Name_ Sangharsh Sitaram Narwade

Mob No_8308407924

One To One Mapping Program

1)Professor.class

```
package com.demo;


import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.OneToOne;

@Entity

public class Professor {

@Id

@@GeneratedValue

private int id;

@Column(name = "Professor_Name")

private String name;

@OneToOne(targetEntity = Subject.class,cascade = CascadeType.ALL,orphanRemoval = true)


private Subject subject;

public Subject getSubject() {

return subject;

}

public void setSubject(Subject subject) {

this.subject = subject;
}
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;
}
}
```

2)Subject.class

```
package com.demo;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;

@Entity
public class Subject {
@Id
@GeneratedValue
private int id;
private String subjectName;
public int getId() {
return id;
}
public void setId(int id) {
this.id = id;
}
public String getSubjectName() {
return subjectName;
}
public void setSubjectName(String subjectName) {
this.subjectName = subjectName;
}
}
```

3)School.main.class

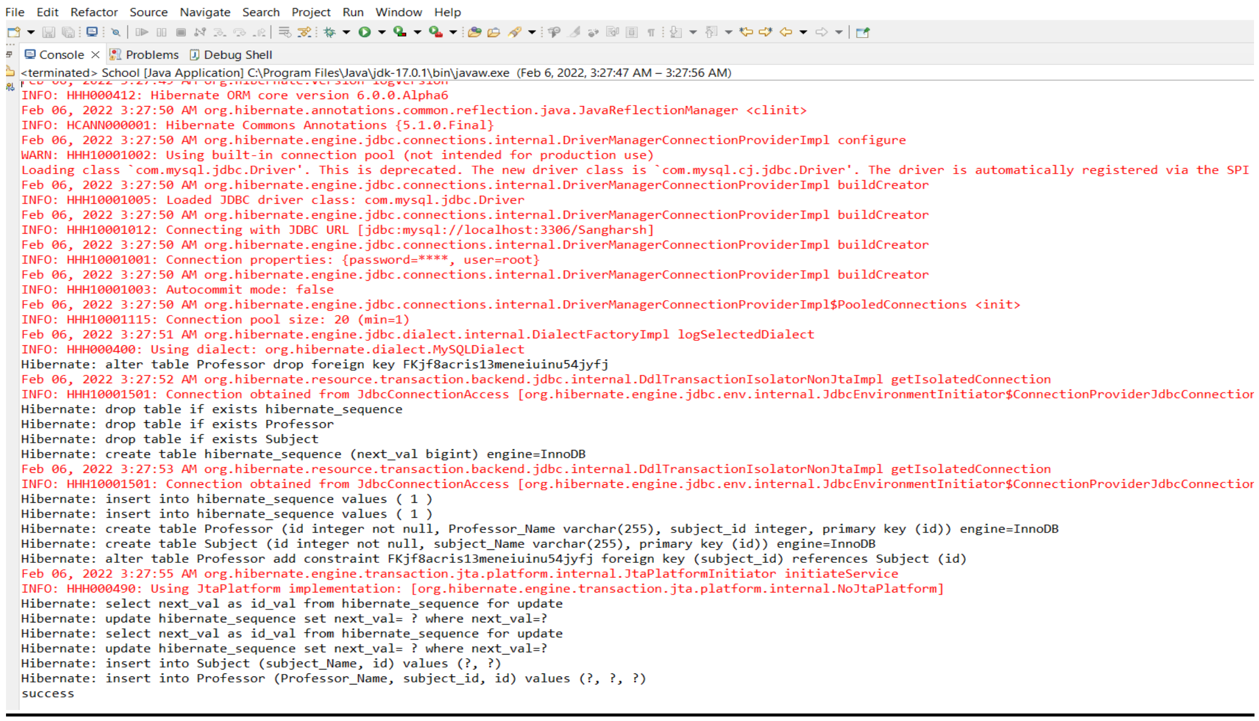
```
package com.demo;
import org.hibernate.Session;
import org.hibernate.Transaction;
import org.hibernate.cfg.Configuration;

public class School {
public static void main(String[] args) {
Configuration cfg=new Configuration().configure("hibernate.cfg.xml");
Session session=cfg.buildSessionFactory().openSession();
Transaction txn=session.beginTransaction();
Professor p=new Professor();

p.setName("sangharsh");

Subject s=new Subject();

s.setSubjectName("java");

p.setSubject(s);

session.save(p);

session.save(s);

txn.commit();

System.out.println("success");

session.close();

}

}
```

