L3 – UE 10 Fondement des bases de données

TP4 : PL/SQL

Un compte rendu est à déposer sur Spiral sous la forme d'un fichier .sql commenté avant le 14/11, 23h59.

Nous considérons la base de données d'Oracle stockée dans le compte MPLANTEVIT :

Employee**(EmpNo, EName, Job, Mgr, Hiredate, Sal , Comm, DeptNo);**
SalaryGrade**(Grade, LoSal, HiSal);**
Department**(DeptNo, DName, Loc);**

Création de tables et ajout de contraintes :

1. Etudiez les tables.
2. Copier dans votre compte les trois tables de Scott présentées ci dessus.
3. Vous pouvez remarquer que la création par copie n'a pas ajouter de contraintes. EmpNo, Grade et DeptNo sont respectivement clés des tables Employee, SalaryGrade, Departement. De plus, on souhaite de Ename ne soit pas nul, Employee(DeptNo) C Departement(DeptNo). Enfin, LoSal doit toujours être inférieur à HiSal.

Premiers pas en PL/SQL :

1. Exécuter le bloc PL/SQL suivant :

```
SET SERVEROUTPUT ON
/*(commande d'environnement de SQL rendant visible le
     fonctionnement des procédures de DBMS_OUTPUT)*/
DECLARE
/*section déclarative des variables et curseurs du bloc PL/SQL,
facultative*/
     Nb NUMBER;
     BEGIN -- section obligatoire des instructions
     DBMS_OUTPUT.PUT_LINE('Hello World !!!') ;
 --  procédure du package DBMS_OUTPUT visualisant une donnée
     Nb := 10;
     DBMS_OUTPUT.PUT_LINE('Nb est ' || Nb);
-- || opérateur de concaténation
     END;
     /   /*(Ce caractère slash  lance l'exécution du bloc PL/SQL
dans la mémoire tampon SQL. Ne pas mettre d'espace avant).*/
```

2. Ecrire un bloc PL/SQL affichant le nombre de lignes de la table Emp. Il contiendra la commande SQL SELECT assignant une valeur à une variable (ou commande PL/SQL SELECT … INTO maVariable FROM …;) . On notera que la commande de SQL SHOW ERRORS est utile pour mettre au point un programme PL/SQL.

3. Ecrire une fonction (mémorisée) de paramètre un numéro de département et qui retourne le nombre de métiers (différents) de ce département. On pourra typer le numéro de département par Emp.DeptNo %TYPE. Puis appeler cette fonction stockée en utilisant un bloc PL/SQL.
Qu'indique la vue USER_OBJECTS ? La fonction créée et le bloc PL/SQL sont-ils des procédures stockées dans votre compte (stored procedure ou plus précisément stored function) ?

4. De la même façon, écrire une fonction qui étant donné un numéro de département retourne le nombre d'employés dans le département. Lever une exception lorsque le département n'existe pas et/ou le nombre d'employés est inférieur à 1.

Curseurs

1. A l'aide d'un curseur, pour chaque ligne de la table EMP, afficher une « phrase résumé » indiquant la situation de l'employé (ex: Employé MILLER(num:7782) travaille dans le département 10 depuis 23/01/82 pour 1300 euros par mois.)
2. Ajouter la colonne NbMétiers , qui contiendra le nombre des métiers de chaque département. On mettra à jour NbMétiers à partir de la table MPLANTEVIT.Employee d'autant de façons possibles : en utilisant la fonction stockée définie dans précédemment ou sans l'utiliser, en utilisant 0, 1 ou 2 curseurs associés à Emp ou Dept suivant l'exemple suivant :

Triggers

1. Mettre en oeuvre un déclencheur et une exception déclarée pour l'assertion « le salaire d'un employé ne peut pas diminuer ». On vérifiera son fonctionnement, puis on commentera ses propriétés à partir de la vue USER_TRIGGERS.
2. De la même façon, créer un déclencheur qui pour chaque mise à jour (insertion ou suppression ou modification) de l'affectation d'un employé, le déclencheur incrémente ou décrémente le nombre d'employés par département. Ajouter d'abord une colonne nbemployé dans la table Departement et la mettre à jour.
3. Créer un déclencheur qui pour chaque mise à jour de l'affectation d'un employé, modifiera le nombre de métiers par département.