**Lab 11**

**1.     CREATE HIBERNATE CRUD OPERATIONS USING entity of your choice. Get the details from respective table using SQL. Define the necessary tables/entities to represent relevant information. Perform update and delete operation.**

**Program :-**

**(Student.java)**

package FirstProject;

import javax.persistence.Entity;

import javax.persistence.Id;

// Defines the class as an entity for Hibernate

@Entity

public class student {

// Specifies the primary key field

@Id

private int id;

private String name;

private String city;

public certificate getCerti() {

return certi;

}

public void setCerti(certificate certi) {

this.certi = certi;

}

private certificate certi;

// Parameterized constructor

public student(int id, String name, String city) {

super();

this.id = id;

this.name = name;

this.city = city;

}

// Default constructor

public student() {

super();

}

// Getter for id

public int getId() {

return id;

}

// Setter for id

public void setId(int id) {

this.id = id;

}

// Getter for name

public String getName() {

return name;

}

// Setter for name

public void setName(String name) {

this.name = name;

}

// Getter for city

public String getCity() {

return city;

}

// Setter for city

public void setCity(String city) {

this.city = city;

}

// Overridden toString method for string representation

@Override

public String toString() {

return this.id + " : " + this.name + " : " + this.city;

}

}

**(Certificate.java)**

package FirstProject;

import javax.persistence.Embeddable;

@Embeddable

public class certificate {

private String course;

private String duration;

public certificate(String course, String duration) {

super();

this.course = course;

this.duration = duration;

}

public certificate() {

super();

}

public String getCourse() {

return course;

}

public void setCourse(String course) {

this.course = course;

}

public String getDuration() {

return duration;

}

public void setDuration(String duration) {

this.duration = duration;

}

}

**(EmDemo.java)**

package FirstProject;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.Transaction;

import org.hibernate.cfg.Configuration;

public class EmDemo {

public static void main(String[] args) {

// Configure Hibernate and build the session factory

Configuration cfg = new Configuration();

cfg.configure();

SessionFactory factory = cfg.buildSessionFactory();

// Create first student object

student std1 = new student();

std1.setId(100);

std1.setName("Mithila");

std1.setCity("Dombivali");

// Create certificate for first student

certificate certi1 = new certificate();

certi1.setCourse("Java");

certi1.setDuration("2 Months");

std1.setCerti(certi1);

// Create second student object

student std2 = new student();

std2.setId(101);

std2.setName("Pranay");

std2.setCity("Vileparle");

// Create certificate for second student

certificate certi2 = new certificate();

certi2.setCourse("Java Fullstack");

certi2.setDuration("4 Months");

std2.setCerti(certi2);

// Open a new session and begin a transaction

Session s = factory.openSession();

Transaction tx = s.beginTransaction();

// Save both student objects to the database

s.save(std1);

s.save(std2);

// Retrieve and print the first student

student stdMithila = (student) s.get(student.class, 100);

System.**out**.println(stdMithila);

// Retrieve and print the second student

student stdPranay = (student) s.get(student.class, 101);

System.**out**.println(stdPranay);

// Update the details of the first student

student stdToUpdate = (student) s.get(student.class, 100);

if (stdToUpdate != null) {

stdToUpdate.setName("Mithila Updated");

stdToUpdate.setCity("Dombivali Updated");

s.update(stdToUpdate);

System.**out**.println("Updated Student: " + stdToUpdate);

} else {

System.**out**.println("Student with ID 100 not found.");

}

// Delete the second student from the database

student stdToDelete = (student) s.get(student.class, 101);

if (stdToDelete != null) {

s.delete(stdToDelete);

System.**out**.println("Deleted Student: " + stdToDelete);

} else {

System.**out**.println("Student with ID 101 not found.");

}

// Commit the transaction to save changes

tx.commit();

