Yasin Hasim Tamboli

```
package com.app.bean;

import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name = "employee_hql")
public class Employee {
	@Id
	private int empId;
	private String empName;
	private String empAddress;
	private double empSal;

	public int getEmpId() {
		return empId;
	}

	public void setEmpId(int empId) {
		this.empId = empId;
	}

	public String getEmpName() {
		return empName;
	}

	public void setEmpName(String empName) {
		this.empName = empName;
	}

	public String getEmpAddress() {
		return empAddress;
	}

	public void setEmpAddress(String empAddress) {
		this.empAddress = empAddress;
	}

	public double getEmpSal() {
		return empSal;
	}

	public void setEmpSal(double empSal) {
		this.empSal = empSal;
	}

	public Employee(int empId, String empName, String empAddress, double empSal) {
		super();
		this.empId = empId;
		this.empName = empName;
		this.empAddress = empAddress;
		this.empSal = empSal;
	}

	public Employee() {
		super();
		// TODO Auto-generated constructor stub
	}

}

—--------------------------------------------
package com.app.dao;

import java.util.List;

import com.app.bean.Employee;

public interface EmployeeDao {
	int updateData(Employee emp);
	int insertData(Employee emp);
	int deleteData(int id);
	List<Employee> listEmployee();
	List<Employee> getEmployee(int id);

}

—--------------------------------------------------
package com.app.dao.impl;

import java.util.List;

import org.hibernate.Session;
import org.hibernate.Transaction;
import org.hibernate.query.Query;

import com.app.bean.Employee;
import com.app.dao.EmployeeDao;
import com.app.utility.EmployeeUtil;

public class EmployeeDaoImpl implements EmployeeDao{

	public int updateData(Employee emp) {
		// TODO Auto-generated method stub
		Session session=EmployeeUtil.getSession();
		Transaction tx=null;
		try {
			tx=session.beginTransaction();
			Query<Employee>query=session.createQuery("update Employee set
empAddress='"+emp.getEmpAddress()+"' where empId="+emp.getEmpId());
			session.update(emp);
			tx.commit();
			EmployeeUtil.closeSession();
			return 1;

		} catch (Exception e) {
			// TODO: handle exception
			e.printStackTrace();
			tx.rollback();
			return 0;
		}

	}

	public int insertData(Employee emp) {
		// TODO Auto-generated method stub
Session session=EmployeeUtil.getSession();
Transaction tx=null;
try {
	tx=session.beginTransaction();
	session.persist(emp);
	tx.commit();
	EmployeeUtil.closeSession();
	return 1;

} catch (Exception e) {
	// TODO: handle exception
	e.printStackTrace();
```

```
        tx.rollback();
        return 0;
    }

        }

        public int deleteData(int id) {
            // TODO Auto-generated method stub
            Session session=EmployeeUtil.getSession();
            Transaction tx=null;
            try {
                tx=session.beginTransaction();
    String hql="delete from Employee where empId ="+id;
            Query<Employee>query=session.createQuery(hql);

            int row=query.executeUpdate();
            tx.commit();
            EmployeeUtil.closeSession();
            return row;
            } catch (Exception e) {
                // TODO: handle exception
                e.printStackTrace();
                tx.rollback();
                return 0;
            }
        }

        public List<Employee> listEmployee() {
            // TODO Auto-generated method stub
            Session session=EmployeeUtil.getSession();
            Transaction tx=null;
            String hql="From Employee";
            Query<Employee>query=session.createQuery(hql);
            List<Employee> list=query.list();
            EmployeeUtil.closeSession();
            return list;

        }

        public  List<Employee> getEmployee(int id) {
            // TODO Auto-generated method stub
            // TODO Auto-generated method stub
            Session session=EmployeeUtil.getSession();
            Transaction tx=null;
```

```
		String hql="From Employee Where empId ="+id;
		Query<Employee>query=session.createQuery(hql);
		//query.setParameter(1, id);
		List<Employee> list=query.list();
		EmployeeUtil.closeSession();
		return list;

	}

}

—-------------------------------------
package com.app.factory;

import com.app.dao.EmployeeDao;
import com.app.dao.impl.EmployeeDaoImpl;

public class EmployeeFactory {
	public static EmployeeDao getEmployeeDao() {
		return new EmployeeDaoImpl();
	}

}

—------------------------------------
package com.app.utility;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class EmployeeUtil {
	private static SessionFactory factory;
	static {
		try {
			factory = new
Configuration().configure("com/app/config/employee.cfg.xml").buildSessionFactory();

		} catch (Exception e) {
			e.printStackTrace();
		}
	}
	static ThreadLocal<Session> local = new ThreadLocal();
	static Session session = null;
	public static Session getSession() {
		try {
			if (local.get() == null) {
				session = factory.openSession();
				local.set(session);
				return session;
			} else {
				return local.get();
			}
		} catch (Exception e) {
			// TODO: handle exception
			return null;
		}
	}

	public static void closeSession() {
		try {
			session.close();
		} catch (Exception e) {
			// TODO: handle exception
			e.printStackTrace();
		}
	}
}

—-----------------------------------
package com.app.test;

import java.util.List;
import java.util.Scanner;

import com.app.bean.Employee;
import com.app.dao.EmployeeDao;
import com.app.factory.EmployeeFactory;

public class Client {

	public static void main(String[] args) {
		// TODO Auto-generated method stub
		EmployeeDao empDao = EmployeeFactory.getEmployeeDao();
		Scanner sn = new Scanner(System.in);
		int choice;
		String conti;
		do {
			System.out.println("!----------------HQL Operation-------!");
			System.out.println("1. insert data");
			System.out.println("2. update data");
			System.out.println("3. delete data");
			System.out.println("4. Get All data");
			System.out.println("5. get Single data");
			System.out.println("!----------------End-------!");
			System.out.println("Enter you choice:");
			choice = sn.nextInt();

			switch (choice) {
			case 1:
				System.out.println("Enter your id:");
				int id = sn.nextInt();
				System.out.println("Enter your name:");
				String name = sn.next();
				System.out.println("Enter your Address:");
				String address = sn.next();
				System.out.println("Enter your Salary:");
				double sal = sn.nextDouble();
				Employee emp = new Employee(id, name, address, sal);
				int i = empDao.insertData(emp);
				if (i == 1) {
					System.out.println("Data inserted successfully.");
				} else {
					System.out.println("Data Not Inserted something went
wrong..!");
				}
				break;
			case 2:
				System.out.println("Enter your id:");
				int id2 = sn.nextInt();
				System.out.println("Enter your name:");
				String name1 = sn.next();
				System.out.println("Enter your Address:");
				String address1 = sn.next();
				System.out.println("Enter your Salary:");
				double sal1 = sn.nextDouble();
				Employee emp5 = new Employee(id2, name1, address1, sal1);
				int i2 = empDao.updateData(emp5);
				if (i2 == 1) {
					System.out.println("Data update successfully.");
				} else {
						System.out.println("Data Not Inserted something went
wrong..!");
					}
					break;
				case 3:
					System.out.println("Enter your id:");
					int id1 = sn.nextInt();

					int row = empDao.deleteData(id1);
					if (row == 1) {
						System.out.println("Data deleted successfully.");
					} else {
						System.out.println("Data Not Inserted something went
wrong..!");
					}
					break;
				case 4:

					List<Employee> list = empDao.listEmployee();
					if (list != null) {
						for (Employee e : list) {
							System.out.println(
									e.getEmpId() + "\t" +
e.getEmpName() + "\t" + e.getEmpAddress() + "\t" + e.getEmpSal());
						}
					} else {
						System.out.println("something went wrong..!");
					}
					break;
				case 5:
					System.out.println("Enter your id:");
					int empId = sn.nextInt();

					List<Employee> emp1 = empDao.getEmployee(empId);
					if (emp1 != null) {
						for (Employee e : emp1) {
							System.out.println(
									e.getEmpId() + "\t" +
e.getEmpName() + "\t" + e.getEmpAddress() + "\t" + e.getEmpSal());
						}
					} else {
						System.out.println("Data Not Inserted something went
wrong..!");
					}
```

