Base de Data

Ouam uniquement

12 décembre 2014

Dans les exemples qui suivent cette magnifique fiche, on utilisera les tables de l'ENS2004.

1 Fonctions et procédures stockées

1.1 Partie PL/SQL

Donc comme vous êtes tous des GROSSES mer..., je vais vous expliquer ce que sont les fonctions et les procédures en PL/SQL. Tout d'abord, à quoi servent les fonctions et les procédures STOCKEES ?
Elle servent à enregistrer des programmes dans le noyau Oracle. Elles peuvent être utilisées par tout le monde (selon les droits) et sont stockées sous forme de *pseudo-code*, c'est-à-dire qu'elles ne sont compilées qu'UNE SEULE FOIS. C'est génial non ?
Trêve de bavardages, on va voir leurs syntaxes.
Une procédure en PL/SQL se déclare de la manière suivante :

```
1 //declaration d'une procedure
2 CREATE [OR REPLACE] PROCEDURE nom_prodecure ([liste des parametres]) IS|AS
3         [declaration des varibles]
4 BEGIN
5         \\corps de la procedure
6 EXCEPTION \\facultatif
7         \\definition des exceptions
8 END;
```

Bon ce n'est pas très clair mais on va voir ça petit à petit.
Donc, la première ligne correspond à une déclaration de procédure basique avec son nom et ses différents paramètres (ne pas oublier le IS ou le AS).
La deuxième ligne correspond à la déclaration des variables.
A partir du BEGIN, on écrit ce que va faire notre procédure. EXCEPTION est facultatif mais correspond à la zone d'exception du code.
Enfin le END, bah END.
En ce qui concerne les paramètres dans une procédure (ou une fonction), il faut leur donner un nom (gicLo), spécifié si c'est un paramètre **d'entrée** (IN, donc non modifiable) ou **de sortie** (OUT, modifiable par la procédure) ou les deux et enfin spécifié son type(une procédure ou une fonction n'a pas obligatoirement de paramètres).

```
1 CREATE OR REPLACE PROCEDURE numActeur (nom IN VARCHAR2, prenom IN VARCHAR2,
    num OUT INTEGER)
```

Ici, la procédure numActeur qui donne le numéro d'un acteur, prend en paramètres **d'entrée**, de type VARCHAR 2, *nom* et *prenom*. En paramètre **de sortie**, de type INTEGER, on a *num*.

Voici un exemple de procédure stockée.

```
// on definit la procedure unTitre qui renvoie le nom d'un film
// en entree, on a le numero du film et du realisateur
// en sortie, le titre du film
CREATE OR REPLACE PROCEDURE unTitre (numFilm IN INTEGER, realisateur IN
    INTEGER, titreFilm OUT VARCHAR2) IS
BEGIN
        // on selectionne le titre du film DANS titreFilm (le parametre de
            sortie)
        // sinon ca ne marche pas
        SELECT f.titre INTO titreFilm
        FROM ens2004.film f
        WHERE f.numFilm = numFilm
        AND f.realisateur = realisateur;
END;
```

Pas besoin d'expliquer ce qui différencie une fonction d'une procédure. Donc syntaxe.

```
//declaration d'une fonction
CREATE [OR REPLACE] FUNCTION nom_fonction([liste des parametres])
        RETURN type_retour IS|AS
        [variable de retour]
        [declaration des varibles]
BEGIN
        \\corps de la fonction
        RETURN variable_retour;
EXCEPTION \\facultatif
        \\definition des exceptions
END;
```

La principale différence c'est que l'on spécifie la variable que l'on va renvoyé. En ce qui concerne les paramètres d'une fonction, c'est la même chose qu'une procédure SAUF que l'on préfère ne spécifier que des paramètres d'entrées (IN).
Vu que vous êtes trop fort, on peut passer directement à un exemple ofc.

```
// on definit la fonction numFilm qui renvoie le numero d'un film
// en entree, on a le titre du film et le numero du realisateur
CREATE OR REPLACE FUNCTION numFilm (titreFilm IN VARCHAR2, realisateur IN
    INTEGER)
        RETURN INTEGER IS
          numFilm INTEGER;
BEGIN
        // on selectionne le numero du film dans numFilm
        SELECT f.numFilm INTO numFilm
        FROM ens2004.film f
        WHERE f.titre = titreFilm
        AND f.realisateur = realisateur
        // et on retourne numFilm qui contient le numero du film
        RETURN numFilm;
END;
```

1.2 Partie Java