TD N°2 : Bases de Données (4ème année)

Exercice 1 (Création et modification de la structure d'une table)

Soit la table suivante : ***Etudiant*** *(NumEtud, NomEtud, PrenEtud, AnneeUniv, Moyenne)*

Donner les requêtes SQL qui permettent de :

1- Créer la table *Etudiant*, sachant que :
   - *NumEtud* est un entier de 6 chiffres au maximum.
   - *AnneeUniv* (Année universitaire) est un entier de *4* chiffres.
   - *Moyenne* est la moyenne de fin d'année de l'étudiant.
   - Le champ *NomEtud* est obligatoire (définir une contrainte en lui donnant un nom).

2- Ajouter une contrainte d'intégrité qui permet d'imposer une valeur de *Moyenne* entre 0 et 20.

3- Ajouter à la table, le champ *DateNaisEtud* (Date de naissance de l'étudiant).

4- Supprimer le champ *DateNaisEtud*.

Exercice 2 (Insertion des données et manipulation des contraintes)

En considérant la table *Etudiant* de l'exercice précédent, donner les requêtes SQL permettant de :

1- Insérer dans la table, l'enregistrement (tuple) suivant : *(3618, 'Alami', 'Mohamed', 2020, 15.25)*

2- Désactiver la contrainte sur le champ *NomEtud*, définie dans l'exercice précédent, pour qu'il ne soit plus obligatoire.

3- Insérer le deuxième tuple suivant : *NumeroEtud=4253, PrenEtud='Said', Moyenne=13.75* (on laissera le champ *NomEtud* vide).

4- Activer la contrainte déjà désactivée à la question 2. Cette requête pourrait-elle avoir lieu ? Justifier. Que faire pour qu'elle ait lieu ?

5- Supprimer la contrainte définie sur le champ *NomEtud* de l'exercice précédent.

6- Supprimer la table *Etudiant*.

Exercice 3 (Requêtes de sélection de données (mono et multi-tables) vérifiant des critères)

Soit une Entreprise marocaine composée de plusieurs filiales (sous-sociétés) dont chacune opère dans un secteur industriel différent et sise dans une ville donnée.

Considérant les deux tables suivantes de son schéma relationnel :

Employes *(NumEmp, NomEmp, Fonction, Salaire, DateRecrut, NumSup#, CodeFil#)*

Filiale *(CodeFil, NomFil, DateCreatFil, Secteur, CapitalFil, VilleFil)*

Notons que *NumSup* est le numéro du supérieur hiérarchique direct de l'employé. Normalement, au niveau du MCD, on avait une association réflexive *SupHierarch* de la table *Employes* avec elle-même (ayant comme cardinalités : *(0,1)* et *(0,n)*), qui indiquait une association entre un employé et son

supérieur hiérarchique direct (un seul), employé lui aussi. Au passage au MLD, l'association a disparu et la clé *NumEmp* a migré dans la table *Employes* pour devenir clé étrangère. Elle a été surnommée *NumSup*.

Donner les requêtes SQL permettant de :

1- Créer, à partir de la table *Employes*, une nouvelle table *EmpInovIndus* contenant uniquement les employés de la filiale *InovIndus* (*NomFil='InovIndus'*). Sa structure est la même que *Employes* et le contenu est extrait à partir de la table *Employes* en vérifiant la condition adéquate.

2- Afficher les filiales pour lesquelles on n'a pas encore recruté d'employés (nouvelles filiales).

3- Afficher les employés qui ne sont pas encore affectés à aucune filiale.

4- Afficher tous les employés qui travaillent dans la ville de '*Tanger'*.

5- Donner le nom de chaque employé avec le nom et la fonction de son supérieur hiérarchique.

6- Afficher, en utilisant les requêtes imbriquées, les employés qui gagnent plus que monsieur '*Alami'*. On suppose qu'il y a un seul employé dont le nom est *'Alami'*. Traiter le cas contraire.

7- Répondre à la question précédente en utilisant les jointures.

8- Afficher les employés qui ne travaillent pas dans la même filiale que leur supérieur hiérarchique (direct).

9- Afficher les employés qui ont des subordonnés ;

10- Trier les employés par ordre décroissant de leurs salaires. En cas d'égalités des salaires, classer, par ordre alphabétique, les employés concernés.

Exercice 4 (Fonctions d'agrégats (groupes))

1- Afficher les fonctions de chaque filiale.

2- Afficher le nombre d'employés par filiale en triant le résultat par ordre croissant du *CodeFil.*

3- Afficher le nombre de fonctions de l'entreprise (de toutes les filiales).

4- Afficher le nombre d'employés de chaque fonction.

5- Afficher le nombre d'employés, recrutés à partir de 01/01/2012, pour chaque fonction.

6- Afficher le nombre de fonctions de chaque filiale (codeFil : nb de fonctions)

7- Afficher le nombre de fonctions de chaque filiale (NomFil : nb de fonctions)

8- Afficher le min, le max, la somme et la moyenne des salaires par filiale (*CodeFil*)

9- Afficher le min et la moyenne des salaires par Fonction.

10- Afficher le nombre de subordonnés de chaque employé (nombre d'employés dont il est supérieur hiérarchique direct). On affichera le *nom* (et non pas son numéro) de l'employé et le nombre de ses subordonnés.

11- Afficher les codes des filiales ayant plus de 100 employés. Le résultat est à trier par ordre croissant des codes des filiales.

12- Afficher les codes des filières disposant de plus de 50 ingénieurs. Le résultat est à trier par ordre croissant des codes des filiales.

Exercice 5 (Requêtes de mise à jour de données)

1- Augmenter de *15%*, les salaires inférieurs à *15000 DH* des employés travaillant dans la ville de *Casablanca* ;
2- Supprimer tous les employés de la filiale *InovIndus* (*NomFil= 'InovIndus*').

Exercice 6 : (Examen ENSAM-Meknès 2019/2020)

Nous nous intéressons à une partie de la base de données d'une clinique composée des tables suivantes :

Patient (NUMPAT, NOMPAT, PRENPAT, DATENAISPAT, VILLE) ;
Operation (CODEOP, NOMOP, PRIXOP) ;
Medecin (CODEMED, NOMMED, PRENMED, SPECIALITE, TELMED, MAILMED) ;
Faire_Op (NUMPAT, CODEOP , DATEOP, DUREEOP, CODEMED).
……..

Donner les requêtes SQL permettant de :

1. Créer les tables (avec les contraintes des clés primaires et étrangères) *Patient*, *Operation*, et *Faire_OP*. (NB : *DUREEOP* est donnée en minutes).
2. Afficher la liste des opérations triées par ordre décroissant de leurs *prix*, et en cas d'égalité triées par ordre croissant de leurs *noms*.
3. Afficher les opérations dont le nom contient le mot "*fibroscopie*" et dont le prix est supérieur ou égal *1000* DH.
4. Diminuer de *10%* les prix des opérations qui n'ont jamais été effectuées.
5. Afficher pour chaque opération le nombre de fois qu'elle a été effectuée.
6. En utilisant les jointures, afficher le nom de chaque opération effectuée, le nom du patient concerné, et le nom de son médecin responsable.
7. Afficher, en utilisant les requêtes imbriquées, les *noms* et *prénoms* des patients qui ont fait des opérations dont le *prix* est inférieur ou égal à 5000 DH et dont la *durée* a dépassé *1h30*.
8. Afficher les médecins ayant effectué, plus de *20* opérations au profit des patients de la ville de "*MEKNES*" durant l'année 2019.