

# La FAQ Hibernate

Date de publication : 8/11/2006

Dernière mise à jour : 17/05/2009

Cette FAQ a été réalisée à partir des questions fréquemment posées sur les **forums java** de **www.developpez.com** et de l'expérience personnelle des auteurs. Je tiens à souligner que cette FAQ ne garantit en aucun cas que les informations qu'elle propose sont correctes. Les auteurs font le maximum, mais l'erreur est humaine.

Cette FAQ ne prétend pas non plus être complète. Si vous trouvez une erreur, ou que vous souhaitez nous aider en devenant rédacteur, lisez les **modalités de participation**.

Bonne lecture à tous,

**L'équipe Java**

## **Ont contribué à cette FAQ :**

Baptiste Wicht - minosis - enok37 -

1. Informations générales (5) .....	4
2. Généralités (5) .....	6
3. Les fichiers de config (7) .....	8
4. Les fichiers de mapping (7) .....	11
4.1. La génération (3) .....	13
5. Les objets (4) .....	15
6. Utilisation (23) .....	16
6.1. Le langage HQL (6) .....	19
6.2. Les Criteria (9) .....	21
7. Divers (10) .....	24
7.1. Exceptions (6) .....	25

[Sommaire > Informations générales](#)**Comment bien utiliser cette FAQ ?****Auteurs :** [Baptiste Wicht](#) ,

**Le but :** Cette FAQ a été conçue pour être la plus simple possible d'utilisation. Elle tente d'apporter des réponses simples et complètes aux questions auxquelles sont confrontés tous les débutants (et les autres).

**L'organisation :** Les questions sont organisées par thème, les thèmes pouvant eux-mêmes contenir des sous-thèmes. Lorsqu'une question porte sur plusieurs thèmes, celle-ci est insérée dans chacun des thèmes rendant la recherche plus facile.

**Les réponses :** Les réponses contiennent des explications et des codes sources. Certaines sont complétées de fichier à télécharger contenant un programme de démonstration. Ces programmes sont volontairement très simples afin qu'il soit aisé de localiser le code intéressant. Les réponses peuvent également être complétées de liens vers d'autres réponses, vers la documentation en ligne de Sun ou vers un autre site en rapport.

**Nouveautés et mises à jour :** Lors de l'ajout ou de la modification d'une question/réponse, un indicateur est placé à côté du titre de la question. Cet indicateur reste visible pour une durée de 15 jours afin de vous permettre de voir rapidement les modifications apportées.

J'espère que cette FAQ pourra répondre à vos questions. N'hésitez pas à nous faire part de tous commentaires/remarques/critiques.

**lien :** [Comment participer à cette FAQ ?](#)**Comment participer à cette FAQ ?****Auteurs :** [Baptiste Wicht](#) ,

Cette faq est ouverte à toute collaboration. Pour éviter la multiplication des versions, il serait préférable que toutes collaborations soient transmises aux administrateurs de la FAQ.

**Plusieurs compétences sont actuellement recherchées pour améliorer cette FAQ :**

**Rédacteur :** Bien évidemment, toute nouvelle question/réponse est la bienvenue.

**Correcteur :** Malgré nos efforts, des fautes d'orthographe ou de grammaire peuvent subsister. Merci de contacter les administrateurs si vous en débusquez une... Idem pour les liens erronés.

**lien :** [Quels sont les droits de reproduction de cette FAQ ?](#)**Quels sont les droits de reproduction de cette FAQ ?****Auteurs :** [Baptiste Wicht](#) ,

Les codes sources présentés sur cette page sont libres de droits et vous pouvez les utiliser à votre convenance. Pour le reste, ce document constitue une oeuvre intellectuelle protégée par les droits d'auteurs.

**Pour toutes les contributions de cette page : Copyright © 2004 Developpez LLC : Tous droits réservés Developpez LLC. Aucune reproduction ne peut en être faite sans l'autorisation expresse de Developpez LLC. Sinon vous encourez selon la loi jusqu'à 3 ans de prison et jusqu'à 300 000 E de dommages et intérêts. Cette page est déposée à la SACD.**

### Où trouver d'autres sources d'information ?

**Auteurs : Baptiste Wicht ,**

**Pour plus d'infos sur Hibernate, la plus grande source d'informations est la documentation de référence en français d'Hibernate, que vous pouvez consulter [ici](#).**

### Remerciements

**Auteurs : Baptiste Wicht ,**

**Un grand merci à tous ceux qui ont pris de leur temps pour la réalisation de cette FAQ.**

**Remerciements également aux personnes qui ont relu les textes pour supprimer un maximum de fautes de français.**

**Aux visiteurs : Remerciements enfin à tous ceux qui ont consulté cette FAQ, et qui, par leurs remarques, nous ont aidé à la perfectionner.**

[Sommaire > Généralités](#)**Qu'est ce qu'Hibernate****Auteurs :** [Baptiste Wicht](#) ,

Hibernate est un framework de mapping objet/relationnel. Ce framework est de plus en plus souvent utilisé.