4)hibernate.cfg.xml

```
<?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">create</property>

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

<!-- Assume test is the database name -->

<property name =
"hibernate.connection.url">jdbc:mysql://localhost:3306/Sangharsh</property>

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

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

<!-- List of XML mapping files -->

<mapping class="com.demo.Professor"/>
<mapping class="com.demo.Subject"/>

</session-factory>
</hibernate-configuration>
```

Output


File Edit Refactor Source Navigate Search Project Run Window Help
Console
Problems
Debug Shell

```
<terminated> School [Java Application] C:\Program Files\Java\jdk-17.0.1\bin\javaw.exe  (Feb 6, 2022, 3:27:47 AM – 3:27:56 AM)
INFO: HHH000412: Hibernate ORM core version 6.0.0.Alpha6
Feb 06, 2022 3:27:50 AM org.hibernate.annotations.common.reflection.java.JavaReflectionManager <clinit>
INFO: HCANN000001: Hibernate Commons Annotations {5.1.0.Final}
Feb 06, 2022 3:27:50 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
Feb 06, 2022 3:27:50 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001005: Loaded JDBC driver class: com.mysql.jdbc.Driver
Feb 06, 2022 3:27:50 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001012: Connecting with JDBC URL [jdbc:mysql://localhost:3306/Sangharsh]
Feb 06, 2022 3:27:50 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001001: Connection properties: {password=****, user=root}
Feb 06, 2022 3:27:50 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001003: Autocommit mode: false
Feb 06, 2022 3:27:50 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PooledConnections <init>
INFO: HHH10001115: Connection pool size: 20 (min=1)
Feb 06, 2022 3:27:51 AM org.hibernate.engine.jdbc.dialect.internal.DialectFactoryImpl logSelectedDialect
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQLDialect
Hibernate: alter table Professor drop foreign key FKjf8acris13meneiuinu54jyfj
Feb 06, 2022 3:27:52 AM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbcConnectior
Hibernate: drop table if exists hibernate_sequence
Hibernate: drop table if exists Professor
Hibernate: drop table if exists Subject
Hibernate: create table hibernate_sequence (next_val bigint) engine=InnoDB
Feb 06, 2022 3:27:53 AM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbcConnectior
Hibernate: insert into hibernate_sequence values ( 1 )
Hibernate: insert into hibernate_sequence values ( 1 )
Hibernate: create table Professor (id integer not null, Professor_Name varchar(255), subject_id integer, primary key (id)) engine=InnoDB
Hibernate: create table Subject (id integer not null, subject_Name varchar(255), primary key (id)) engine=InnoDB
Hibernate: alter table Professor add constraint FKjf8acris13meneiuinu54jyfj foreign key (subject_id) references Subject (id)
Feb 06, 2022 3:27:55 AM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select next_val as id_val from hibernate_sequence for update
Hibernate: update hibernate_sequence set next_val= ? where next_val=?
Hibernate: select next_val as id_val from hibernate_sequence for update
Hibernate: update hibernate_sequence set next_val= ? where next_val=?
Hibernate: insert into Subject (subject_Name, id) values (?, ?)
Hibernate: insert into Professor (Professor_Name, subject_id, id) values (?, ?, ?)
success
```

