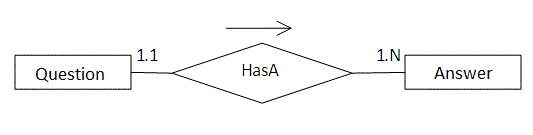
# Hibernate One to Many Example using Annotation

In this section, we will perform one-to-many association to map the list object of persistent class using annotation.

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 information that is why we have used list in the persistent class (containing the reference of Answer class) to represent a collection of answers.

## Example of One to Many mapping using annotation

### Create the Persistent class

This persistent class defines properties of the class including List.

**Question.java**

**package** com.javatpoint;

**import** javax.persistence.\*;

**import** java.util.List;

@Entity

@Table(name="q5991")

**public** **class** Question {

@Id

@GeneratedValue(strategy=GenerationType.TABLE)

**private** **int** id;

**private** String qname;

@OneToMany(cascade = CascadeType.ALL)

@JoinColumn(name="qid")

@OrderColumn(name="type")

**private** List<Answer> answers;

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** String getQname() {

**return** qname;

}

**public** **void** setQname(String qname) {

**this**.qname = qname;

}

**public** List<Answer> getAnswers() {

**return** answers;

}

**public** **void** setAnswers(List<Answer> answers) {

**this**.answers = answers;

}

}

**Answer.java**

**package** com.javatpoint;

**import** javax.persistence.\*;

    @Entity

    @Table(name="answer")

**public** **class** Answer {

    @Id

    @GeneratedValue(strategy=GenerationType.TABLE)

**private** **int** id;

**private** String answername;

**private** String postedBy;

**public** **int** getId() {

**return** id;

    }

**public** **void** setId(**int** id) {

**this**.id = id;

    }

**public** String getAnswername() {

**return** answername;

    }

**public** **void** setAnswername(String answername) {

**this**.answername = answername;

    }

**public** String getPostedBy() {

**return** postedBy;

    }

**public** **void** setPostedBy(String postedBy) {

**this**.postedBy = postedBy;

    }

    }

### Add project information and configuration in pom.xml file.

Open pom.xml file and click source. Now, add the below dependencies between <dependencies> .... </dependencies> tag.

<dependency>

    <groupId>org.hibernate</groupId>

    <artifactId>hibernate-core</artifactId>

    <version>5.3.1.Final</version>

</dependency>

<dependency>

    <groupId>com.oracle</groupId>

    <artifactId>ojdbc14</artifactId>

    <version>10.2.0.4.0</version>

</dependency>

### 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 5.3//EN"

          "http://hibernate.sourceforge.net/hibernate-configuration-5.3.dtd">

<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">jtp</property>

        <property name="connection.driver\_class">oracle.jdbc.driver.OracleDriver</property>

     <mapping **class**="com.javatpoint.Question"/>

    </session-factory>

</hibernate-configuration>

### 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.Session;

**import** org.hibernate.SessionFactory;

**import** org.hibernate.Transaction;

**import** org.hibernate.boot.Metadata;

**import** org.hibernate.boot.MetadataSources;

**import** org.hibernate.boot.registry.StandardServiceRegistry;

**import** org.hibernate.boot.registry.StandardServiceRegistryBuilder;

**public** **class** StoreData {

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

    StandardServiceRegistry ssr=**new** StandardServiceRegistryBuilder().configure("hibernate.cfg.xml").build();

    Metadata meta=**new** MetadataSources(ssr).getMetadataBuilder().build();

    SessionFactory factory=meta.getSessionFactoryBuilder().build();

    Session session=factory.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();

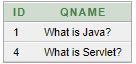
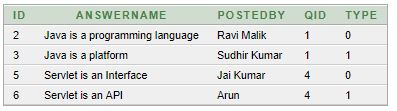
    System.out.println("success");

}

}

#### **Note - Using these annotations in a similar way, we can also perform one-to-many association for set, map and bag objects.**

#### **Output**

## 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.

**package** com.javatpoint;

**import** java.util.\*;

**import** javax.persistence.TypedQuery;

**import** org.hibernate.\*;

**import** org.hibernate.boot.Metadata;

**import** org.hibernate.boot.MetadataSources;

**import** org.hibernate.boot.registry.StandardServiceRegistry;

**import** org.hibernate.boot.registry.StandardServiceRegistryBuilder;

**public** **class** FetchData {

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

    StandardServiceRegistry ssr=**new** StandardServiceRegistryBuilder().configure("hibernate.cfg.xml").build();

    Metadata meta=**new** MetadataSources(ssr).getMetadataBuilder().build();

    SessionFactory factory=meta.getSessionFactoryBuilder().build();

    Session session=factory.openSession();

    TypedQuery query=session.createQuery("from Question");

    List<Question> list=query.getResultList();

    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())

       {

        Answer a=itr2.next();

            System.out.println(a.getAnswername()+":"+a.getPostedBy());

        }

    }

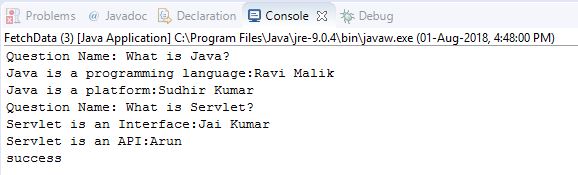
    session.close();

    System.out.println("success");

}

}

#### **Output**



[Download this hibernate example (Developed using Eclipse IDE)](https://www.javatpoint.com/src/hb/onetomany.zip)