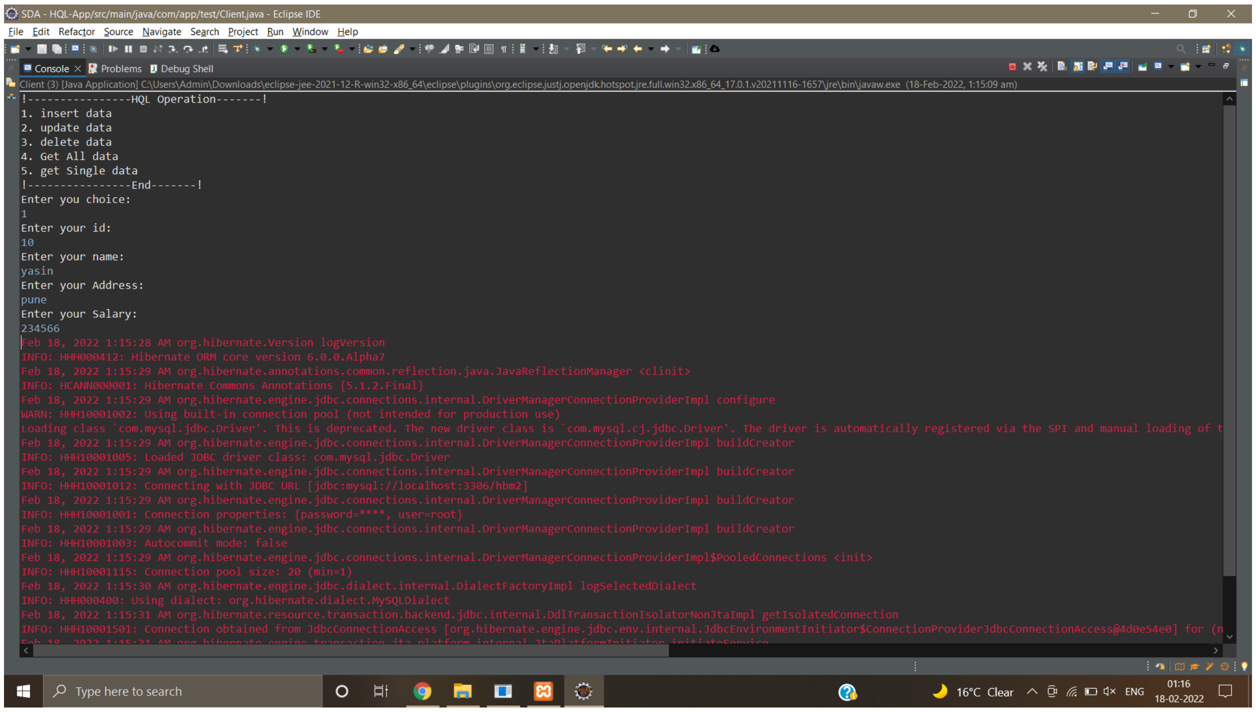
```
                                        break;
                                default:
                                        break;
                                }

                                System.out.println("do you want to continue...!");
                                conti = sn.next();

                        } while (conti.equalsIgnoreCase("y"));

                }

        }
```

