PL (ProceduraL) /SQL

1 Préliminaires

1.1 Tables

On considère ici les tables suivantes :

- ETUDIANT=(ID_ETUDIANT, NOM_ETUDIANT, TUTEUR) qui, pour un étudiant identifié de manière unique par ID_ETUDIANT donne, quand il est désigné, le nom de l'enseignant qui est son tuteur de stage.
- ENS=(NOM_ENS, NB_ETU) qui donne pour chaque enseignant le nombre d'étudiants qu'il encadre en stage.

1. Créez ces tables, avec les contraintes d'intégrité qui vous paraissent appropriées, ainsi qu'une séquence qui sera utilisée pour générer des identifiants pour les étudiants.

1.2 PL/SQL et SQL*Plus

Dans SQL*Plus, vous devez commencer par executer la commande suivante :

```
SET serveroutput ON
```

Ceci vous permettra d'afficher des informations à l'écran depuis vos programmes PL/SQL, en utilisant :

```
DBMS_OUTPUT.PUT_LINE
```

1. Ecrivez un bloc PL/SQL qui *declare* une variable x, lui *affecte* la valeur '**Hello World!**' et *affiche* son contenu à l'écran.
2. Executez-le dans SQL*Plus. Comment le réexecuter sans avoir à le redéfinir ?
3. En cas d'erreur, utilisez l'ordre SQL*Plus : SHOW ERRORS.

2 Blocs, fonctions et procédures stockées.

Nous testons ici la possibilité offerte par le language PL/SQL de définir des fonctions et des procédures stockées et de les appeler dans des blocs PL/SQL.

1. Ecrivez une *procédure stockée* PL/SQL **Nouvel_Etudiant** prenant comme paramètre le nom d'un étudiant et l'insérant dans la table ETUDIANT, mais sans lui attribuer de tuteur. Quel est le type le plus adapté pour la déclaration du paramètre ? Testez cette procédure dans un bloc PL/SQL.
2. Ecrivez et testez de même une *fonction stockée* PL/SQL **Nb_Etudiants** prenant comme paramètre un nom d'enseignant et calculant à partir des données de la relation ETUDIANT le nombre d'étudiants qu'il encadre.

3. Manuellement, insérez des enseignants et affectez-leur des étudiants. Testez la fonction Nb_Etudiants dans un bloc PL/SQL, en affichant ses résultats à l'écran.
4. Déconnectez-vous de la base de données, reconnectez-vous et renouvelez les tests des questions précédentes. Qu'en concluez-vous ?
5. Vous pouvez consulter à partir de maintenant les tables systèmes concernées :

```
USER_SOURCE(NAME, TYPE, LINE, TEXT)
USER_ERRORS
```

3 Triggers et contraintes

Nous allons voir maintenant comment utiliser les triggers pour s'assurer que des contraintes complexes restent vérifiées par la base. C'est d'ailleurs l'une des utilisation principales des triggers. Ici, on veut s'assurer que le nombre d'étudiants encadrés par un tuteur de la table ENS est cohérent avec les données de la table ETUDIANT.

1. Créez un trigger qui, lors de l'enregistrement du tuteur d'un étudiant (jusqu'alors sans tuteur) dans la table ETUDIANT, met à jour automatiquement dans la table ENS le nombre d'étudiants que cet enseignant encadre en stage.
2. Créez plus généralement un trigger qui, lors de toute insertion / modification / suppression d'affectation de tuteur dans la table ETUDIANT, met à jour l'attribut NB_ETU dans la table ENS.
3. Testez tous les cas de figure possibles et assurez-vous que votre trigger remplit bien sa fonction.

4 Packages

Les packages permettent de regrouper des fonctions et procédures apparentées dans une seule unité et de contrôler leur visibilité. Par exemple DBMS_OUTPUT est un package.

1. Vous allez créer un package permettant de gérer les étudiants et leurs tuteurs, en proposant les opérations suivantes :
   (a) Ajouter un nouvel enseignant/un nouvel étudiant (Vous savez déjà faire). Gérez les erreurs qui peuvent survenir.
   (b) Affecter un enseignant sans stagiaire comme tuteur d'un étudiant sans tuteur.
   (c) Donner le tuteur d'un étudiant dont on donne le nom ou le numéro.
   (d) Incrémenter ou décrémenter l'attribut NB_ETU d'un enseignant
2. Modifier le trigger de la question 2 en réutilisant les opérations de votre package.

5 Curseurs

Les curseurs permettent de gérer des opérations (sélections, mises à jour) sur des ensembles de n-uplets.

1. Créer une procédure stockée PL/SQL qui affiche à l'écran la liste des étudiants encadrés par un tuteur donné.
2. Créer une procédure stockée PL/SQL de "consolidation", dont le rôle est de s'assurer que chaque tuteur a un nombre d'étudiants cohérent avec la table ETUDIANT.