Laboratoire 2

Dates de remises : le 17 ou le 18 septembre . Selon le groupe.

Objectifs :

- Introduction à Transact-SQL,
- Écrire et exécuter des procédures stockées
- Expérimenter le concept des transactions

Consignes obligatoires

- Toutes vos procédures doivent être parfaitement identifiées par leur numéro et doivent être enregistrées dans un fichier de nom Laboratoire2.sql.
- Vous devez également remettre un fichier d'exécution identifié par le nom Execution.sql. Ce fichier d'exécution doit contenir toutes vos requêtes EXECUTE et SELECT
- Vous devez le déposer dans la boite à la date et à l'heure indiquée.

Attention !!

Pour ce travail, vous devez utiliser uniquement les concepts vus en classe ou dans le cours de KB6 (Introduction aux bases de données)

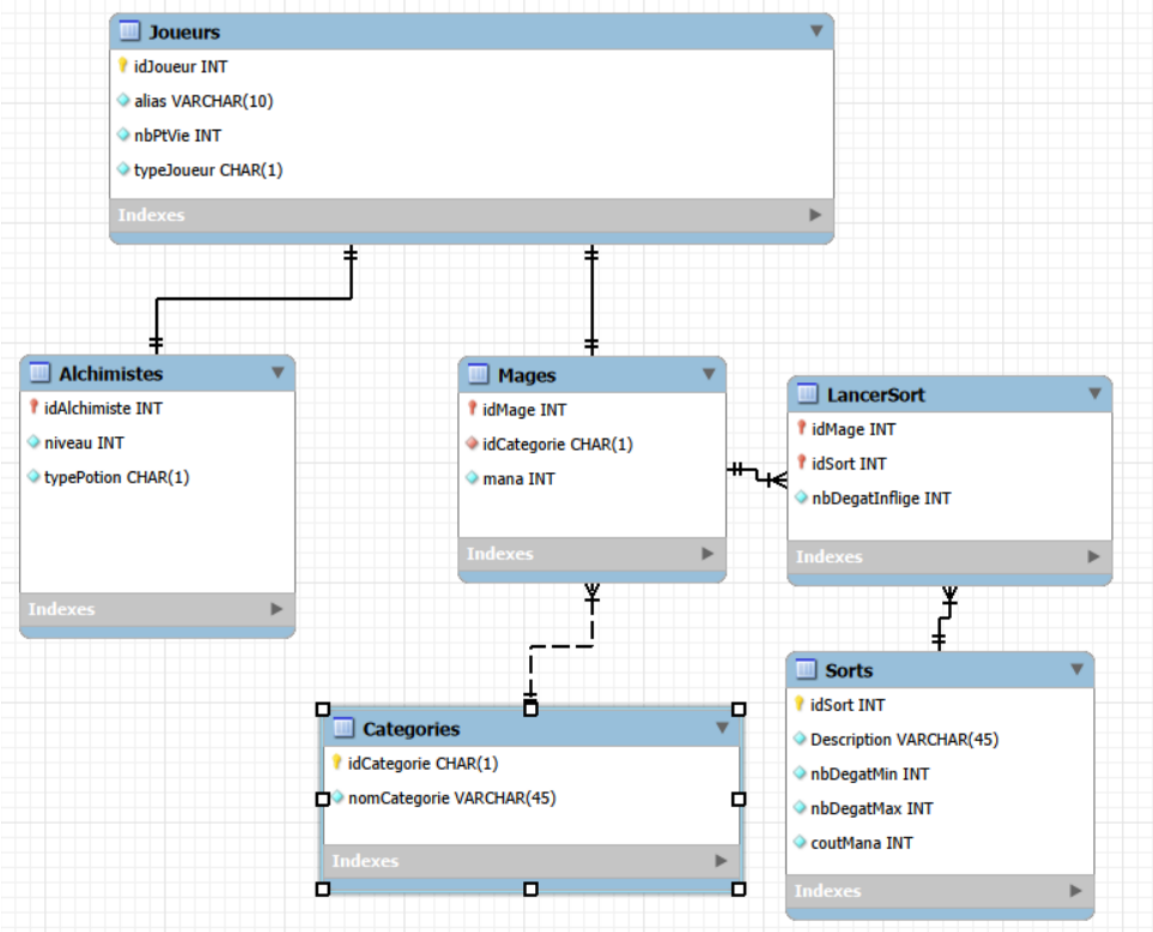
Si vous avez besoin d'utiliser autre que les concepts présentés en classe alors vous devez :

