Student> childs=factulty.getStudentsInfo();
factulty.setStudentsInfo(null);
childs.forEach(ch->{
ch.setFacultiesInfo(null);
});
facultyRepo.save(factulty);
System.out.println("Faculty is removed from certain students");
}
else
{
System.out.println("Faculty not found");
}
}
}//class
```

Runner class

```
=========
package com.nt.runners;
import org.springframework.beans.factory.annotation.Autowired; import
org.springframework.boot.Command Line Runner;
import org.springframework.stereotype.Component;
import com.nt.service.ICollegeMgmtService;
@Component
public class ManyToManyAssociation MappingTestRunner implements CommandLineRunner {
@Autowired
private ICollegeMgmtService collegeService;
@Override
```

```
public void run(String... args) throws Exception {
/*try {
collegeService.saveData UsingParent();
}
catch (Exception e) {
e.printStackTrace();
}
*/
/*try {
}
collegeService.loadDataUsingParent();
catch (Exception e) { e.printStackTrace();
}*/
try {
collegeService.deleteDataUsingParent();
}
catch (Exception e) {
e.printStackTrace();
}
}//method
}//class
```

Example app on Doctor and Patient Usecase

```
===
BootDataJpaProj15-ManyToManyAssociation [boot]
src/main/java
<
#com.nt
› BootDataJpaProj15MToMAssociationApp.java
#com.nt.entity
> Doctor.java
> Patient.java
#com.nt.repository
> IDoctorRepository.java
>
IPatientRepository.java
W com.nt.runners
*
```

> ManyToManyAssociation TestRunner.java

com.nt.service

> HospitableMgmtServiceImpl.java

> HospitalMgmtService.java

src/main/resources

application.properties

> src/test/java

target/generated-sources/annotations

target/generated-test-sources/test-annotations

>

JRE System Library [JavaSE-17]

>

Maven Dependencies

>

>

>

src

>

target

HELP.md

mvnw

========

//Doctor.java (parent class)

package com.nt.entity;

import java.util.List;

import jakarta.persistence.CascadeType;

import jakarta.persistence.Column; import jakarta.persistence.Entity; import jakarta.persistence.FetchType; import jakarta.persistence.GeneratedValue; import jakarta.persistence.GenerationType; import jakarta.persistence.Id; import jakarta.persistence.JoinColumn; import jakarta.persistence.JoinTable; import jakarta.persistence.ManyToMany; import jakarta.persistence.Table;

import lombok.Getter;

import lombok.Setter;

@Table(name="JPA_MTM_DOCTOR")

@Entity @Setter

@Getter

package com.nt.entity;

import java.util.List;

import jakarta.persistence.CascadeType;

```
import jakarta.persistence.Column;
import jakarta.persistence.Entity;
import jakarta.persistence.FetchType; import jakarta.persistence.GeneratedValue; import
jakarta.persistence.GenerationType; import jakarta.persistence.Id; import jakarta.persistence.JoinColumn;
import jakarta.persistence.Join Table; import jakarta.persistence.ManyToMany;
import jakarta.persistence.SequenceGenerator;
import jakarta.persistence.Table;
import lombok.Getter;
import lombok.Setter;
@Table(name="JPA_MTM_PATIENT")
@Entity
@Setter
public class Doctor {
@Id
@GeneratedValue(strategy = GenerationType.AUTO)
private Integer did;
@Column(length = 30)
private String name;
@Column(length = 30)
private String specilaization;
@ManyToMany(targetEntity = Patient.class, cascade = CascadeType.ALL, fetch = FetchType.LAZY)
}
@JoinTable(name="JPA_MTM_PATIENTS_DOCTORS",joinColumns =
@JoinColumn(name="DOCTOR_ID",referencedColumnName = "DID"), inverseJoinColumns =
@JoinColumn(name="PATIENT_ID",referencedColumnName = "PATID"))
private List<Patient> patients;
//toString (alt+shit+s)
@Override
public String toString() {
return "Doctor [did=" + did + ", name=" + name + ", specilaization=" + specilaization + "]";
}
@Getter
public class Patient {
@SequenceGenerator(name="gen1",sequenceName = "PATID_SEQ",initialValue = 1000,allocationSize = 1)
@GeneratedValue(generator = "gen1",strategy = GenerationType.SEQUENCE)
@Id
private Integer patid;
@Column(length = 30)
```

```
private String pname;
@Column(length = 30)
private String problem;
@ManyToMany(targetEntity = Doctor.class,cascade = CascadeType.ALL,fetch = FetchType.LAZY)
@JoinTable(name="JPA_MTM_PATIENTS_DOCTORS",joinColumns =
@JoinColumn(name="PATIENT_ID",referencedColumnName = "PATID"),
inverseJoinColumns = @JoinColumn(name="DOCTOR_ID", referencedColumnName = "DID"))
private List<Doctor> doctors;
//toString() alt+shift+s,s
@Override
public String toString() {
return "Patient [patid=" + patid +
}
}
```

Service Interface

```
====
package com.nt.service;
import java.util.List;
"
pname=" + pname + ", problem=" + problem + "]";
import com.nt.entity.Doctor;
import com.nt.entity.Patient; public interface IHospital MgmtService {
public String saveDoctorsAndPatients (List<Doctor> list); //parent to child
public String save PatientsAndDoctors (List<Patient> list); // child to parent public List<Doctor>
ShowDoctorsAndThierPatients(); //parent to child public List<Patient> showPatientsAndTheir Doctors();
//child to parent public String removeCertain Patients From Doctor(int did, int pat1,int pat2);
}
```

Service Impl class

