SMI-S5
SGBD ORACLE : Introduction PL/SQL

Pr. Issam QAFFOU

Laboratoire Ingénierie des Systèmes d'Information
Département d'Informatique
FSSM-UCA

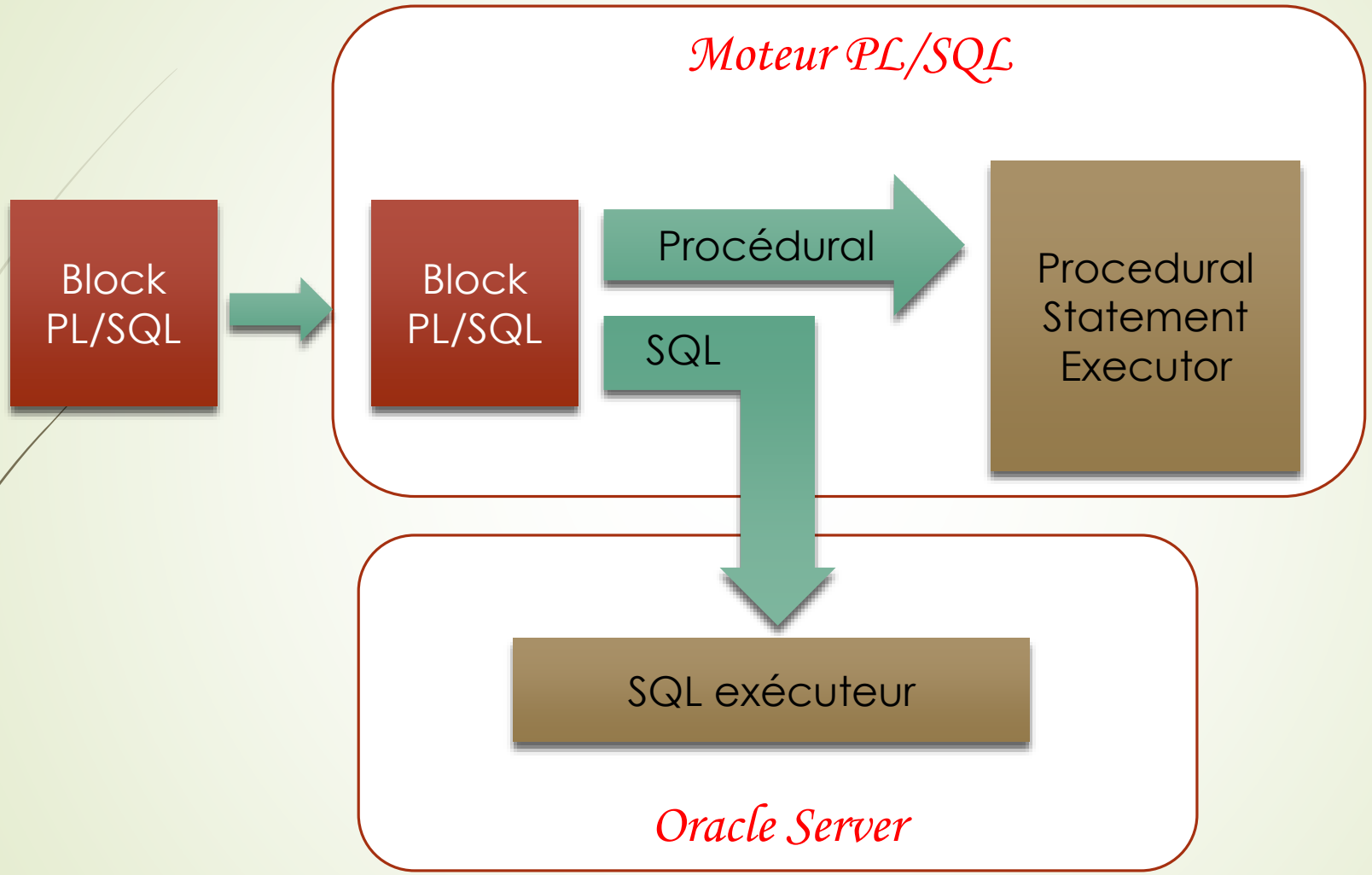
Pourquoi PL/SQL?

- SQL ne traite qu'un ensemble de données dotées de critère de recherche.
- Les notions de programmation ne sont pas valides dans SQL
- PL/SQL est une extension procédurale de SQL incluant:
  - Les variables et les types
  - Les structures de contrôle et les boucles.
  - Les procédures et les fonctions.
  - Les types d'objets et les méthodes.
- PL/SQL est une extension procédurale sur Oracle du SQL.
- Permet de combiner les déclarations SQL avec les déclarations procédurales comme IF, structures de boucle, etc.
- PL/SQL utilise SQL pour la manipulation des données et utilise ses propres déclarations pour le traitement

C'est quoi PL/SQL

- Les unités d'un programme PL/SQL sont catégorisées comme suit:
  - blocs anonymes: apparaissent lorsque des instructions SQL peuvent apparaître.
  - procédures stockées: elles le sont dans la base de données avec un nom. Des programmes peuvent les exécuter en utilisant ce nom.
- Oracle permet de créer des fonctions, similaires aux procédures mais elles retournent une valeur.
- Il permet aussi de créer des packages.

