# SQL - TP 1

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

      A l'exécution, il est possible que vous ayez une erreur. En effet, le mot "order" est un mot réservé en SQL. Il est donc nécessaire de l'échapper dans la requête en utilisant le symbole backquote `

    - Une table **Product** 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 **Order_Product** 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. **Modification de la table Address**

   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.

5. **Ajout d'une table Categorie**

   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