LAB 10

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

# Program files:

## pom.xml

**Program:**

<project xmlns="<http://maven.apache.org/POM/4.0.0>" xmlns:xsi="<http://www.w3.org/2001/XMLSchema-instance>"

xsi:schemaLocation="<http://maven.apache.org/POM/4.0.0> <http://maven.apache.org/xsd/maven-4.0.0.xsd>">

<modelVersion>4.0.0</modelVersion>

<groupId>com.demo</groupId>

<artifactId>college\_management\_system</artifactId>

<version>0.0.1-SNAPSHOT</version>

<packaging>jar</packaging>

<name>college\_management\_system</name>

<url>[http://maven.apache.org](http://maven.apache.org/)</url>

<properties>

<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>

</properties>

<dependencies>

<!-- https://mvnrepository.com/artifact/org.hibernate/hibernate-core -->

<dependency>

<groupId>org.hibernate</groupId>

<artifactId>hibernate-core</artifactId>

<version>5.6.15.Final</version>

</dependency>

<!-- https://mvnrepository.com/artifact/mysql/mysql-connector-java -->

<dependency>

<groupId>mysql</groupId>

<artifactId>mysql-connector-java</artifactId>

<version>8.0.30</version>

</dependency>

<dependency>

<groupId>junit</groupId>

<artifactId>junit</artifactId>

<version>3.8.1</version>

<scope>test</scope>

</dependency>

</dependencies>

</project>

## hibernateUtil.java

**program:**

package com.demo.college\_management\_system; import org.hibernate.SessionFactory; import org.hibernate.cfg.Configuration; public class HibernateUtil {

private static final SessionFactory = *buildSessionFactory*(); private static SessionFactory buildSessionFactory() {

try { return new

Configuration().configure("hibernate.cfg.xml").addAnnotatedClass(Student.class)

.buildSessionFactory();

} catch (Throwable ex) {

throw new ExceptionInInitializerError(ex);

}

}

public static SessionFactory getSessionFactory() { return ***sessionFactory***;

}

}

## hibernate.cfg.xml

**program:**

<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE hibernate-configuration PUBLIC

"-//Hibernate/Hibernate Configuration DTD 3.0//EN" ["http://www.hibernate.org/dtd/hibernate](http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd)-[configuration-3.0.dtd](http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd)">

<hibernate-configuration>

<session-factory>

<!-- Database connection settings -->

<property name="connection.driver\_class">com.mysql.cj.jdbc.Driver</property>

<property name="connection.url">jdbc:mysql://localhost:3306/college\_management\_system</prop erty>

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

<property name="connection.password">Abbu@05</property>

<!-- Automatically create or update database schema -->

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

<!-- Specify the dialect for your database -->

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

<!-- Enable or disable showing SQL statements in the console -->

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

<property name="format\_sql">true</property>

<!-- Specify the package(s) where your entity classes are located →

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

<! Additional mappings can be included here -->

</session-factory>

</hibernate-configuration>

## Student.java

**Program:**

package com.demo.college\_management\_system; import java.time.LocalDate;

import java.time.LocalDate;

import javax.persistence.Column; import javax.persistence.Entity; import javax.persistence.Id; import javax.persistence.Temporal;

import javax.persistence.TemporalType; @Entity

public class Student { @Id

@Column(name = "StudentId", length = 10) private String studentId;

@Column(name = "FirstName", length = 50) private String firstName;

@Column(name = "LastName", length = 25) private String lastName;

@Column(name = "DateOfBirth") private LocalDate dateOfBirth;

@Column(name = "Gender", length = 25) private String gender;

@Column(name = "Email", length = 30) private String email;

@Column(name = "Phone", length = 25) private String phone;

//Setter and Getter

public String getStudentId() { return studentId;

}

public void setStudentId(String studentId) { this.studentId = studentId;

}

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 LocalDate getDateOfBirth() { return dateOfBirth;

}

public void setDateOfBirth(LocalDate dateOfBirth) { this.dateOfBirth = dateOfBirth;

}

public String getGender() { return gender;

}

public void setGender(String gender) { this.gender = gender;

}

public String getEmail() { return email;