```
package com.nt.service;
import java.util.List;
import java.util.Optional;
import java.util.stream.Collectors;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;
import com.nt.entity.Doctor;
import com.nt.entity.Patient;
import com.nt.repository.IDoctorRepository;
import com.nt.repository.IPatientRepository;
```

```
@Service
public class HospitableMgmtServiceImpl implements Hospital MgmtService {
@Autowired
private IDoctorRepository docRepo;
@Autowired
private IPatientRepository patRepo;
@Override
public String save DoctorsAndPatients(List<Doctor> list) {
//save objects (parent to child)
List<Doctor> saved List=docRepo.saveAll(list);
// get saved doctor ids
List<Integer> ids=saved List.stream().map(Doctor::getDid).collect(Collectors.toList());
return ids+" Doctors and their patients are saved";
}
@Override
public String save PatientsAndDoctors(List<Patient> list) {
//save objs
List<Patient> saved List-patRepo.saveAll(list);
//get the ids of saved objs
List<Integer> ids=saved List.stream().map(Patient::getPatid).collect(Collectors.toList());
return ids+" Patients and their Doctors are saved";
}
@Override
public List<Doctor> ShowDoctorsAndThierPatients() {
return docRepo.findAll();
}
@Override
public List<Patient> showPatientsAndTheir Doctors() {
return patRepo.findAll();
}
@Override
public String removeCertain Patients From Doctor(int did, int pat1, int pat2) {
Optional<Doctor> opt=docRepo.findByld(did);
//Load the given Doctor
{
Doctor
doc=opt.get();
```

```
//get Patients of the Doctor
if(opt.isPresent())
List<Patient> list=doc.getPatients();
list.forEach(pat->{
if(pat.getPatid()==pat1 || pat.getPatid()==pat2) {
pat.setDoctors(null);
}
});
//update the doctor info
docRepo.save(doc);
return "Patients are removed from the Doctor";
}
return "Doctor not found";
}
}
```

Runner class

```
=========
package com.nt.runners;
import java.util.List;
import java.util.Set;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.Command Line Runner;
import org.springframework.stereotype.Component;
import com.nt.entity.Doctor;
import com.nt.entity.Patient;
import com.nt.service.IHospital MgmtService;
@Component
public class ManyToManyAssociation TestRunner implements CommandLineRunner {
@Autowired
private HospitalMgmtService hospitalService;
@Override
public void run(String... args) throws Exception {
/*try {
//prepare parent objs
Doctor doc1=new Doctor(); doc1.setName("raja"); doc1.setSpecilaization("Cardio");
Doctor doc2=new Doctor(); doc2.setName("karan"); doc2.setSpecilaization("Nuero");
Patient pat1=new Patient(); pat1.setPname("ramesh"); pat1.setProblem("heart"); Patient pat2=new Patient();
```

```
pat2.setPname("naresh"); pat2.setProblem("brain");
Patient pat3=new Patient(); pat3.setPname("rajesh"); pat3.setProblem("stomoch");
doc1.setPatients (List.of(pat1, pat2));
doc2.setPatients(List.of(pat1, pat2, pat3));
//invoke the method
String msg=hospitalService.saveDoctorsAndPatients(List.of(doc1, doc2));
System.out.println(msg);
}//try
catch(Exception e) {
e.printStackTrace();
}*/
/*try {
//prepare parent objs
Doctor doc1=new Doctor(); doc1.setName("mahesh");doc1.setSpecilaization("ortho");
Doctor doc2=new Doctor(); doc2.setName("nikhil"); doc2.setSpecilaization("pulamalogist");
Patient pat1=new Patient(); pat1.setPname("ramesh1"); pat1.setProblem("knee pain"); Patient pat2=new Patient();
pat2.setPname("kiran"); pat2.setProblem("brain"); Patient pat3=new Patient(); pat3.setPname("mukesh");
pat3.setProblem("throat");
pat1.setDoctors(List.of(doc1,doc2));
pat2.setDoctors(List.of(doc1));
pat3.setDoctors(List.of(doc1,doc2));
//invoke the method
String msg=hospitalService.save PatientsAndDoctors (List.of(pat1,pat2, pat3));
System.out.println(msg);
}
catch (Exception e) {
e.printStackTrace();
/*
try {
hospitalService.ShowDoctorsAndThierPatients().forEach(doc->{ System.out.println("parent::"+doc);
//get childs of each parent
List<Patient> list=doc.getPatients();
list.forEach(pat->{
System.out.println("child::"+pat);
});
});
}//try
catch(Exception e) {
```

```
e.printStackTrace();
}
/*
try {
hospitalService.showPatientsAndTheir Doctors().forEach(pat->{
System.out.println("child ::"+pat);
List<Doctor> list=pat.getDoctors();
System.out.println("parent::"+doc);
list.forEach(doc->{
});
});
}
catch(Exception e) {
e.printStackTrace();
}*/
try {
String msg=hospitalService.removeCertain Patients From Doctor (1, 1000, 1001);
System.out.println(msg);
}
catch(Exception e) {
e.printStackTrace();
}
}//main
}//class
```