Mapping Set in Collection Mapping

If our persistent class has Set object, we can map the Set by set element in the mapping file. The set element doesn't require index element. The one difference between List and Set is that, it stores only unique values.

Let's see how we can implement the set in the mapping file:

```
<class name="com.javatpoint.Question" table="q102">
    ...
      <set name="answers" table="ans102">
      <key column="qid"></key>
      <element column="answer" type="string"></element>
      </set>

    ...
</class>
```

Example of mapping set in collection mapping

In this example, we are going to see full example of collection mapping by set. This is the example of set that stores value not entity reference that is why are going to use element instead of one-to-many.

1) create the Persistent class

This persistent class defines properties of the class including Set.

```
package com.javatpoint;


import java.util.Set;


public class Question {
private int id;
private String qname;
private Set<String> answers;
```

```
//getters and setters


}
```

2) create the Mapping file for the persistent class

Here, we have created the question.hbm.xml file for defining the list.

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE hibernate-mapping PUBLIC
        "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
        "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<hibernate-mapping>
 <class name="com.javatpoint.Question" table="q102">
   <id name="id">
     <generator class="increment"></generator>
   </id>
   <property name="qname"></property>

   <set name="answers" table="ans102">
     <key column="qid"></key>
     <element column="answer" type="string"></element>
   </set>

 </class>

</hibernate-mapping>
```

3) create the configuration file

This file contains information about the database and mapping file.

```
<?xml version='1.0' encoding='UTF-8'?>
<!DOCTYPE hibernate-configuration PUBLIC
        "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
```

```
        "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

<!-- Generated by MyEclipse Hibernate Tools.                   -->
<hibernate-configuration>

    <session-factory>
        <property name="hbm2ddl.auto">update</property>
        <property name="dialect">org.hibernate.dialect.Oracle9Dialect</property>
        <property name="connection.url">jdbc:oracle:thin:@localhost:1521:xe</property>
        <property name="connection.username">system</property>
        <property name="connection.password">oracle</property>
        <property name="connection.driver_class">oracle.jdbc.driver.OracleDriver</property>
    <mapping resource="question.hbm.xml"/>
    </session-factory>

</hibernate-configuration>
```

4) Create the class to store the data

In this class we are storing the data of the question class.

```
package com.javatpoint;

import java.util.ArrayList;

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

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



    HashSet<String> set1=new HashSet<String>();
```

```
    set1.add("java is a programming language");
    set1.add("java is a platform");

    HashSet<String> set2=new HashSet<String>();
    set2.add("Servlet is an Interface");
    set2.add("Servlet is an API");

    Question question1=new Question();
    question1.setQname("What is Java?");
    question1.setAnswers(set1);

    Question question2=new Question();
    question2.setQname("What is Servlet?");
    question2.setAnswers(set2);

    session.persist(question1);
    session.persist(question2);

    t.commit();
    session.close();
    System.out.println("success");
 }
}
```

download this hibernate example

How to fetch the data of Set

Here, we have used HQL to fetch all the records of Question class including answers. In such case, it fetches the data from two tables that are functional dependent.

```
package com.javatpoint;

import java.util.*;

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

```
public class FetchData {
public static void main(String[] args) {

    Session session=new Configuration().configure("hibernate.cfg.xml")
                    .buildSessionFactory().openSession();

    Query query=session.createQuery("from Question");
    List<Question> list=query.list();

    Iterator<Question> itr=list.iterator();
    while(itr.hasNext()){
        Question q=itr.next();
        System.out.println("Question Name: "+q.getQname());

        //printing answers
        Set<String> set=q.getAnswers();
        Iterator<String> itr2=set.iterator();
        while(itr2.hasNext()){
            System.out.println(itr2.next());
        }

    }
    session.close();
    System.out.println("success");

}
}
```

download this hibernate example (developed using MyEclipse IDE)
download this hibernate example (developed using Eclipse IDE)

← prev next →