**Lab 10**

**1.     Develop a basic Create, Read operation using Hibernate for a simple entity, such as Student**

**1. Set Up Hibernate Configuration**

<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate Configuration DTD 3.0//EN" "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

<hibernate-configuration>

<session-factory>

<!-- Database connection settings -->

<property name="hibernate.connection.driver\_class">org.h2.Driver</property>

<property name="hibernate.connection.url">jdbc:h2:mem:testdb</property>

<property name="hibernate.connection.username">root</property>

<property name="hibernate.connection.password">Sankey$8990</property>

<!-- Specify dialect -->

<property name="hibernate.dialect">org.hibernate.dialect.H2Dialect</property>

<!-- Echo all executed SQL to stdout -->

<property name="hibernate.show\_sql">true</property>

<!-- Drop and re-create the database schema on startup -->

<property name="hibernate.hbm2ddl.auto">update</property>

<!-- Mention annotated class -->

<mapping class="com.example.Student"/>

</session-factory>

</hibernate-configuration>

**2. Create the Student Entity Class**

package lab10;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

@Entity

public class Student {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private Long id;

private String name;

private int age;

// Constructors

public Student() {}

public Student(String name, int age) {

this.name = name;

this.age = age;

}

// Getters and Setters

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public int getAge() {

return age;

}

public void setAge(int age) {

this.age = age;

}

@Override

public String toString() {

return "Student [id=" + id + ", name=" + name + ", age=" + age + "]";

}

}

**3. Create a Hibernate Utility Class**

package lab10;

import org.hibernate.SessionFactory;

import org.hibernate.cfg.Configuration;

public class HibernateUtil {

private static final SessionFactory sessionFactory = buildSessionFactory();

private static SessionFactory buildSessionFactory() {

try {

// Create the SessionFactory from hibernate.cfg.xml

return new Configuration().configure().buildSessionFactory();

} catch (Throwable ex) {

// Make sure you log the exception, as it might be swallowed

System.err.println("Initial SessionFactory creation failed." + ex);

throw new ExceptionInInitializerError(ex);

}

}

public static SessionFactory getSessionFactory() {

return sessionFactory;

}

public static void shutdown() {

// Close caches and connection pools

getSessionFactory().close();

}

}

**4. Perform Create and Read Operations**

package lab10;

import org.hibernate.Session;

import org.hibernate.Transaction;

public class Main {

public static void main(String[] args) {

// Create a new student

Student student = new Student("John Doe", 20);

// Save the student

saveStudent(student);

// Read the student

readStudent(student.getId());

}

public static void saveStudent(Student student) {

// Obtain a session

Session session = HibernateUtil.getSessionFactory().openSession();

Transaction transaction = null;

try {

// Start a transaction

transaction = session.beginTransaction();

// Save the student object

session.save(student);

// Commit the transaction

transaction.commit();

System.out.println("Student saved successfully!");

} catch (Exception e) {

if (transaction != null) {

transaction.rollback();

}

e.printStackTrace();

} finally {

session.close();

}

}

public static void readStudent(Long studentId) {

// Obtain a session

Session session = HibernateUtil.getSessionFactory().openSession();

try {

// Fetch the student object

Student student = session.get(Student.class, studentId);

if (student != null) {

System.out.println("Student found: " + student);

} else {

System.out.println("Student not found with id: " + studentId);

}

} finally {

session.close();

}

}

}

**2.     Use get() method to fetch a student object with an ID that doesn't exist in the database. What will be the result, and how would you handle it?**

public static void readStudent(Long studentId) {

// Obtain a session

Session session = HibernateUtil.getSessionFactory().openSession();

try {

// Fetch the student object

Student student = session.get(Student.class, studentId);

if (student != null) {

// Student found

System.out.println("Student found: " + student);

} else {

// Student not found

System.out.println("No student found with ID: " + studentId);

}

} finally {

// Close the session

session.close();

}

}

**3.     Also demonstrate use of load() method.**

public static void loadStudent(Long studentId) {

// Obtain a session

Session session = HibernateUtil.getSessionFactory().openSession();

try {

// Load the student object (may throw ObjectNotFoundException if entity is not found)

Student student = null;

try {

student = session.load(Student.class, studentId);

// Accessing a property of the proxy will trigger loading

System.out.println("Student found: " + student.getName());

} catch (org.hibernate.ObjectNotFoundException e) {

// Handle the case where the entity is not found

System.out.println("No student found with ID: " + studentId);

}

} finally {

// Close the session

session.close();

}

}