—---

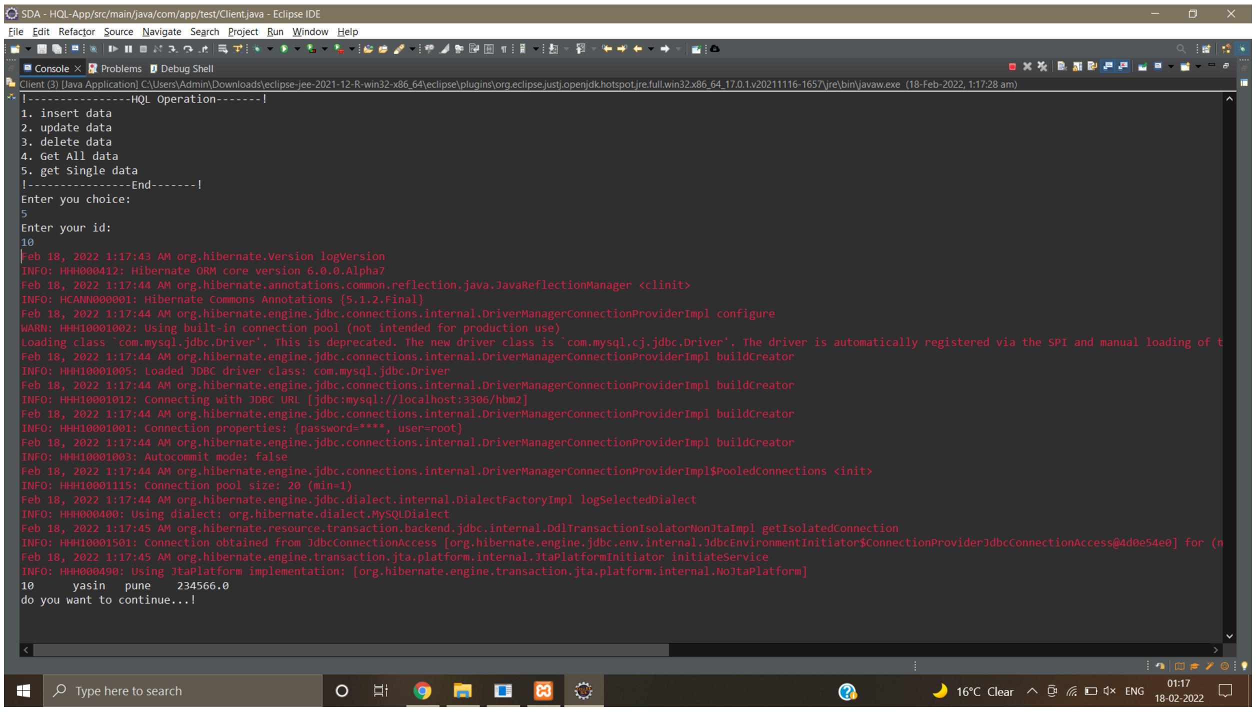
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
        "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
        "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>
        <session-factory>
                <property name="hibernate.dialect">
                        org.hibernate.dialect.MySQLDialect
                </property>
                <property name="hbm2ddl.auto">
                        update</property>