Name _Sangharsh Sitaram Narwade

Mob No_8308407924

Hibernate Many to One Mapping Program

1)EmpDetails.class

```
package com.app.model;

import javax.persistence.CascadeType;
import javax.persistence.Entity;
import javax.persistence.FetchType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;

@Entity
public class EmpDetails {
@Id
private int eNo;
public EmpDetails(int eNo, String eName, long salary, Department department) {
super();
this.eNo = eNo;
this.eName = eName;
this.salary = salary;
this.department = department;
}

public int geteNo() {
return eNo;
}
public void seteNo(int eNo) {
this.eNo = eNo;
}
public String geteName() {
return eName;
}
public void seteName(String eName) {
```

```
this.eName = eName;
}
public long getSalary() {
return salary;
}
public void setSalary(long salary) {
this.salary = salary;
}
public Department getDepartment() {
return department;
}
public void setDepartment(Department department) {
this.department = department;
}
private String eName;
private long salary;
@ManyToOne(targetEntity = Department.class, cascade = CascadeType.ALL, fetch =
FetchType.EAGER)
@JoinColumn(name = "deptNo", referencedColumnName = "deptNo")
private Department department;
}
```

2)Department.class

```
package com.app.model;
import javax.persistence.Entity;
import javax.persistence.Id;

@Entity
public class Department {
@Id
private int deptNo;
private String deptName;

public Department(int deptNo, String deptName, String deptHead) {
super();
this.deptNo = deptNo;
this.deptName = deptName;
this.deptHead = deptHead;
}
private String deptHead;
public int getDeptNo() {
return deptNo;
}
public void setDeptNo(int deptNo) {
this.deptNo = deptNo;
}
public String getDeptName() {
return deptName;
}
public void setDeptName(String deptName) {
this.deptName = deptName;
}
public String getDeptHead() {
return deptHead;
}
public void setDeptHead(String deptHead) {
this.deptHead = deptHead;
}
}
```

3)Many to One Dao.class

```
package com.app.dao;

public interface ManyToOneDao {
void addEmployeeWithDept();
void displayEmpAndDept();
}
```

4)Many to One Dao Impl.class

```
package com.app.dao.impl;

import java.util.ArrayList;
import java.util.List;
import org.hibernate.Session;
import org.hibernate.Transaction;
import org.hibernate.query.Query;

import com.app.dao.ManyToOneDao;
import com.app.model.Department;
import com.app.model.EmpDetails;
import com.app.model.User;
import com.app.util.UtilityClass;

public class ManyToOneDaoImpl implements ManyToOneDao{

public void addEmployeeWithDept() {

Session session=UtilityClass.getSession();
Department dept=new Department(1, "HR", "Akash");
Department dept2=new Department(2, "devloper", "shubham");
EmpDetails emp1=new EmpDetails(2001, "sangharsh", 200020, dept);
EmpDetails emp2=new EmpDetails(2002, "Ajay", 30000, dept2);
Transaction tx=session.beginTransaction();
//session.save(emp1);
session.update(emp2);
tx.commit();
UtilityClass.closeSession();
List<EmpDetails>list=new ArrayList<EmpDetails>();
list.add(emp1);
list.add(emp2);
}
public void displayEmpAndDept() {
Session session=UtilityClass.getSession();
Query<EmpDetails>query=session.createQuery("from EmpDetails");
List<EmpDetails>list=query.list();
for(EmpDetails emp : list ) {
System.out.println(emp.geteName() + "\t" + emp.geteNo() + "\t" + emp.getSalary());
}
UtilityClass.closeSession();

}
}
```

5)Many to One Dao Factory.class

```
package com.app.factory;

import com.app.dao.ManyToOneDao;
import com.app.dao.impl.ManyToOneDaoImpl;

public class ManyToOneFactory {
public static ManyToOneDao getManyInstance() {
return new ManyToOneDaoImpl();
}
}
```

