5. PL/SQL

Exercice 5.0 –

Écrire les déclarations et l'exécution d'un programme qui affiche les nombres de 1 à 5 puis d'une procédure qui affiche les nombres de 1 à n (paramètre) et enfin d'une fonction qui incrémente n de j (2 paramètres).

Exercice 5.1 –L'entreprise employant les personnes de la table EMP est délocalisée des États-Unis en France (si, si). Il est donc nécessaire de convertir leur salaire et leur commission en euros (pour simplifier, on admettra 1,4 USD = 1€). Tous les employés voient également augmenter leur salaire de 15 % après conversion.

1) Écrire un programme PL/SQL qui place dans 2 variables a et b, les noms et job de l'employé 7521 et affiche : « l'employé s'appelle WARD et son travail est Salesman »
2) Écrire un programme qui affiche tous les noms des employés
3) Créer une nouvelle table vide EMP_FR de même structure que EMP. On pourra, par soucis de rapidité, recopier la table EMP dans EMP_FR pour en obtenir la structure, puis effacer tous les tuples de EMP_FR.
4) Écrire un programme PL/SQL permettant de recopier tous les tuples de la table EMP dans la table EMP_FR en effectuant au passage les opérations nécessaires sur le salaire et la commission. Traiter le cas où la table EMP est vide comme une exception.

Considérons la base de données dont le schéma et l'extension sont donnés ci-dessous.
EMP (**EMPNO**, ENAME, JOB, *MG*R, HIREDATE, SAL, COMM, *DEPTN*O)
DEPT (**DEPTNO**, DNAME, LOC)
Clés primaires , *Clés étrangères*
Exemple de la table EMP :

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17/12/90	800.00	NULL	20
7499	ALLEN	SALESMAN	7698	20/02/91	1600.00	300.00	30
7521	WARD	SALESMAN	7698	22/02/91	1250.00	500.00	30
7566	JONES	MANAGER	7839	02/04/91	2975.00	NULL	20
7654	MARTIN	SALESMAN	7698	28/09/91	1250.00	1400.00	30
7698	BLAKE	MANAGER	7839	01/05/91	2850.00	NULL	30
7782	CLARK	MANAGER	7839	09/06/91	2450.00	NULL	10
7788	SCOTT	ANALYST	7566	09/11/91	3000.00	NULL	20
7839	KING	PRESIDENT	NULL	17/11/91	5000.00	NULL	10
7844	TURNER	SALESMAN	7698	08/09/91	1500.00	0.00	30
7876	ADAMS	CLERK	7788	23/09/91	1100.00	NULL	20
7900	JAMES	CLERK	7698	03/12/91	950.00	NULL	30
7902	FORD	ANALYST	7566	03/12/91	3000.00	NULL	20
7934	MILLER	CLERK	7782	23/01/92	1300.00	NULL	10

Exercice 5.2-

On désire pouvoir consulter à tout instant les effectifs des différents groupes de TD en Licence d'Informatique. Il s'agit de définir un programme PL/SQL permettant l'insertion automatique de tuples dans une relation (table) GROUPE. Cette information est implicitement contenue dans la relation ETUDIANT. Le programme PL/SQL doit permettre de l'extraire et de réaliser les insertions correspondantes dans la relation GROUPE préalablement créée.

Pour définir ce programme, *ne pas utiliser de curseur* mais réaliser les étapes suivantes.
1) Définir en SQL*Plus la structure de la relation EDUDIANT. Elle doit au minimum comporter les champs suivants : numéro d'étudiant, nom et numéro de groupe. Peupler la relation ETUDIANT.
2) Définir en SQL*Plus la structure de la relation GROUPE, qui doit comporter seulement deux champs : numéro de groupe et effectif du groupe.
3) Définir un bloc PL/SQL intégrant les requêtes SQL nécessaires et permettant d'insérer dans la relation GROUPE les tuples constitués du numéro de groupe et de son effectif calculé.
5) Terminer le programme en validant la transaction par la commande COMMIT.
NB : Tenir compte de la possibilité de n'avoir aucun tuple dans la relation ETUDIANT. Dans ce cas, un groupe de valeurs nulles (NULL) doit être inséré dans la relation GROUPE.

Exercice 5.3 -
On souhaite gérer les résultats d'examens de l'UFR d'Informatique. Il s'agit de définir un programme PL/SQL permettant l'insertion automatique d'informations dans les relations RESULTAT et CLASSEMENT, à partir des données des relations NOTATION et MATIERE, qui contiennent respectivement des renseignements sur les notes obtenues par les étudiants et les coefficients affectés aux matières.
Pour définir ce programme, suivre les étapes suivantes.
1) Définir en SQL*Plus la structure des relations NOTATION et MATIERE :
• la relation MATIERE a pour champs un code matière, un coefficient pour le contrôle continu et un coefficient pour l'examen (par exemple, 0,2 CC / 0,8 exam.) ;
• la relation NOTATION a pour champs un numéro d'étudiant, un code matière, une note de contrôle continu et une note d'examen.
Peupler ces deux tables.
2) Définir en SQL*Plus la structure des relations RESULTAT et CLASSEMENT :
• la relation RESULTAT a pour champs un numéro, un nom d'étudiant, un code matière, ainsi qu'un champ note globale pour cet étudiant ;
• la relation CLASSEMENT a pour champs un numéro, un nom d'étudiant, une moyenne générale et un rang (place au classement).
3) Définir un programme PL/SQL permettant d'insérer dans RESULTAT tous les tuples constitués du numéro d'un étudiant, de son nom, du code d'une matière et de la note obtenue par *cet* étudiant dans *cette* matière. Le calcul de cette note doit tenir compte des coefficients de contrôle continu et d'examen définis pour la matière en question, ainsi que de la possibilité d'avoir des valeurs nulles pour les notes des étudiants, qui sont alors assimilées à 0 (utiliser la fonction NVL). Les tuples considérés doivent être extraits des tables NOTATION et MATIERE de manière itérative, grâce à un curseur adapté.
4) Terminer le traitement en réalisant l'insertion dans la relation CLASSEMENT des tuples constitués du numéro d'un étudiant, de son nom, de son prénom, de la moyenne générale obtenue dans toutes les matières par *cet* étudiant (ces informations doivent être extraites de la table RESULTAT) et de son rang (place), qui doit être calculé. Pour simplifier, on considère que toutes les matières sont équivalentes en termes de notes. Utiliser un curseur dans lequel les enregistrements sont triés.
Questions subsidiaires : Modifier le programme afin de prendre en compte :
• le cas où plusieurs étudiants ont le même rang
• le cas d'étudiants n'ayant aucune note (par défaut, leur moyenne générale est 0) ;
• la possibilité de n'avoir aucun tuple dans la relation ETUDIANT. Dans ce cas, un couple de valeurs

(0,'Aucun étudiant') doit être inséré dans la relation RESULTAT et le traitement doit s'arrêter.