Travail pratique no_1
Les chasseurs de reliques

Automne 2020

- Ce travail sera réalisé individuellement.
- À remettre le 22 octobre 2020
- Le travail compte pour 20 % de la note finale.

Objectifs

Ce travail vise l'atteinte des objectifs suivants :

1. Produire un modèle de données normalisé
2. Écrire des procédures stockées
3. Écrire des triggers.

Mise en contexte :

Nous sommes dans un environnement du jeu vidéo de type médiéval, et nous souhaitons développer une application ***jeuRelique*** où chaque joueur pourrait acheter des reliques.

- Les reliques peuvent être des ouvrages, des armes ou des talismans (bijoux).
- Toutes les reliques sont identifiées par un **numéro unique (IDENTITY)**, ont une brève description, un prix unitaire, un flag de disponibilité , une quantité en stock et une quantité stock limite. Le flag est tout le temps à 1 aussi longtemps que la relique n'est pas supprimée. Sinon le flag est égal à 0.
- Les armes ont en plus une efficacité (nombre de points de dommage), un genre (une main ou deux mains ou autre) et une description
- Les talismans ont en plus la matière dont ils sont faits (le métal qui compose le talisman : ivoire, or, argent, bronze etc), le type (bague, bracelet anneau, chaîne etc) et un pouvoir magique.
- Les ouvrages ont en plus un auteur, l'année où l'ouvrage est écrit, et un type d'ouvrage (ouvrages pour jeter des sorts, fabriquer des potions, faire des antidote etc..)

Dans l'environnement du jeu, il y a plusieurs joueurs, chaque joueur peut acheter plusieurs reliques . Des joueurs différents peuvent acheter des reliques identiques.

Un joueur a un numéro **unique**, un alias **unique**, un nom, un prénom et un montant initial en écus.

Tous les joueurs peuvent acheter n'importe quelle relique, à condition qu'il soit disponible en inventaire. La quantité d'une relique achetée peut-être supérieure à 1.

Les achats sont conservés dans un panier. Le panier d'achats contient la quantité achetée de chaque relique pour un joueur donné. Au fur et à mesure que les reliques sont ajoutées dans le panier, l'inventaire de la relique est mis à jour. Les reliques s'accumulent dans le panier jusqu'au moment où le joueur passe à la caisse pour payer le panier. Au moment de payer le panier le solde en écu du joueur est déduit du montant total des achats. Après que le joueur ait payé le conteneur du panier, le panier est supprimé.

Votre base de données doit-être conçue de sorte que l'on puisse conserver l'historique des achats des joueurs. Cet historique contient le contenu des paniers d'achats payés. On a pour chaque historique d'achats un numéro unique, la date à laquelle les achats ont été payés ainsi que la liste des reliques achetés .

C'est au moment de payer le panier que l'historique des achats est mis à jour avec le contenu du panier.

Questions :

Conception de la base de données :

1. Donner le modèle relationnel de votre BD en 3FN. La remise du modèle a lieu le 13 octobre 2020.
2. Créer la base de données : BDReliquesVosInitiales : Exemple BDReliquesSY.
3. Créer toutes les tables de la base de données.
4. Peupler votre base de données avec des données initiales.

Exploitation de la base de données : Procédures stockées et triggers.

Groupe 1 :

Écrire les procédures stockées suivantes pour la gestion des Reliques.
(Toutes les procédures et fonctions doivent-être exécutées et testées. Vos données doivent être cohérentes en tout temps)

1. Une procédure ***ajouterArme***, cette procédure permet d'ajouter une arme dans la base de données. Insérer toutes les informations d'une arme.

2. Une procédure ***ajouterTalisman***, cette procédure permet d'ajouter un talisman dans la base de données. Insérer toutes les informations d'un talisman.

3. Une procédure ***ajouterOuvrage***, cette procédure permet d'ajouter un ouvrage dans la base de données. Insérer toutes les informations de l'ouvrage

4. Une procédure ***afficherReliques*** cette procédure permet d'afficher les reliques selon le **type de relique**. Afficher toutes les informations des reliques. Exemple pour une arme, nous allons afficher les informations (nom, quantité en stock, le prix unitaire, le nombre de points de dommage et le genre). Pour cette question, il est conseillé de créer des **vues** que vous allez utiliser dans votre procédure. Le code sera plus propre.

5. Une fonction ***prixMoyenReliques*** qui retourne sous forme de table le prix moyen de chaque type de relique.

6. Une procédure ***supprimerRelique,*** qui permet de supprimer une relique étant donné un numéro de la relique. La suppression d'une relique implique la mise à jour du flag de disponibilité à 0. Une relique supprimée qui est contenu dans des paniers doit aussi être supprimé.