6)Utility.class

```
    package com.app.util;

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

    public class UtilityClass {
    private static SessionFactory factory;
    static {
    try {

    factory = new Configuration().configure("hibernate.cfg.xml").buildSessionFactory();

    } catch (Exception e) {
    // TODO: handle exception
    e.printStackTrace();
    }
    }
    static ThreadLocal<Session> threadLocal = new ThreadLocal<Session>();
    static Session session = null;

    public static Session getSession() {
```

```
try {
if(threadLocal.get()==null) {
session=factory.openSession();
return session;
}
else {
return threadLocal.get();
}
}catch (Exception e) {
// TODO: handle exception
return null;
}

}
public static void closeSession() {
try {
session.close();

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

7)hibernate.cfg.xml

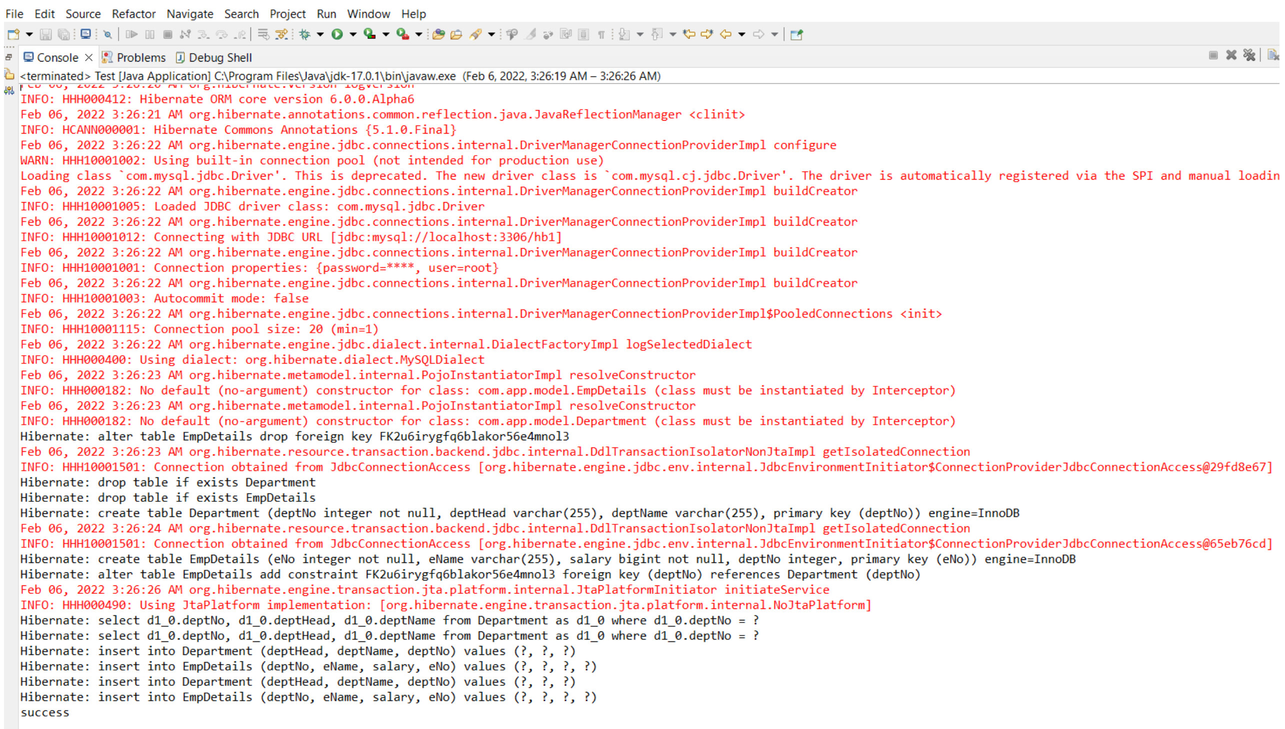
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
"-//Hibernate/Hibernate Configuration DTD 3.0//EN"
"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>
<session-factory>

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

<property name="hbm2ddl.auto">create</property>
```

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

<!-- Assume test is the database name -->

<property name = "hibernate.connection.url">jdbc:mysql://localhost:3306/hb1</property>
<property name = "hibernate.connection.username">root</property>

<property name = "hibernate.connection.password"></property>
<property name="show_sql">True</property>


<!--  <property name="show_sql">true</property>
<property name="format_sql">true</property> -->
<!-- List of XML mapping files -->
<mapping class="com.app.model.EmpDetails"/>
<mapping class="com.app.model.Department"/>
</session-factory>
</hibernate-configuration>
```

8)Test.class

