1to1

Bi-Directional 1-1

class Question

{

**private** **int** q\_id;

**private** String ques;

@OneToOne

Answer answer;

}

class Answer

{

**private** **int** a\_id;

**private** String ans;

@OneToOne

Question question;

}

Default column name

mappingObject\_PKcolname

Rename the column by

@OneToOne

@JoinColumn(name=”aid”)

Answer answer;

UniDirectional 1-1

If you want Only Question class should ONLY be the Owner entity then like as follows ,

|  |  |  |
| --- | --- | --- |
| Question\_no | Question\_description | answer |
| 1 | What is Mobile Used for | 1 |
| 2 | What is Car Used for | 2 |

class Question

{

**private** **int** q\_id;

**private** String ques;

@OneToOne

Answer **answer**;

}

|  |  |
| --- | --- |
| Answer\_no | Answer\_Description |
| 1 | Communication |
| 2 | Transportation |

class Answer

{

**private** **int** a\_id;

**private** String ans;

@OneToOne(mappedBy=”**answer**”)

Question question;

}

NOTE : //where u add "mappedby" in that table FK column will not be created.

1-M

@Entity

**public** **class** Department {

@Id

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

**private** **int** dno;

**private** String dname;

@OneToMany(mappedBy = "dept", cascade = CascadeType.***ALL***)

**private** List<Employee> employees;

@Entity

**public** **class** Employee {

@Id

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

**private** **int** eno;

**private** String ename;

**private** **double** sal;

@ManyToOne

**private** Department dept;

M:M Actor Movie

Owner Entity is Actor

**class Actor**

{

...

@ManyToMany()

@JoinTable(

name = "bollywood",

joinColumns = @JoinColumn(name = "ano"),

inverseJoinColumns = @JoinColumn(name = "mno")

)

List <Movies> movies;

}

**class Movie** //routine POJO class

{

}

cascade = CascadeType.ALL,

fetch = FetchType.EAGER

LifeCycle states

Session Object database table

Student s1=new Student(101,’Alice’,800); //Transient State

When object has been initialized (either by constructor / setter)

Session.save(s1) // Persistant State

If you make any changes to the object and do a commit , it will reflect in the session & db

Sop(Session.contains(s1)) //will return true

Session.Clear() or Session.close() //detached State

Object is not in the session but the value is in the db

Session.delete() //removed State

It wil be in the session

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Transient | Persistent | Detached | Removed |
| Session | N | Y | N | Y |
| Database | N | Y | Y | N |