**LAB-4:**

**TODO:1**

**Todo1.1:Create Instructor Table with column id as a primary key(pk),Firstname,Lastname,Email,**

**Todo1.2:Create course table with cpolumn as id(pk),Title,forgien key column named as Instrctor\_id as a reference to the Instrcutor table**

**Todo1.3:Create Course Instrcutor table with forgien key column course-id referenced to course table primarykey column id ,Instrcutor-id referenced to Instructor table primarykey column id**

**TODO:2**

**Todo2.1:Create com.hiberntae.practice,com.hibernate.practice.Entity package under src/main/java folder**

**Todo2.2:Create Instrctor.java,Course.java classes under com.hibernate.practice.Entity**

**Todo 2.3:CreateDemo.java,DeleteDemo.java classes under com.hibernate.practice**

**TODO:3**

**Todo3.1:Create com.hibernate.utils package under src/main/resource folder**

**Todo 3.2:Create hibernate.cfg.xml file under src/main/resource folder**

**Todo 3.3:Do the necessary Configuration for DB**

**Instructor.Java**

package com.hibernate.demo.entity;

import java.util.ArrayList;

import java.util.List;

import javax.persistence.CascadeType;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.FetchType;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.JoinColumn;

import javax.persistence.JoinTable;

import javax.persistence.ManyToMany;

import javax.persistence.ManyToOne;

import javax.persistence.OneToMany;

import javax.persistence.Table;

@Entity

@Table(name="course")

public class Course {

    // define our fields

    // define constructors

    // define getter setters

    // define tostring

    // annotate fields

    @Id

    @GeneratedValue(strategy=GenerationType.IDENTITY)

    @Column(name="id")

    private int id;

    @Column(name="title")

    private String title;

    @ManyToMany(fetch=FetchType.LAZY,

            cascade= {CascadeType.PERSIST, CascadeType.MERGE,

             CascadeType.DETACH, CascadeType.REFRESH})

    @JoinTable(

            name="course\_Instructor”

            joinColumns=@JoinColumn(name="course\_id"),

            inverseJoinColumns=@JoinColumn(name="Instructor\_id")

            )

    private List<Instrcutor> Instrcutors;

    public Course() {

    }

    public Course(String title) {

        this.title = title;

    }

    public int getId() {

        return id;

    }

    public void setId(int id) {

        this.id = id;

    }

    public String getTitle() {

        return title;

    }

    public void setTitle(String title) {

        this.title = title;

    }

    public Instructor getInstructor() {

        return instructor;

    }

    public void setInstructor(Instructor instructor) {

        this.instructor = instructor;

    }

    public List<Instrcutor> getInstrcutors() {

        return instrcutors;

    }

    public void setInstrcutor(List<Instructor> instructors) {

        this.instructors = instructors;

    }

    // add a convenience method

    public void addInstructor(Instructor theInstructor) {

        if (instructors == null) {

            instructors = new ArrayList<>();

        }

        instructors.add(theInstructor);

    }

    //Generate the toString method

    @Override

    public String toString() {

        return "Course [id=" + id + ", title=" + title + "]";

    }

    }

**Instructor.Java**

package com.hibernate.demo.entity;

import java.util.ArrayList;

import java.util.List;

import javax.persistence.CascadeType;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.FetchType;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.JoinColumn;

import javax.persistence.OneToMany;

import javax.persistence.OneToOne;

import javax.persistence.Table;

@Entity

@Table(name="instructor")

public class Instructor {

    // annotate the class as an entity and map to db table

    // define the fields

    // annotate the fields with db column names

    // \*\* set up mapping to InstructorDetail entity

    // create constructors

    // generate getter/setter methods

    // generate toString() method

    @Id

    @GeneratedValue(strategy=GenerationType.IDENTITY)

    @Column(name="id")

    private int id;

    @Column(name="first\_name")

    private String firstName;

    @Column(name="last\_name")

    private String lastName;

    @Column(name="email")

    private String email;

 @ManyToMany(fetch=FetchType.LAZY,

            cascade= {CascadeType.PERSIST, CascadeType.MERGE,

             CascadeType.DETACH, CascadeType.REFRESH})

    @JoinTable(

            name="course\_instrutor”

            joinColumns=@JoinColumn(name="Instructor\_id"),

            inverseJoinColumns=@JoinColumn(name="Course\_id")

    private List<Course> courses;

    public Instructor() {

    }

    public Instructor(String firstName, String lastName, String email) {

        this.firstName = firstName;

        this.lastName = lastName;

        this.email = email;

    }

    public int getId() {

        return id;

    }

    public void setId(int id) {

        this.id = id;

    }

    public String getFirstName() {

        return firstName;

    }

    public void setFirstName(String firstName) {

        this.firstName = firstName;

    }

    public String getLastName() {

        return lastName;

    }

    public void setLastName(String lastName) {

        this.lastName = lastName;

    }

    public String getEmail() {

        return email;

    }

    public void setEmail(String email) {

        this.email = email;

    }

    @Override

    public String toString() {

        return "Instructor [id=" + id + ", firstName=" + firstName + ", lastName=" + lastName + ", email=" + email"]";

    }

    public List<Course> getCourses() {

        return courses;

    }

    public void setCourses(List<Course> courses) {

        this.courses = courses;

    }

    // add convenience methods for bi-directional relationship

    public void add(Course tempCourse) {

        if (courses == null) {

            courses = new ArrayList<>();

        }

        courses.add(tempCourse);

        tempCourse.setInstructor(this);

    }

}

**CreateCourseInstructor.java**

package com.hibernate.demo;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.cfg.Configuration;

import com.hibernate.demo.entity.Course;

import com.hibernate.demo.entity.Instructor;

public class CreateCourseAndStudentsDemo {

    public static void main(String[] args) {

        // create session factory

        SessionFactory factory = new Configuration()

                                .configure("hibernate.cfg.xml")

                                .addAnnotatedClass(Instructor.class)

                                .addAnnotatedClass(Course.class)

                                .buildSessionFactory();

        // create session

        Session session = factory.getCurrentSession();

        try {

            // start a transaction

            session.beginTransaction();

            // create a course

            Course tempCourse = new Course("Java");

            // save the course

            System.out.println("\nSaving the course ...");

            session.save(tempCourse);

            System.out.println("Saved the course: " + tempCourse);

            // create the Instructor

            Instructor tempInstructor1 = new Instructor("John", "private", "john@gmail.com");

            Instructor tempInstructor2 = new Student("Mary", "Public", "mary@yahoo.com");

            // add instructor to the course

            tempCourse.addInstructor(tempInstructor1);

            tempCourse.addInstructor(tempInstructor2);

            // save the Instructor

            System.out.println("\nSaving instructors ...");

            session.save(tempInstructor1);

            session.save(tempInstructor2);

            System.out.println("Saved instructors: " + tempCourse.getInstructors());

            // commit transaction

            session.getTransaction().commit();

            System.out.println("Done!");

        }

        finally {

            // add clean up code

            session.close();

            factory.close();

        }

    }

}