}

public void setEmail(String email) { this.email = email;

}

public String getPhone() { return phone;

}

public void setPhone(String phone) { this.phone = phone;

}

//All argument Constructor

public Student(String studentId, String firstName, String lastName, LocalDate dateOfBirth, String gender,String email, String phone) {

super();

this.studentId = studentId; this.firstName = firstName; this.lastName = lastName; this.dateOfBirth = dateOfBirth; this.gender = gender; this.email = email;

this.phone = phone;

}

//Default Constructor public Student() { super();

}

//ToString method @Override

public String toString() {

return "Student [studentId=" + studentId + ", firstName=" + firstName + ",lastName=" + lastName+ ", dateOfBirth=" + dateOfBirth + ", gender=" + gender + ",email=" + email + ", phone=" + phone + "]";

}

}

## App.java

**Program:**

package com.demo.college\_management\_system; import java.time.LocalDate;

import org.hibernate.Session;

import org.hibernate.SessionFactory; import org.hibernate.Transaction; public class App {

public static void main(String[] args) {

// Obtain a Hibernate SessionFactory

SessionFactory factory = HibernateUtil.getSessionFactory();

// Create a new Student

LocalDate date1 = LocalDate.of(1988, 1, 13);

Student Student1 = new Student("S111", "Oliver", "Henry", date1, "M","[oliver@gmail.com](mailto:oliver@gmail.com)", "6742906745");

// Open a new session

Session session = factory.openSession();

// Begin a transaction

Transaction transaction = session.beginTransaction();

// Save the student to the database session.save(Student1);

// Commit the transaction transaction.commit();

// Close the Session session.close();

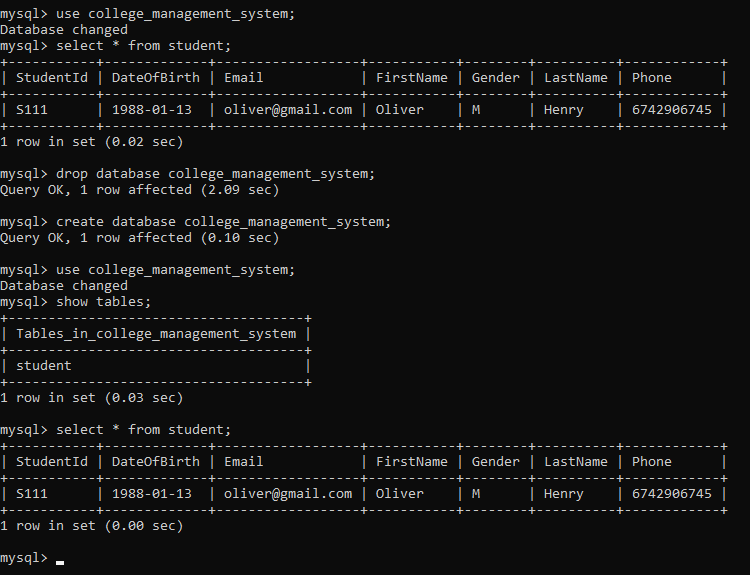
// Close the Session Factory factory.close();