// Close the session and factory to release resources

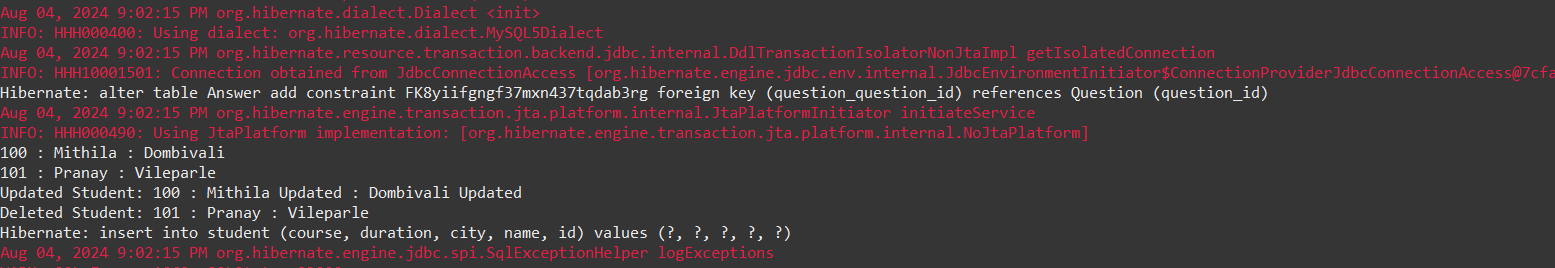
s.close();

factory.close();

}

}

**Output :-**



**2.     You are working on a Java application to manage information about students and their respective addresses. Implement a one-to-one association between the Student and Address entities using Hibernate.**

**Program :-**

**(Question.java)**

package map;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Id;

import javax.persistence.OneToOne;

import javax.persistence.JoinColumn;

@Entity

public class Question {

@Id

@Column(name="question\_id")

private int questionId;

private String question;

@OneToOne

@JoinColumn(name = "answer\_id") // Foreign key in the Question table

private Answer answer;

public Question() {

super();

}

public Question(int questionId, String question, Answer answer) {

super();

this.questionId = questionId;

this.question = question;

this.answer = answer;

}

public int getQuestionId() {

return questionId;

}

public void setQuestionId(int questionId) {

this.questionId = questionId;

}

public String getQuestion() {

return question;

}

public void setQuestion(String question) {

this.question = question;

}

public Answer getAnswer() {

return answer;

}

public void setAnswer(Answer answer) {

this.answer = answer;

}

}

**(Answer.java)**

package map;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Id;

import javax.persistence.OneToOne;

@Entity

public class Answer {

@Id

@Column(name="answer\_id")

private int answerId;

private String answer;

@OneToOne(mappedBy = "answer") // This indicates the inverse side of the relationship

private Question question;

public Answer() {

super();

}

public Answer(int answerId, String answer) {

super();

this.answerId = answerId;

this.answer = answer;

}

public int getAnswerId() {

return answerId;

}

public void setAnswerId(int answerId) {

this.answerId = answerId;

}

public String getAnswer() {

return answer;

}

public void setAnswer(String answer) {

this.answer = answer;

}

public Question getQuestion() {

return question;

}

public void setQuestion(Question question) {

this.question = question;

}

}

**(MapDemo.java)**

package map;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.Transaction;

import org.hibernate.cfg.Configuration;

public class MapDemo {

public static void main(String[] args) {

Configuration cfg = new Configuration();

cfg.configure("hibernate.cfg.xml");

SessionFactory factory = cfg.buildSessionFactory();

// Creating question1 object

Question q1 = new Question();

q1.setQuestionId(1212);

q1.setQuestion("What is java ?");

// Creating answer1 object

Answer answer1 = new Answer();

answer1.setAnswerId(343);

answer1.setAnswer("Java is a programming language");

q1.setAnswer(answer1);

answer1.setQuestion(q1); // Set the reference back to Question

// Creating question2 object

Question q2 = new Question();

q2.setQuestionId(242);

q2.setQuestion("What is collection framework ?");

// Creating answer2 object

Answer answer2 = new Answer();

answer2.setAnswerId(344);

answer2.setAnswer("API to work with objects in java");

q2.setAnswer(answer2);

answer2.setQuestion(q2); // Set the reference back to Question

Session session = factory.openSession();

Transaction tx = session.beginTransaction();

session.save(answer1); // Save Answer first to maintain referential integrity

session.save(q1);

session.save(answer2); // Save Answer first to maintain referential integrity

session.save(q2);

tx.commit();