```
package com.app.client;

import com.app.dao.ManyToOneDao;

import com.app.factory.ManyToOneFactory;

public class Test {

public static void main(String[] args) {

ManyToOneDao dao = ManyToOneFactory.getManyInstance();
dao.addEmployeeWithDept();
System.out.println("success");
}
}
```

Output:


File Edit Source Refactor Navigate Search Project Run Window Help
Console
Problems
Debug Shell

```
<terminated> Test [Java Application] C:\Program Files\Java\jdk-17.0.1\bin\javaw.exe  (Feb 6, 2022, 3:26:19 AM – 3:26:26 AM)
INFO: HHH000412: Hibernate ORM core version 6.0.0.Alpha6
Feb 06, 2022 3:26:21 AM org.hibernate.annotations.common.reflection.java.JavaReflectionManager <clinit>
INFO: HCANN000001: Hibernate Commons Annotations {5.1.0.Final}
Feb 06, 2022 3:26:22 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 loadin
Feb 06, 2022 3:26:22 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001005: Loaded JDBC driver class: com.mysql.jdbc.Driver
Feb 06, 2022 3:26:22 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001012: Connecting with JDBC URL [jdbc:mysql://localhost:3306/hb1]
Feb 06, 2022 3:26:22 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001001: Connection properties: {password=****, user=root}
Feb 06, 2022 3:26:22 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001003: Autocommit mode: false
Feb 06, 2022 3:26:22 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PooledConnections <init>
INFO: HHH10001115: Connection pool size: 20 (min=1)
Feb 06, 2022 3:26:22 AM org.hibernate.engine.jdbc.dialect.internal.DialectFactoryImpl logSelectedDialect
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQLDialect
Feb 06, 2022 3:26:23 AM org.hibernate.metamodel.internal.PojoInstantiatorImpl resolveConstructor
INFO: HHH000182: No default (no-argument) constructor for class: com.app.model.EmpDetails (class must be instantiated by Interceptor)
Feb 06, 2022 3:26:23 AM org.hibernate.metamodel.internal.PojoInstantiatorImpl resolveConstructor
INFO: HHH000182: No default (no-argument) constructor for class: com.app.model.Department (class must be instantiated by Interceptor)
Hibernate: alter table EmpDetails drop foreign key FK2u6irygfq6blakor56e4mnol3
Feb 06, 2022 3:26:23 AM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbcConnectionAccess@29fd8e67]
Hibernate: drop table if exists Department
Hibernate: drop table if exists EmpDetails
Hibernate: create table Department (deptNo integer not null, deptHead varchar(255), deptName varchar(255), primary key (deptNo)) engine=InnoDB
Feb 06, 2022 3:26:24 AM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbcConnectionAccess@65eb76cd]
Hibernate: create table EmpDetails (eNo integer not null, eName varchar(255), salary bigint not null, deptNo integer, primary key (eNo)) engine=InnoDB
Hibernate: alter table EmpDetails add constraint FK2u6irygfq6blakor56e4mnol3 foreign key (deptNo) references Department (deptNo)
Feb 06, 2022 3:26:26 AM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select d1_0.deptNo, d1_0.deptHead, d1_0.deptName from Department as d1_0 where d1_0.deptNo = ?
Hibernate: select d1_0.deptNo, d1_0.deptHead, d1_0.deptName from Department as d1_0 where d1_0.deptNo = ?
Hibernate: insert into Department (deptHead, deptName, deptNo) values (?, ?, ?)
Hibernate: insert into EmpDetails (deptNo, eName, salary, eNo) values (?, ?, ?, ?)
Hibernate: insert into Department (deptHead, deptName, deptNo) values (?, ?, ?)
Hibernate: insert into EmpDetails (deptNo, eName, salary, eNo) values (?, ?, ?, ?)
success
```

Name:Sangharsh Sitaram Narwade
Mob No: 8308407924

Hibernate One to Many Mapping program

1) User.class