}

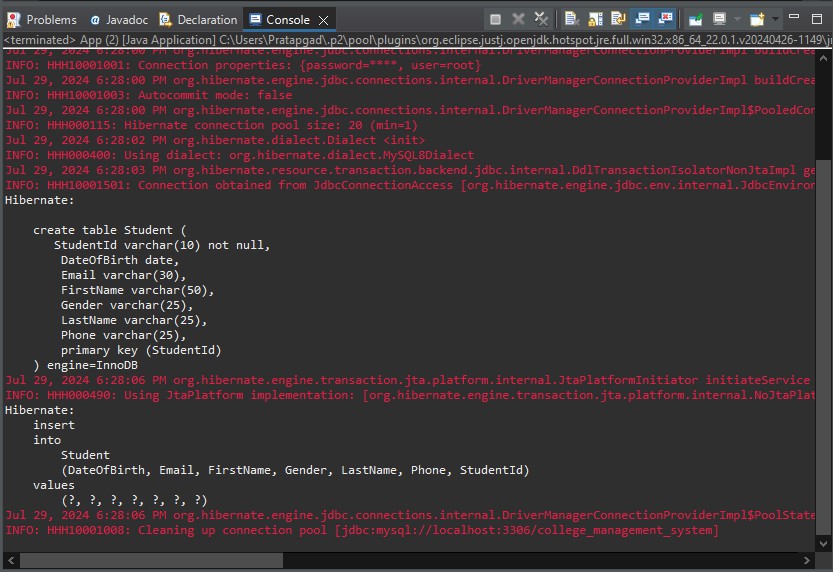
}

## Output Screen:

**Database Screen:**



**Eclipse Screen:**



* 1. 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?

## Program Files:

**RetrieveRecord.java**

import org.hibernate.Session; import org.hibernate.SessionFactory; import org.hibernate.Transaction;

import com.demo.college\_management\_system.HibernateUtil; import com.demo.college\_management\_system.Student; public class RetrievingRecord {

public static void main(String[] args) {

// Obtain a Hibernate SessionFactory

SessionFactory factory = HibernateUtil.*getSessionFactory*();

// Open a new session

Session session = factory.openSession();

// Begin a transaction

Transaction transaction = session.beginTransaction();

// Retrieve the object using the primary key Student student=session.get(Student.class, "S111");

//display data using toString() method System.***out***.println(student);

// Commit the transaction transaction.commit();

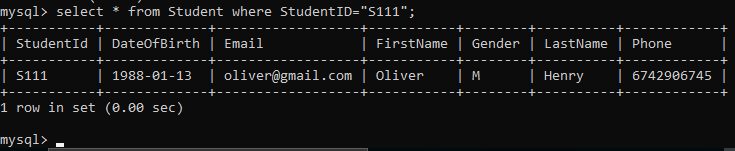
// Close the session session.close();

}

}

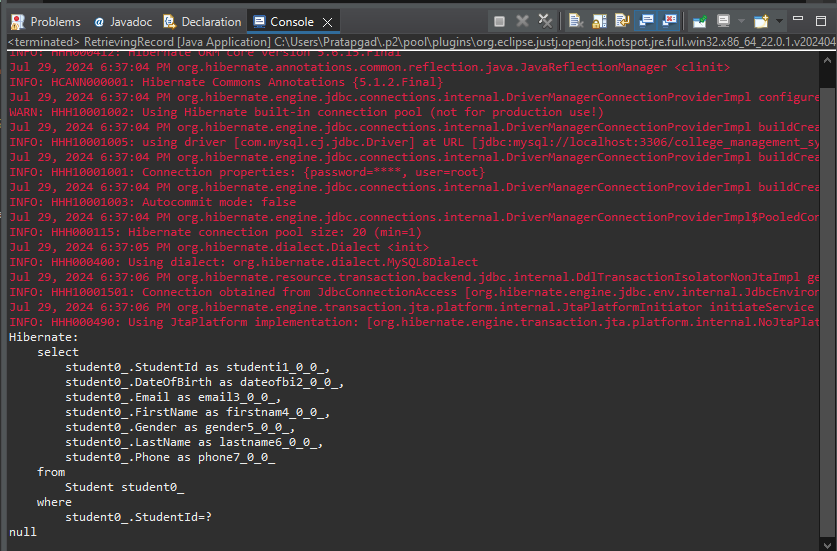
# Output Screen:

**Database:**



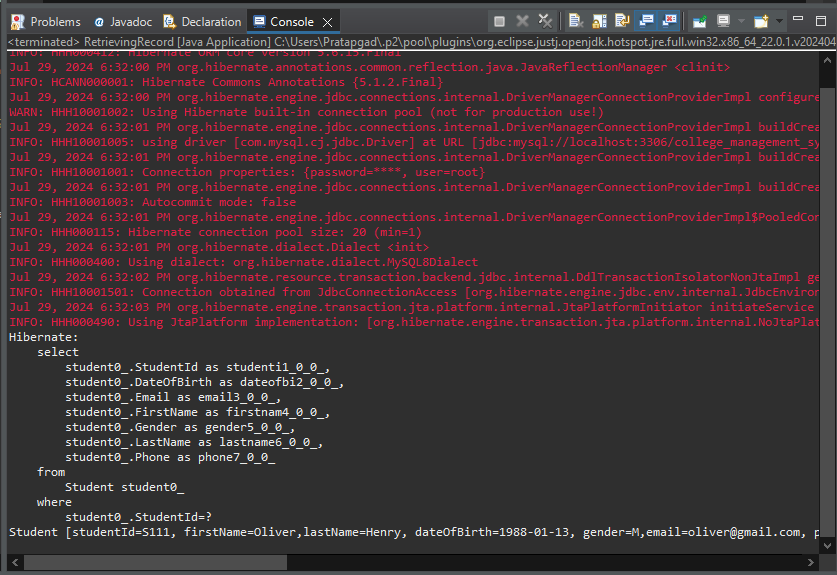
**Eclipse:** If the parameter **(StudentID=”S112”)** of Object is not match with

database’s record then eclipse shows null:



**Eclipse:** If the parameter **(StudentID=”S111”)** of Object is match with database’s

record then eclipse ran the program successfully and shows the record:



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

# Program Files:

**RetrieveRecord.java**

import org.hibernate.Session; import org.hibernate.SessionFactory; import org.hibernate.Transaction;

import com.demo.college\_management\_system.HibernateUtil; import com.demo.college\_management\_system.Student; public class RetrievingRecord {

public static void main(String[] args) {

// Obtain a Hibernate SessionFactory

SessionFactory factory = HibernateUtil.*getSessionFactory*();

// Open a new session

Session session = factory.openSession();

// Begin a transaction

Transaction transaction = session.beginTransaction();

// Retrieve the object using the primary key Student student=session.load(Student.class, "S111");

//display data using toString() method System.***out***.println(student);

// Commit the transaction transaction.commit();

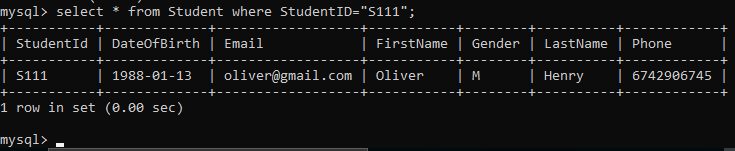
// Close the session session.close();

}

}

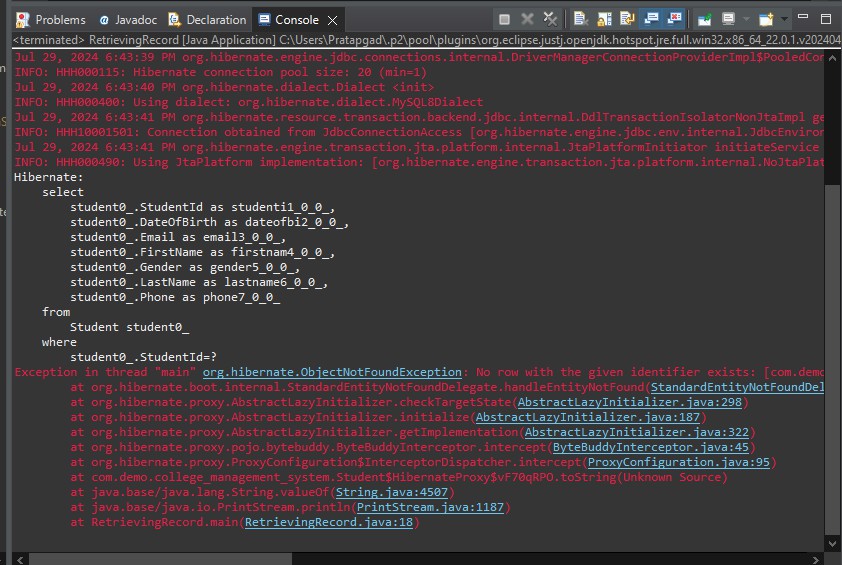
# Output Screen:

**Database:**



**Eclipse:** If the parameter **(StudentID=”S112”)** of Object is not match with

database’s record then eclipse shows an exception:



**Eclipse:** If the parameter **(StudentID=”S111”)** of Object is match with database’s

record then eclipse ran the program successfully and shows the record:

