TP4 : Hibernate (version 5.4)
avec annotations et PostgreSQL

1 Installation de PostgreSQL

1.1 Procédure d'installation (sous Linux)

1. Sur votre machine, installez le paquet `postgresql`
2. L'installation ajoute un nouvel utilisateur linux et un utilisateur de la base de données nommé *postgres.* Il est seul autorisé, pour l'instant, à se connecter à la base de données. Il ne possède pas de mot de passe, et par mesure de sécurité nous ne lui en attribuerons pas. Nous allons créer un nouvel utilisateur. Connectez avec l'utilisateur postgres en utilisant la commande suivante :
   ```
   sudo -i -u postgres
   ```
3. Lancez l'invite de commande Postgresql à l'aide de la commande `psql`.
4. Vous êtes désormais connecté à Postgresql en mode administrateur.
5. Créer un utilisateur portant le même nom que votre utilisateur linux (on supposera « test » dans la suite)
   ```
   CREATE USER test ;
   ```
6. Donnez-lui le droit de créer un base de données.
   ```
   ALTER ROLE test WITH CREATEDB ;
   ```
7. Créez un base portant le même nom que l'utilisateur :
   ```
   CREATE DATABASE test OWNER test ;
   ```
8. Ajoutez un mot de passe à votre utilisateur
   ```
   ALTER USER test WITH ENCRYPTED PASSWORD 'password' ;
   ```
   Personnalisez ici votre mot de passe pour sécuriser votre base.
9. Quittez Postgresl avec \q
   Quittez l'utilisateur postgres avec `exit`
   Lancez Postgresql avec la command `psql` (normalement vous êtes automatiquement connecté avec la base de données test).

1.2 Aide sur les commandes de base

Une fois connecté avec votre compte à Postgresql, vous pouvez utiliser les commandes suivantes :
\h pour obtenir de l'aide sur les commandes SQL
\? pour obtenir de l'aide sur les commandes Postgresql
\q pour quitter

1.3 Installation d'un client graphique

Les dépôts contiennent une application graphique d'administration d'une base de données : PgAdmin. Installez le paquet `pgadmin3` sur votre machine.

2 Objectifs de ce TP

- Découvrir la gestion de la **persistance des objets** Java à l'aide du framework **Hibernate** et des annotations JPA (Java Persistence API). Nous n'utiliserons pas le moteur de base de données proposé avec Hibernate, mais une base PostgreSQL.
  - *Documentation officielle Hibernate :* http://hibernate.org/orm/documentation/5.4/
  - *Hibernate Annotations : http://docs.jboss.org/hibernate/orm/5.4/quickstart/html_single/*

3 Configuration de base du projet

- Sur votre serveur de base de données PostgreSQL, créer une base sans aucune table ;

- Dans votre IDE, créer un projet Maven - Java Application comme dans le premier TD.
- Ajouter les dépendances Maven pour PostgreSQL et pour Hibernate 5.4 (core ORM)
  A chercher ici : https://mvnrepository.com/
- Ecrivez un classe `Employee.java` dans le package `tsi.ensg.jee.hibernate.ex1` qui complète le squelette ci-dessous avec les constructeurs, getters et setters ainsi que les méthodes `toString`,`equals` et `hashCode`.

```
public class Employee {
    private long id;
    private String firstName;
    private String lastName;
    private int salary;

    ...

}
```

4 Configuration d'Hibernate

- Créer un fichier « XML » nommé « hibernate.cfg.xml » dans le dossier src/main/resources

  Le fichier contiendra la configuration de la connexion à la base de données :

```
<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
"http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">
<hibernate-configuration>

<session-factory>

  <property name="connection.driver_class">org.postgresql.Driver</property>

  <property name="connection.url">jdbc:postgresql://localhost:5432/mabase</property>

  <property name="connection.username">test</property>

  <property name="connection.password">test</property>


  <property name="hibernate.dialect">org.hibernate.dialect.PostgreSQL10Dialect</property>

</session-factory>

</hibernate-configuration>
```

- Ajouter les lignes suivantes dans le fichier XML :

```
<property name="hibernate.hbm2ddl.auto">create</property>
<property name="hibernate.show_sql">true</property>
<mapping class="tsi.ensg.jee.hibernate.ex1.Employee"/>
```

- Essayer de comprendre le rôle de ces trois lignes ;

5 Ajout d'un enregistrement dans la base de données

En vous aidant du support de cours :

- Ajouter les annotations hibernate dans la classe Employee ; Attention de bien importer le package JPA pour les annotations `javax.persistence.*` et non celui d'Hibernate qui est déprécié.
- Créer la classe `HibernateUtils` dans le package `tsi.ensg.jee.hibernate.ex1` La classe `HibernateUtils` implémente le *design pattern* singleton et fournit une unique `SessionFactory` qui sera utilisée par votre application.
- Ecrire une classe `Application` principale qui ajoute un `Employee` Harry Potter payé 1000 euros par mois dans la BD.
- Vérifier que la table contient bien l'employé.

6 CRUD (Create, Read, Update, and Delete)

On veut maintenant réaliser une classe de DAO (Data Access Object) pour réaliser les opérations de base de persistence pour nos objets de la classe `Employee`. Cette classe `EmployeeDAO` contient les méthodes suivantes:

```
public class EmployeeDAO {

   private final SessionFactory sessionFactory = HibernateUtils.getSessionFactory();


    public long create(String firstName, String lastName, int salary) {
        // TODO
    }

    public boolean delete(long id) {
        // TODO
    }

    public boolean update(long id, int salary) {
        //TODO
    }


    public Optional<Employee> get(long id) {
       // TODO
    }

    public List<Employee> getAll() {
        // TODO
    }

}
```

1. Complétez la classe `EmployeeDAO`

2. Ecrivez une classe `Application` qui:

   - crée 5 employés : Boby Lapointe (500), Boris Vian (600), Lisa Simpson (100), Marge Simpson (1000) et Homer Simpson (450),
   - supprime Lisa Simpson de la base de données,
   - augmente le salaire de Homer Simpson de 100,
   - enfin affiche tous les employées de la base de données.

3 Proposez une nouvelle signature pour la méthode `update` qui permette de réaliser simplement:

   - l'augmentation du salaire de 10%,
   - ajout de 100 euros si le salaire est inférieur à 1000 euros.

4 Dans votre classe `Application`, rajouter 100 euros à tous les employés qui gagnent moins de 550 euros.

5 Rajoutez dans la classe `EmployeeDAO` une méthode `List<Employee> getAllByFirstName(String firstName)` qui renvoie la liste des employés ayant un prénom donné.
Vous utiliserez un requête HQL préparée.