- Chaque bloc PL/SQL est exécuté par un moteur PL/SQL. Ce moteur est chargé de compiler et exécuter ces blocs.
- Ce moteur est disponible dans des outils de Oracle Server comme: Oracle Forms et Oracle Reports.
- Durant son exécution, il envoie des commandes SQL à l'exécuteur de commandes dans le SGBD Oracle.
- Càd, il sépare les commandes SQL de celles de PL/SQL.
- Les commandes PL/SQL sont exécutées par "Procedural Statement Executor".


Moteur PL/SQL
Block PL/SQL
Block PL/SQL
Procédural
SQL
Procedural Statement Executor
SQL exécuteur
Oracle Server

Caractéristiques de PL/SQL

- Structure de Bloc
  - Tout programme PL/SQL est écrit en blocs,
  - Les blocs peuvent aussi être imbriqués.
  - Chaque bloc est conçu pour une tâche particulière.
- Variables et constantes
  - PL/SQL permet de déclarer des variables et des constantes.
  - Les variables sont utilisées pour stocker des valeurs temporairement.

- ## Structures de contrôle

  PL/SQL permet d'utiliser IF, la boucle FOR, la boucle WHILE dans le bloc. Ces structures constituent une importante extension du SQL au PL/SQL. Elle permettent tout traitement de données en PL/SQL.

- ## Gestion des exceptions

  PL/SQL permet de détecter des erreurs et de les gérer. Chaque fois qu'il y ait une erreur prédéfinie, PL/ SQL soulève automatiquement une exception. Ces exceptions peuvent être manipulées pour récupérer les erreurs.

- modularité

  PL / SQL permet à un processus d'être divisés en différents modules. Les sous-programmes appelés procédures et fonctions peuvent être définies et appelées à l'aide d'un nom. Ces sous-programmes peuvent également prendre des paramètres.

- curseurs

  Un curseur est une zone SQL privée utilisée pour exécuter des instructions SQL et stocker des informations de traitement. PL/SQL utilise implicitement les curseurs pour toutes les commandes DML et la commande SELECT qui retourne une seule ligne. Il permet également de définir un curseur explicite pour faire face aux multiples requêtes de ligne.

Bloc PL/SQL

- Les programmes PL/SQL sont écrits sous forme de blocs. Un bloc permet de grouper les déclarations liées de manière logique. Un bloc est constitué de trois parties suivantes:
  - Partie déclarative
  - Partie exécutable
  - Partie de gestion des exceptions

[DECLARE

déclaration des variables
déclaration des curseurs
déclaration des exceptions]

BEGIN

Commandes exécutables

[EXCEPTION

gestionnaires d'exceptions]

END;

```
declare
        v_rollno      number(5);
        v_count       number(2) := 0;
        v_name        étudiant.nom%type;
        done          boolean := FALSE;
```

Note: Si le nom d'une variable et le nom d'une colonne sont les mêmes alors Oracle suppose que la colonne est référencée lorsque le nom est utilisé dans les commandes SQL. Cela signifie que les noms de colonnes ont la priorité sur les noms des variables dans les instructions SQL.

Écrire son premier bloc PL/SQL

```
declare
 v_rollno étudiant.rollno%type;
begin
--obtenir le numéro des étudiants qui sont inscrits récemment
 select max(rollno) into v_rollno from étudiant;
-- insérer une nouvelle ligne dans la table des paiements
 insert into payement values (v_rollno,sysdate,1000);
-- transaction de validation
 commit;
end;
/
```

- Exemple: L'exemple suivant copie le montant total payé par l'étudiant ayant rollno 102 dans la variable V_SUM, qui est déclarée comme NUMBER (5).

```
declare
    v_sum      number(5);
    v_frais    cours.frais%type;
    v_dur      cours.durée%type;

begin
    select sum(amount) into v_sum from payement where rollno = 102;
    -- récupérer les frais et la durée du cours Oracle
    select frais, durée into v_frais, v_duration from cours
    where ccode = 'ora';
    . . .
end;
```

- Le nombre de variables et de types de données proposés après INTO doit correspondre au nombre de colonnes dans SELECT et leurs types de données.
- Déclaration des constantes

variable CONSTANT datatype [precision , scale] := expression;

Exemple: bonus constant number(3) := 500;

