La base Bibliothèque

Pour commencer

Exercice 1 : Création de la base de données

Créez une nouvelle base qui portera le nom de `biblio_votrenom`, puis connectez vous sur cette nouvelle base, et enfin chargez le fichier de création et de peuplement des tables :

```
biblio# create database biblio_votrenom;
CREATE DATABASE
biblio=# \c biblio_votrenom
Vous êtes maintenant connecté à la base de données « biblio_votrenom ».
biblio_votrenom=#\i 'bibliotheque.sql'
psql:biblio2.sql:5: INFO:  CREATE TABLE créera des séquences implicites ...
psql:biblio2.sql:5: INFO:  CREATE TABLE / PRIMARY KEY créera un index ...
...
INSERT 0 1
biblio_votrenom=#
```

Écrire des fonctions SQL et PL/SQL

Ça y est, vous êtes prêts à travailler. Pour les exercices suivants, vous devrez choisir d'écrire des fonctions soit en SQL, soit en PL/SQL. Vous devrez toujours tester vos fonctions (donc écrire une requête appelant votre fonction et vérifier son résultat).

Vous devrez travailler en créant vos fonctions dans un fichier `mesFctsBiblio.sql` que éditerez à côté (à l'aide de votre éditeur préféré, `gedit` par exemple) que vous re-chargerez à chaque nouvelle fonction dans votre base.

Exercice 1 :

Écrire une fonction `nbre_exemplaires` qui retourne le nombre de livres associés à une oeuvre.

Exercice 2 :

Écrire une fonction `est_emprunte` qui retourne un booléen suivant qu'un livre est actuellement emprunté ou non.

Exercice 3 :

Écrire une fonction `les_livres_presents` qui retourne les livres actuellement présents sur les rayons de la bibliothèque.

Exercice 4 :

Écrire une fonction `nbre_emprunts` qui retourne le nombre de livres empruntés (depuis l'ouverture de la bibliothèque) associés à une même oeuvre.

Exercice 5 :

Écrire une fonction `les_plus_empruntes` qui retourne les n oeuvres les plus empruntées depuis l'ouverture de la bibliothèque. En cas d'égalité, on rangera les oeuvres par ordre alphabétique sur leur titre.

Exercice 6 :

Écrire une fonction `infos_oeuvres` qui, pour une oeuvre donnée, retourne une chaîne de caractères qui contient des informations sur cette oeuvre. Si l'oeuvre est écrite par une seule personne, alors la chaîne retournée devra être *Titre : Le titre de l'oeuvre - Auteur : Le nom de l'auteur*; si l'oeuvre a été écrite par plusieurs personnes, alors la chaîne retournée sera *Titre : Le titre de l'oeuvre - Auteurs : Le nom du premier auteur et al.*. Appelez cette fonction sur toutes les oeuvres de la base.

Les déclencheurs

Tous les exercices sont à réaliser et *à tester* sur la base `biblio_votrenom` que vous venez de créer. Les exercices suivants sont sous la forme de règle de gestion. La plupart de ces règles doivent être réalisés par un déclencheur, mais certaines peuvent n'être qu'une contrainte de table. Dans ce cas, vous devrez modifier la table. Il est important que vous testiez chacune de vos réponses.

Si vous avez un doute sur une syntaxe SQL, n'oubliez pas la commande `\h`. Par exemple :

```
biblio# \h alter table
```

Exercice 1 : Un adhérent ne peut pas emprunter plus de trois livres.

Exercice 2 : On ne peut pas emprunter un livre qui est déjà emprunté.

Exercice 3 : Lorsqu'un livre est rendu au retour d'un prêt, alors l'emprunt qui vient de se terminer doit être enregistré dans `histoemprunt`. La seule exception est le cas où la date d'emprunt est également la date du retour.

Exercice 4 : Un adhérent ne peut pas emprunter s'il a une dette, ou s'il a un livre en retard (la durée du prêt est de trois semaines).

Exercice 5 : Lorsqu'un adhérent rend un livre, il aura une amende de deux euros par semaine de retard (le cas échéant).

Exercice 6 : Lorsqu'un livre est supprimé de la base (et des rayons), dans l'historique des emprunts (`histoemprunt`), on remplace sa référence par `null`.

Exercice 7 : On ne peut rien supprimer dans `histoemprunt`, sauf les enregistrements dont date la date d'emprunt est la même que la date du retour ou bien les enregistrements dont la référence du livre est `null`.

Exercice 8 : Pour savoir plus rapidement si un livre est emprunté, on décide d'ajouter un champ booléen `sorti` dans la table `livre`. Ajoutez ce champ, avec comme valeur par défaut `false`.

Exercice 9 : Proposez des solutions pour que le champ `sorti` soit toujours à jour.

Exercice 10 : On décide maintenant d'ajouter un champ `reserve_adh` (qui est une clé étrangère pour `adherent`) dans la table `livre`. Ajoutez ce champ. Ils permettra d'indiquer si un livre est réservé par un adhérent.

Exercice 11 : Maintenant on ne peut pas emprunter un livre s'il est déjà réservé, à moins que l'emprunteur ne soit celui qui a fait la réservation. Dans ce cas, le champ `reserve_adh` doit être remis à `null` lors de l'emprunt.