Concrètement, cela veut dire qu'Hibernate nous permet de manipuler les données d'une base de données relationnelle sous forme d'objet.

Pour faire cela, Hibernate utilise des fichiers pour relier la base aux objets.

**lien :** [Que nous apporte Hibernate ?](#)**Que nous apporte Hibernate ?****Auteurs :** [Baptiste Wicht](#) ,

Le fait de manipuler directement les données d'une base sous forme d'objet est beaucoup plus pratique, cela nous permet de nous défaire de toute la couche SQL.

De plus, cela permet de définir clairement la limite entre la persistance et la couche métier, ce qui se révèle très utile dans le cas d'une application trois-tiers.

**Où puis-je télécharger Hibernate ?****Auteurs :** [Baptiste Wicht](#) ,

Pour télécharger la dernière version d'Hibernate (3.2.0), il vous suffit d'aller le chercher sur [sourceforge](#) [ici](#).

**lien :** [De quoi a besoin Hibernate pour fonctionner ?](#)**Où trouver plus de documentations sur Hibernate ?****Auteurs :** [Baptiste Wicht](#) ,

Voici certains liens qui pourraient vous être bien utile dans votre apprentissage d'Hibernate :

- [Débuter avec Hibernate sous Eclipse](#) : Ce tutorial vous expliquera de manière très intuitive, comment réaliser un mapping O/R grâce au plugin Hibernate Synchronizer.
- [JSF et Hibernate](#) : Ce document est destiné à expliquer l'intégration de JSF et Hibernate dans JOnAS 4.0.0 / Tomcat 5.0.21.
- [Le site officiel d'Hibernate](#)
- [La FAQ officielle](#)
- [La javadoc d'Hibernate 3.2.0](#)

**De quoi a besoin Hibernate pour fonctionner ?****Auteurs :** [Baptiste Wicht](#) ,

Hibernate a aussi besoin d'autres api pour fonctionner. En premier lieu, il ne faut pas oublier le driver JDBC de votre base de données.

**En plus de votre driver, il vous faut aussi les librairies suivantes :**

- **Jakarta Commons Logging**
- **Jakarta Commons Collections**
- **Log4j**
- **dom4j**
- **Jta**
- **asm**
- **cglib.jar**

**Il est à noter aussi que les dernières versions d'Hibernate fournissent déjà toutes ces librairies. Pour les utiliser, il vous suffira des les ajouter au classPath.**

Sommaire > Les fichiers de config

## Qu'est ce que le fichier Hibernate.properties ?

Auteurs : [Baptiste Wicht](#) ,

Le fichier `hibernate.properties` sert à configurer l'accès à la base de données. On va donc y configurer les différentes infos nécessaires à une connexion JDBC. On peut aussi configurer le pool de connexion via ce fichier.

Les infos indispensables sont les suivantes :

- `hibernate.connection.driver_class` = Le chemin vers le driver JDBC (par exemple `org.postgresql.Driver`)
- `hibernate.connection.url` = Le chemin d'accès à la base (par exemple `jdbc:postgresql://localhost/mydatabase`)
- `hibernate.connection.username` = Le nom d'utilisateur de la connexion
- `hibernate.connection.password` = Le mot de passe de la connexion
- `hibernate.dialect` = Le dialecte de votre base de données (par exemple `net.sf.hibernate.dialect.PostgreSQLDialect`), voir le lien pour plus d'infos

lien : [Qu'est ce que le dialect SQL ?](#)

## Qu'est ce que le dialect SQL ?

Auteurs : [Baptiste Wicht](#) ,

La propriété `hibernate.dialect` du fichier `hibernate.properties` vous sert à configurer le dialecte SQL de votre base de données. Ce dialecte va servir à Hibernate pour optimiser certaines parties de l'exécution en utilisant les propriétés spécifiques à la base. Cela se révèle très utile pour la génération de clé primaire et la gestion de concurrence tel que le `pessimist locking`.

Voici une liste des dialectes utilisables :

Base de données	Dialecte
DB2	net.sf.hibernate.dialect.DB2Dialect
DB2 AS/400	net.sf.hibernate.dialect.DB2400Dialect
DB2 OS390	net.sf.hibernate.dialect.DB2390Dialect
PostgreSQL	net.sf.hibernate.dialect.PostgreSQLDialect
MySQL	net.sf.hibernate.dialect.MySQLDialect
Oracle (toute version)	net.sf.hibernate.dialect.OracleDialect
Oracle 9/10g	net.sf.hibernate.dialect.Oracle9Dialect
Sybase	net.sf.hibernate.dialect.SybaseDialect
Sybase Anywhere	net.sf.hibernate.dialect.SybaseAnywhereDialect
MS SQL Server	net.sf.hibernate.dialect.SQLServerDialect
SAP DB	net.sf.hibernate.dialect.SAPDBDialect
Informix	net.sf.hibernate.dialect.InformixDialect
HypersonicSQL	net.sf.hibernate.dialect.HSQLDialect
Ingres	net.sf.hibernate.dialect.IngresDialect
Progress	net.sf.hibernate.dialect.ProgressDialect
Mckoi SQL	net.sf.hibernate.dialect.MckoiDialect
Interbase	net.sf.hibernate.dialect.InterbaseDialect
Pointbase	net.sf.hibernate.dialect.PointbaseDialect
FrontBase	net.sf.hibernate.dialect.FrontbaseDialect
Firebird	net.sf.hibernate.dialect.FirebirdDialect

