One to Many mapping in Hibernate by List Example (using xml file)

If the persistent class has list object that contains the entity reference, we need to use one-to-many association to map the list element.

Here, we are using the scenario of Forum where one question has multiple answers.

In such case, there can be many answers for a question and each answer may have its own informations that is why we have used **list** in the persistent class (containing the reference of Answer class) to represent a collection of answers.

Let's see the persistent class that has list objects (containing Answer class objects).

```
package com.javatpoint;

import java.util.List;

public class Question {
private int id;
private String qname;
private List<Answer> answers;
//getters and setters

}
```

The Answer class has its own informations such as id, answername, postedBy etc.

```
package com.javatpoint;

public class Answer {
private int id;
```

```
private String answername;
private String postedBy;
//getters and setters

}
}
```

The Question class has list object that have entity reference (i.e. Answer class object). In such case, we need to use **one-to-many** of list to map this object. Let's see how we can map it.

```
<list name="answers" cascade="all">
        <key column="qid"></key>
        <index column="type"></index>
        <one-to-many class="com.javatpoint.Answer"/>
</list>
```

Full example of One to Many mapping in Hibernate by List

In this example, we are going to see full example of mapping list that contains entity reference.

1) create the Persistent class

This persistent class defines properties of the class including List.

Question.java

```
package com.javatpoint;

import java.util.List;

public class Question {
private int id;
private String qname;
private List<Answer> answers;

//getters and setters
```

```
}
```

Answer.java

```
package com.javatpoint;

public class Answer {
private int id;
private String answername;
private String postedBy;
//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="q501">
        <id name="id">
        <generator class="increment"></generator>
        </id>
        <property name="qname"></property>

        <list name="answers" cascade="all">
        <key column="qid"></key>
        <index column="type"></index>
        <one-to-many class="com.javatpoint.Answer"/>
        </list>
```

```
        </class>

        <class name="com.javatpoint.Answer" table="ans501">
        <id name="id">
        <generator class="increment"></generator>
        </id>
        <property name="answername"></property>
        <property name="postedBy"></property>
        </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();

    Answer ans1=new Answer();
    ans1.setAnswername("java is a programming language");
    ans1.setPostedBy("Ravi Malik");

    Answer ans2=new Answer();
    ans2.setAnswername("java is a platform");
    ans2.setPostedBy("Sudhir Kumar");

    Answer ans3=new Answer();
    ans3.setAnswername("Servlet is an Interface");
    ans3.setPostedBy("Jai Kumar");

    Answer ans4=new Answer();
    ans4.setAnswername("Servlet is an API");
    ans4.setPostedBy("Arun");

    ArrayList<Answer> list1=new ArrayList<Answer>();
    list1.add(ans1);
    list1.add(ans2);
```

```
    ArrayList<Answer> list2=new ArrayList<Answer>();
    list2.add(ans3);
    list2.add(ans4);

    Question question1=new Question();
    question1.setQname("What is Java?");
    question1.setAnswers(list1);

    Question question2=new Question();
    question2.setQname("What is Servlet?");
    question2.setAnswers(list2);

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

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

OUTPUT

ID	QNAME
1	What is Java?
2	What is Servlet?

ID	ANSWERNAME	POSTEDBY	QID	TYPE
1	java is a programming language	Ravi Malik	1	0
2	java is a platform	Sudhir Kumar	1	1
3	Servlet is an Interface	Jai Kumar	2	0
4	Servlet is an API	Arun	2	1

How to fetch the data of List

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. Here, we are direct printing the object of answer class, but we have overridden the **toString()** method in the Answer class returning answername and poster name. So it prints the answer name and postername rather than reference id.

FetchData.java

```
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
        List<Answer> list2=q.getAnswers();
        Iterator<Answer> itr2=list2.iterator();
        while(itr2.hasNext()){
            System.out.println(itr2.next());
        }

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

OUTPUT

```
Question Name: What is Java?
java is a programming language by: Ravi Malik
java is a platform by: Sudhir Kumar
Question Name: What is Servlet?
Servlet is an Interface by: Jai Kumar
Servlet is an API by: Arun
success
```

download this hibernate example (developed using MyEclipse IDE)

download this hibernate example (developed using Eclipse IDE)

← prev

next →