```
package com.app.model;
import java.util.List;
import javax.persistence.CascadeType;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.OneToMany;
import javax.persistence.OrderColumn;
import javax.persistence.Table;
@Entity
@Table(name = "user_table")
public class User {
@Override
public String toString() {
return "User [UserId=" + UserId + ", fName=" + fName + ", lName=" + lName + "]";
}
@Id
@Column(name = "User_id")
private int UserId;
@Column(name = "First_name")
private String fName;
@Column(name = "Last_name")
@OneToMany(targetEntity = PhoneNumber.class, cascade = CascadeType.ALL,
orphanRemoval = true)
@JoinColumn(name = "unid", referencedColumnName = "user_id")
@OrderColumn(name = "list_index")
private List<PhoneNumber> phoneNumbers;
public int getUserId() {
return UserId;
}
public void setUserId(int userId) {
UserId = userId;
}
public String getfName() {
return fName;
}

public void setfName(String fName) {
this.fName = fName;
}

public String getlName() {
return lName;
}

public void setlName(String lName) {
this.lName = lName;
}

private String lName;

public List<PhoneNumber> getPhoneNumbers() {
return phoneNumbers;
}

public void setPhoneNumbers(List<PhoneNumber> phoneNumbers) {
this.phoneNumbers = phoneNumbers;
}
}
```

2) <u>**PhoneNumber.class**</u>

```
package com.app.model;
import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Table;
@Entity
@Table(name = "phoneNumber_table")
public class PhoneNumber {
@Id
private long phone;
@Column(name = "number_type")
private String numberType;
public long getPhone() {
return phone;
}
public void setPhone(long phone) {
this.phone = phone;
}
public String getNumberType() {
return numberType;
}
public void setNumberType(String numberType) {
this.numberType = numberType;
}
@Override
public String toString() {
return "PhoneNumber [phone=" + phone + ", numberType=" + numberType + "]";
}
}
```

3) One to many Dao.class

```
package com.app.dao;
public interface OneToManyDao {
void insertData();
}
```

4) One to many Dao Impl.class

```
package com.app.dao.impl;
import java.util.ArrayList;
import java.util.List;

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

import com.app.dao.OneToManyDao;
import com.app.model.PhoneNumber;
import com.app.model.User;
import com.app.util.UtilityClass;
```

```
public class OneToManyDaoimpl implements OneToManyDao {

public void insertData() {
Session session=UtilityClass.getSession();
Transaction tx=session.beginTransaction();
PhoneNumber phoneNumber=new PhoneNumber();
phoneNumber.setPhone(976882325);
phoneNumber.setNumberType("office");

PhoneNumber phoneNumber1=new PhoneNumber();
phoneNumber1.setPhone(830847223);
phoneNumber1.setNumberType("home");

List<PhoneNumber>list= new ArrayList<PhoneNumber>();
list.add(phoneNumber1);
list.add(phoneNumber);

User user=new User();
user.setfName("sangharsh");
user.setlName("Narwade");
user.setUserId(101);
user.setPhoneNumbers(list);
session.save(user);
tx.commit();
UtilityClass.closeSession();

}

}
```

5)One to many Dao Factory.class

```
package com.app.factory;
import com.app.dao.OneToManyDao;
import com.app.dao.impl.OneToManyDaoimpl;
public class OneToManyFactory {
public static OneToManyDao getInstance() {
return new OneToManyDaoimpl();
}
}
```

6)Utility.class

```
package com.app.util;

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

public class UtilityClass {
private static SessionFactory factory;
static {
try {
 factory = new Configuration().configure("hibernate.cfg.xml").buildSessionFactory();

} catch (Exception e) {
// TODO: handle exception
e.printStackTrace();
}
}
static ThreadLocal<Session> threadLocal = new ThreadLocal<Session>();
static Session session = null;

public static Session getSession() {
try {
if(threadLocal.get()==null) {
session=factory.openSession();
return session;
}
else {
return threadLocal.get();
}
}catch (Exception e) {
// TODO: handle exception
return null;
}
}
public static void closeSession() {
try {
session.close();