### Comment configurer un pool de connexion ?

Auteurs : **Baptiste Wicht** ,

Hibernate possède son propre algorithme de pool de connexion, mais il reste très rudimentaire et n'est pas conseillé pour un programme en production.

Pour utiliser le pool de connexion d'Hibernate, il vous suffit de rajouter cette ligne dans le fichier `Hibernate.properties` :

```
hibernate.connection.pool_size=nombre maximum de connexion simultanées
```

lien : [Comment configurer un pool de connexion C3P0?](#)

### Comment configurer un pool de connexion C3P0?

Auteurs : **Baptiste Wicht** ,

Hibernate ayant un algorithme très basique pour les pools de connexion, on peut aussi utiliser d'autres pools de connexion. Par exemple, C3P0.

C3P0 est un pool de connexion JDBC open-source distribué avec Hibernate. C3P0 possède un algorithme plus évolué que celui d'Hibernate, vous pourrez donc l'utiliser directement dans un programme en production.

Pour l'utiliser, il vous faudra ajouter quelques lignes dans votre fichier `hibernate.properties` :

```
hibernate.c3p0.min_size=Taille minimale du pool
hibernate.c3p0.max_size=Taille maximale du pool
hibernate.c3p0.timeout=Temps pendant lequel une connexion peut être utilisée avant d'être libérée. 0
signifie qu'une connexion n'expire pas.
```

```
hibernate.c3p0.max_statements=La taille du cache de statements de C3P0. 0 signifie qu'on désactive le cache.
```

**Vous pouvez aussi configurer C3P0 dans le fichier hibernate.cfg.xml. La seule différence réside dans le fait qu'une propriété se présente ainsi :**

```
<property name="c3p0.max_size">100</property>
```

## Comment utiliser une datasource JNDI ?

**Auteurs :** Baptiste Wicht ,

**Dans le cadre d'une utilisation avec un serveur d'application, il faudrait toujours configurer Hibernate pour qu'il aille chercher ses DataSource dans le serveur enregistré dans le JNDI.**

**Pour faire cela, il vous suffit de configurer au moins une des propriétés suivantes :**

```
hibernate.connection.datasource=Le nom JNDI de la datasource (obligatoire)
hibernate.jndi.url=L'url du fournisseur JNDI (facultatif)
hibernate.connection.username=Le nom d'utilisateur de la base de données (facultatif)
hibernate.connection.password=Le mot de passe de l'utilisateur de la base de données (facultatif)
hibernate.jndi.class=La classe de l'InitialContextFactory du JNDI (facultatif)
```

## Qu'est ce que le fichier Hibernate.cfg.xml

**Auteurs :** Baptiste Wicht ,

**Le fichier Hibernate.cfg.xml a presque la même utilité que le fichier Hibernate.properties. Soit on configure la connexion JDBC dans le fichier properties soit on le configure ici. Les deux cas sont équivalents.**

**La seule chose ou il change, est que ce fichier sert aussi à mapper les différents fichiers de mapping de l'application.**

## Comment ajouter un nouveau fichier de mapping dans Hibernate ?

**Auteurs :** Baptiste Wicht ,

**A chaque fois que vous créez un nouveau fichier de mapping, il faudra que Hibernate sache ou il se trouve pour qu'il mappe aussi ces fichiers.**

**Pour cela, il va falloir dire dans le fichier hibernate.cfg.xml quels sont les fichiers de mapping à charger dans l'application. Pour cela, on utilise la propriété suivante :**

```
<mapping resource="Dossier/Fichier.hbm.xml"/>
```

**Vous ajouterez cette ligne autant de fois que vous aurez de fichier de mapping.**

[Sommaire](#) > Les fichiers de mapping

## Qu'est ce qu'un fichier de mapping

Auteurs : [Baptiste Wicht](#) ,

Un fichier de mapping est tout simplement un fichier XML qui va permettre à Hibernate de faire le lien entre vos objets Java et la base de données.

