TP PL/SQL

(Énoncé)

Environnement de travail

Le TP se déroule en ligne de commande, sous Linux sur le serveur **delos.cnam.fr**. Pour se connecter il y a deux possibilités :

A. Vous êtes au CNAM : dans ce cas connectez-vous sous une machine Linux avec votre login et mot de passe CNAM. Ensuite ouvrez un terminal en ligne de commande (par exemple 'konsole' dans le menu KDE) et connectez-vous au serveur **delos.cnam.fr** (toujours avec votre login et mot-de-passe CNAM) avec la commande **ssh** :

```
ssh -Y nom_utililisateur@delos.cnam.fr
```

Vous êtes dans un terminal en ligne de commande connecté sur **delos.**

B. Vous êtes à l'extérieur du CNAM : Il faut se connecter d'abord sur la passerelle d'entrée **vlad.cnam.fr**. Si vous êtes sous Windows, utilisez PUTTY pour faire une connexion par **ssh** sur **vlad.cnam.fr**. PUTTY se trouve ici :

```
http://the.earth.li/~sgtatham/putty/latest/x86/putty.exe
```

Si vous êtes sous Linux ou Mac, ouvrez un terminal en ligne de commande et connectez-vous par **ssh** d'abord sur **vlad** :

```
ssh -Y nom_utililisateur@vlad.cnam.fr
```

Une fois sur **vlad.cnam.fr**, connectez-vous au serveur **delos.cnam.fr** (toujours avec votre login et mot-de-passe CNAM) avec la commande **ssh** :

```
ssh -Y nom_utililisateur@delos.cnam.fr
```

Vous êtes dans un terminal en ligne de commande connecté sur **delos.**

Attention : le -Y dans la commande **ssh** est en majuscule (Linux est sensible à la casse). Après la connexion vous vous retrouvez dans un terminal en ligne de commande (il n'y a pas d'interface graphique). Toutes les commandes sont données à la main et elles affichent leurs résultats dans le terminal. Pour ceux qui n'ont pas utilisé Linux depuis un moment, il est utile de réviser les principales commandes : **cd** (change directory), **ls** (list directory content), **rm** (remove file), **mkdir** (make directory), **rmdir** (remove directory) etc.

Important : on utilise deux sessions en ligne de commande connectées sur delos : une pour éditer et sauvegarder les fichiers SQL et l'autre pour l'interpréteur SQLPLUS, qui est un utilitaire en ligne de commande d'Oracle qui permet aux utilisateurs de se connecter à une base de données et d'exécuter interactivement des commandes SQL et des scripts PL/SQL.

Commencez donc par ouvrir deux sessions en ligne de commande sur **delos** en suivant la procédure décrite en haut.

Dans la 1ere session en ligne de commande : On suppose que les travaux PLSQL se dérouleront dans un répertoire NFA011 (pour ne pas les mélanger avec tous vos autres fichiers). Si c'est votre première connexion pour ce TP, créez un répertoire NFA011 et placez-vous dedans :

```
mkdir NFA011
cd NFA011
```

Si ce n'est pas votre 1ere connexion pour ce TP, ou si le répertoire NFA011 existe déjà, il suffit de se placer dans ce répertoire :

```
cd NFA011
```

Encore une fois : attention à la casse, NFA011 est en majuscules.

Cette 1ere session en ligne de commande est utilisée pour éditer des fichiers SQL et exécuter de commandes Linux. Le code SQL pour chaque exercice de TP doit être placé dans un fichier SQL (par exemple : ex01.sql pour l'exercice 1). Pour éditer des fichiers utilisez l'éditeur **gedit.** Par exemple, pour créer et/ou éditer le fichier 01.sql donnez la commande :

gedit ex01.sql &

Attention : n'oubliez pas le caractère **&** après la commande, il permet de lancer une commande en tache de fond, et vous donne la main sur le terminal pour pouvoir donner d'autres commandes.

