Université Mohammed V - Rabat

Ecole Mohammadia d'Ingénieurs

Filière Réseaux et Télécom

Bases de Données

Programmation PL/SQL

Mr N.EL FADDOULI

2023-2024

Plan

- Introduction
- Bloc PL/SQL
- Déclaration des variable
- Structure de contrôle
- Curseurs
- Les exceptions
- Les fonctions et procédures
- Les packages
- Les triggers

Procédure, Fonction et Package

- Définition
- Utilité
- Structure
- Exemples

Procédures et fonctions

- PL/SQL est aussi utilisé pour définir des procédures et fonctions stockées dans la BD.
- Syntaxe: **Procédure**

  CREATE [OR REPLACE] PROCEDURE
  Nom_procédure [**(** *liste d'arguments* **)**] { **IS** | **AS**}
  [***variables locales***]
  Corps PL-SQL
- syntaxe: **Fonction**

  CREATE [OR REPLACE] FUNCTION
  Nom_fonction [**(** ***liste d'arguments*** **)**] **RETURN** *type* { **IS** | **AS**}
  [***variables locales***]
  Corps PL-SQL

Procédures et fonctions

- L'option **OR REPLACE** permet de spécifier au système le remplacement de la procédure ou de la fonction si elle existe déjà dans la BD.
- Liste d'arguments: ***nom_arg*** [**IN** | **OUT** | **IN OUT**] ***Type***
  - **IN**: la variable est passée en entrée
  - **OUT**: la variable est renseignée par la procédure puis renvoyée à l'appelant
  - **IN OUT**: passage par référence.
- le mot **RETURN** permet de spécifier le type de la donnée de retour.
- Le corps PL/SQL doit commencer par le mot clé **BEGIN** et se termine par **END**. Il peut être composé d'une partie déclarative , d'un corps de la procédure et d'un gestionnaire d'erreurs.

Exemples (1/3)

Exemple : Créer une **procédure** qui permet de baisser le prix d'un produit de la table: Produit (Nump, nomp, pu, qstock)

```
CREATE PROCEDURE baisse_prix ( nprod IN NUMBER,
                               Taux IN NUMBER) IS
BEGIN
    if Taux <100 then
        UPDATE produit SET PU= PU* ( 1 – Taux/100)
        WHERE Nump= nprod ;
    end if;
END ;
```

Exemples (2/3)

Exemple : Créer une **fonction** qui retourne le nombre de commandes d'un client

Client (CliNo, CliName, …) Commande(ComNo, CliNo, ….)

```
CREATE FUNCTION NbCommande ( N IN Number) RETURN Number IS
        C Number;
BEGIN
Select count(*) into C from Commande Where CliNo = N;     Return (C);
END ;      -- Fonction qui retourne le nom d'un client
CREATE FUNCTION NomClient( N IN Number) RETURN Client.CliName%type IS
        S Client.CliName%type ;
BEGIN
Select CliName into S from Client Where CliNo = N;     Return (S);
Exception
   When No_Data_Found Then Return ('Aucun');
END ;
```


Exemples (3/3)

Appel de procédure et fonction dans un bloc PL/SQL

```
SQL>  Declare
            A  Client.CliNo%type;  S  varchar2(30);
      BEGIN
      A := &Numero_Client;  S := NomClient(A);  dbms_output.put_line( S );
      END ;
```

Exécution les procédures avec exec: SQL> **exec** baisse_prix (123, 10) ;

Appel d'une fonction dans une requête

Client (CliNo , CliName , …) Commande (ComNo , CliNo, ….)

```
SQL>  Select  CliName , NbComande (CliNo)
      From  Client;
SQL>  Select  NbComande (23)
      From  dual ;
```

Modification d'une procédure (fonction)

- **Si la base de données évolue, il faut recompiler les procédures existantes pour qu'elles tiennent compte de ces modifications.**

Exemple:

- La commande est la suivante:

ALTER { FUNCTION | PROCEDURE } ***nom*** **COMPILE**

Exemple:

```
ALTER   PROCEDURE   Baisse_Prix   COMPILE;
ALTER   FUNCTION    NomClient     COMPILE;
```


Suppression d'une procédure (fonction)

- Pour supprimer une procédure

DROP { FUNCTION |PROCEDURE } ***nom***

Les sous-blocs

- Un bloc PL/SQL peut contenir un sous bloc:

```
Declare
    Variables
Begin
        …..
        Declare
            Variables
        Begin
            ….
        End;
        …..
End;
```


Les sous-blocs: Exemple

```
Declare
    a number;   b varchar2(30); c number;
Begin
        a:=10; b:='Imagination'; c:= 20;
        Declare
            a number;   b varchar2(30); d number;
        Begin
            d:=c+5;
            a:=40; b:='programmation';
        End;
        b:=b||' Esprit';  a:=a+d;
        dbms_output.put_line(a||'  '||b||'  '||c);
End;
```

Exercices

1. Ecrire une fonction **Montant** qui retourne le montant réalisé par une formation dont le code est donné comme paramètre

   N.B: Traiter le cas où la formation n'existe pas (on retourne par exemple **-1**)
2. Ecrire une requête select pour afficher le titre et le montant de chaque formation en appelant la fonction **Montant**.
3. Ecrire une fonction **Reste** qui retourne le montant qui reste à payer pour un participant dont le code est donné comme paramètre.
4. Ecrire un bloc qui lit le code d'un participant, appelle la fonction **Reste** et affiche le résultat.
5. Ecrire une procédure qui affiche toutes les formations (titre, prix, **montant réalisé**) qui ont réalisé un montant inférieur à un montant X donnée comme argument.

