PLSQL

Le langage PL/SQL (Procedural Language /SQL) est une extension du langage SQL qui offre un environnement procédural au langage SQL. Les fonctionnalités de PL/SQL sont les suivantes :
- Définition de variables, Traitements conditionnels, Traitements répétitifs, Traitements des curseurs, Traitements des erreurs
Les programmes PL/SQL sont organisés et sont interprétés en blocs. Un bloc est un ensemble de commandes, il est structuré en trois sections comme suit :

```
--BLOC PLSQL
DECLARE
/* Déclaration des variables, des types, des curseurs, fonctions et procédures */
BEGIN
/* Instructions PLSQL ; toute instruction est terminée par ; */
EXCEPTION
/* Traitement des erreurs */
END;
- -  Fin du bloc PL/SQL
```

Remarque :
Le traitement des erreurs se fait en initialisant une variable de type EXCEPTION et ensuite l'utiliser dans la partie EXCEPTION.
Exemple :
Afficher les noms des produits par rang ensuite afficher le nombre des produits existant.

```
DECLARE
        cursor cr is select distinct refprod, nomprod
                        from produit
                        order by refprod, nomprod ;
        i integer;
        vide EXCEPTION;
BEGIN
        i:=1;
        for item in cr loop
                dbms_output.put_line('Le produit N°' ||i||' est '||item.refprod||' '||item.nomprod);
                i:=i+1;
        end loop;

        if (i<2) then
                 RAISE vide;
        else
                i := i-1;
                dbms_output.put_line('La base de données contient ' ||i||' produits ');
        end if;
EXCEPTION
        WHEN vide THEN dbms_output.put_line('La base de données ne contient aucun produit');

END;
/
```

Pour afficher un texte vous utilisez le package DBMS_OUTPUT. Pour rendre les affichages visibles dans SQLPLUS, il faut utiliser la commande suivante : **SET SERVEROUTPUT ON**

Fonctions et procédures

Le code PlSQL peut être sauvegardé dans une procédure ou fonction avec ou sans paramètres.

```
CREATE [OR REPLACE] PROCEDURE Nom_de_procedure (arg1 type, arg2 type, …) IS
Declaration de variables locales
BEGIN
Instructions;
END;
```

Pour exécuter une procédure :

```
SQL> EXECUTE Nom_de_procedure(valeurs des arguments);
```

Remarque : pour voir les erreurs syntaxiques commises lors de la déclaration une procédure, il faut utiliser l'instruction :

show errors

Questions

Supposons que les tables des TP précédents sont crées et remplies.

1. Écrire un bloc PL/SQL qui affiche la *culture_principale* et la *taille_exploitation* de l'agriculteur **'Benali Ahmed'**. Si taille_exploitation est supérieure à 10 hectares, afficher : "Grande exploitation" sinon afficher "Petite exploitation".

   Exemple : '**Benali Ahmed** a une **Petite** exploitation (**5** hectars) de **tomates**'

   Gérer l'exception NO_DATA_FOUND et afficher : "Aucun agriculteur trouvé avec ce nom."

2. Refaire la question 1 **pour** l'ensemble des agriculteurs. Utiliser un curseur qui parcourt la table Agriculteur.

3. Créer un bloc PL/SQL qui :
   a. insère une nouvelle ligne dans la table Production (6, 4, 3, 10000, 'hiver 2024') .
   b. Met à jour la quantité_produite de la production 3 à 15000.
   c. Supprimer de la table Approvisionnement toutes les lignes dont la date_approvisionnement est antérieure à '01-03-2024'

4. Créer une fonction total_production_agriculteur qui retourne la somme totale des quantités produites pour un agriculteur donné. Afficher un message d'erreur si l'agriculteur mentionné n'existe pas. Exécuter la fonction pour tous les agriculteurs.

5. Créer une procédure qui permet d'insérer **une production** à partir de tous les attributs nécessaires. N'oublier pas de vérifier l'unicité de la clé et l'existence de clé étrangère vers **les tables référencées**. Affichez les messages d'erreurs en cas de problèmes.