SQL - TP 1

Pour ce TP, il vous sera demandé de fournir un fichier de script SQL répondant aux différentes questions.

Utilisez le commentaire SQL (--) pour préfixer la ligne où vous indiquerez le numéro de la question.

Une fois terminé, partagez-moi votre script sur https://gist.github.com/ afin que je puisse vous apporter la correction.

Pour les questions 5 et 6, n'hésitez pas à ajouter des commentaires afin d'expliquer votre logique.

1. Commencez par **créer une BDD** "**shop_db**" avec les paramètres par défaut (Moteur de stockage InnoDB et le charset par défaut (DEFAULT COLLATION, en principe UTF-8)

2. **(Facultatif) Créer 2 utilisateurs** avec des droits différents
    - Un utilisateur "admin" qui possède **tous les privilèges** sur la BDD **shop_db**
    - Un utilisateur "developer" qui possède uniquement les privilèges **ALTER, CREATE, DROP, INDEX, UPDATE** sur la BDD **shop_db**

3. **Créer les tables suivantes**
    - Une table **Customers** avec les propriétés suivantes
        - id : INT : la primary key avec incrémentation automatique
        - username : VARCHAR(16) non-nullable et unique
        - email : VARCHAR(255) non-nullable et unique
        - password : VARCHAR(32) non-nullable
        - create_time : TIMESTAMP

    - Une table **Addresses** avec les propriétés suivantes :
        - id : INT : la primary key avec incrémentation automatique
        - road_number : INT
        - road_name : VARCHAR(100) non-nullable
        - zip_code : CHAR(5) non-nullable
        - city_name: VARCHAR(100) non-nullable
        - country_name : VARCHAR(100) non-nullable

    - Une table **Orders** avec les propriétés suivantes :
        - id : INT : la primary key avec incrémentation automatique
        - ref : VARCHAR(45) unique et non-nullable
        - date : DATE non-nullable
        - shipping_cost : DECIMAL(6,2) avec 0.00 pour valeur par défaut
        - total_amount : DECIMAL(6,2) avec 0.00 pour valeur par défaut

Le mot “order” étant un mot réservé en SQL, j’ai choisi de mettre les noms de table au pluriel. Il aurait été possible de l’échapper dans les requêtes en utilisant le symbole backquote ` mais cela aurait pu causer des problèmes.

- Une table **Products** avec les propriétés suivantes
  - ref : CHAR(20) primary key
  - name : VARCHAR (100) non-nullable
  - price : DECIMAL(6,2) non-nullable
  - description : LONGTEXT
  - stock : INT avec 0 pour valeur par défaut

- Une table **Orders_Products** qui est une table associative de type Many-to-Many entre les tables Order et Product
  - ref_product : Foreign key vers table product
  - id_order : Foreign key vers la table order
  - quantity : INT avec 0 pour valeur par défaut

4. **Ajout des relations**

Comme vous le constatez, il manque des relations, notamment entre les tables Customers, Addresses et Orders.

Ce n'est pas forcément le mieux au niveau conception, mais nous voulons créer une relation One to Many entre un client (customer) et une adresse. Un client pourra donc posséder 1 adresse, mais une adresse pourra être partagée par plusieurs clients. Il faudra donc créer une foreign key dans la table customer.

Nous voulons aussi créer une relation One to Many entre une adresse et un order, la commande possédant une et une seule adresse de livraison et le l’adresse pouvant être partagée entre plusieurs commandes.

5. **Ajout d’une table Categories**

Nous souhaitons qu’un Product puisse appartenir à plusieurs catégories de produits.

Une catégorie possède un nom (VARCHAR), une description (LONGTEXT), et peut ou non être associée à une catégorie Parente (parent_category_id).

Créer les différentes requêtes permettant d’implémenter ces modifications.

6. **Mise en place d’un système de gestion des stocks et magasins**

- Les magasins font office d'entrepôt, le stock de chaque produit est donc rattaché au magasin.
- Les magasins ont un nom et une adresse

Proposez le script permettant l’implémentation de cette modification

7. Bonus : Modification de la table Addresses

Si on analyse la table Address, on peut constater qu'elle ne nous permettra pas de respecter les formes normales.

En effet, les attributs concernant la rue, le pays ou la ville vont engendrer de nombreuses duplications de données.

Une bonne pratique pour gérer ce cas est d'extraire ces données dans des tables spécifiques.

Nous allons donc modifier la table Address pour y extraire les données zip_code et city_name dans une table City et la donnée country_name dans une table Country.

Les associations seront les suivantes :

- Une adresse n'est composée que d'une seule ville
- Une ville peut apparaître dans plusieurs adresses.
- Une ville appartient à un seul pays.
- Un pays possède plusieurs villes.