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

7) hibernate.cfg.xml

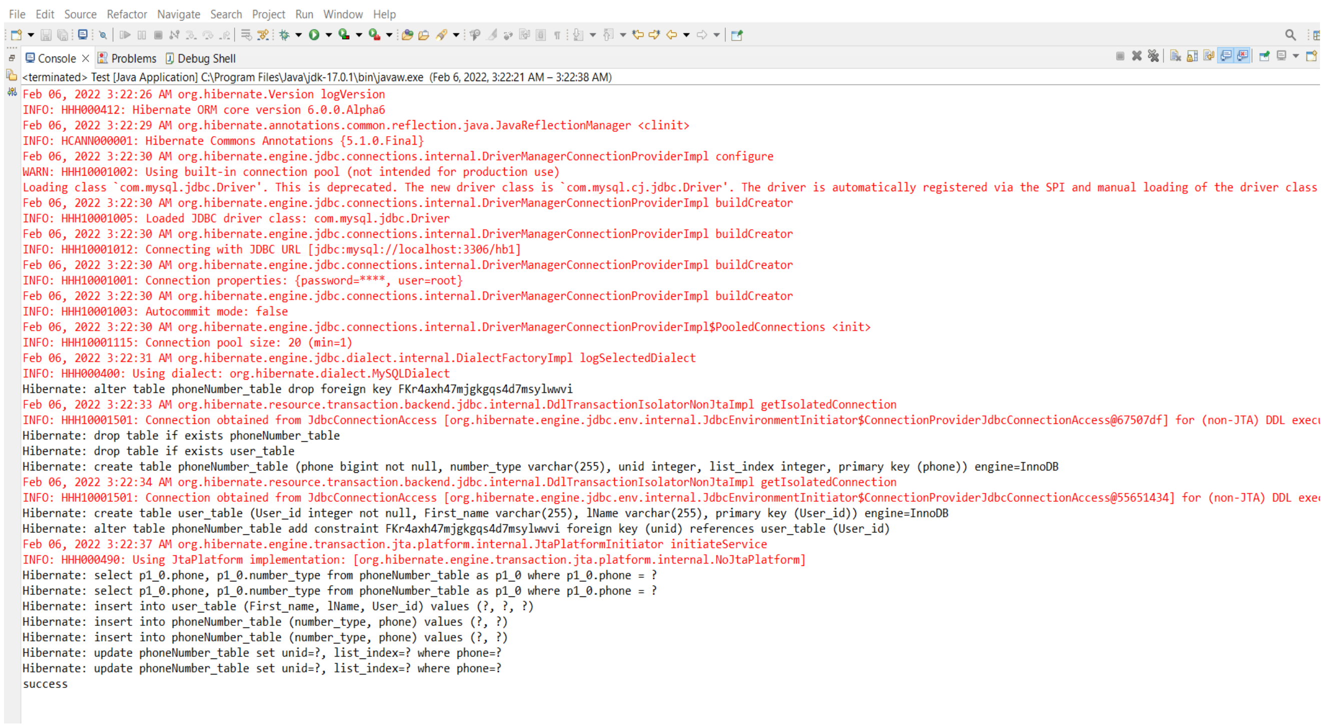
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
"-//Hibernate/Hibernate Configuration DTD 3.0//EN"
"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>
<session-factory>

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

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

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

<!-- Assume test is the database name -->

<property name = "hibernate.connection.url">jdbc:mysql://localhost:3306/hb1</property>
<property name = "hibernate.connection.username">root</property>

<property name = "hibernate.connection.password"></property>
<property name="show_sql">True</property>


<!--  <property name="show_sql">true</property>
<property name="format_sql">true</property> -->
<!-- List of XML mapping files -->
```

```
<mapping class="com.app.model.User"/>
<mapping class="com.app.model.PhoneNumber"/>
</session-factory>
</hibernate-configuration>
```

8)Test.class

