Mathissart Nicolas

HIBERNATE

Création des classes métier

1) Comment est déclarée la table assurant la persistance d'un objet ?

Pour déclarée une table assurant la persistance d'un objet, il suffit d'ajouter la ligne de code ci-dessous qui permet de définir le nom de la table.
Il faut mettre la ligne de code au dessus de la classe :

|h Déclaration d'une classe de mapping

```
@Table(name="Categorie") // C'est cette ligne qui permet de définir la table
public class Categorie {
    @Id
    @Column(name="id")
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    private int id;
```

2) Comment est déclaré le mapping entre un membre de la classe et un champ de la table relationnelle ?

Afin de déclaré un mapping entre une membre de la classe et un champ de la table relationnelle,
il suffit d'ajouter une ligne de code qui permet de définir le nom de notre champ et de définir son type.
voici la ligne de code qu'il vous faut :

|hChamp

```
    @Column(name="libelle") //On indique que notre champ s'appellera Libelle
    private String libelle; // On indique que notre champ sera un String
```

@Column(name=“monChamp”)

private monType monChamp

3) Comment est déclarée la clé primaire de la table ?

Pour déclarer une clé primaire sur Hibernate, il suffit de r'ajouter la balise “@Id” au-dessus de la colonne concerné. Par exemple :

|h Champ

```
    @Id // C'est cette balise qui permet de définir la clé primaire.
    @Column(name="id")
    private int id;
```

```
```

4) Quelles sont les possibilités de déclaration des clés primaires ?

Sur hibernate, il y à plusieurs méthodes pour déclarer les clés primaires.

- Afin d'auto incrémenter à partir de la base donnée une clé primaire, il suffit d'utiliser cette ligne de code ;

|h Auto

```
@GeneratedValue(strategy=generationType.AUTO)
```

-

|h Identity

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

- Afin d'auto incrémenter à partir de Hibernate une clé primaire, il suffit d'utiliser cette ligne de code ;

|h Sequence

```
@GeneratedValue(strategy=GenerationType.SEQUENCE)
```

-

|h Table

```
@GeneratedValue(strategy=GenerationType.TABLE)
```

5) tableau montrant la correspondance de type (entier, chaine, etc.) entre les propriétés d'une classe et les champs d'une table

Voici un tableau recapitulant, la correspondance de type entre les propriétés d'une classe et les champs d'une table :

Type Java	Type Sql
String	Varchar
int	Integer
float	Float

6) lien bidirectionnel entre deux classes

7) À quoi correspond la méthode persist() ?

La méthode persist sur Hibernate permet d'ajouter un élément dans la base de base de données :

|h Méthode Persist

```
Categorie aCategorie=new Categorie("Presse"); // On instancie un produit, donc l'objet que l'on insérera dans la base de donnée
session.persist(aCategorie); // Une fois l'objet instancier, on l'insère à la BDD
```

8) À quoi correspond la méthode commit() ?

La méthode commit permet de mettre à jour la base de donnée avec les objets persistants. Autrement dit, la méthode commit permet d'insérer dans la base de donnée les objets qui sont valide.

Code de la méthode commit :

|h Méthode Commit

```
Transaction trans =  session.beginTransaction();


trans.commit();
```

8) Comment ont été traduits les liens objet entre le membre categorie et produits entre ces classes dans les tables de la base ?

Lors de l'insertion d'un produit et d'une catégorie dans la base de donnée, nous avions indiquer au produit qu'il fera partit de la catégorie “aCatégorie” que nous avons instancier au dessus (Catégorie : Presse) .

|h Produit

```
Produit aProduit=new Produit("Programmez!", 3.0f, aCategorie) //On ne passe
pas l'id de la catégorie mais directement la catégorie concerné;
```

Lors de l'insertion dans la base de donnée, l'iDCategorie (clé étrangère) de notre produit à donc prit comme valeur la clé primaire de la catégorie “aCatégorie”.

9) Quelles requêtes SQL ont été créées par Hibernate pour réaliser la persistance ?