Groupe 2

Écrire les procédures stockées suivantes pour la gestion des Achats.
(Toutes les procédures et fonctions doivent-être exécutées et testées. Les transactions doivent être cohérentes)

1. Une fonction **montantPanier** qui retourne le montant total du panier d'achats d'un joueur étant donné son **alias**. S'il n'y a pas de panier pour le joueur, la fonction retourne 0;

2. Une procédure ***ajouterReliquePanier*** qui permet d'ajouter une relique au panier avec une quantité donnée pour un joueur étant donné son **alias**.

    - Si la quantité en inventaire de la relique est insuffisante, l'ajout est annulé.
    - Si le solde du joueur est insuffisant pour couvrir le montant total du panier plus l'ajout de la relique, l'ajout est annulé.
    - La procédure ajoute la relique au panier et prend soin de mettre à jour l'inventaire de la relique.
    - Si la relique existe déjà dans le panier du joueur, la quantité est mise à jour uniquement.

3. Une procédure ***modifierReliquePanier*** qui permet de modifier la quantité achetée d'une relique dans le panier d'achats d'un joueur étant donné son **alias**.

    - Si la quantité en inventaire est insuffisante pour couvrir la nouvelle quantité, en tenant compte du retour, la modification est annulée;
    - La procédure met à jour la quantité et prend soin de mettre à jour l'inventaire de la relique.

4. Une procédure ***supprimerReliquePanier*** qui permet de supprimer une relique du panier d'achats d'un joueur étant donné son **alias**.

    - La procédure supprime la relique du panier et prend soin de mettre à jour l'inventaire de la relique.

5. Une fonction **table *afficherPanier*** qui retourne le contenu du panier d'un joueur. La fonction retourne sous forme d'une table le nom de la relique et son type (**'Arme', 'Talisman', 'Ouvrage'**), la quantité et le prix total.

6. Une procédure ***payerPanier*** qui permet de payer le panier d'achats d'un joueur étant donné son **alias**.
    - Mettre à jour le solde du joueur avec le montant du panier.
    - Mettre à jour l'historique des achats du joueur.
    - Supprimer le panier d'achats du joueur;
7. Une procédure ***SupprimerPanier*** qui permet de supprimer le panier d'achats d'un joueur étant donné son **alias**;
    - Remettre en inventaire la quantité achetée de chaque relique contenue dans le panier du joueur;
    - Une fois l'inventaire mis à jour, on supprime le panier d'achats du joueur;
8. Une fonction **table *afficherAchats*** qui retourne la liste de tous les achats payés étant donné **l'alias** d'un joueur. La table retournée contient l'alias du joueur, le nom de la relique, son type ('Arme', 'Talisman', 'Ouvrage') et la quantité.

Groupe 3, écrire les triggers suivants :

1. Le trigger ***beforeInsertTalisman*** qui permet de garantir qu'un numéro inséré dans la table Talismans ne sera pas utilisé (inséré) dans la table Armes ni dans la table ouvrages.
2. Un trigger ***beforeInsertArme*** qui permet de garantir qu'un numéro inséré dans la table Armes ne sera pas utilisé (inséré) dans la table Talismans ni la table ouvrages.
3. Un trigger ***updatestockRelique*** qui permet d'augmenter la quantité en stock d'une Relique lorsque la quantité limite est atteinte. Ce qui veut dire si la quantité en stock ne doit pas être plus petite que la quantité limite. On doit augmenter la quantité du triple de la quantité limite.
4. Le trigger ***cascadeDeleteRelique*** qui fait la suppression en cascade d'une relique qui n'est pas dans l'historique des achats. Les reliques supprimées qui sont contenus dans des paniers doivent aussi être supprimés.

5. Tester TOUS VOS triggers par les requêtes DML appropriées.

Ce qu'il faut remettre dans un dossier Zippé portant votre nom :

1. Le modèle de la base de données en PDF;
2. Un script SQL contenant la création des tables;
3. Un script SQL contenant les procédures et les fonctions stockées;
4. Un script SQL contenant TOUS les triggers;
5. Un script SQL contenant les requêtes d'exécution des procédures, des fonctions et de la vérification du déclenchement des triggers;

Évaluation

Élément évalués	Pondération
Groupe 1 (3*6)	18
Groupe 2, toutes les questions sur 4	32
Groupe 3, chaque trigger sur 4 points	16
Modèle de données	10
Script d'Exécution	10
Création des tables avec les contraintes	10
La base de données est bien peuplée	4
Total	100