Fondements des bases de données

Programmation en PL/SQL Oracle – procédures, fonctions et exceptions

Équipe pédagogique BD

`http://liris.cnrs.fr/~mplantev/doku/doku.php?id=lif10_2016a`

Version du 13 octobre 2016

Bloc anonyme ou nommé

- Un bloc anonyme PL/SQL est un bloc `DECLARE ...BEGIN ...END` comme dans les exemples précédents.
- On peut exécuter directement un bloc PL/SQL anonyme.
- On passe plutôt une procédure ou une fonction nommée pour réutiliser le code.

Procédure sans paramètre

Exemple

```
CREATE OR REPLACE PROCEDURE list_nom_emps IS
BEGIN
  DECLARE
    CURSOR c_nom_emps IS
      SELECT nom, adresse
      FROM emp;
  BEGIN
    FOR v_emp IN c_nom_emps LOOP
      dbms_output.put_line(
        'Client : ' || UPPER(v_emp.nom) ||
        ' Ville : ' || v_emp.adresse);
    END LOOP;
  END;
END;
/

CALL list_nom_emps();
```

Procédure avec paramètres

Exemple

```
CREATE OR REPLACE PROCEDURE
   liste_nom_emps(ville IN varchar2, v_result OUT number) IS
BEGIN
   BEGIN
        SELECT COUNT(*) INTO v_result
        FROM emp
        WHERE adresse LIKE ('%' || ville || '%');
   END;
END;
/

DECLARE
   v_result number;
BEGIN
   liste_nom_emps('Manchester',v_result);
   dbms_output.put_line(v_result);
END;
/
```

IN	lecture seule
OUT	écriture seule
IN OUT	lecture et écriture

Fonctions sans paramètre

Exemple

```
CREATE OR REPLACE FUNCTION nb_emps
   RETURN NUMBER -- Type de retour
   IS
BEGIN
   DECLARE
      i NUMBER;
   BEGIN
      SELECT COUNT(*) INTO i
      FROM emp;
      RETURN i;
   END;
END;
/

DECLARE
BEGIN
   dbms_output.put_line(nb_emps());
END;
/
```

Fonctions avec paramètres

Exemple

```
CREATE OR REPLACE FUNCTION euro_to_fr(v_somme IN number)
   RETURN NUMBER
   IS
BEGIN
   DECLARE
      taux CONSTANT number := 6.55957;
   BEGIN
      RETURN v_somme * taux;
   END;
END;
/

DECLARE
BEGIN
   dbms_output.put_line(euro_to_fr(15.24));
END;
/
```

Seuls les paramètres `IN` (en lecture seule) sont autorisés

Un peu plus ...

- Les procédures et fonctions peuvent être utilisées dans d'autres procédures ou fonctions ou dans des blocs PL/SQL anonymes
- Les fonctions peuvent aussi être utilisées dans les requêtes
- Table système contenant les procédures et fonctions : `user_source`
  `SELECT * FROM user_source`

Pratique

- Déclaration d'une variable globale avec le préfixe « : »
- Description des paramètres : `DESC nom_procedure`
- Suppression de procédures ou fonctions :
  - `DROP PROCEDURE nom_procedure`
  - `DROP FUNCTION nom_fonction`
- `DUAL` est une pseudo table avec une seule colonne.

Les exceptions

Une exception est une erreur qui survient durant une exécution, elle est soit :

- prédéfinie par Oracle,
- définie par le programmeur.

Exceptions prédéfinies

`NO_DATA_FOUND` quand `SELECT ...INTO` ne retourne aucune ligne.

`TOO_MANY_ROWS` quand `SELECT ...INTO` retourne plusieurs lignes.

`VALUE_ERROR` érreur numérique.

`ZERO_DIVIDE` division par zéro

`OTHERS` toutes erreurs non interceptées.

Traitement des exceptions

Saisir une exception

- Une exception ne provoque pas nécessairement l'arrêt du programme : elle peut être saisie par une partie `EXCEPTION`.
- Une exception non saisie remonte dans la procédure appelante (où elle peut être saisie).

Exemple

```
BEGIN
   ...
EXCEPTION
   WHEN NO_DATA_FOUND THEN
      ...
   WHEN TOO_MANY_ROWS THEN
      ...
   WHEN OTHERS THEN --optionnel
      ...
END;
```

Exceptions utilisateur

- Elles doivent être déclarées avec le type `EXCEPTION`
- On les lève avec l'instruction `RAISE`
- On peut aussi utiliser `raise_application_error` quand on veut que l'exception remonte à l'utilisateur (sans la saisir)

Exemple

```
DECLARE
   e_custom EXCEPTION;
   PRAGMA EXCEPTION_INIT(e_custom, -20001 );
BEGIN
   raise_application_error(-20001, 'My custom error');
   -- RAISE e_custom;
EXCEPTION
   WHEN e_custom THEN
      dbms_output.put_line('A / custom / error was encountered - '
         ||SQLCODE|| ' : ' ||SQLERRM);
   WHEN OTHERS THEN
      dbms_output.put_line('An error was encountered - '
         ||SQLCODE|| ' : ' ||SQLERRM);
END;
/
```

Exceptions utilisateur

Exemple

```
DECLARE
   v_salaire emp.sal%TYPE;
   e_trop_bas EXCEPTION;
   PRAGMA EXCEPTION_INIT(e_trop_bas, -20001 );
BEGIN
   SELECT sal INTO v_salaire
   FROM emp
   WHERE matr = 1;
   IF v_salaire < 2000 THEN
      RAISE e_trop_bas;
   END IF;
EXCEPTION
   WHEN e_trop_bas THEN
      dbms_output.put_line('A / custom / error was encountered - '
         ||SQLCODE||' : salaire trop bas');
   WHEN OTHERS THEN
      dbms_output.put_line('An error was encountered - '
         ||SQLCODE||' : '||SQLERRM);
END;
/
```

Fin du cours.