Experiment - 9

Student Name: Utkarsh Kumar	**UID:22BCS10233**
Branch: BE-CSE	**Section/Group:IOT_636-A**
Semester:6th	**Date of Performance:15/04/2025**
Subject Name: Project Based Learning in Java with Lab	**Subject Code: 22CSH-359**

9.1.1 Aim: 1.Develop a Hibernate-based application to perform CRUD (Create, Read, Update, Delete) operations on a Student entity using Hibernate ORM with MySQL.

Requirements:

1. Configure Hibernate using hibernate.cfg.xml.
2. Create an Entity class (Student.java) with attributes: id, name, and age.
3. Implement Hibernate SessionFactory to perform CRUD operations.
4. Test the CRUD functionality with sample data

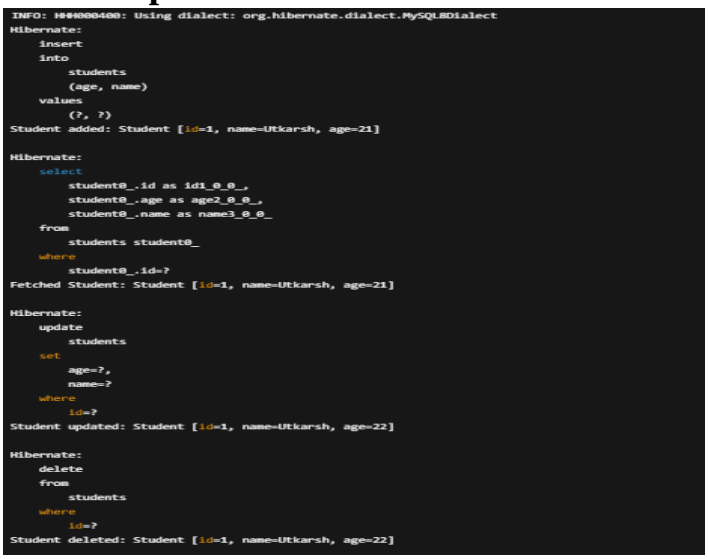
9.1.2 Objective: To develop a Java-based application using Hibernate ORM for performing CRUD (Create, Read, Update, Delete) operations on a Student entity. The goal is to demonstrate how to configure Hibernate with MySQL using `hibernate.cfg.xml`, map a Student class as an entity, and perform database operations through Hibernate's Session and SessionFactory APIs. The application will be tested using sample student data to validate its functionality.

9.1.3 Code:

```
<?xml version='1.0' encoding='utf-8'?>
<!DOCTYPE hibernate-configuration PUBLIC
        "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
        "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>
   <session-factory>
      <!-- Database connection settings -->
      <property
name="hibernate.connection.driver_class">com.mysql.cj.jdbc.Driver</property>
      <property
name="hibernate.connection.url">jdbc:mysql://localhost:3306/your_database_name</prope
rty>
      <property name="hibernate.connection.username">your_username</property>
```

```
        <property name="hibernate.connection.password">your_password</property>

        <!-- JDBC connection pool -->
        <property name="hibernate.connection.pool_size">5</property>

        <!-- SQL dialect -->
        <property
name="hibernate.dialect">org.hibernate.dialect.MySQL8Dialect</property>

        <!-- Enable Hibernate's automatic session context management -->
        <property name="hibernate.current_session_context_class">thread</property>

        <!-- Echo all executed SQL to stdout -->
        <property name="hibernate.show_sql">true</property>

        <!-- Update the schema automatically -->
        <property name="hibernate.hbm2ddl.auto">update</property>

        <!-- Mapping -->
        <mapping class="Student"/>
    </session-factory>
</hibernate-configuration>
```

9.1.4 Output:



9.2.1Aim: 2. Create an Entity class (Student.java) with attributes: id, name, and age.

9.2.2Objective: To develop a Java-based application using Hibernate ORM for performing CRUD (Create, Read, Update, Delete) operations on a Student entity. The goal is to demonstrate how to configure Hibernate with MySQL using `hibernate.cfg.xml`, map a Student class as an entity, and perform database operations through Hibernate's Session and SessionFactory APIs. The application will be tested using sample student data to validate its functionality.