Si vous êtes sous Microsoft Windows l'éditeur **gedit** ne marchera pas, car il a besoin d'un système graphique X-Windows pour fonctionner. Dans ce cas, utilisez un éditeur en ligne de commande, par exempla **nano** :

nano ex01.sql

Attention : dans ce cas ne mettez pas la caractère & après la commande, sinon votre éditeur se lancera en tache de fond.

Préparation de la session SQLPLUS

Dans la 2ere session en ligne de commande : Si c'est votre 1re connexion pour ce TP, exécuter les commandes suivantes :

```
cd NFA011
wget http://cedric.cnam.fr/~ferecatu/NFA011/setOracle.sh
```

Si ce n'est pas votre 1ere connexion pour ce TP il suffit de se placer dans le répertoire NFA011 :

```
cd NFA011
```

Ensuite utilisez votre login et mot-de-passe pour vous connecter à la base de donnée :

```
source setOracle.sh
rlwrap sqlplus
```

Le script **setOracle.sh** crée plusieurs variables d'environnement necessaires pour la connexion à la base de donnés. Attention : le mot de passe utilisé pour les comptes ORACLE est celui envoyé par le CNAM, même si vous l'avez changé entre temps (le CNAM ne peut pas connaître votre nouveau mot de passe, car il est crypté par le système). Si votre login et mot de passe Unix ne marchent pas, contactez votre enseignant.

Important : Il faut exécuter la commande **'source setOracle.sh'** avant de lancer **sqlplus** chaque fois que vous ouvrez un nouveau terminal sur **delos** (à chaque connexion).

`rlwrap` est un utilitaire qui permet d'avoir un retour d'historique des commandes sur les flèches du clavier. Une fois connecté, vous vous retrouvez dans l'invité de commandes SQLPLUS, où on donne des commandes et on exécute des fichiers sources SQL.

Pour résumer : La 1ere session connectée à **`delos`** est utilisée pour éditer les fichiers SQL, pour ne pas entrer le code directement à l'invite de commandes, ce qui risque d'être fastidieux, car il y a toujours des erreurs à corriger. Il est recommandé d'utiliser des noms de fichiers qui correspondent aux exercices du TP, par exemple, **`ex01.sql`** pour l'exercice 1 du TP. La 2eme session connectée à **`delos`** est utilisée pour la connexion à la base de données avec l'utilitaire **sqlplus**.

Vous êtes prêt(e) à commencer le TP.

Commandes SQLPLUS à connaître :

CONNECT/DISCONNECT
DROP TABLE nomTable PURGE
DROP PROCEDURE nomProcedure
SET SERVEROUTPUT ON – active affichage à l'intérieur des procédures et fonctions
@nomFichier – execute nomFichier.sql
SHOW ALL – affiche tous les paramétrés internes SQLPLUS
SHOW ERRORS – affiche les erreurs
DESCRIBE nomTable
SELECT table_name FROM USER_TABLES ;– affiche les tables crées par l'utilisateur
SELECT * FROM USER_SOURCE ;
SELECT * FROM USER_OBJECTS ;

Regardez de plus près les commandes en gras. Pour exécuter un script SQL utilisez '@' sous **sqlplus**. Par exemple, pour exécuter le code de l'exercice 1 après l'avoir écrit dans le fichier **ex01.sql** utilisez

@ex01.sql ;

Si la commande ne s'exécute pas ajouter un '/' suivi par la touche entrée.
Si une fonction ou procédure n'affiche rien lors d'une instruction DBMS_OUTPUT, exécuter avant la commande '**SET SERVEROUTPUT ON**'.

Le principe est d'écrire/éditer le code source dans un fichier SQL, exécuter le fichier avec '**@nom_fichier.sql ;**' visualiser les erreurs avec '**SHOW ERRORS**', les corriger et ré-itérer la boucle autant de fois que nécessaire, jusqu'à quand le code s'exécute sans erreur.

