**One to Many with Annotation**

Question.java

@Entity

@Table(name = "one")

**public** **class** Question {

@Id

@GeneratedValue(strategy = GenerationType.***IDENTITY***)

**int** qid;

@Column

String qname;

@OneToMany(cascade = CascadeType.***PERSIST***)

List<Answer> ans; Getter and Setter

}

Answer.java

@Entity

@Table(name = "two")

**public** **class** Answer {

@Id

@GeneratedValue(strategy=GenerationType.***IDENTITY***)

**int** aid;

@Column

String answer;

@Column

String postedby; Getter and Setter

}

Client Answer a = **new** Answer();

a.setAnswer("Language");

a.setPostedby("Pam");

Answer a1 = **new** Answer();

a1.setAnswer("Program");

a1.setPostedby("Harry");

List<Answer> li = **new** ArrayList<Answer>();

li.add(a1);

li.add(a);

Question q = **new** Question();

q.setQid(1);

q.setQname("Java");

q.setAns(li);

**s.save(q);**

It will create seperate table by combining both table name and show the linked on that table

**Now Suppose if we want to change the table and column name which its gonna created byDefault so add this below @OneToMany Annotation.**

@**JoinTable**(**name** = "ot",**joinColumns** = **@JoinColumn**(name = "qid"),

**inverseJoinColumns** = **@JoinColumn**(name = "aid"))

**ot is Table name , qid and aid is new column name**

**Many to One with Annotation ( UniDirectional )**

Consider **One to Many with Annotation** on that we have One Question and Multiple Answer for that Question so it will be One to Many from Question side and Many to One from Answer side. But in this it will be from Answer side only not from Question side.

Question.java

@Entity

@Table(name = "one")

**public** **class** Question {

@Id

@GeneratedValue(strategy = GenerationType.***IDENTITY***)

**int** qid;

@Column

String qname;

@OneToMany

List<Answer> ans; Getter and Setter

}

Answer.java

@Entity

@Table(name = "two")

**public** **class** Answer {

@Id

@GeneratedValue(strategy=GenerationType.***IDENTITY***)

**int** aid;

@Column

String answer;

@Column

String postedby;

@ManyToOne

Question q; Getter and Setter

}

Client Answer a = **new** Answer();

a.setAnswer("Language");

a.setPostedby("Pam");

Answer a1 = **new** Answer();

a1.setAnswer("Program");

a1.setPostedby("Harry");

List<Answer> li = **new** ArrayList<Answer>();

li.add(a1);

li.add(a);

Question q = **new** Question();

q.setQid(1);

q.setQname("Java");

q.setAns(li);

a1.setQ(q);

a.setQ(q);

**s.save(a); s.save(a1); s.save(q);**

**Many to One with Annotation ( BiDirectional )**

Consider **One to Many with Annotation** on that we have One Question and Multiple Answer for that Question so it will be One to Many from Question side and Many to One from Answer side. So this time **Question is giving One to Many RelationShip** and **Answer is Giving Many to One RelationShip**

Question.java

@Entity

@Table(name = "one")

**public** **class** Question {

@Id

@GeneratedValue(strategy = GenerationType.***IDENTITY***)

**int** qid;

@Column

String qname;

@OneToMany

List<Answer> ans; Getter and Setter

}

Answer.java

@Entity

@Table(name = "two")

**public** **class** Answer {

@Id

@GeneratedValue(strategy=GenerationType.***IDENTITY***)

**int** aid;

@Column

String answer;

@Column

String postedby; Getter and Setter

}

Client Answer a = **new** Answer();

a.setAnswer("Language");

a.setPostedby("Pam");

Answer a1 = **new** Answer();

a1.setAnswer("Program");

a1.setPostedby("Harry");

List<Answer> li = **new** ArrayList<Answer>();

li.add(a1);

li.add(a);

Question q = **new** Question();

q.setQid(1);

q.setQname("Java");

q.setAns(li);