                <property name="hibernate.connection.driver_class">
                        com.mysql.jdbc.Driver
                </property>

                <property name="hibernate.connection.url">
                        jdbc:mysql://localhost:3306/hbm2
                </property>
                <property name="hibernate.connection.username">
                        root
                </property>
                <property name="hibernate.connection.password">

                </property>
                <mapping class="com.app.bean.Employee" />
        </session-factory>
</hibernate-configuration>
```

—---

Output:



```
Client (3) [Java Application] C:\Users\Admin\Downloads\eclipse-jee-2021-12-R-win32-x86_64\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1657\jre\bin\javaw.exe  (18-Feb-2022, 1:15:09 am)
!---------------HQL Operation-------!
1. insert data
2. update data
3. delete data
4. Get All data
5. get Single data
!---------------End-------!
Enter you choice:
1
Enter your id:
10
Enter your name:
yasin
Enter your Address:
pune
Enter your Salary:
234566
Feb 18, 2022 1:15:28 AM org.hibernate.Version logVersion
INFO: HHH000412: Hibernate ORM core version 6.0.0.Alpha7
Feb 18, 2022 1:15:29 AM org.hibernate.annotations.common.reflection.java.JavaReflectionManager <clinit>
INFO: HCANN000001: Hibernate Commons Annotations {5.1.2.Final}
Feb 18, 2022 1:15:29 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl configure
WARN: HHH10001002: Using built-in connection pool (not intended for production use)
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new driver class is `com.mysql.cj.jdbc.Driver'. The driver is automatically registered via the SPI and manual loading of t
Feb 18, 2022 1:15:29 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001005: Loaded JDBC driver class: com.mysql.jdbc.Driver
Feb 18, 2022 1:15:29 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001012: Connecting with JDBC URL [jdbc:mysql://localhost:3306/hbm2]
Feb 18, 2022 1:15:29 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001001: Connection properties: {password=****, user=root}
Feb 18, 2022 1:15:29 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001003: Autocommit mode: false
Feb 18, 2022 1:15:29 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PooledConnections <init>
INFO: HHH10001115: Connection pool size: 20 (min=1)
Feb 18, 2022 1:15:30 AM org.hibernate.engine.jdbc.dialect.internal.DialectFactoryImpl logSelectedDialect
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQLDialect
Feb 18, 2022 1:15:31 AM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbcConnectionAccess@4d0e54e0] for (n
Feb 18, 2022 1:15:31 AM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
```



```
Client (3) [Java Application] C:\Users\Admin\Downloads\eclipse-jee-2021-12-R-win32-x86_64\eclipse\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.1.v20211116-1657\jre\bin\javaw.exe  (18-Feb-2022, 1:17:28 am)
!---------------HQL Operation-------!
1. insert data
2. update data
3. delete data
4. Get All data
5. get Single data
!---------------End-------!
Enter you choice:
5
Enter your id:
10
Feb 18, 2022 1:17:43 AM org.hibernate.Version logVersion
INFO: HHH000412: Hibernate ORM core version 6.0.0.Alpha7
Feb 18, 2022 1:17:44 AM org.hibernate.annotations.common.reflection.java.JavaReflectionManager <clinit>
INFO: HCANN000001: Hibernate Commons Annotations {5.1.2.Final}
Feb 18, 2022 1:17:44 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl configure
WARN: HHH10001002: Using built-in connection pool (not intended for production use)
Loading class `com.mysql.jdbc.Driver'. This is deprecated. The new driver class is `com.mysql.cj.jdbc.Driver'. The driver is automatically registered via the SPI and manual loading of t
Feb 18, 2022 1:17:44 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001005: Loaded JDBC driver class: com.mysql.jdbc.Driver
Feb 18, 2022 1:17:44 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001012: Connecting with JDBC URL [jdbc:mysql://localhost:3306/hbm2]
Feb 18, 2022 1:17:44 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001001: Connection properties: {password=****, user=root}
Feb 18, 2022 1:17:44 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001003: Autocommit mode: false
Feb 18, 2022 1:17:44 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PooledConnections <init>
INFO: HHH10001115: Connection pool size: 20 (min=1)
Feb 18, 2022 1:17:44 AM org.hibernate.engine.jdbc.dialect.internal.DialectFactoryImpl logSelectedDialect
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQLDialect
Feb 18, 2022 1:17:45 AM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbcConnectionAccess@4d0e54e0] for (n
Feb 18, 2022 1:17:45 AM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
10      yasin   pune    234566.0
do you want to continue...!
```