Blocs imbriqués

- Il est possible de définir un bloc à l'intérieur d'un autre bloc. Lorsque les blocs sont définis à l'intérieur de l'un l'autre, ils sont dits imbriqués.
- Ci-après un exemple de bloc imbriqué. Le bloc principal contient une seule variable X et le bloc imbriqué contient une seule variable Y.

```
declare
    x number(5);
begin
    -- mettre le code exécutable du bloc principal ici
    declare /* début du bloc imbriqué */
     y number(3);
    begin
     -- mettre le code exécutable du bloc imbriqué
     exception
     -- la gestion des exceptions pour le bloc imbriqué
    end;
     -- code de bloc principal se poursuit.
     Exception
     -- la gestion des exceptions pour le bloc principal
end;/
```

Portée et visibilité des variables

- Le champ d'application d'une variable indique la région du programme dans laquelle la variable peut être utilisée. Une variable est dite visible quand elle peut être renvoyée sans aucune qualification.

```
Declare
   num1   number(5);
Begin

   Declare
      num2  number(5);
   Begin
      ...
   End;

End;
```

La variable NUM1 peut être consultée à partir du moment de la déclaration à la fin du bloc externe.
La variable NUM2 peut être consultée à partir du moment de la déclaration à la fin de bloc interne.

- La différence entre la portée et la visibilité

```
Declare
   n   number(5);
Begin


   Declare
      n  number(5);
   Begin
      ...
   End;


End;
```

Il est possible d'accéder à la variable n déclarée dans le bloc externe même si elle est masquée en utilisant l'étiquette du bloc externe.

Étiquette d'un bloc

- Une étiquette peut être utilisée pour nommer un bloc. L'étiquette est placée à l'intérieur de << et >> juste avant le début du bloc.

```
<<mainblock>>
declare
...
Begin
...
end;
```

- Pour accéder à la variable cachée du bloc principal à partir du bloc interne, on donne une étiquette au bloc principal qui peut être utilisée pour faire référence à des objets cachés pour le bloc interne.

```
<<mainblock>>
Declare
    n number(5);
Begin
    <<nestedblock>>
    Declare
     n number(5);
    Begin
     ...
     n := 20; -- stocke 20 dans N block imbriqué.
     /* stocke 50 dans n du bloc principal*/
     mainblock.n := 50;
    End;
End;
```

- Ce bloc PL/SQL modifier les frais du cours VBNET par la plus grande valeur entre la moyenne des frais de tous les cours et les frais du cours Oracle.

```
declare
    v_avgfee cours.frais%type;
    v_orafee cours.frais%type;
begin
    -- moyenne des frais de tous les cours
    select avg(frais) into v_avgfee  from cours;
    -- les frais de Oracle
    select frais into v_orafee  from cours where ccode = 'ora';
    -- mettre à jour les frais de VB
    update cours set frais = max( v_avgfee, v_orafee) where ccode = 'vbnet';
    -- valider
    commit;
end;
/
```

Affichage en PL/SQL

- Pour afficher une sortie en PL/SQL, on utilise le package DBMS_OUTPUT.
- De ce package on peut utiliser deux procédures:
  - Put: permet de mettre plusieurs morceaux qui composent une ligne.
  - Put_line: place les données, suivis du caractère de fin de ligne.
- Les deux affichent des valeurs de type: NUMBER, VARCHAR2 ou DATE.

- Pour voir la sortie envoyée à l'aide de ces procédures, les éléments suivants doivent être satisfaits:
  - Les unités de programme desquelles elles sont appelées doivent être accomplies.
  - SERVEROUTPUT doit être déclenché en SQL*PLUS: SQL> SET SERVEROUTPUT ON

```
declare
    v_amtcltd number(6);
    v_totamt number(6);
    v_orafee cours.frais%type;
    v_studcnt number(3);
    v_diff number(6);
begin
    -- Retourne la somme totale assemblée par les étudiants Oracle
    select sum(total) into v_amtcltd from payement  where rollno in ( select rollno from étudiant
    where bcode in( select bcode from batches  where ccode = 'ora'));
    -- retourner d'abord les frais du cours oracle
    select frais into v_orafee from cours where ccode = 'ora';
    -- retourner les no. des étudiants de Oracle batches
    select count(*) into v_studcnt from étudiant where bcode in ( select bcode from batches
    where ccode = 'ora');
    -- calcul de la somme totale à assembler par les étudiants Oracle
    v_diff := v_orafee * v_studcnt - v_amtcltd;
    dbms_output.put_line('Oracle arrears : ' || v_diff );
end;
```

Exercices

1. Dans la partie ……. Du block PL/SQL, les erreurs sont manipulées.
2. La partie de Oracle Server qui exécute les commandes SQL est appelée: ….
3. …… est un exemple d'outils de Oracle contenant le moteur PL/SQL.
4. …… est utilisé pour un commentaire d'une ligne.
5. Écrire un bloc PL/SQL qui change la durées des cours C++ à celle des cours de JAVA.
6. Insérer la ligne suivante à la table Cours_faculté:

   Code du cours est 10g, le code de la faculté est FSSM et le grade est B.