Voici sa structure la plus basique :

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC
    "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
    "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">

<hibernate-mapping>

    <class name="NomDeLaClasse" table="NOMDELATABLE">

    </class>

</hibernate-mapping>
```

La seule chose que nous lui disons pour le moment est quelle classe (un simple javaBean) correspond à quelle table.

Le nom de ces fichiers doit se terminer par `.hbm.xml` pour qu'Hibernate sache que c'est un fichier de mapping. De plus, par convention, on écrit un fichier de mapping par classe de persistance.

[lien](#) : Comment doivent être faites mes classes de persistance ?

## Comment ajouter une propriété à un objet ?

Auteurs : [Baptiste Wicht](#) ,

Pour ajouter une propriété à un de vos objets, il vous faut modifier le fichier de mapping de cet objet.

L'ajout d'une propriété est très simple, il vous suffit d'ajouter une ligne comme celle-ci :

```
<property name="nomDeLaVariable" type="typeDeLaValeur" column="ColonneOuTrouverCetteValeur" />
```

Comme vous le voyez, c'est très simple, il vous juste associer le nom de la variable du Javabeau au nom de la colonne de la base de données.

## Comment doivent être faites mes classes de persistance ?

Auteurs : [Baptiste Wicht](#) ,

Comme il est conseillé de bien séparer la couche de persistance de la couche métier, vos classes de persistance, seront de simples classes avec des getters et des setters, des Javabeans. Il faudrait éviter d'implémenter des opérations métiers dans ces classes, mais cette restriction est très relative et plusieurs opinions existent sur le sujet. L'important étant que vous sépariez correctement les couches de votre application.

Voici un simple exemple d'une de ces classes :

```
public class OdAuteur {
    private String nom = null;
    private String prenom = null;

    public String getNom(){
        return nom;
    }

    public void setNom(String nom){
        this.nom = nom;
    }

    public String getPrenom(){
        return prenom;
    }

    public void setPrenom(String prenom){
        this.prenom = prenom;
    }
}
```

**Il faut faire très attention à la casse des variables et de leurs getters/setters. Un nom de variable commence par une minuscule et le nom de son getter est formé de get suivi du nom de la variable avec la première lettre en majuscule. Si vous ne respectez cette casse, Hibernate ne va pas trouver vos propriétés et ne pourra donc pas fonctionner correctement.**

**lien : Pourquoi est-ce que j'obtiens une erreur lorsque j'utilise un type primitif dans mon javabean ?**

### Peut-on mapper les classes qui n'ont pas de clé primaire ?

**Auteurs : enok37 ,**

**La réponse est oui. Effectivement, Hibernate permet de mapper les classes qui n'ont pas de clé primaire.**

**Lorsqu'une table n'a pas de clé primaire, le tag <generator ..> à l'intérieur du tag <id ..> </id> n'est pas indispensable car par défaut, il est à "assigned" . Mais le tag <id ..> </id> est obligatoire car Hibernate se sert de ce tag pour rechercher (loader vos objets). Dans vos fichiers de mapping, vous devez avoir ceci :**

```
<id
    column="nomColumn"
    name="unNom"
    type="LeType"
>
    <generator class="assigned" />
</id>
<property ... />
```

**Ou Ceci :**

```
<id
    column="nomColumn"
    name="unNom"
    type="LeType"
>
</id>
<property ... />
```

**lien : Plus d'informations**

Sommaire > Les fichiers de mapping > La génération

## Comment générer les fichiers de mapping depuis les classes ?

Auteurs : [Baptiste Wicht](#) ,

**Vous pouvez générer directement les fichiers de mapping depuis vos classes Java avec un outil intégré dans Hibernate (MapGenerator). L'emploi de cet utilitaire est très simple, il vous suffit de lancer la ligne de commande suivante :**

```
java -cp classpath_contenant_hibernate_et_vos_classes net.sf.hibernate.tool.class2hbm.MapGenerator  
options et noms_des_classes
```

**La principale option est la configuration du fichier de sortie :**  
**--output=Fichier\_de\_sortie**

**lien : [Plus d'infos sur les outils Hibernate](#)**

## Comment générer les classes Java depuis les fichiers de mapping ?

Auteurs : [Baptiste Wicht](#) ,

**Vous pouvez générer directement le code des classes Java depuis vos fichiers de mapping avec un outil intégré dans Hibernate (CodeGenerator). L'emploi en est très simple, il vous suffit de lancer la ligne de commande suivante :**

```
java -cp classpath_hibernate net.sf.hibernate.tool.hbm2java.CodeGenerator options  
fichiers_de_mapping
```

**Les options possibles sont :**  
**--output=repertoire : Répertoire des fichiers générés**  
**--config=fichier\_config : fichier de configuration de la génération**

**Ce fichier permet entre autres, de configurer le package des classe générées et de modifier la portée des setters. Pour plus d'infos sur ce fichier de configuration (il est optionnel), je vous invite à lire ce qu'en dit la [documentation officielle](#)**

**lien : [Plus d'infos sur les outils Hibernate](#)**

## Comment générer la structure de la base de données à partir des fichiers de mapping ?

Auteurs : [Baptiste Wicht](#) ,

**Pour commencer, il vous faut paramétrer vos colonnes. C'est-à-dire que dans vos fichiers de mapping, pour chaque propriété, vous pouvez encore ajouter des options : length, precision, not-null et unique. Ainsi, lors de la création, vous aurez précisément le bon schéma qui sera créé. Vous pouvez aussi ajouter des contraintes sur la balise column : default, check (pour ajouter une condition d'insertion) et sql-type pour mettre un type précis et non pas le type par défaut d'Hibernate.**

**Il est impératif de renseigner la propriété de dialecte SQL dans le fichier de propriété sans ca, la génération ne sera pas possible.**

**Pour lancer la génération, vous avez 2 choix, soit par programmation :**

