ECOLE MAROCAINE DES SCIENCES DE L'INGENIEUR

Membre de HONORIS UNITED UNIVERSITIES

École Marocaine des Sciences de l'Ingénieur

Filière: IIR

Les Bases de Données

Prof. Zakaria KHATAR

PL/SQL : Procedural Language for SQL

Langage Procédural pour le Langage de Requête Structuré

1) Introduction sur langage procédural PL/SQL :

PL/SQL (**Procedural Language/Structured Query Language**) est un langage de programmation conçu pour être utilisé avec le système de gestion de base de données Oracle. Il permet de créer des programmes qui gère les instructions SQL de manière plus avancée que ce que permet le SQL seul.

PL/SQL offre de nombreuses fonctionnalités de programmation procédurale, comme :

- **La déclaration de variables**
- **Les boucles et les conditions**
- **Les déclencheurs et les curseurs**
- **La gestion des erreurs…etc.**

2) Blocs et sections de PL/SQL :

Un bloc de PL/SQL est une unité de code PL/SQL qui peut être exécutée indépendamment dans la base de données Oracle. Un bloc de PL/SQL peut inclure une ou plusieurs instructions PL/SQL, ainsi que des déclarations de variables et des exceptions.

Un bloc PL/SQL est défini par la clause BEGIN et se termine par END. Voici à quoi ressemble :

```
DECLARE      -- mes déclarations de variables
BEGIN        -- début de la section exécutable
     Les requêtes SQL et PL/SQL
END;         --la fin du bloc exécutable
```

DECLARE n'est pas **OBLIGATOIRE**

2-1) Section déclaration DECLARE :

Cette section contient la description des variables utilisées dans le bloc. Il s'agit d'une section **facultative** **(n'es pas obligatoire)** qui commence par la clause **DECLARE**.

Pour affecter une valeur à une variable, on utilise la syntaxe suivante :

```
DECLARE
        v_nom-variable type de données := Valeur;
```

Pour demander à l'utilisateur de saisir une valeur et puis l'affecter à une variable, on utilise la syntaxe suivante :

```
DECLARE
        v_nom-variable type de données := &v_nom-variable;
```

2-1) Section déclaration DECLARE :

Il est recommandé d'utiliser un préfixe comme **v_** devant chaque nom de variable pour le différencier des autres noms de colonne, comme : **v_nom, v_prenom**, **v_salaire**….

2-1) Section déclaration DECLARE :

Exemple 1 :

Le bloc suivant déclare les variables : **v_nom, v_salaire** et **v_adresse** puis leur affecte les valeurs suivantes : 'Ahmed' et 10000.

Dans la section **BEGIN - END,** on peut affecter des valeurs aux variables comme monter l'exemple suivant avec la variable **v_adresse.**

```
DECLARE
        v_nom VARCHAR2(50) := 'Ahmed';
        v_salaire NUMBER := 10000;
        v_adresse VARCHAR2(50);
BEGIN
        v_adresse := 'Casablanca'
END;
/
```

2-1) Section déclaration DECLARE :

Exemple 2 :

Le bloc suivant déclare les variables : **v_nom, v_salaire** et **v_adresse,** puis demande à l'utilisateur de saisir le nom et le salaire et les affectent aux variables : **v_nom, v_salaire. v_adresse** est une variable sans valeur (vide).

```
DECLARE
        v_nom VARCHAR2(50) := &v_nom;
        v_salaire NUMBER := &v_salaire;
        v_adresse VARCHAR2(50);
BEGIN
        Opérations de traitement
END;
/
```

2-1) Section déclaration DECLARE :

Les chaînes de caractères et les dates doivent être saisies entre **guillemets** **'Valeur_entrée'** comme dans l'exemple suivant : 'Ahmed', 'Karim'...

