Université Ibn Tofail **Département d'informatique**
Faculté des sciences – Kénitra **2017/2018**

Examen de la session de printemps - Rattrapage
Base de données II
Durée : 1 h : 30 min

N. B. : La qualité de la rédaction et la rigueur des raisonnements seront pris en compte dans la notation.

Exercice 1 : Gestion des notes

On souhaite gérer les résultats d'examens de la faculté des sciences. Il s'agit de définir un programme PL/SQL permettant l'insertion automatique d'information dans les relations RESULTAT et CLASSEMENT, à partir des données des relations ETUDIANT, NOTATION et MATIERE, qui contiennent respectivement des renseignements sur les étudiants, les notes obtenues par les étudiants et les coefficients affectés aux matières. Schéma de relation :

Etudiant (numetu, nom, prenom, datenaiss, rue, cp, ville, email)
Matiere (codemat, libelle, coefcc, coefexam)
Notation (numetu, codemat, notecc, notexam)

Pour définir ce programme, suivre les étapes suivantes :

1) Définir en SQL la structure des relations RESULTAT et CLASSMENT.
   - La relation RESULTAT a pour attributs un numéro d'étudiant, un nom d'étudiant, un code matière, ainsi qu'un attribut note globale pour cet étudiant ;
   - La relation CLASSEMENT a pour attributs un numéro d'étudiant, un nom d'étudiant, une moyenne générale et un rang (classement).
2) Définir un block PL/SQL anonymes, en utilisant un curseur, permettant d'insérer dans **RESULTAT** tous les n-uples 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.

   Note = (notecc*coefcc+notexam*coefexam)/(coefcc+coefexam)

   Terminer le traitement en réalisant l'insertion dans la relation **CLASSEMENT** des n-uplets 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 (classement), 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.

Exercice 2 : collection TABLE

Dans un bloc anonyme, définir un type collection de réels TABLE qui contiendra les notes de contrôle continu stockées dans la relation NOTATION de l'exercice 1. Définir une variable de ce type.
Dans la section de code, charger en mémoire, dans la collection, les notes de contrôle continues contenant dans la relation NOTATION à l'aide d'un curseur. Afficher les notes au fur et à mesure.
Calculer et afficher la moyenne des notes contenues dans la collection.
Afficher à l'écran l'indice du premier élément de la collection, l'indice du dernier élément et le nombre d'éléments dans la collection.
Supprimer les éléments d'indices1, 10 et 16.
Afficher tous les éléments de la collection.

Exercice 3 : Numérotation automatique de clé primaire à l'aide d'un déclencheur

Soit une table quelconque TABL, dont la clé primaire CLENUM est de type numérique. Définir un déclencheur avant insertion permettant d'implémenter une numérotation automatique de la clé. Le premier numéro doit être 1.

Exercice 4

Soit la table suivante :
METEO(NOM_VILLE, Température, Humidité)

1) Écrire une fonction PL/SQL qui prend en entrée le nom d'une ville et retourne la température et l'humidité de cette ville. Gérer aussi par une exception le cas où la ville n'existe pas.
2) Écrire un déclencheur qui avant l'insertion d'une nouvelle ville dans la table METEO vérifie :
   a. Si la température est la plus grande de toutes les villes, afficher un message d'avertissement.
   b. Si la ville existe déjà dans la table, ne pas l'insérer une nouvelle fois mais faire la mise à jour seulement.