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

Code:-

This is Main class (App.class).

**package** com.hms;

**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*();

// Open a new session

Session session = factory.openSession();

// Begin a transaction

Transaction transaction = session.beginTransaction();

// create Address object

Room r1 = **new** Room("106", "Occupied", "Dirty", "2500", "Double Bed");

// create Employee Object

session.save(r1);

// Commit the transaction

transaction.commit();

// retrieve the data

Room rooms = session.get(Room.**class**, "106");

// display the data

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

// Close the Session

session.close();

// Close the Session Factory

factory.close();

}

}

Room.class

Code:-

**package** com.hms;

**import** java.time.LocalDate;

**import** javax.persistence.Column;

**import** javax.persistence.Entity;

**import** javax.persistence.Id;

@Entity

**public** **class** Room {

@Id

@Column(name = "RoomNumber", length = 10)

**private** String roomnumber;

@Column(name = "Availablity", length = 50)

**private** String availability;

@Column(name = "Clean\_Status", length = 25)

**private** String cleanstatus;

@Column(name = "Price")

**private** String price;

@Column(name = "Bed\_Type", length = 25)

**private** String bedtype;

//Setter And Getter

**public** String getStudentId() {

**return** roomnumber; }

**public** **void** setStudentId(String roomnumber) {

**this**.roomnumber = roomnumber; }

**public** String getAvailability() {

**return** availability; }

**public** **void** setAvailability(String availability) {

**this**.availability = availability;

}

**public** String getCleanStatus() {

**return** cleanstatus;

}

**public** **void** setCleanStatus(String cleanstatus) {

**this**.cleanstatus = cleanstatus;

}

**public** String getPrice() {

**return** price;

}

**public** **void** setPrice(String price) {

**this**.price = price;

}

**public** String getBedType() {

**return** bedtype;

}

**public** **void** setGender(String bedtypes) {

**this**.bedtype = bedtypes;

}

//All argument Constructor

**public** Room(String roomnumber, String availability, String cleanstatus, String price,String bedtype) {**super**();

**this**.roomnumber = roomnumber;

**this**.availability = availability;

**this**.cleanstatus = cleanstatus;

**this**.price = price;

**this**.bedtype = bedtype;

//this.email = email;

//this.phone = phone;

}

//Default Constructor

**public** Room() {

**super**();

}

//ToString method

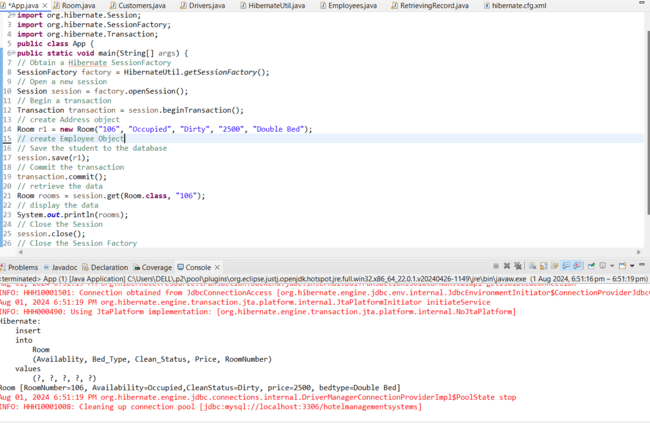
@Override

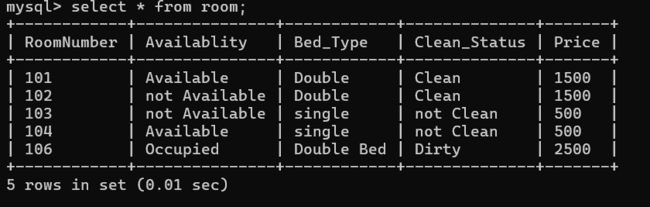
**public** String toString() {

**return** "Room [RoomNumber=" + roomnumber + ", Availability=" + availability + ",CleanStatus=" + cleanstatus+ ", price=" + price + ", bedtype=" + bedtype + "]";

}}

Output:-





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?