```
SchemaExport export = new SchemaExport(config);  
export.create(true, true);
```

**Soit en ligne de commande :**

```
java -cp classpath_hibernate net.sf.hibernate.tool.hbm2ddl.SchemaExport options_de_lancement  
fichiers_de_mapping
```

**Vous pouvez aussi utiliser SchemaUpdate pour mettre à jour le schéma et SchemaValidator pour vérifier qu'il est correct.**

**lien : [Plus d'infos sur cet outil](#)**

**Sommaire > Les objets****Qu'est ce qu'un objet détaché ?****Auteurs : Baptiste Wicht ,**

Un objet détaché est un objet qui n'est plus lié à la session Hibernate. On ne peut donc plus effectuer de save ou d'update dessus. Cela arrive quand la session à laquelle il était lié a été fermée.

**Qu'est ce qu'un objet persistant ?****Auteurs : Baptiste Wicht ,**

Un objet persistant est un objet qui a une existence dans la base de données et qui est référencé par un id. Toutes les modifications effectuées sur cet objet seront synchronisées avec la base lors de la fin de la transaction. Vous pouvez aussi employer update() pour valider ces changements.

**Qu'est ce qu'un objet transient (éphémère) ?****Auteurs : Baptiste Wicht ,**

Un objet transient est un objet qui n'existe pas en base et qui a été créé via l'opérateur new. Il n'existe donc nulle part d'autre que dans la JVM. Il n'a donc aucune référence dans la base de données et n'est référencé par aucun id. Il faut utiliser la méthode save() pour le faire passer à un état persistant.

**Comment attacher un objet détaché à une session ?****Auteurs : Baptiste Wicht ,**

Vous pouvez soit employer update si vous êtes sûrs que cette session ne contient pas cet objet ou alors merge() pour mettre à jour l'objet déjà existant dans la session avec vos modifications. Vous pouvez aussi utiliser saveOrUpdate qui va directement effectuer ce qu'il faut en détectant l'état de l'objet que vous lui donnez. Ainsi plus besoin de faire attention à l'état de votre objet.

[Sommaire > Utilisation](#)**Comment ouvrir une session ?****Auteurs : Baptiste Wicht ,**

L'obtention d'une session Hibernate est très simple. Il faut tout d'abord créer une session factory :

```
SessionFactory factory = new Configuration().configure().buildSessionFactory();
```

`new Configuration().configure()` charge tout simplement les paramètres de configuration d'Hibernate dans la classe. Ensuite, à partir de cette classe, on crée le factory.

Pour obtenir une session à partir de ce factory, il suffit d'appeler `openSession` :

```
Session session = factory.openSession();
```

Vous disposez maintenant d'une session sur laquelle vous allez pouvoir travailler.

**Comment sauvegarder une nouvelle instance d'un objet ?****Auteurs : Baptiste Wicht ,**

Pour sauvegarder une instance d'un objet que vous créez directement dans le code, il vous faut ensuite appeler la méthode `save` de la session pour que cet objet soit créé en base.

```
Livre livre = new Livre();  
livre.setTitle("Livre");  
livre.setAuteur("Auteur");  
  
session.save(livre);
```

Il ne faut pas oublier de committer les changements de la transaction via la méthode `commit()` :

```
session.getTransaction().commit();
```

**lien : Qu'est ce qu'une transaction ?****Comment rafraîchir une instance d'un objet ?****Auteurs : Baptiste Wicht ,**

Dans certains cas (un trigger qui modifie des données ou après l'exécution d'une requête directement sur le serveur (native)), les données qui sont dans votre objet ne sont plus à jour; il vous faut donc les rafraîchir.

Pour cela, il vous faut employer la méthode `refresh` sur votre objet :

```
session.refresh(votreObjet);
```

### Comment exécuter du SQL natif ?

Auteurs : [Baptiste Wicht](#) ,

En plus du HQL, vous pouvez aussi utiliser du SQL natif. Pour cela, il vous faut employer la méthode `createSqlQuery` de votre session.

