Bases de données relationnelles
Prof. Nastaran Fatemi
Assistante : Fatemeh Borran

Labo4 : Opérations avancées SQL

Objectifs

Dans ce laboratoire vous allez pratiquer les concepts avancés de SQL, notamment les triggers, les vues et les procédures/fonctions stockées.

Indications

- Utiliser la base de données *Sakila* (du Labo3) pour effectuer les différentes opérations.
- Chaque opération doit être correctement formatée et indentée. Les 20 premiers tuples du résultat de chaque opération seront reportés dans le rapport (screenshot, ou résultat obtenu), avec le nombre de tuples total.
- Tous les mots clés SQL sont en majuscule, le reste en minuscule.

Le laboratoire est à rendre :

- Le **mardi 2 decembre avant 9h** pour la **classe BDR-1-B-L1**.
- Le **vendredi 5 decembre avant 14h50** pour la **classe BDR-1-A-L1**.

Triggers (déclencheurs)

Vérifier la syntaxe MySQL sur http://dev.mysql.com/doc/refman/5.6/en/triggers.html.
Pour chaque question, montrer que l'opération s'est bien déroulée : Donner toutes les requêtes utilisées pour vérifier le bon fonctionnement.

1. Utiliser un Trigger sur la table `payment`, pour qu'à chaque insertion, le paiement soit majoré de 8% et la date de paiement soit mise à jour à la date courante du serveur.

2. Créer une nouvelle table "staff_creation_log" avec les attributs "username" et "when_created".
   Créer un Trigger pour insérer une nouvelle ligne dans "staff_creation_log", à chaque fois qu'un tuple est inséré dans la table "staff".

3. Utiliser un Trigger qui permet de mettre à jour l'adresse email d'un membre du personnel de manière automatique à partir de son prénom et de son nom selon le format "*prénom.nom@sakilastaff.com*".

4. Créer une nouvelle table "customer_store_log" avec les attributs "customer_id", "last_store_id", "register_date" et "unregister_date" dont le but est d'archiver les anciens magasins fréquentés par un client. Le "register_date" représente la date d'inscription d'un client dans un magasin et le "unregister_date" représente la date de sa désinscription. Créer un Trigger pour enregistrer une nouvelle ligne dans la table "customer_store_log", à chaque fois qu'un client change de magasin.

Events (événements)

5. Créer un événement qui permet de nettoyer une fois par minute (pour pouvoir le tester !) la table "customer_store_log" pour éliminer de cette table les enregistrements des clients qui ont changé au moins deux fois de magasin et dont leur plus récent changement date de plus d'une année.

Vues

Vérifier la syntaxe MySQL sur http://dev.mysql.com/doc/refman/5.6/en/create-view.html.

6. On aimerait envoyer des cartes du Nouvel An à l'adresse postale du personnel. On voudrait déléguer cette tâche à un employé, Franklin, mais pour des questions de protection des données personnelles, on voudrait donner accès à Franklin seulement aux numéros de téléphone des membres du personnel et de leurs adresses postales, mais pas leurs adresses e-mail (ou, d'ailleurs, les mots de passe). Créer une vue sur la table du personnel qui permettra à Franklin de réaliser la tâche demandée.
   Est-ce que Franklin pourra mettre à jour la base de donnée à travers cette vue?

7. On aimerait demander à Franklin d'envoyer un rappel par email à tous les clients qui ont du retard. Définir la vue qu'il a besoin pour effectuer cette tache. L'email envoyé à chaque client doit contenir le titre du film qu'ils n'ont pas rendu, ainsi que le nombre de jours de leur retard.

8. Utiliser la vue créée au point 7 pour afficher le nombre de clients ayant plus de trois jours de retard.

9. On aimerait demander à Franklin de calculer le nombre de location par client. Définir la vue qu'il a besoin pour effectuer cette tache. Donner la requête pour afficher les 20 premiers clients avec plus de location en utilisant la vue définie.

10. Créer une vue pour calculer le nombre total de location par jour.
    Combien de locations sont effectués en 1er août 2005 ? Donner la requête SQL.

Procédures/Fonctions stockées

Vérifier la syntaxe MySQL sur http://dev.mysql.com/doc/refman/5.6/en/create-procedure.html.

11. Créer une fonction qui calcule le nombre de films proposés par le magasin. La fonction prend en entrée l'ID de magasin et retourne en sortie le nombre de films proposés par le magasin. Utiliser la fonction pour afficher le nombre de films proposés par magasin 1 et 2. Confirmer la réponse avec une requête SQL.

12. Créer une fonction qui calcule le nombre de clients par le magasin. La fonction prend en entrée un ID de magasin et retourne en sortie le nombre de clients par le magasin. Utiliser la fonction pour afficher le nombre de clients par magasin 1 et 2. Confirmer la réponse avec une requête SQL.

13. Créer une fonction qui calcule le revenu de chaque magasin. La fonction prend en entrée l'ID de magasin et retourne en sortie le revenu total du magasin. Utiliser la fonction pour afficher le revenu de magasin 1 et 2. Confirmer la réponse avec une requête SQL.

14. Créer une procédure pour mettre à jour la date de la mis à jour de tous les films à la date d'exécution de la procédure. Quelle est la date de la mis à jour des films avant et après cette procédure ?

15. Créer une procédure qui calcule le nombre d'exemplaires physiques des films ainsi que le nombre de locations dans chaque magasin. Utiliser la procédure pour afficher le résultat de magasin 1 et 2.