UNIVERSITÉ ASSANE SECK DE ZIGUINCHOR
UFR DES SCIENCES ET TECHNOLOGIES
DÉPARTEMENT D'INFORMATIQUE

CHAPITRE II
PL-SQL SOUS MYSQL

LICENCE 2 INGÉNIERIE INFORMATIQUE

ANNÉE ACADÉMIQUE 2021 – 2022

SEMESTRE 4

DR SERIGNE DIAGNE

PLAN DU COURS

Introduction

I. La structure d'un programme
   1. Configuration du serveur de données
   2. L'implémentation de bases de données

II. Les bases du PLSQL
   1. Affectation
   2. Opérateurs et commentaires
   3. Instructions conditionnelles
   4. Instructions itératives

III. Les triggers

IV. Les routines
   1. Procédure stockée
   2. Fonction

V. Les curseurs

INTRODUCTION

- ✓Le langage SQL n’est pas un langage procédural. Il ne permet, donc pas :
  - de déclarer des variables ;
  - de créer des sous-programmes ;
  - de programmer l'exécution d'une action suivant un événement ;
  - de répéter l'exécution d'une ou plusieurs instructions.
- ✓Un langage procédural, repose sur des procédures, des fonctions, des sous-routines, etc. ;
- ✓Il spécifie toutes les étapes que l'ordinateur doit suivre pour atteindre l'état ou la sortie souhaité ;
- ✓Il sépare un programme au sein de variables, fonctions, instructions et opérateurs conditionnels.

INTRODUCTION

- ✓ Ainsi, le langage PL/SQL (Procedural Language/SQL) est créé pour permettre de faire de la programmation procédurale sous MySQL et Oracle ;
- ✓ Il est basé sur le langage SQL donc il emprunte les types, les opérateurs, etc.

I. STRUCTURE D’UN PROGRAMME PLSQL

- ✓ Un programme PL/SQL est composé généralement de trois parties :
  - L’entête ;
  - La partie déclarative ;
  - Le corps du programme.

I. STRUCTURE D'UN PROGRAMME PLSQL

I. 1. L'ENTÊTE

- ✓ L'entête d'un module PL/SQL :
  - contient le type de programme ;
  - son nom ;
  - ses arguments éventuels (fonction et procédure) ;
  - le domaine de la valeur de retour (fonction) ;
  - le moment et l'évènement ainsi que la table sur laquelle elle porte (trigger) ;
  - etc.

I. STRUCTURE D'UN PROGRAMME PLSQL

I. 2. LA PARTIE DÉCLARATIVE

- ✓ Les déclarations sont faites dans la deuxième partie du programme après l'entête.
- ✓ C'est dans cette partie que toutes les déclarations sont faites : variable, constante, curseur, enregistrement, etc.
- ✓ Elle est ouverte par le mot clé **Declare**.

Exemple :

Declare id_variable Domaine ;

I. STRUCTURE D'UN PROGRAMME PLSQL

I. 3. LE CORPS DU PROGRAMME

- ✓ Le corps d'un programme PL/SQL est ouvert par le mot-clé **Begin** et fermé par le mot-clé **End**.
- ✓ Toutes les instructions à exécuter dans le programme doivent être écrites entre ces deux mots-clés.

Exemple :

```
Begin
        Instruction_1 ;
        Instruction_2 ;
End ;
```

II. LES BASES DU PLSQL

II.1. L'Affectation

Il existe trois types d'affectation sous PL-SQL :

- ✓ **Initialisation pendant la déclaration :**

Declare id_variable type **Default** Valeur ;

- ✓ **Affectation d'une valeur scalaire dans le programme :**

Set id_variable **=** valeur ;

- ✓ **Affectation du résultat d'une requete**

Select Attribut **Into** id_variable **From** Table **Where** condition(s) ;

II. LES BASES DU PLSQL

II.1. L’Affectation

Exemple :

- ✓ **Declare** a integer Default 25 ;
- ✓ **Set** a = 15 ;
- ✓ **Select** Age **Into** a **From** Etudiant **Where** Matricule = 'MA00254' ;

II. LES BASES DU PLSQL

II.2. Les opérateurs

- ✓ Opérateurs ensemblistes : Not ; Is Null ; Like ; Between ; In ; Any
- ✓ Opérateurs booléens : And ; Or
- ✓ Opérateurs athmetiques : + ; - ; * ; / ; @ ; = ; <> ; <= ; >=

II. LES BASES DU PLSQL

II.3. Les commentaires

- ✓ Commentaires sur une ligne : -- Sur une seule ligne
- ✓ Commentaires sur plusieurs lignes : /* Sur plusieurs ligne */

II. LES BASES DU PLSQL

II.4. Les instructions conditionnelles

a. L'instruction IF

```
If  condition  Then
        Instruction (s) ;
ElseIf  condition Then
        Instruction(s) ;
Else
        Instruction(s) ;
End If ;
```

II. LES BASES DU PLSQL

II.4. Les instructions conditionnelles

Exemple :

```
Declare a integer ;
Declare b varchar(50) ;
Begin
        Select Count(*) Into a From Etudiant Where Age = 22 ;
        If a = 5 then
        Set b = 'Il y a cinq étudiants ayant 22 ans' ;
        ElseIf a = 10 Then
        Set b = 'Il y a dix étudiants ayant 22 ans' ;
        End If ;
        Select b ;
End ;
```