```
Query query = session.createSQLQuery("Requête en SQL natif");
```

Si c'est une requête de mise à jour, vous pouvez utiliser `executeUpdate` pour l'exécuter ou sinon, utiliser `list()` pour récupérer les résultats

lien : [Pour plus d'infos](#)

### Comment insérer un fichier dans la base de données ?

Auteurs : [Baptiste Wicht](#) ,

Vous devez pour cela, employer un objet de type `Blob`. Pour créer un blob dans votre application, il vous suffit de faire :

```
Blob blob = Hibernate.createBlob(inputStreamVersLeFichier);
```

Ensuite, vous pouvez le mettre comme tout autre objet dans une classe persistante et la sauver.

Il est évident que le champ dans le fichier de mapping doit être de type `blob`.

### Pourquoi rien n'est modifié sur la base bien que j'utilise `save` ?

Auteurs : [Baptiste Wicht](#) ,

La méthode `save` est là pour dire à Hibernate que cet objet va être sauvé, mais il faut encapsuler ces sauvegardes dans une transaction. Avant tout changement, il vous faut donc démarrer une transaction et committer cette transaction à la fin :

```
Transaction tc = session.beginTransaction() ;  
  
//Différentes modifs  
  
tc.commit() ;
```

Ainsi, les changements seront répercutés dans la base de données.

lien : [Qu'est ce qu'une transaction ?](#)

### Qu'est ce qu'une transaction ?

Auteurs : [Baptiste Wicht](#) ,

Une transaction est un objet qui définit une unité de travail atomique. Une transaction encapsule différentes opérations. On démarre une transaction par la session et pour valider les changements intervenus durant cette transaction, il faut procéder à un commit de celle-ci.

### Quel est la différence entre save et saveOrUpdate() ?

Auteurs : [Baptiste Wicht](#) ,

Save va juste s'occuper de sauvegarder un objet qui n'est pas encore présent dans la base de données, alors que saveOrUpdate, va faire une vérification sur l'état de l'objet et en fonction de son état transient, détaché ou persistant, et en fonction de cet état, il va effectuer l'action correcte. Vous n'avez ainsi plus de problèmes pour savoir l'état de votre objet.

[Sommaire](#) > [Utilisation](#) > [Le langage HQL](#)**Qu'est ce que le HQL ?****Auteurs :** [minosis](#) ,

Hibernate Query Langage est un langage d'interrogation des classes persistantes avec une syntaxe proche du SQL. Cependant les objets manipulés sont les classes et membres du mapping contrairement au SQL qui réalise les requêtes directement sur la base de données. Aussi, le HQL contient des fonctionnalités spécifiques au modèle objet.

Les requêtes HQL peuvent être formées des éléments principaux suivants :

- les clauses (from, select, where, order by, ...)
- les fonctions d'agrégation (count, sum, avg, max, ...)
- les sous-requêtes (il s'agit de requêtes HQL dont le résultat est utilisé dans une requête HQL principale)

**Exemple :**

```
FROM infos.Contact contact WHERE contact.nom = 'Dupont'
```

Cette requête sélectionne tous les contacts, dans la classe infos.Contact, ayant comme nom Dupont. Vous remarquerez l'utilisation d'un alias essentiel si on fait référence à la classe dans différents endroits de la requête.

**lien :** [Plus d'infos](#)

**lien :** [Comment exécuter du code HQL ?](#)

**lien :** [Les requêtes HQL sont elles sensibles à la casse ?](#)

**Comment exécuter du code HQL ?****Auteurs :** [Baptiste Wicht](#) ,

Pour exécuter une requête HQL, il vous faut employer la méthode `createQuery()` sur votre session :

```
Query query = session.createQuery("Requête HQL");
```

Ensuite, si c'est une requête de sélection, vous pouvez récupérer les résultats avec la méthode `list()` ou alors exécuter la requête avec la méthode `executeUpdate()`.

**Les requêtes HQL sont elles sensibles à la casse ?****Auteurs :** [Baptiste Wicht](#) ,

Non, les requêtes HQL ne sont pas sensibles à la casse, sauf pour tout ce qui concerne le nom des classes Java et des propriétés.

Ainsi `FrOm` est tout à fait égal à `FROM` et à `from`. Mais `from Cat` n'est pas égal à `from cAt`.

**Comment insérer des quote (') dans une requête HQL ?****Auteurs :** [Baptiste Wicht](#) ,

Pour pouvoir insérer des quotes dans une requête, il suffit de doubler tous les ' dans votre requête. Par exemple :

```
FROM table WHERE field LIKE 'C'est'
```

### Comment récupérer la taille d'une collection sans l'initialiser ?

Auteurs : **Baptiste Wicht**,

**Vous pouvez directement employer next() sur le résultat de votre requête HQL pour ne pas avoir à initialiser votre collection et simplement compter le nombre de résultats en employant count(\*) :**

```
((Integer) session.iterate("select count(*) from ...").next() ).intValue();
```

### Comment limiter le nombre de résultats d'une requête ?

Auteurs : **Baptiste Wicht**,

**Il suffit d'employer la méthode setMaxResults de votre Query :**

