SAÉ S2.04 : Exploitation d'une base de données

A) Objectifs de la SAÉ

À partir d'un cahier des charges, l'étudiant devra réaliser et étudier une base de données relationnelle. L'étudiant devra aborder les aspects de conception, implémentation, administration, exploitation d'une base de données relationnelle et visualiser les résultats de son analyse.

B) Sujet

Il s'agit de poursuivre la SAE S1.04 sur la base de données BD'Lire pour une analyse plus poussée des données pour la gestion des ventes, l'affectation des droits d'accès aux différents intervenants, et la visualisation des données agrégées de la base dans Libre Office.

1. Rappel de la base de données

Le modèle logique des données correspondant (MLD) est le suivant :

Serie (numSerie, nomSerie,#numEditeur)
BD(isbn,titre,prixActuel,numTome,#numserie,#numAuteurDessinateur, #numAuteurScenariste)
Editeur(numEditeur, nomEditeur, adrEditeur, numTelEditeur, mailEditeur)
Auteur(numAuteur, nomAuteur, prenomAuteur, dteNaissAuteur, dteDecesAuteur, pays)
Vente (numVente, dteVente, #numClient)
Client (numClient, nomClient, numTelClient, mailClient)
Concerner(#isbn, #numVente, prixVente, quantite)

2. Fonction PlpgSQL

L'objectif est de réaliser et d'exécuter les procédures stockées des questions qui suivent en PGSQL ; on utilisera pour cela au maximum des types « basés » (c'est-à-dire reprenant les types déjà définis sur les attributs des tables de la base) et on pourra également créer des types ad hoc si besoin (via la création de nouveaux types par exemple pour correspondre aux types composites retournés par les fonctions).

NB : pour les questions où cela se présente, vous devrez exécuter les deux cas de figure : un cas qui renvoie un résultat positif et un autre négatif (qui génère les messages d'avertissement ou d'erreur).
Pour nommer les procédures, vous pourrez utiliser un nom systématique pour chaque question composé de la base 'proc_' suivie du nom de la question.
Ex : 'proc_f' pour la procédure de la question f.

1) Ecrire une procédure qui prend en paramètre un numéro d'auteur dessinateur et qui renvoie pour chaque titre de BD de l'auteur, le nombre d'exemplaires vendus de cette BD.
2) Écrire une fonction qui prend en paramètre le nom d'une série de BD et qui renvoie pour chaque titre de la série le nombre d'exemplaires vendus et le chiffre d'affaire réalisé par titre.
3) Écrire une procédure qui prend en paramètre un nom d'éditeur et un nom d'auteur dessinateur et un nom d'auteur scénariste, et qui renvoie la liste des BD de ces auteurs éditées par l'éditeur choisi. Si l'éditeur n'a pas édité de BD de ces auteurs, ou qu'il n'existe pas de BD de ces deux auteurs, on devra générer le message suivant « l'éditeur % n'a pas édité de BD des auteurs % et %» où on remplacera les « % » par les noms correspondants.
4) Créer une procédure stockée qui prend en paramètre le nom d'une série de BD et qui renvoie les clients ayants acheté tous les albums de la série (utiliser des boucles FOR et/ou des curseurs).
   Si aucun client ne répond à la requête alors on affichera un message d'avertissement 'Aucun client n'a acheté tous les exemplaires de la série %', en complétant le ' %' par le nom de la série.
5) Créer une procédure qui prend en paramètre un nombre nbBD de BD et une année donnée, et qui renvoie la liste des éditeurs ayant vendu au moins ce nombre de BD dans l'année en question. Si aucun éditeur ne répond à la requête, le signaler par un message approprié.
6) Écrire une procédure qui prend en paramètre une année donnée, et un nom d'éditeur et qui renvoie le(s) tuple(s) comportant l'année et le nom de l'éditeur d'une part, associé au nom et email du(des) client(s) d'autre part ayant acheté le plus de BD cette année-là chez cet éditeur.
7) Écrire une procédure SQL utilisant un curseur, qui classe pour un éditeur dont le nom est donné en entrée, les clients de cet éditeur en trois catégories selon le nombre de BD qu'ils leur ont achetées : les « très bons clients » (plus de 10 achats strictement), les « bons clients » (entre 2 et 10 BD), les « mauvais clients » (moins ou égal à 2 BD)
8) Ecrire une procédure qui renvoie pour tous les clients sa plus petite quantité achetée (min) et sa plus grande quantité achetée (max) et la somme totale de ses quantités achetées de BD.
   Vous devrez donc créer un type composite 'clientBD' comportant quatre attributs: l'identifiant du client, son nom, sa plus petite quantité achetée, sa plus grande quantité achetée, et la somme totale de ses quantités achetées. Votre procédure devra retourner des éléments de ce type de données.
   On rajoutera le comportement suivant : si le minimum est égal au maximum pour un client, on affichera le message 'Egalité du minimum et maximum pour le client %' en précisant le nom du client.
   NB : utiliser une boucle FOR ou un curseur...
9) Ecrire une procédure qui supprime une vente dont l'identifiant est passé en paramètre.