Afin de réaliser la persistance, Hibernate à effectuer de requetes sql INSERT ;

Ajout de la catégorie :

|h INSERT Catégorie

```
    INSERT
    INTO
        Categorie
        (libelle)
    VALUES
        (?)
```

Ajout du produit :

|h INSERT Produit

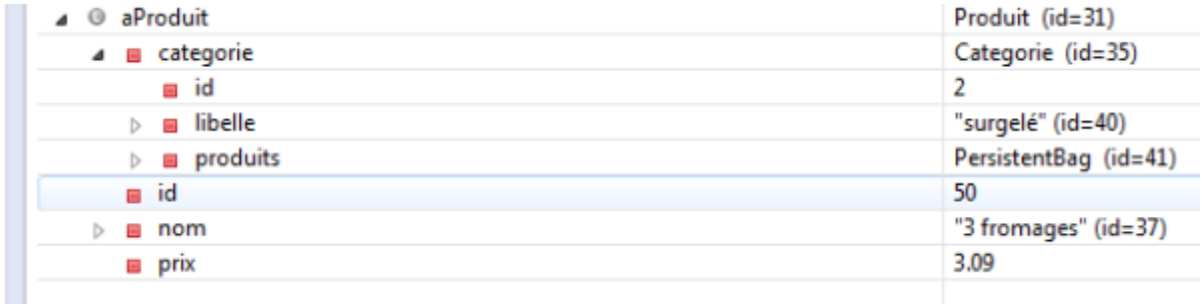
```
  INSERT
    INTO
        Produit
        (idCategorie, nom, prix, id)
    VALUES
        (?, ?, ?, ?)
```

10) Pourquoi comportent t-elles des points d'interrogation ?

Les requêtes comportent des points d’interrogation car hibernate a mit en place des requêtes paramétrables. Autrement, dit on peut utiliser la requête plusieurs-fois avec les paramètres concernés. Les paramètres sont dont affichés avec des “?”.

11) Qu'est ce que charge exactement Hibernate lors du chargement d'un Objet ?

Lors d'un chargement d'objet, Hibernate charge automatiquement tous les objets en relation avec l'objet concerné.
Par exemple si l'on charge un produit, on charge aussi sa catégorie :



On peut voir que la catégorie est bien chargé

12) Comment sont chargés les instances liées à un objet chargé pour les liens oneToMany et manyToOne ?

Pour charger les instances liées à un objet chargé pour les liens oneToMany, Hibernate charge l'objet concerné ainsi que les objets dont il est en relation. Par exemple, lorsque l'on charge un produit, il charge aussi sa catégorie :

|h jj

```
    SELECT
        produit0_.id AS id1_1_,
        produit0_.idCategorie AS idCatego4_1_1_,
        produit0_.nom AS nom1_1_,
        produit0_.prix AS prix1_1_,
        categorie1_.id AS id0_0_,
        categorie1_.libelle AS libelle0_0_
    FROM
        Produit produit0_
    LEFT OUTER JOIN
        Categorie categorie1_
            ON produit0_.idCategorie=categorie1_.id
    WHERE
        produit0_.id=?
```

Pour charger les instances liées à un objet chargé pour les liens ManyToOne,Hibernate charge seulement l'objet concerné. Par exemple, lorsque l'on charge une catégorie, il charge seulement la catégorie et il prépare le chargement des produits. Hibernate:

```
select
    categorie0_.id as id0_0_,
    categorie0_.libelle as libelle0_0_
from
    Categorie categorie0_
where
    categorie0_.id=?
```

Hibernate:

```
select
    produits0_.idCategorie as idCatego4_1_,
    produits0_.id as id1_,
    produits0_.id as id1_0_,
    produits0_.idCategorie as idCatego4_1_0_,
    produits0_.nom as nom1_0_,
```

```
    produits0_.prix as prix1_0_
from
    Produit produits0_
where
    produits0_.idCategorie=?
```

</code>

En quoi consiste le chargement paresseux d'Hibernate et la qualification lazy ?

Le chargement paresseux consiste donc à charger objet par objet.
Par exemple, avec l'exemple de la catégorie, il charge seulement la catégorie et non les produits, si on ne lui demande pas.