**Relation ManyToMany**

Une relation plusieurs à plusieurs se produit lorsque plusieurs enregistrements dans une table sont associés à plusieurs enregistrements dans une autre table. Par exemple, une relation plusieurs-à-plusieurs existe entre les clients et les produits : les clients peuvent acheter divers produits et les produits peuvent être achetés par de nombreux clients.

Voici les étapes d’implémentation de cette relation entre l’entité Employe et l’entité Projet :

**Etape 1 :** Création de la base de données et les tables :

**CREATE DATABASE IF NOT EXISTS DBPROJECTS;**

**USE DBPROJECTS;**

**CREATE TABLE IF NOT EXISTS EMPLOYE**

**(**

**ID INT AUTO\_INCREMENT PRIMARY KEY,**

**NOM VARCHAR(30),**

**PRENOM VARCHAR(30)**

**);**

**CREATE TABLE IF NOT EXISTS PROJET**

**(**

**ID INT AUTO\_INCREMENT PRIMARY KEY,**

**NOMPROJET VARCHAR(30)**

**);**

**CREATE TABLE IF NOT EXISTS EMPLOYEE\_PROJET**

**(**

**ID\_PROJET INT,**

**ID\_EMPLOYE INT,**

**PRIMARY KEY(ID\_PROJET, ID\_EMPLOYE),**

**CONSTRAINT fk1 FOREIGN KEY (ID\_PROJET) REFERENCES PROJET(ID),**

**CONSTRAINT fk2 FOREIGN KEY (ID\_EMPLOYE) REFERENCES EMPLOYE(ID)**

**);**

**Etape 2 :** Les classes de mapping

1. La classe **Employe**

@Entity

@Table(name="employe")

**public** **class** Employe {

@Id

@GeneratedValue

@Column(name="id")

**private** **int** id;

@Column(name="nom")

**private** String nom;

@Column(name="prenom")

**private** String prenom;

@ManyToMany

@JoinTable(name="employee\_projet",

joinColumns = @JoinColumn(name="id\_employe"), inverseJoinColumns=@JoinColumn(name="id\_projet"))

**private** Set<Projet> projets=**new** HashSet<>();

**// Les contructeurs**

**// getters et setters**

**}**

1. La classe Projet :

@Entity

@Table(name="projet")

**public** **class** Projet {

@Id

@GeneratedValue

@Column(name="id")

**private** **int** id;

@Column(name="nomProjet")

**private** String nomProjet;

@ManyToMany(mappedBy="projets")

**private** Set<Employe> employes=**new** HashSet<>();

// Ajouter les constructeurs

// Ajouter les getters et setters

}

**Etape 3 : Ajouter des données dans la table**

Session s = HibernateUtils.*getSessionFactory*().getCurrentSession();

Transaction tr = s.beginTransaction();

Projet p1 = **new** Projet(1,"AXA Assurance");

Projet p2 = **new** Projet(2,"Carrefour");

Projet p3 = **new** Projet(3,"Total");

Employe e1 = **new** Employe(1,"Hamid","Riad");

e1.getProjets().add(p1);

e1.getProjets().add(p3);

s.save(p1);

s.save(p2);

s.save(p3);

s.save(e1);

tr.commit();

**Etape 4 :** Récupérer un employé par son identifier en utilisant la méthode get qui permet de retourner un employé en se basant sur son ID.

Session s = HibernateUtils.*getSessionFactory*().getCurrentSession();

Transaction tr = s.beginTransaction();

Employe e = s.get(Employe.**class**, 8);

System.***out***.println(e.getNom());

tr.commit();