```
Query query = sess.createQuery("Votre requête HQL");  
query.setMaxResults(10);  
List results = query.list();
```

**10 étant le nombre maximal d'enregistrements retournés par la requête.**

[Sommaire](#) > [Utilisation](#) > [Les Criteria](#)

### Qu'est ce que Criteria ?

**Auteurs :** [Baptiste Wicht](#) ,

Criteria est une API de recherche orientée objet. Elle permet de faire des recherches dans la base de données de manière très simple. On peut ajouter des tris, des filtres, ...

HQL a beau être très puissant, beaucoup de développeurs préfèrent avoir une approche plus orientée objet, ils trouvent donc leur bonheur dans Criteria.

**lien :** [Comment effectuer une requête avec Criteria ?](#)

### Comment effectuer une requête avec Criteria ?

**Auteurs :** [Baptiste Wicht](#) ,

Tout d'abord il vous faut une session d'ouverte pour créer la Criteria. Ensuite, Criteria étant une recherche sur une classe donnée, il vous faut connaître le nom de cette classe et finalement, vous pouvez créer la requête :

```
Criteria criteria = session.createCriteria(Od.class);
```

Ensuite, pour récupérer les résultats de la recherche, vous pouvez employer la méthode list() qui va vous renvoyer les résultats sous forme d'une liste (List).

```
List results = criteria.list();
```

Vous avez maintenant tous vos résultats sous forme d'objet dans votre liste.

**lien :** [Comment récupérer le résultat d'un Criteria dans un Set ?](#)

### Comment restreindre les résultats de la requête ?

**Auteurs :** [Baptiste Wicht](#) ,

Avec Criteria, vous pouvez ajouter des restrictions qui vous permettront de limiter les résultats en fonctions de différents critères.

Pour cela, nous allons ajouter des Criterions, que va nous fournir la classe restriction. Par exemple pour un like :

```
Criteria criteria = sess.createCriteria(Od.class)
    .add(Restrictions.like("description", "Attention%"));
List results = criteria.list();
```

Vous pouvez employer encore d'autres restrictions, telles que between ou lt (less than).

**lien :** [Comment grouper les restrictions de manière logique ?](#)**lien :** [La classe Restrictions](#)

### Comment grouper les restrictions de manière logique ?

**Auteurs :** [Baptiste Wicht](#) ,

Vous pouvez aussi regrouper différents critères de manière logique avec les opérateurs ou et et.

Voici un exemple avec un opérateur ou :

```
Criteria criteria = session.createCriteria(Od.class)
    .add(Restrictions.between("weight", 100,200) )
    .add(Restrictions.or(
        Restrictions.isNotNull("etiquette"),
        Restrictions.isNull("description")
    ) )
List results = criteria.list();
```

## Comment effectuer une recherche à partir d'un exemple ?

Auteurs : Baptiste Wicht ,

Une autre possibilité qui nous est donnée par Criteria est de pouvoir effectuer une recherche à partir d'un exemple.

Exemple, vous avez une classe Film avec un réalisateur et une année de sortie et que vous voulez trouver tous les autres films de ce même réalisateur :

```
Film film = new Film();
film.setRealisateur("Frères Wachowsky");
film.setAnnee(2005);

Example example = Example.create(film)
    .excludeProperty("annee") //On ne compare pas avec l'année
    .ignoreCase(); //On ne tient pas compte de la casse

Criteria criteria = session.createCriteria(Film.class)
    .add(example);

List results = criteria.list();
```

## Comment trier les résultats d'une requête ?

Auteurs : Baptiste Wicht ,

Vous pouvez trier les résultats d'une requête Criteria par propriété dans l'ordre croissant ou décroissant.

Pour cela, nous allons employer la méthode addOrder sur le Criteria et la classe Order :

```
Criteria criteria = session.createCriteria(Od.class)
    .add(Restrictions.between("weight", 200, 500))
    .addOrder(Order.asc("weight"))
    .addOrder(Order.desc("nom"));
List results = criteria.list();
```

## Comment savoir le nombre d'enregistrements retournés par une requête Criteria ?

Auteurs : Baptiste Wicht ,

Vous pouvez ajouter une Projection à votre Criteria :

```
criteria.setProjection(Projections.rowCount());
```

Et dire ensuite que vous voulez récupérer un seul résultat :

```
Integer result = (Integer) criteria.uniqueResult();
```

Donc pour une requête complète, vous pouvez faire :

```
Criteria criteria = session.createCriteria(OdPersonne.class);
criteria.setProjection(Projections.rowCount());
Integer result = (Integer) criteria.uniqueResult();
System.out.println("Nombre d'enregistrements : " + result);
```

## Comment limiter le nombre de résultats retournés par une requête Criteria

Auteurs : [Baptiste Wicht](#) ,

Comme pour une requête HQL, il vous suffit d'employer la méthode `setMaxResults` sur votre Criteria pour que la recherche ne retourne par plus de résultats que voulu.

