Travail pratique #3
Conception du serveur d'application et du client

1 Introduction

L'objectif de ce travail est de vous familiariser avec la conception d'une application à trois niveaux (three-tier application) en utilisant, dans un premier temps, une couche présentation, une couche de contrôle et finalement des données transactionnelles stockées dans une BD relationnelle Oracle. En particulier, ce laboratoire portera sur l'utilisation du framework de mappage objet/relationnel (ORM) Hibernate pour assurer la persistance transparente des données.

Le travail à réaliser dans le cadre de ce TP se base sur celui accompli lors des travaux précédents. Ainsi, l'application à développer devra utiliser le schéma relationnel conçu dans les travaux précédents de même que les données qui y ont été insérées.

2 Travail à réaliser

Tâche 1 : Modélisation des classes de l'application en Java

En se basant sur votre diagramme de classe UML élaboré précédemment, vous devrez tout d'abord coder en Java toutes les classes métier utilisées par l'application (ex : Film, Client, Location, etc.). Ces classes doivent modéliser les attributs et les relations des différentes entités du système.

Consigne 1 : Vous pouvez, si nécessaire, apporter des modifications à votre diagramme de classe ou votre schéma relationnel.

Consigne 2 : Bien que vous n'êtes pas tenu de mettre des commentaires pour chaque élément de votre code, vous serez pénalisé si celui-ci n'est pas suffisamment compréhensible. En particulier, choisissez des noms appropriés et significatifs pour vos classes, attributs et méthodes.

Tâche 2 : Mappage des objets à la BD avec Hibernate

Pour cette seconde tâche, vous devrez définir le mappage de vos objets aux tables de la BD en utilisant le framework ORM Hibernate.

Dans un premier temps, vous devrez modifier le fichier de configuration **hibernate.cfg.xml** se trouvant dans le répertoire **src/main/resources**, afin de mettre les bons paramètres de connexion (username, password et default schema)

```
<hibernate-configuration>
  <session-factory name="Hibernate">
 <property name="hibernate.connection.driver_class">
        oracle.jdbc.driver.OracleDriver
 </property>
 <property name="hibernate.connection.url">
          jdbc:oracle:thin:@dim-oracle.uqac.ca:1521/dimdb.uqac.ca
 </property>
 <property name="hibernate.connection.username">
    trd157XX
 </property>
 <property name="hibernate.connection.password">
   MotDePasse
 </property>
<property name="dialect">
         org.hibernate.dialect.OracleDialect
</property>
</session-factory>
</hibernate-configuration>
```

Ensuite, vous devrez créer, pour chaque classe de votre application, un fichier de mappage (hibernate-mapping) définissant :

- Le mappage de la classe vers une table ;
- Le mappage de l'identifiant de la classe vers une clé primaire, et le mécanisme de génération si la clé est artificielle ;
- Le mappage des propriétés de la classe vers des colonnes de sa table ;
- Le mappage des relations d'association (ex : 1-1, 1-plusieurs, plusieurs-1, etc.) ;
- Le mappage des relations de spécialisation (ex : une table par hiérarchie, par sous-classe, etc.)

Voici un exemple de fichier de mappage `Film.hbm.xml` :

```
<?xml version="1.0"?>
<!DOCTYPE hibernate-mapping PUBLIC
      "-//Hibernate/Hibernate Mapping DTD//EN"
      "http://hibernate.org/dtd/hibernate-mapping-3.0.dtd">
<hibernate-mapping>
      <class name="Film" table="FILM">
            <id name="id" column="ID_FILM"/>
                  <generator class="native"/>
            </id>
            <property name="titre" column="TITRE"/>
            <property name="annee"column="ANNEE_SORTIE"/>
            ---
            <set name="lesActeurs"inverse="true" cascade="all">
                  <key column="ID_FILM"/>
                  <one-to-many class="Personne"/>
            </set>
      </class>
</hibernate-mapping>
```

Consigne 1 : Basez-vous sur les acétates présentées en classe ainsi que sur l'exemple fourni (ExempleHibernate.zip) pour la définition des fichiers de mappage. De plus, vous pouvez générer ceux-ci automatiquement avec Netbeans tel que démontré en classe. Finalement, vous pouvez également vous référer à la documentation en-ligne d'Hibernate.

Tâche 3 : Implémentation de la couche présentation

Dans cette troisième tâche, vous aurez à développer la couche présentation de votre application sous la forme d'un client mince. Dans le but de simplifier la tâche, ce client doit pouvoir traiter **uniquement les cas d'utilisation 2, 3 et 4** du document de mise en situation, soit :

- Cas 2 : connexion d'un utilisateur au système ;
- Cas 3 : consultation interactive des films ;
- Cas 4 : location de films.

Pour le développement de votre client, vous avez deux options :

1. Client Web à l'aide de Servlet Java (ou technologie équivalente au choix);
2. Interface GUI au choix (ex : Swing)

Si vous choisissez le choix 1, référez-vous au document « Application Web et Servlet Java » disponible avec cet énoncé pour des instructions détaillées.

À noter : toute opération transactionnelle de l'application doit être faite à l'intérieur d'une session Hibernate (et à l'intérieur d'un courtier). Par exemple :

```
Session uneSession = HibernateUtil.getSessionFactory().openSession();
uneSession.beginTransaction();

List lesClients = uneSession.createQuery("FROM Client c WHERE c.courriel =
'toto@uqac.ca'").list();
Client unClient = (Client) lesClients.next();

List lesCopies = uneSession.createQuery("FROM Copie c WHERE c.titre = 'La
revanche de toto' AND c.statut = 'disponible'").list();
Copie uneCopie = (Copie) lesCopies.next();

Client.getLocations().add(uneCopie);
uneCopie.setEmprunteur(unClient);
uneCopie.setStatut("louee");

uneSession.getTransaction().commit();
uneSession.close();
```

Consigne 1 : Ne passez pas trop de temps sur le côté esthétique de l'application, car celui-ci ne sera pas évalué.

Consigne 2 : Assurez de bien tester toutes les fonctionnalités de votre application, y compris les utilisations non-prévues et autres cas d'erreurs.

Consigne 3 : Les mêmes règles d'affaires (contraintes) que pour le travail #2 s'appliquent. Par exemple, un client ne peut toujours pas louer plus de films que le permet son forfait. Ces règles peuvent être validées soit au niveau de la BD (ex : les TRIGGER développées au TP 2) ou au niveau de l'application.

Tâche 4 : Rédaction du rapport

Enfin, la dernière tâche du travail sera de rédiger un rapport décrivant votre travail et justifiant vos décisions de conception.

Note : utilisez le gabarit prévu à cet effet.

3 Consignes de remise

Avant la séance du 3 décembre 2018, vous devrez déposer votre travail sur le Moodle du cours, à savoir :

1. Le gabarit de rapport complété ;
2. Tous les fichiers de configuration et de mappage Hibernate ;
3. Le code source de votre application (serveur et couche présentation);

Note : Tous les fichiers de remise doivent être dans un répertoire compressé ayant comme nom 8TRD157-TP3-[Noms des équipiers].

4 Barème de correction

L'évaluation du laboratoire sera faite sur un total de 100 points, distribués comme suit :

COMPOSANTE	POINTS
Rapport :	(45 pts)
Manuel utilisateur	5
Stratégie de test	5
Patrons de conception	5
Validation des contraintes et de l'accès	5
Code de mappage et justification	10
Question 1	5
Question 2	5
Autres items	5
Code source :	(30 pts)
Code Hibernate	10
Code d'application Java	20
Fonctionnalité et interface utilisateur :	(25 pts)
Correction interactive	25
TOTAL	100