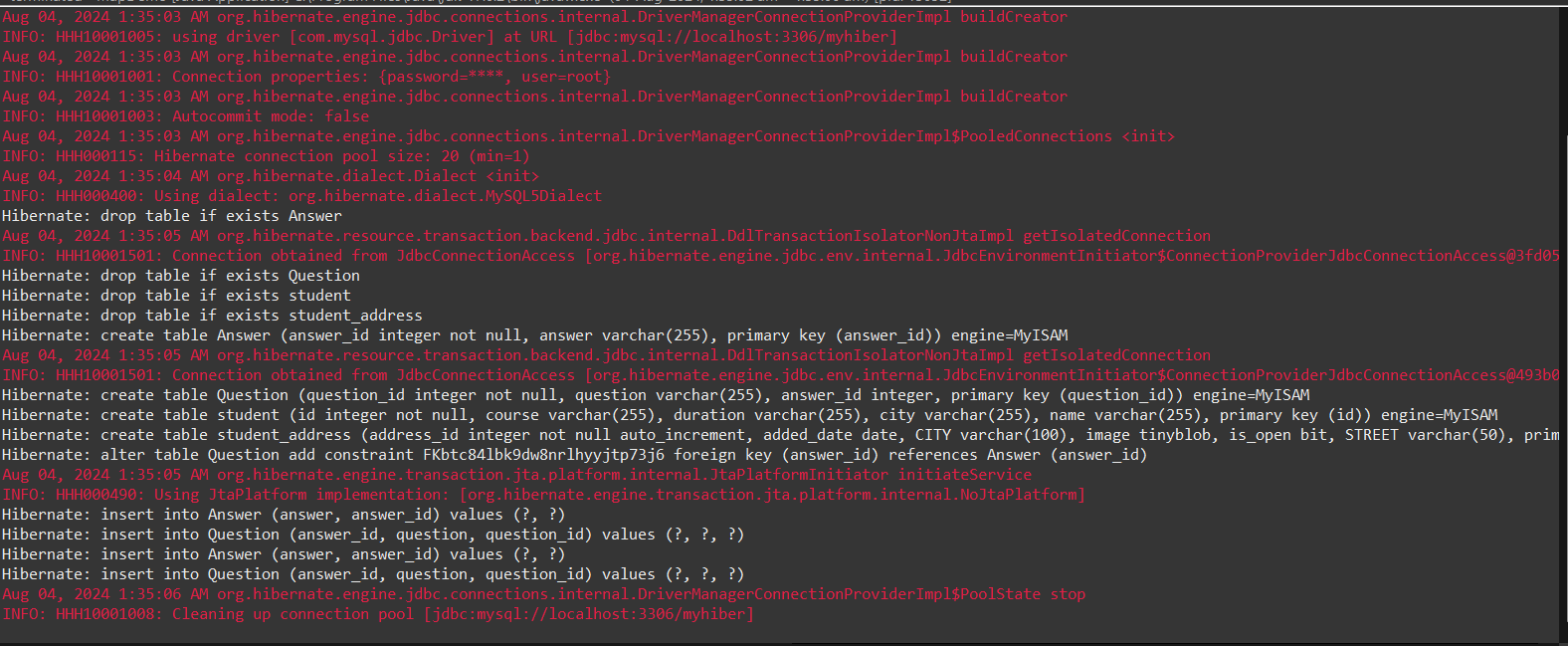
session.clear();

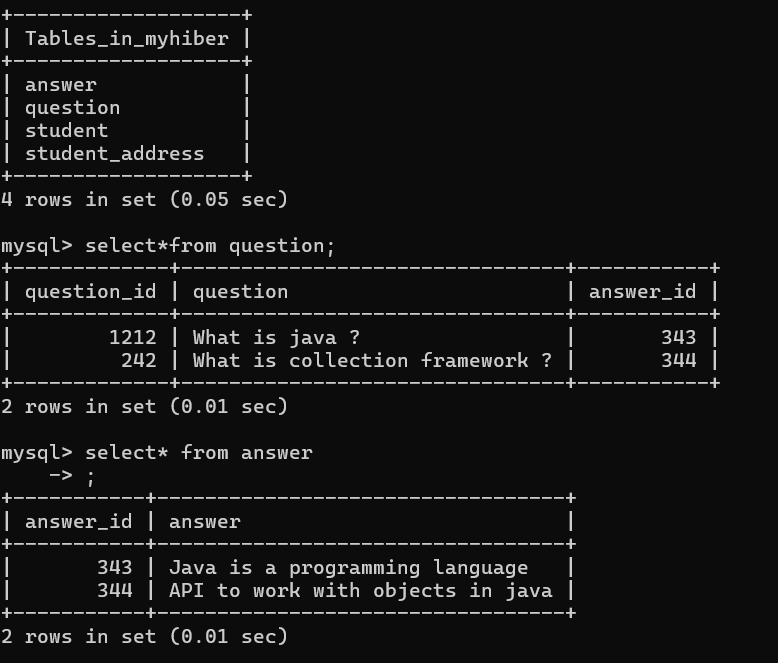
factory.close();