```
package com.app.client;

import com.app.dao.OneToManyDao;
import com.app.factory.OneToManyFactory;

public class Test {
public static void main(String[] args) {
OneToManyDao dao = OneToManyFactory.getInstance();
dao.insertData();
System.out.println("success");
}
}
```

Output:


File Edit Source Refactor Navigate Search Project Run Window Help
Console
Problems
Debug Shell

```
<terminated> Test [Java Application] C:\Program Files\Java\jdk-17.0.1\bin\javaw.exe (Feb 6, 2022, 3:22:21 AM – 3:22:38 AM)
Feb 06, 2022 3:22:26 AM org.hibernate.Version logVersion
INFO: HHH000412: Hibernate ORM core version 6.0.0.Alpha6
Feb 06, 2022 3:22:29 AM org.hibernate.annotations.common.reflection.java.JavaReflectionManager <clinit>
INFO: HCANN000001: Hibernate Commons Annotations {5.1.0.Final}
Feb 06, 2022 3:22:30 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 the driver class
Feb 06, 2022 3:22:30 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001005: Loaded JDBC driver class: com.mysql.jdbc.Driver
Feb 06, 2022 3:22:30 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001012: Connecting with JDBC URL [jdbc:mysql://localhost:3306/hb1]
Feb 06, 2022 3:22:30 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001001: Connection properties: {password=****, user=root}
Feb 06, 2022 3:22:30 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl buildCreator
INFO: HHH10001003: Autocommit mode: false
Feb 06, 2022 3:22:30 AM org.hibernate.engine.jdbc.connections.internal.DriverManagerConnectionProviderImpl$PooledConnections <init>
INFO: HHH10001115: Connection pool size: 20 (min=1)
Feb 06, 2022 3:22:31 AM org.hibernate.engine.jdbc.dialect.internal.DialectFactoryImpl logSelectedDialect
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQLDialect
Hibernate: alter table phoneNumber_table drop foreign key FKr4axh47mjgkgqs4d7msylwwvi
Feb 06, 2022 3:22:33 AM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbcConnectionAccess@67507df] for (non-JTA) DDL exec
Hibernate: drop table if exists phoneNumber_table
Hibernate: drop table if exists user_table
Hibernate: create table phoneNumber_table (phone bigint not null, number_type varchar(255), unid integer, list_index integer, primary key (phone)) engine=InnoDB
Feb 06, 2022 3:22:34 AM org.hibernate.resource.transaction.backend.jdbc.internal.DdlTransactionIsolatorNonJtaImpl getIsolatedConnection
INFO: HHH10001501: Connection obtained from JdbcConnectionAccess [org.hibernate.engine.jdbc.env.internal.JdbcEnvironmentInitiator$ConnectionProviderJdbcConnectionAccess@55651434] for (non-JTA) DDL exe
Hibernate: create table user_table (User_id integer not null, First_name varchar(255), lName varchar(255), primary key (User_id)) engine=InnoDB
Hibernate: alter table phoneNumber_table add constraint FKr4axh47mjgkgqs4d7msylwwvi foreign key (unid) references user_table (User_id)
Feb 06, 2022 3:22:37 AM org.hibernate.engine.transaction.jta.platform.internal.JtaPlatformInitiator initiateService
INFO: HHH000490: Using JtaPlatform implementation: [org.hibernate.engine.transaction.jta.platform.internal.NoJtaPlatform]
Hibernate: select p1_0.phone, p1_0.number_type from phoneNumber_table as p1_0 where p1_0.phone = ?
Hibernate: select p1_0.phone, p1_0.number_type from phoneNumber_table as p1_0 where p1_0.phone = ?
Hibernate: insert into user_table (First_name, lName, User_id) values (?, ?, ?)
Hibernate: insert into phoneNumber_table (number_type, phone) values (?, ?)
Hibernate: insert into phoneNumber_table (number_type, phone) values (?, ?)
Hibernate: update phoneNumber_table set unid=?, list_index=? where phone=?
Hibernate: update phoneNumber_table set unid=?, list_index=? where phone=?
success
```