Tables nécessaires pour le TP :

Pour générer les tables nécessaires au bon déroulement du TP exécutez les commandes suivantes (vous êtes déjà dans le répertoire NFA011) :

1ere session :
`wget http://cedric.cnam.fr/~ferecatu/NFA011/tp_tables.sql`

2eme session :
`@tp_tables ; – execute le fichier à l'invite SQLPLUS`

Pour ceux qui ne sont pas familiers avec le système Linux voici une liste des commandes de base :
`ls, cd, mkdir, rmdir, pwd, less, cat, more, cp, mv, find, ps, kill, killall`
Exécuter **`man 1 nom_commande`** pour lire la page de manuel de chaque commande.

Liens outils :
http://cedric.cnam.fr/~ferecatu/NFA011/ED_Corrige_PLSQL.pdf

Exercices PLSQL

Exercice 1. Expliquez les résultats obtenus par l'exécution du programme PL/SQL suivant :

```
DECLARE
    n NUMBER := 10;
    PROCEDURE etudeNocopy
        (n1 IN NUMBER, n2 IN OUT NUMBER, n3 IN OUT NOCOPY NUMBER) IS
    BEGIN
        n2 := 20;
        DBMS_OUTPUT.PUT_LINE(n1);
        n3 := 30;
        DBMS_OUTPUT.PUT_LINE(n1);
    END etudeNocopy;
BEGIN
    etudeNocopy(n, n, n);
    DBMS_OUTPUT.PUT_LINE(n);
END;
```

Exercice 2. Exécutez sous SQL*PLUS les solutions des exercices 1, 2, 3, 4, 5, 6 et 7 de la partie ED. Si des tables sont nécessaires il faut aussi les créer.

Exercice 3. Écrire une fonction PL/SQL qui retourne le n-ème numéro de Fibonacci :
$x_n = x_{n-1} + x_{n-2}$ où $x_0 = 1$ et $x_1 = 1$
Écrire une version récursive et une version itérative.

Exercice 4. Écrivez un programme PL/SQL qui affiche tous les vols arrivant à Paris. Utilisez un curseur.

Exercice 5. Écrivez un programme PL/SQL qui calcule la moyenne des trois plus grands salaires de pilote.

Exercice 6. Écrivez un déclencheur qui vérifie pour chaque nouveau vol inséré dans la table VOL que la ville de départ est bien différente de la ville d'arrivée. Si c'est le cas, on affiche un message d'erreur et l'insertion ne doit pas être faite.

Exercice 7. Exécutez sous SQL*PLUS les solutions des exercices 18, 19, 20 de la partie ED. Si des tables sont nécessaires il faut aussi les créer.

Exercice 8. Créez une nouvelle table VOLG qui, en plus de la table VOL, a un attribut de plus : OuestEst qui vaut vrai si un vol se déroule géographiquement de l'ouest vers l'est sur la carte et faux autrement. Cette table nous permettra de trouver de vrai tours du monde dans les exercices suivants. Peuplez cette table avec les mêmes vols que la table VOL mais avec la mis a jour correspondante.

Exercice 9. Reprendre l'exercice 20 de l'ED mais en utilisant la table VOLG crée dans l'exercice précédent : VOLG(Numvol, Heure_départ, Heure_arrivée, Ville_départ, OuestEst)
Écrivez une procédure PL/SQL capable de faire des propositions de tours du monde, prenant en entrée la ville de départ (qui est aussi la destination finale) et deux bornes (supérieure et inférieure) pour le nombre d'escales. Dans ce cas il n'y a pas de liste prédéfinie d'escales et on ne s'intéresse pas à la durée des escales. La procédure doit afficher les vols pour chaque tour du monde proposé. Utiliser une procédure ou fonction récursive. Un tour du monde valide contient que des vols qui se déroulent dans la direction ouest vers est.