**s.save(a);**

**s.save(a1);**

**s.save(q);**

**One to One with Annotation ( UniDirectional )**

Now Suppose we have Question and Answer related to that Question So UniDirectional we can get data from Question Side only not from Answer side means Association is from Question side not from Answer

Question.java

@Entity

@Table(name = "one")

**public** **class** Question {

@Id

@GeneratedValue(strategy = GenerationType.***IDENTITY***)

**int** qid;

@Column

String qname;

@OneToOne

@Cascade(value = CascadeType.***ALL***)

@PrimaryKeyJoinColumn

Answer ans; Getter and Setter

}

Answer.java

@Entity

@Table(name = "two")

**public** **class** Answer {

@Id

@GeneratedValue(strategy = GenerationType.**IDENTITY**)

**int** aid;

@Column

String answer;

@Column

String postedby;

}

Client Answer a = **new** Answer();

a.setAnswer("Language");

a.setPostedby("Pam");

Answer a1 = **new** Answer();

a1.setAnswer("Program");

a1.setPostedby("Harry");

Question q = **new** Question();

q.setQid(1);

q.setQname("Java");

q.setAns(a);

q.setAns(a1);

**s.save(q);**

FetchData.java

ServiceRegistry ssr = **new** StandardServiceRegistryBuilder().configure().build();

MetadataSources mds = **new** MetadataSources(ssr);

Metadata md = mds.getMetadataBuilder().build();

SessionFactory sf = md.getSessionFactoryBuilder().build();

Session s = sf.openSession();

Question q = s.load(Question.**class**, 1);

System.***out***.println(q.getQ().getQid());

System.***out***.println(q.getQ().getQname());

**System.*out*.println(q.getAid());**

**System.*out*.println(q.getAnswer());**

**We cannot call Answer directly we need to call Answer through Question class.**

**One to One with Annotation ( BiDirectional )**

Now Suppose we have One Question and One Answer for that Question means One Answer for one Question and one Question for One Answer So if i call Answer Hbm should tell me that Answer is related to which Question and if i call Question Hbm should tell me associate answer of it

Question.java

@Entity

@Table(name = "one")

**public** **class** Question {

@Id

@GeneratedValue(strategy = GenerationType.***IDENTITY***)

**int** qid;

@Column

String qname;

@OneToOne(mappedBy = "q")

@Cascade(value = CascadeType.***ALL***)

Answer ans; Getter and Setter

}

Answer.java

@Entity

@Table(name = "two")

**public** **class** Answer {

@Id

@Column(name="a\_id", unique=**true**, nullable=**false**)

@GeneratedValue(generator="gen")

@GenericGenerator(name = "gen", strategy = "foreign",

parameters = @Parameter(name="property", value="q"))

**int** aid;

@Column

String answer;

@Column

String postedby;

@OneToOne

@PrimaryKeyJoinColumn

Question q; Getter and Setter

}

Client Answer a = **new** Answer();

a.setAnswer("Language");

a.setPostedby("Pam");

Answer a1 = **new** Answer();

a1.setAnswer("Program");

a1.setPostedby("Harry");

List<Answer> li = **new** ArrayList<Answer>();

li.add(a1);

li.add(a);

Question q = **new** Question();

q.setQid(1);

q.setQname("Java");

q.setAns(li);

**s.save(a);**

**s.save(a1);**

**s.save(q);**

FetchData.java

ServiceRegistry ssr = **new** StandardServiceRegistryBuilder().configure().build();

MetadataSources mds = **new** MetadataSources(ssr);

Metadata md = mds.getMetadataBuilder().build();

SessionFactory sf = md.getSessionFactoryBuilder().build();

Session s = sf.openSession();

Answer a = s.load(Answer.**class**, 1);

System.***out***.println(a.getQ().getQid());

System.***out***.println(a.getQ().getQname());

System.***out***.println(a.getAid());

System.***out***.println(a.getAnswer());

**So we are calling Answer so we are getting Question Also and if we call Question then we get Answer also that we call BiDirectional One-To-One**