   N.B: Cette procédure doit appeler la fonction **Montant**.

Les packages

- **Encapsuler des procédures, des fonctions, des curseurs et des variables comme une unité dans la base de données.**
- Etapes de création:
  - **Création des spécifications du package**
    - spécifier la partie **publique** du package (**fonctions, procédures, types, variables, constantes, exceptions et curseurs**).
  - **Création du corps du package**
    - définir les procédures, les fonctions, les curseurs et les exceptions qui sont déclarés dans les spécifications du package.
    - définir d'autres objets non déclarés dans les spécifications. Ces objets sont alors **privés**.

Les packages

- **Création des spécifications du package**

CREATE [**OR REPLACE**] **PACKAGE** ***nom_package***

{ **IS** | **AS** }

spécifications PL/SQL

END ***nom_package;***

spécification PL/SQL::=declaration_de_variable |
declaration_d_enregistrement |
declaration_de_curseur |
declaration_d_execption |
declaration_de_procedure |
declaration_de_fonction…

Exemple

```
CREATE OR REPLACE PACKAGE gest_empl IS
   -- Variables publiques
                Sal    EMP.sal%Type ;
  -- Fonctions et procédures publiques
   FUNCTION Augmentation (NumEmp   IN  EMP.empno%Type,
                          Pourcent  IN  NUMBER ) Return NUMBER ;
   PROCEDURE  Test_Augmentation  ( NumEmp   IN  EMP.empno%Type,
                                   Pourcent  IN OUT  NUMBER ) ;
END  gest_empl;
```

Les packages

- **Création du corps du package**

CREATE [**OR REPLACE**] **PACKAGE BODY** ***nom_package***
{ **IS** | **AS** }
spécifications PL/SQL
END ***nom_package;***

spécification PL/SQL::=declaration_de_variable |
declaration_d_enregistrement |
declaration_de_curseur |
declaration_d_execption |
définition_de_procedure |
définition_de_fonction…

Exemple

```
CREATE OR REPLACE PACKAGE BODY gest_empl IS
E    EMP%Rowtype ;    --Variables  privées dans le package
  -- Fonctions publiques
FUNCTION Augmentation  ( Numemp IN EMP.empno%Type ,
  Pourcent IN NUMBER        ) Return NUMBER     IS
      Salaire   EMP.sal%Type ;
BEGIN
  Select  sal  Into  Salaire From  EMP  Where  empno = Numemp ;
  Salaire := Salaire * (Pourcent + 1 ); -- augmentation virtuelle
  Sal := Salaire ;     -- Affectation de la variable globale publique
  Return( Salaire ) ;       -- retour de la valeur
END Augmentation;
```

```
-- Procédure privée  dans le package
PROCEDURE Affiche_Salaires  IS
   CURSOR    C_EMP   IS select * from EMP ;
BEGIN
   OPEN   C_EMP ;  FETCH   C_EMP   Into   E ;
    While   C_EMP%FOUND  Loop
          dbms_output.put_line(    'Employé    '    ||    E.ename    ||
                                   ' -> '|| To_char( E.sal ) );
          FETCH   C_EMP   Into   E ;
     End loop ;
     CLOSE C_EMP ;
END Affiche_Salaires ;
```

PL/SQL - N.EL FADDOULI 74

```
-- Procédures publiques
PROCEDURE   Test_Augmentation   (   Numemp  IN
EMP.empno%Type ,   Pourcent IN OUT NUMBER   ) IS
     Salaire   EMP.sal%Type ;
BEGIN
 Select sal  Into Salaire From EMP Where   empno = Numemp ;
Pourcent := Salaire * (Pourcent + 1) ; -- augmentation virtuelle
  Affiche_Salaires ;     -- appel procédure privée
 END Test_Augmentation;
END gest_empl;   /
```

PL/SQL - N.EL FADDOULI 75

Package

- L'accès à un objet d'un paquetage :

***nom_paquetage.nom_objet*[(*liste paramètres*)]**

- Exemple: Appel de la fonction Augmentation du paquetage

```
SQL> Declare
        A   emp.sal%Type ;
   Begin
        A  := gest_empl.Augmentation( 100, 0.2 )  ;
        …..
        dbms_output.put_line(gest_empl.sal);
        …..
  End ;   /
```


Exercice

1. Modifier la table **Participant** en ajoutant une colonne **NF** de type entier.

```
SQL> alter table apprenant add NF number;
Table modifiÚe.
```

2. Ecrire un package contenant :

- La fonction **Montant**
- La fonction **Reste**
- La procédure **FuturFormation** qui affiche les formations qui n'ont pas encore démarré.
- La procédure **NF_Participant** qui modifie le champ **NF** pour chaque apprenant. Ce champ représente le nombre de formation de l'apprenant.

3. Tester les fonctions **Montant** et **Reste** dans un bloc

4. Tester les procédures **FuturFormation** et **NF_Participant** directement par la commande **EXEC** dont l syntaxe est:

SQL> **exec** *ma_procédure* [(*paramètres*)];

5. Vérifier le colonne **NF** de la table **Apprenant** .