```
Criteria crit = session.createCriteria(Od.class);
crit.setMaxResults(10);
```

## Comment récupérer le résultat d'un Criteria dans un Set ?

Auteurs : [Baptiste Wicht](#) ,

Par défaut, cela n'est pas possible, mais il suffit tout simplement de rajouter dans un Set les éléments de la liste renvoyée par le Criteria :

```
HashSet set = new HashSet();
set.addAll(session.createCriteria(MonOd.class).list());
```

Ainsi, tous les éléments de votre liste se trouveront exempts de doublons dans votre Set.

[Sommaire > Divers](#)

### Qu'est ce que NHibernate ?

**Auteurs :** [Baptiste Wicht](#) ,

NHibernate est le pendant d'Hibernate pour .NET. Vous pouvez ainsi faire du mapping objet/relationnel sur vos bases de données avec .NET.

NHibernate étant directement inspiré d'Hibernate, toutes les connaissances et la documentation sur ce dernier lui est aussi applicable.

**lien :** [Plus d'infos sur NHibernate](#)

### Comment récupérer la connexion JDBC de votre session ?

**Auteurs :** [Baptiste Wicht](#) ,

Il est possible que vous ayez besoin de récupérer la connexion JDBC pour l'utiliser directement. Pour cela, vous pouvez utiliser la méthode `connection()` de votre session :

```
Connection connection = session.connection();
```

### Comment executer un PreparedStatement sur Hibernate ?

**Auteurs :** [Baptiste Wicht](#) ,

Il vous faut tout simplement récupérer la connexion JDBC d'Hibernate et ensuite l'utiliser pour créer un PreparedStatement, vous pouvez ensuite l'exécuter comme vous avez l'habitude de le faire sur JDBC, soit avec `executeUpdate` si c'est une requête d'update, soit avec `executeQuery()` si c'est une méthode de sélection.

```
PreparedStatement ps = session.connection().prepareStatement("Requête SQL");
```

**lien :** [Comment récupérer la connexion JDBC de votre session ?](#)

### Que faut-il faire pour migrer de la version 2 à la version 3 ?

**Auteurs :** [Baptiste Wicht](#) ,

La meilleure chose que je puisse vous conseiller de faire pour migrer d'Hibernate version 2 à la version 3 est de lire le [guide de migration officiel](#).

Ce guide recense tous les changements de la version 3 et toutes les choses qu'il va falloir modifier dans votre code pour qu'il fonctionne aussi avec la version 3. Si vous suivez correctement ce guide, vous n'aurez aucun problème pour passer de la version 2 à la 3.

Sommaire > Divers > Exceptions

### Comment résoudre l'erreur HQL : « net.sf.hibernate.QueryException: undefined alias » ?

Auteurs : [minosis](#) ,

Un alias de classe doit être créé. Contrairement au SQL, le HQL impose la plupart du temps d'utiliser des alias des objets sur lesquels on travaille pour en manipuler leurs membres.

```
SELECT produits.reference,produits.titre FROM stock.Produits produits WHERE produits.prix > 50
```

### Pourquoi est-ce que j'obtiens une erreur lorsque j'utilise un type primitif dans mon javabean ?

Auteurs : [Baptiste Wicht](#) ,

Tout simplement, parce qu'il ne faut pas employer des types primitifs, Hibernate n'acceptant que les types Serializable. Il vous faut employer des types sérialisables, Integer par exemple. C'est pareil pour boolean, double, long, #. Il vous faut toujours employer un type qui soit sérializable.

### Que faire en cas de : IllegalArgumentException: Removing a detached instance ?

Auteurs : [Baptiste Wicht](#) ,

Cela veut dire que l'objet n'est plus attaché à la session Hibernate. Il vous faut donc le rattacher à la session.

Pour cela, vous pouvez consulter la question qui en parle.

lien : [Comment attacher un objet détaché à une session ?](#)

lien : [Qu'est ce qu'un objet détaché ?](#)

### Pourquoi est-ce qu'Hibernate ne fonctionne pas avec toute les version de MySql ?

Auteurs : [Baptiste Wicht](#) ,

Certaines versions de Mysql ne fonctionnent pas correctement avec les preparedStatement. Il faut donc désactiver les preparedStatement.

Pour cela, il vous faut ajouter `useServerPrepStmts=false` à l'url de connexion à votre base de données.

### Que faire si je vois le log : Parsing XML: unknown system id ?

**Auteurs :** Baptiste Wicht ,

Rien du tout, cette erreur est tout à fait normale lors du chargement des ressources.

**lien :** Comment attacher un objet détaché à une session ?

### Pourquoi est-ce que j'obtiens une OutOfMemory lors de l'insertion de beaucoup d'objets ?

**Auteurs :** Baptiste Wicht ,

Tout simplement parce qu'Hibernate va devoir stocker tout vos objets dans le cache et que cela va demander énormément de mémoire si vous avez beaucoup d'objet.

Comment y remédier ? Il faut procéder à une insertion par paquet. C'est-à-dire que tous les 20 (taille d'un paquet JDBC, mais vous pouvez mettre autre chose) insertions, il faut faire un flush de la session et un clear du cache :

```
for(int index = 0 ; index < 100000 ; index++){  
    Od objet = new Od(/* Paramètres divers */);  
    session.save(objet);  
  
    if(index % 20 == 0){  
        session.flush();  
        session.clear();  
    }  
}
```

Ainsi vous aurez des petits paquets et un risque quasi nul de dépassement mémoire.