Instructions générales

1. Les instructions générales du projet de SQL restent d'application pour l'examen.
    - Utilisez un schéma nommé **examen**.
    - Pour toute fonctionnalité demandée :
        - Si c'est possible, implémentez-la sur base de contraintes d'intégrité.
        - Sinon, si c'est possible, implémentez-la sur base d'un trigger.
        - Sinon, si c'est possible, implémentez-la sur base d'une vue.
        - Sinon, si c'est possible, implémentez-la sur base d'une procédure stockée.
    - Utilisez le modèle transactionnel et évitez d'ignorer silencieusement les erreurs.
    - Attention aux injections de SQL.
    - Chaque PreparedStatement ne doit être préparé qu'une seule fois.
    - Il ne faut, en Java, faire des SELECT **que** sur une seule chose à la fois. Cette chose peut être une table, une vue ou une procédure stockée.
2. Vous n'avez pas accès à Internet, mais vous trouverez dans le répertoire Z: une copie du contenu du cours qui se trouve sur moodle.
3. Vous devez créer :
    - un fichier script.sql contenant toutes les commandes SQL à exécuter au serveur ;
    - un projet Eclipse contenant la ou les classes Java.
4. 15 minutes avant la fin de l'examen :
    a. Vous recopiez le fichier script.sql à la racine de votre projet Eclipse.
    b. Vous créez un fichier NOM_PRENOM.zip (par exemple DAMAS_CHRISTOPHE.zip) contenant l'intégralité de votre projet Eclipse.
    c. Vous copiez ce fichier à la racine du disque Z:.
    d. Finalement vous vérifiez que ce fichier contient bien le projet Eclipse contenant le fichier script.sql et le répertoire src avec vos sources Java.

ATTENTION PGADMIN III PLANTE PARFOIS : SAUVEZ SOUVENT !

Enoncé

Une société désire développer un système de réservations de tickets de concert. Pour éviter la revente illégale de tickets, les clients ne pourront pas réserver plus de 4 tickets par concert. De plus, les clients ne pourront pas réserver des tickets pour deux concerts se déroulant à la même date.

Tables

Il y aura trois tables : concerts, clients et reservations.

Pour un concert, on retiendra sa date, le nom de l'artiste, le nom de la salle ainsi que le nombre de tickets mis en vente. Pour chaque concert, il faut également retenir s'il est complet ou non, c.-à-d. si le nombre total de tickets réservés est égal au nombre de tickets mis en vente. Attention, dans le cadre de cet examen, vous ne pouvez pas avoir de champ contenant le nombre total de tickets réservés dans la table concerts. Pour connaître le nombre total de tickets réservés, il faudra faire une requête.

Pour un client, on retiendra son nom, son prénom et son sexe ('M' ou 'F').

Pour une réservation, on retiendra le concert, un numéro de réservation, le client et le nombre de tickets réservés. Pour un concert particulier, la première réservation aura comme numéro de réservation '1', la deuxième aura comme numéro '2', … Il y aura donc plusieurs réservations avec le même numéro de réservation. Par contre, il ne peut y avoir deux réservations pour le même concert avec le même numéro de réservation.

À développer :

- Il faut créer une procédure stockée pour réserver des tickets pour un concert. Cette procédure prend 3 paramètres : le client, le concert et le nombre de tickets demandés. La méthode renverra vrai si le concert est désormais complet, faux sinon. La procédure lance un message d'erreur dans les cas suivants :
  - Lorsque le client a déjà réservé des tickets pour un autre concert se déroulant à la même date.
  - Lorsque le nombre de tickets demandés est plus grand que le nombre de tickets encore disponibles pour ce concert.
  - Lorsque le nombre total de tickets réservés par le client pour le concert est supérieur à 4. Exemple : si un client a déjà réservé 2 tickets pour un concert et qu'il désire réserver de nouvelles places, il pourra au maximum en réserver 2.
- Il faut créer un mécanisme qui permet de mettre à jour le champ de la table concerts qui retient si un concert est complet ou non.
- Une classe Java qui se connecte au serveur : Le programme demande à l'utilisateur un nom d'artiste (via un Scanner*) et affichera ensuite tous les concerts de cet artiste, triés par date, à la console. Pour chaque concert, il affichera la date, le nom de la salle et le nombre total de tickets réservés. Les concerts qui n'ont pas encore de réservations auront donc 0 comme nombre total de tickets réservés. Il n'est pas nécessaire de gérer les cas d'erreurs et il ne faut pas perdre de temps à afficher les données de manière jolie.

  * pour rappel, voici deux instructions utiles pour les Scanner :
  Scanner s = new Scanner(System.in); String artiste = s.nextLine();

Si vous voulez tester votre système, vous pouvez insérer des concerts et des clients (via de simples INSERT) et appeler la procédure stockée de réservation dans votre script.