PL/SQL : Les Packages

Les Packages

Un **package** en PL/SQL est une structure qui regroupe **plusieurs objets PL/SQL** (procédures, fonctions, types, curseurs, constantes, etc.) dans une unité logique. Il est divisé en deux parties :

- **Spécification (Specification)** : Contient la déclaration des objets accessibles depuis l'extérieur.
- **Corps (Body)** : Contient les définitions des objets déclarés dans la spécification.

Avantages des packages

- **Organisation** : Regroupe logiquement des objets liés.
- **Encapsulation** : Permet de cacher les détails d'implémentation tout en exposant une interface publique.
- **Performances** : Charge le package en mémoire lors de son premier appel, réduisant ainsi les accès répétés à la base de données.
- **Réutilisabilité** : Favorise la réutilisation du code.

Qui a le droit de créer un package?

- Lorsqu'un package est utilisé par plusieurs sessions, chaque session utilise sa propre copie des variables et des curseurs
- Un utilisateur doit posséder le privilège CREATE PROCEDURE pour créer un package qui utilise ses propres objets
- Un utilisateur doit posséder le privilège CREATE ANY PROCEDURE pour créer un package qui utilise n'importe quels objets

Les Packages

➢ Déclaration d'un package : Syntaxe

Un package se compose de deux parties :

1. **Spécification du package (CREATE PACKAGE)** : Contient uniquement les déclarations des objets (procédures, fonctions, variables, curseurs, etc.).
2. **Corps du package (CREATE PACKAGE BODY)** : Contient les définitions et l'implémentation des objets déclarés.

```
CREATE [OR REPLACE] PACKAGE nom-package[IS | AS]
[déclaration-de-variable;]  [déclaration-de-curseur;]
[déclaration-de-procédure;]  [déclaration-de-fonction;]
[déclaration-d'exception;]
END nom-package;
```

Les Packages

➢ **Syntaxe Spécification :**

```
CREATE [OR REPLACE] PACKAGE nom_package IS
    -- Déclarations publiques (accessibles depuis l'extérieur)
    PROCEDURE nom_procedure(param1 IN NUMBER);
    FUNCTION nom_fonction(param1 IN NUMBER) RETURN NUMBER;
    CONSTANTE CONSTANT NUMBER := 100;
    -- Déclarations de variables et curseurs
END nom_package;
```

Les Packages

➢ **Syntaxe Corps:**

```
CREATE [OR REPLACE] PACKAGE BODY nom_package IS
        PROCEDURE nom_procedure(param1 IN NUMBER) IS
    BEGIN
        -- Instructions
    END nom_procedure;
    FUNCTION nom_fonction(param1 IN NUMBER) RETURN NUMBER IS
    BEGIN
        -- Instructions
        RETURN param1 * 2;
    END nom_fonction;
END nom_package;
```

➢ Exemple complet : Spécification du package

```
CREATE OR REPLACE PACKAGE gestion_salaire IS
    PROCEDURE augmenter_salaire(emp_id IN NUMBER, pourcentage
                                IN NUMBER);
    FUNCTION total_salaire_service(service_id IN NUMBER)
    RETURN NUMBER;
END gestion_salaire;
```

Les Packages

➢ **Corps du package :**

```
CREATE OR REPLACE PACKAGE BODY gestion_salaire IS
    PROCEDURE augmenter_salaire(emp_id IN NUMBER, pourcentage IN NUMBER) IS
    BEGIN
        UPDATE employes
        SET salaire = salaire + (salaire * pourcentage / 100)
        WHERE id = emp_id;
    END augmenter_salaire;

    FUNCTION total_salaire_service(service_id IN NUMBER) RETURN NUMBER IS
        total NUMBER;
    BEGIN
        SELECT SUM(salaire) INTO total FROM employes WHERE id_service =
service_id;
        RETURN total;
    END total_salaire_service;
END gestion_salaire;
```

Les Packages

- Declaration D'un Package :Exemple Partie Corps

```
CREATE OR REPLACE PACKAGE BODY gestionV
IS
FUNCTION leReal(monFilmfilm.titre%TYPE)
RETURN individu.nomIndividu%TYPE IS
R individu.nomIndividu%TYPE;
 BEGIN
 SELECT nomIndividuI NTO R FROM individu
WHERE numIndividu= (SELECT realisateur
FROM film WHERE titre= monFilm);
RETURN R;
END leReal;
PROCEDURE
etablirAgenda(monClientclient.login%TYPE)
IS
CURSOR clientCURIS SELECT * FROM location
WHERE login= monClient;
BEGIN
FOR rIN clientCUR LOOP
IF r.dateLocation<= dateMax THEN
DBMS_OUTPUT.PUT_LINE
(r.dateLocation||r.dateEnvoi);
END IF;
END LOOP;
END etablirAgenda;
END gestionV;
```

Les Packages

Référence à un élément d'un package: pour référencer un élément (procédure, fonction, variable, ou constante) défini dans un **package**, il suffit d'utiliser la syntaxe suivante :

```
nom_package.nom_element;
```

Exemples de référence à des éléments dans un package

a) Appeler une procédure : Si le package s'appelle **gestion_salaire** et contient une procédure augmenter_salaire :

```
BEGIN    gestion_salaire.augmenter_salaire(101, 10);    END;
```

b) Appeler une fonction dans une requête SQL : Si le package contient une fonction total_salaire_service :

```
SELECT gestion_salaire.total_salaire_service(10) AS total FROM dual;
```