1. **Expliquer le concept avant son utilisation**
2. Donner la source de l'information
3. Demander l'autorisation à l'enseignante ou à l'enseignant

La débrouillardise et l'autonomie sont deux qualités recherchées chez un informaticien. **Le plagiat sous toutes ses formes c'est NON et entraine les sanctions prévues par la PIEA**

Énoncé :

Le modèle suivant représente une partie de la base de données des joueurs d'un jeu de types « magie »



Nous avons au moins deux classes de joueurs : les mages et les alchimistes.

- L'ensemble des joueurs ont des attributs communs comme l'alias, le nombre de points de vie.
- La table « Mages » contient uniquement les attributs en rapport avec la classe « Mages »
- La table « Alchimistes » contient uniquement les attributs en rapport avec les « Alchimistes »
- Les trois tables : Joueurs, Mages et Alchimistes ont la même clé primaire (de noms différents mais c'est le même attribut)
- idMage dans la table « Mages » est une FK issue de la table Joueurs.
- idAlchimiste de la table « Alchimistes » est une FK issue de la table Joueurs.
- Un Mage ne peut pas être un Alchimiste et vice versa.

À chaque fois qu'un mage lance un sort, (une insertion dans la table « LancerSort) on doit vérifier :

1. que le nombre de point de dégât (nbPtDegat) est compris entre le nbDegaMax et nbDegaMin . Ceci pourrait être garanti par un trigger → Laboratoire 3
2. que le mage a le mana nécessaire pour lancer le sort. (mana peut couvrir le coutMana)
3. lorsque le sort est lancé, le mana du Mage diminue du coût du mana, et le nombre de points de vie du joueur(du Mage qui lance le sort) baisse de 2.
4. un mage ne peut pas avoir 0 points de vie sinon il est mort.

Question1 :

1. Lisez le script de création des tables contenu dans ScriptLabo2.sql. Le but est de voir les contraintes d'intégrités et les liens entre les tables.
2. Exécuter le script ScriptLabo2.sql pour créer les tables de la Base de données BDWillow et les différentes tables.
3. Faire les insertions dans chacune des tables. (en utilisant le script)

Question2 :

Écrire les procédures stockées suivantes :

1. Écrire la procédure stockée *ajouterMage* qui permet d'ajouter un mage dans la base de données. Ici il faut faire les insertions dans les deux tables Joueurs et Mages
2. Écrire la procédure stockée ajouterAlchimiste qui permet d'ajouter un Alchimiste à la base de données : Ici il faut faire les insertions dans les deux tables Joueurs et Alchimistes.

Pour les questions 1 et 2 vous devez vous assurer que la transaction (les insertions) soit faite uniquement lorsque toutes les données sont valides. : (TRY… CATCH)

3. Écrire une procédure stockée *ListeJoueurs* (@typeJoueur) qui affiche la liste des joueurs selon leur type des joueurs. Afficher l'alias et le nombre de points de vie, puis pour les mages : afficher leur mana et le nom de leur catégorie et pour les alchimistes afficher leur niveau et le typePotion. Lorsque le typePotion est A, afficher « Attaque », lorsque le typePotion est D, afficher « Défense », lorsque le typePotion est T,
afficher « pour Tout »

Pour la question 3, et pour une clarté du code , on vous demande d'utiliser des vues. Les vues doivent être crées à l'extérieur de la procédure stockées.

4. Écrire la procédure stockée *LancerUnSort* (@idMage ,@idSort,@nbDegaInflige) qui permet à un mage de lancer un sort. (faire une insertion dans la table « LancerSort » et faire les mises à jour nécessaires)
5. Écrire la fonction TABLE *listeMages* qui retourne pour chaque Mage, son alias, son mana, ses point de vie et le nom de sa catégorie.
6. Écrire la fonction scalaire *degatTota*l(@alias) qui retourne le nombre total de points de dégâts infligés par un mage d'un alias donné.
7. Donner le script d'exécution de chacune des procédures et fonctions