}

}

Output :-



****

**3.     You are working on a Java application to manage information about employees and their respective departments. Implement a one-to-many association between the Employee and Department entities using Hibernate.**

**Program :-**

**(Question.java)**

package OneToOneMap;

import java.util.List;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Id;

import javax.persistence.OneToOne;

import javax.persistence.JoinColumn;

import javax.persistence.OneToMany;

@Entity

public class Question {

@Id

@Column(name="question\_id")

private int questionId;

private String question;

@OneToMany(mappedBy="question")

private List <Answer> answer;

public Question() {

super();

}

public int getQuestionId() {

return questionId;

}

public void setQuestionId(int questionId) {

this.questionId = questionId;

}

public String getQuestion() {

return question;

}

public void setQuestion(String question) {

this.question = question;

}

public Question(int questionId, String question, List<Answer> answer) {

super();

this.questionId = questionId;

this.question = question;

this.answer = answer;

}

public List<Answer> getAnswer() {

return answer;

}

public void setAnswer(List<Answer> answer) {

this.answer = answer;

}

}

**(Answer.java)**

package OneToOneMap;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.Id;

import javax.persistence.ManyToOne;

import javax.persistence.OneToOne;

@Entity

public class Answer {

@Id

@Column(name="answer\_id")

private int answerId;

private String answer;

@ManyToOne

private Question question;

public Answer() {

super();

}

public Answer(int answerId, String answer) {

super();

this.answerId = answerId;

this.answer = answer;

}

public int getAnswerId() {

return answerId;

}

public void setAnswerId(int answerId) {

this.answerId = answerId;

}

public String getAnswer() {

return answer;

}

public void setAnswer(String answer) {

this.answer = answer;

}

public Question getQuestion() {

return question;

}

public void setQuestion(Question question) {

this.question = question;

}

}

**(MapDemo.java)**

package OneToOneMap;

import java.util.ArrayList;

import java.util.List;

import org.hibernate.Session;

import org.hibernate.SessionFactory;

import org.hibernate.Transaction;

import org.hibernate.cfg.Configuration;

public class MapDemo {

public static void main(String[] args) {

Configuration cfg = new Configuration();

cfg.configure("hibernate.cfg.xml");

SessionFactory factory = cfg.buildSessionFactory();

// Creating question1 object

Question q1 = new Question();

q1.setQuestionId(1212);

q1.setQuestion("What is java ?");

// Creating answer1 object

Answer answer1 = new Answer();

answer1.setAnswerId(343);

answer1.setAnswer("Java is a programming language");

answer1.setQuestion(q1); // Set the reference back to Question

Answer answer2 = new Answer();

answer2.setAnswerId(33);

answer2.setAnswer("Java is fast");

answer2.setQuestion(q1); // Set the reference back to Question

Answer answer3 = new Answer();

answer3.setAnswerId(363);

answer3.setAnswer("Java helps to build software");

answer3.setQuestion(q1); // Set the reference back to Question

List<Answer> list = new ArrayList<Answer>();

list.add(answer1);

list.add(answer2);

list.add(answer3);

q1.setAnswer(list);

Session session = factory.openSession();

Transaction tx = session.beginTransaction();

session.save(q1);

session.save(answer1);

session.save(answer2);

session.save(answer3);

// Fetching..

Question que = (Question)session.get(Question.class, 1212);

System.out.println(que.getQuestion());

for(Answer a:que.getAnswer()) {

System.out.println(a.getAnswer());

}

// Question newQ=(Question)session.get(Question.class, 1212);

// System.out.println(newQ.getQuestion());

// System.out.println(newQ.getAnswer().getAnswer());

tx.commit();

session.clear();

factory.close();

}

}

**Output :-**