Code:-

**package** com.hms;

**import** org.hibernate.Session;

**import** org.hibernate.Transaction;

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

**public** **class** GetMethod {

**public** Room getRoom(String getRoomNumber) {

Transaction transaction = **null**;

Room room = **null**;

**try** (Session session = **new** Configuration().configure().buildSessionFactory().openSession()) {

transaction = session.beginTransaction();

room = session.get(Room.**class**, getRoomNumber);

transaction.commit();

} **catch** (Exception e) {

**if** (transaction != **null**) {

transaction.rollback();

}

e.printStackTrace();

}

**return** room;

}

}

**Main.Class for the handle exceptions.**

Code:-

**package** com.hms;

**public** **class** Main {

**public** **static** **void** main(String[] args) {

GetMethod getMethod = **new** GetMethod();

String getRoomNumber = "999"; // Assuming this ID doesn't exist in the database

Room room = getMethod.getRoom(getRoomNumber);

**if** (room == **null**) {

System.***out***.println("Room number with RoomNo." + getRoomNumber + " does not exist.");

} **else** {

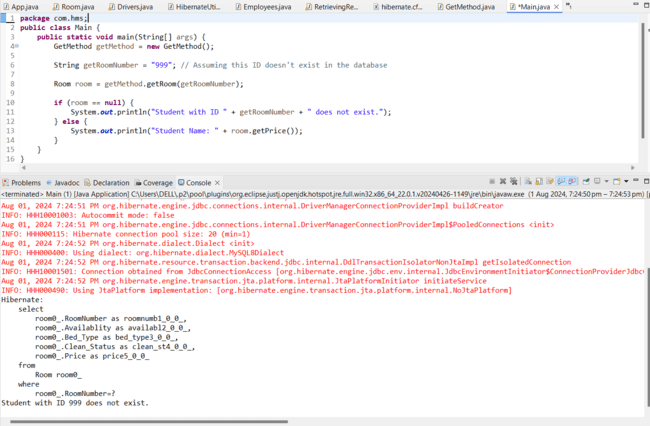
System.***out***.println("Room Price: " + room.getPrice());

}

}

}

Output:-



1. Also demonstrate use of load() method.

Code:-

**package** com.hms;

**import** org.hibernate.Session;

**import** org.hibernate.Transaction;

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

**import** org.hibernate.ObjectNotFoundException;

**public** **class** LoadMethod {

**public** Room loadRoom(String roomNumber) {

Transaction transaction = **null**;

Room room = **null**;

**try** (Session session = **new** Configuration().configure().buildSessionFactory().openSession()) {

transaction = session.beginTransaction();

room = session.load(Room.**class**, roomNumber);

// Trigger loading

room.getRoomNumber();

transaction.commit();

} **catch** (ObjectNotFoundException e) {

System.***out***.println("Room with RoomNumber " + roomNumber + " does not exist.");

} **catch** (Exception e) {

**if** (transaction != **null**) {

transaction.rollback();

}

e.printStackTrace();

}

**return** room;

}

}

**Main.Class for the handle exceptions.**

**package** com.hms;

**import** org.hibernate.ObjectNotFoundException;

**public** **class** Main {

**public** **static** **void** main(String[] args) {

LoadMethod loadMethod = **new** LoadMethod();

String loadRoomNumber = "1315"; // Assuming this ID doesn't exist in the database

**try** {

Room room = loadMethod.loadRoom(loadRoomNumber);

**if** (room != **null**) {

System.***out***.println("Room Price: " + room.getPrice()); // This line will trigger the actual database query }

} **catch** (ObjectNotFoundException e) {

System.***out***.println("Room with RoomNumber " + loadRoomNumber + " does not exist.");

} } }

Output:-