Vérifier d'abord que la vente associée à l'identifiant existe, si elle n'existe pas afficher un message d'erreur le mentionnant; si elle existe on la supprime.
Cette suppression va générer une violation de clé étrangère dans la table 'Concerner'.
Pour gérer cela, on utilisera le code d'erreur FOREIGN_KEY_VIOLATION dans un bloc EXCEPTION dans lequel on supprimera tous les tuples de la table 'Concerner' qui possèdent ce numéro de vente, avant de supprimer la venet elle-même. On pourra au passage afficher aussi un message d'avertissement sur cette exception.

10) On souhaite classer tous les clients par leur quantité totale d'achats de BD. Ainsi on veut associer à chaque client son rang de classement en tant qu'acheteur dans l'ordre décroissant des quantités achetées. Ainsi le client de rang 1 (classé premier) aura totalisé le plus grand nombre d'achats.
Vous devez donc créer un nouveau type de données 'rangClient', qui associe l'identifiant du client, son nom et son classement dans les acheteurs (attribut nommé 'rang').
Ecrire une procédure qui renvoie pour tous les clients, son identifiant, son nom et son classement d'acheteur décrit ci-dessus.
NB : on pourra avantageusement utiliser une boucle FOR ou un curseur...

3. **Visualisation de données**
   a) représenter dans le tableur de Libre Office par une courbe des valeurs d'évolution des ventes de BD (chiffre d'affaires) par année de vente (dans l'ordre croissant des années). Les années sont en abscisse et le chiffre d'affaire en ordonnée.
   b) Représenter le nombre de BD vendues par éditeur de deux manières : par un diagramme de type histogramme (les noms des éditeurs sont en abscisse et le nombre de BD est en ordonnée) et par un diagramme de type secteur (on les placera sur la même feuille).

4. **Administration de la base**
   On considère trois rôles/utilisateurs qui peuvent intervenir sur la base : l'administrateur de la base, l'éditeur, le vendeur. Définir les privilèges de chacun des rôles ci-dessous et les déclarer sur la base de données :
   a) l'administrateur a les droits de création/suppression des bases, des tables, et toutes les opérations de lecture/écriture sur la base et l'affectation des privilèges aux autres utilisateurs/roles
   b) le vendeur a les droits en lecture/écriture sur la table des ventes et des clients et sur la table Concerner entre les deux.
   c) l'éditeur a les droits du vendeur plus les lectures/écritures sur les tables BD, Serie, et Auteur.

   NB : Ecrire seulement les scripts SQL qui attribuent ces privilèges à chacun de ces rôles sans les exécuter (vous n'aurez pas les droits pour le faire sur le SGBD de l'IUT).

C) Livrables .

Les groupes restent ceux définis en phase 1 :

- Fichier de scripts SQL pour les procédures stockées avec l'énoncé des questions et les résultats d'exécution obtenus (question B.2)
- fichier Libre Office pour la visualisation des données (question B.3).
- Fichier des scripts d'administration (commandes SQL de gestion des droits – question B.4).

  Les résultats ne sont pas demandés ici car vous n'aurez pas les droits suffisants pour exécuter ces commandes sur le SGBD PostgreSQL de l'IUT.