II. LES BASES DU PLSQL

II.4. Les instructions conditionnelles

b. L'instruction Case

```
Case  id_variable
        When Valeur_1 Then Instruction(s) ;
        When Valeur_2 Then  Instruction(s) ;
        - - - - - - -
        When Valeur_n Then Instruction(s) ;
        Else     Instruction(s) ;
End Case ;
```

II. LES BASES DU PLSQL

II.4. Les instructions conditionnelles

b. L'instruction Case

```
Case
        When Condition_1 Then Instruction(s) ;
        When Condition_2 Then Instruction(s) ;
          - - - - - - - - - - -
        When Condition_n Then Instruction(s) ;
        Else   Instruction(s) ;
End Case ;
```

II. LES BASES DU PLSQL

II.5. Les instructions itératives

While Condition **Do**

Instruction(s) ;

End While ;

Label : **Loop**

Instruction(s) ;

Leave Label ;

End Loop Label

Repeat

Instruction (s) ;

Until Condition ;

End Repeat ;

II. LES BASES DU PLSQL

II.5. Les instructions itératives


```
Declare a integer ;
Declare b boolean Default False ;
Begin
        Select Count(*) Into a From Etudiant Where VilleNaissance = 'Bignona' ;
        While a < 5 Do
                Set a = a + 1 ;
                Set b = True ;
        End While ;
        Select b ;
End ;
```

II. LES BASES DU PLSQL

II.5. Les instructions itératives

```
Declare a, c integer Default 0 ;
Begin
        Select Count(*) Into a From Etudiant Where VilleNaissance = 'Bignona' ;
        Iterloop : Loop
                If a >= 35 Then
                        Leave iterloop ;
                End If ;
                Set a = a + 5 ;
                Set c = c + 1 ;
        End Loop interloop ;
        Select c ;
End ;
```

II. LES BASES DU PLSQL

II.5. Les instructions itératives

Remarques :

- ✓ La commande **Leave** permet d’arrêter l'exécution de la boucle dans laquelle elle est placée ;
- ✓ La commande **Iterate** permet d'arrêter l'exécution en cours et de passer à une nouvelle.

II. LES BASES DU PLSQL

II.6. Variable globale

- ✓ Une variable globale est une variable dont l’identificateur est précédé du caractère '@'.
- ✓ Elle est visible dans le trigger ou la routine dans lequel elle est déclarée et dans les autres qui seront créées par la suite.
- ✓ Une variable qui n'est pas précédée de ce signe est donc une variable locale.

III. LES TRIGGERS

III. 1. Qu’est-ce qu’un trigger ?

- ✓ Un trigger est un objet de base de données associé à une table ;
- ✓ Il est appelé lorsqu'un événement particulier survient ;
- ✓ Son exécution est déclenchée par la survenance d’une action bien déterminée (Evénement) à un moment bien déterminé (Moment) ;

III. LES TRIGGERS

III. 1. Qu’est-ce qu’un trigger ?

Les triggers doivent respecter les conditions suivantes :

- ✓ Un trigger ne peut être placé sur une vue ou une table temporaire ;
- ✓ Deux triggers de même moment et même événement ne peuvent pas porter sur la même table ;
- ✓ Il est nécessaire d’être super-utilisateur pour créer, modifier ou supprimer un trigger ;
- ✓ Si le trigger échoue et que son instant d’exécution est BEFORE, l’action qui suit ne sera pas exécutée ;
- ✓ La seule manière d’interrompre un trigger est de provoquer une erreur en son sein.

III. LES TRIGGERS

III. 2. Création d’un trigger

La syntaxe de la création d'un trigger est :

Create TRIGGER Nom_Trigger Moment_Trigger Evenement_Trigger
On Nom_Tabble **For Each Row**

Declare

Begin

Instruction (s) ;

End ;

III. LES TRIGGERS

III. 2. Création d’un trigger

Remarques :

- ✓ **Temps_Trigger** est soit **Before** soit **After**.
- ✓ **Evenement_Trigger** est soit **Insert** soit **Update** soit **Delete**
- ✓ Dans le corps d'un trigger, on peut faire référence aux colonnes dans la table associée en utilisant les mots **OLD** et **NEW**.
  - **OLD.*col_name*** fait référence à une colonne d'une ligne existante avant sa modification ou son effacement ;
  - **NEW.*col_name*** fait référence à une colonne d'une ligne après insertion ou modification.

III. LES TRIGGERS

III. 2. Création d’un trigger

Remarque :

- ✓ L'utilisation de **SET NEW.*col_name* = *value*** requiert le droit de **UPDATE** sur la colonne.
- ✓ L'utilisation de **SET *value* = NEW.*col_name*** requiert le droit de **SELECT** sur la colonne.
- ✓ La commande **CREATE TRIGGER** requiert le droit de **SUPER**

III. LES TRIGGERS

III. 2. Création d’un trigger

Exemple :

```
Create Trigger ControleAge Before Insert On Etudiant For Each Row
Begin
        declare n integer ;
        Set n = New.Age ;
        If n > 50 or n < 15 then
        Set New.Age = Null ;
End ;
```