2-1-1) Les types de variables :

Les types de variables qui peuvent être utilisés dans un bloc PL/SQL :

A. **Variables de type Oracle :**
   Char, Varchar2, Number, Date….etc
B. **Variables booléennes :**
   Exemple : v_valide BOOLEAN := true;
C. **Variables de même type qu'une colonne d'une table de BD :**

Nom_variable Table.Colonne%type := 'Valeur';

Exemple : v_nom **employees.LAST_NAME%type := 'Ahmed';**

Cette clause déclare une variable **v_nom** de type **VARCHAR2**, qui est le même type de données de la colonne **LAST_NAME** de la table **employees**. Puis il affecte la valeur **'Ahmed'** à la variable **v_nom.**

2-1-1) Les types de variables :

D. Variables de même type qu'une colonne d'une table de BD :

Nom_variable nom_table%ROWTYPE;

%ROWTYPE elle hérite automatiquement de la structure de la table ou de la vue à laquelle elle est associée. Cela signifie que la variable de type **ROWTYPE** contiendra des sous-champs correspondant à toutes les colonnes de la table ou de la vue.

Exemple :

```
DECLARE
    v_employee employees%ROWTYPE;
```

v_employee est une variable de type **%ROWTYPE** qui peut stocker une ligne complète de la table **employee**. La variable prendra alors tous les champs de la table **employee** et sera utilisée comme un ensemble de champs de données.

2-2) Section corps du bloc BEGIN – END :

Cette section contient **les instructions du programme** et éventuellement, à la fin, la section de **traitement des erreurs**, cette section est **obligatoire** et introduite par la clause **BEGIN** et se termine par la clause **END.**

```
DECLARE      -- mes déclarations de variables
BEGIN        -- début de la section exécutable
     Les requêtes SQL et PL/SQL

END;         --la fin du bloc exécutable
/
```

2-2) Section corps du bloc BEGIN – END :

Pour affecter les données d'une table ou d'une (colonne ou plusieurs) aux variables déclarées, on utilise la commande suivante :

```
DECLARE
        v_var1 VARCHAR2(30);
        v_var2 VARCHAR2(30);
BEGIN
        SELECT col1, col2 INTO v_var1, v_var2
        FROM nom_table
        WHERE condition;
END;
/
```

2-2) Section corps du bloc BEGIN – END :

Exemple :

Dans cet exemple, on a déclaré deux variables **"v_nom"** et **"v_prenom"** qui et puis affecté les données des colonnes **"first_name"** et **"last_name"** de la table **"employees"** aux variables **"v_nom"** et **"v_prenom".**

```
DECLARE
        v_nom employees.last_name%TYPE;
        v_prenom employees.first_name%TYPE;
BEGIN
        SELECT last_name, first_name
        INTO v_nom, v_prenom
        FROM employees
END;
/
```

les deux variables prendront le même type de données dans les colonnes 'first_name' et 'last_name'.

les deux variables prendront le même type de données dans les colonnes 'first_name' et 'last_name'.

2-2) Section corps du bloc BEGIN – END :

Pour afficher le contenu des variables dans un bloc pl/sql, on utilise la commande **DBMS_OUTPUT.PUT_LINE**, cette commande est activée ou désactivée à travers la variable **SERVEROUTPUT**, donc pour l'activer on doit écrire :

```
SET SERVEROUTPUT ON

DECLARE
        v_nom varchar2(30):= 'Amine';
BEGIN
        DBMS_OUTPUT.PUT_LINE('le nom est :'|| v_nom);
END;
/
```

2-2) Section corps du bloc BEGIN – END :

Exemple : Afficher le **nom** et le **prénom** de l'employé dont le '**employee_id**' égal à 100 dans la table '**employees**' :

```
SET SERVEROUTPUT ON      --Pour activer l'affichage
DECLARE
        v_employee_id employees.employee_id%TYPE := 100;
        v_nom employees.last_name%TYPE;
        v_prenom employees.first_name%TYPE;
BEGIN
    SELECT last_name, first_name INTO v_nom, v_prenom FROM employees
    WHERE employee_id = v_employee_id;
    DBMS_OUTPUT.PUT_LINE(v_employee_id||': '|| v_nom ||', '|| v_prenom);
END;
/
```

*-- le résultat de ce bloc : **100: Steven, King***

2-2) Section corps du bloc BEGIN – END :

EXEMPLE 2 :

Ce bloc PL/SQL sélectionne et affiche le nombre total de salaires de la table employees :

```
DECLARE
    v_total_salaries NUMBER;
BEGIN
    SELECT SUM(salary) INTO v_total_salaries
    FROM employees;
    DBMS_OUTPUT.PUT_LINE('Total des salaires : ' || v_total_salaries);
END;
```