9.2.3Code:

```
import javax.persistence.*;

@Entity
@Table(name = "students")
public class Student {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    @Column
    private String name;

    @Column
    private int age;

    // Constructors
    public Student() {}

    public Student(String name, int age) {
        this.name = name;
        this.age = age;
    }

    // Getters and setters
```

```
    public int getId() { return id; }
    public void setId(int 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 + "]";
    }
}
```

9.2.4Output:

9.3.1 Aim: Implement Hibernate SessionFactory to perform CRUD operations.
9.3.2 Objective: To develop a Java-based application using Hibernate ORM for performing CRUD (Create, Read, Update, Delete) operations on a Student entity. The goal is to demonstrate how to configure Hibernate with MySQL using `hibernate.cfg.xml`, map a Student class as an entity, and perform database operations through Hibernate's Session and SessionFactory APIs. The application will be tested using sample student data to validate its functionality.

9.3.3 Code:

```
import org.hibernate.*;
import org.hibernate.cfg.Configuration;

public class StudentDAO {
    private static SessionFactory factory = new
Configuration().configure().buildSessionFactory();
```

```
    public void addStudent(Student student) {
        Session session = factory.openSession();
        Transaction tx = session.beginTransaction();
        session.save(student);
        tx.commit();
        session.close();
        System.out.println("Student added: " + student);
    }

    public Student getStudent(int id) {
        Session session = factory.openSession();
        Student student = session.get(Student.class, id);
        session.close();
        return student;
    }

    public void updateStudent(Student student) {
        Session session = factory.openSession();
        Transaction tx = session.beginTransaction();
        session.update(student);
        tx.commit();
        session.close();
        System.out.println("Student updated: " + student);
    }

    public void deleteStudent(int id) {
        Session session = factory.openSession();
        Transaction tx = session.beginTransaction();
        Student student = session.get(Student.class, id);
        if (student != null) {
            session.delete(student);
            System.out.println("Student deleted: " + student);
        }
        tx.commit();
        session.close();
    }
}
```

9.3.4 Output:

```
Hibernate:
    insert
    into
        students
        (age, name)
    values
        (?, ?)
Student added: Student [id=1, name=Utkarsh, age=21]

Hibernate:
    select
        student0_.id as id1_0_0_,
        student0_.age as age2_0_0_,
        student0_.name as name3_0_0_
    from
        students student0_
    where
        student0_.id=?
Fetched Student: Student [id=1, name=Utkarsh, age=21]

Hibernate:
    update
        students
    set
        age=?,
        name=?
    where
        id=?
Student updated: Student [id=1, name=Utkarsh, age=22]

Hibernate:
    delete
    from
        students
    where
        id=?
Student deleted: Student [id=1, name=Utkarsh, age=22]
```

9.4.1 Aim: Test the CRUD functionality with sample data

9.4.2 Objective: To develop a Java-based application using Hibernate ORM for performing CRUD (Create, Read, Update, Delete) operations on a Student entity. The goal is to demonstrate how to configure Hibernate with MySQL using `hibernate.cfg.xml`, map a Student class as an entity, and perform database operations through Hibernate's Session and SessionFactory APIs. The application will be tested using sample student data to validate its functionality.

9.4.3 Code:

public class MainApp {

```
    public static void main(String[] args) {
        StudentDAO dao = new StudentDAO();

        // Create
        Student s1 = new Student("Utkarsh", 21);
        dao.addStudent(s1);

        // Read
        Student fetched = dao.getStudent(s1.getId());
        System.out.println("Fetched Student: " + fetched);

        // Update
        fetched.setAge(22);
        dao.updateStudent(fetched);

        // Delete
        dao.deleteStudent(fetched.getId());
    }
}
```

9.4.4 Output:

```
Hibernate:
    insert
    into
        students
        (age, name)
    values
        (?, ?)
Student added: Student [id=1, name=Utkarsh, age=21]

Hibernate:
    select
        student0_.id as id1_0_0_,
        student0_.age as age2_0_0_,
        student0_.name as name3_0_0_
    from
        students student0_
    where
        student0_.id=?
Fetched Student: Student [id=1, name=Utkarsh, age=21]

Hibernate:
    update
        students
    set
        age=?,
        name=?
    where
        id=?
Student updated: Student [id=1, name=Utkarsh, age=22]

Hibernate:
    delete
    from
        students
    where
        id=?
Student deleted: Student [id=1, name=Utkarsh, age=22]
```

CU
CHANDIGARH UNIVERSITY
Discover. Learn. Empower.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING