Les Bases de Données : SQL

Série d'exercices N° 3

Exercice 1 :

Soit la base de données FPJ suivante :

- FOURNISSEUR (**codfrs**, nomfrs, villefrs, telfrs)
- PROJET (**codproj**, nomproj, villeproj, budjetproj, #coddirecteur)
- DIRECTEUR (**coddirecteur**, nomdirecteur)
- PIECE (**codpiece**, nompiece, couleurpiece, poidspiece, villepiece)
- FPJ (**#codfrs, #codpiece, #codproj**, qtelivree, dateliv)

1. Créer la base de données ci-dessus tout en créant toutes les contraintes d'intégrités nécessaires.
2. Insérer trois lignes dans chaque table.
3. Formuler les requêtes suivantes en SQL :
   1) Donner les numéros des pièces destinées à tout projet se déroulant dans la même ville que celle où se situe le fournisseur de ces mêmes pièces.
   2) Donner les numéros des projets dont au moins un des fournisseurs ne se trouve pas dans la même ville que celle où le projet se déroule.
   3) Donner les numéros des projets utilisant au moins une des pièces fournies par 'F3'.
   4) Quels sont les projets dont la deuxième lettre de leur nom est 'E' ?
   5) Quels sont les projets dont le nom de leur directeur se termine par 'A' ?
   6) Combien de fois chaque pièce a-t-elle été livrée ?
   7) Combien de livraisons ont été effectuées entre le 01/01/19 et 01/01/20 par le fournisseur 'F2' ?
   8) Quelle est la pièce qui a la plus grande quantité livrée pour le projet 'P1 ' ?
   9) Quel est le poids de la pièce qui a été livrée le plus de fois ?
   10) Quelle sont les pièces qui n'ont été jamais livrées à des projets se déroulant à 'Tunis ' ?
   11) Quels sont les projets auxquels on a livré toutes les pièces ?
   12) Ajouter 1000 aux budgets des projets se déroulant à Tunis et qui ont reçu plus de 10 livraisons de fournisseurs n'habitant pas Tunis.
   13) Changer les couleurs de toutes les pièces rouges en orange.
   14) Supprimer tous les projets pour lesquels il n'y a pas de livraison.
   15) Augmenter de 10% toutes les livraisons effectuées par les fournisseurs de pièces détachées rouges.

Exercice 2 :

Soit la base de données relationnelle « Compte-bancaire » suivante :

1. CLIENT ((**numcli**, entier), (nomcli, text), (prencli, text), (adr, text))
2. PERSONNEL (**numpers**, entier), (nompers, text), (prenpers, text), (manager, entier), (sal, nombre (>=1250)))
3. TYPCOMPT ((**numTypC**, entier), (nomTypC, text))
4. COMPTE ((**numcpt**, entier), (#**numcli**, entier), (#numTypC, entier), (dateOuv, date default date_système), (#numPers, entier))
   - NB : numPers = numéro de l'employé qui s'occupe du compte.
5. TYPOPER ((**numTypO**, entier), (nomTypO, text))
6. OPERATION ((**numOp**, entier), (#numcpt, entier), (#numcli, entier), (#numTypO, entier), (dateOp, date default date_système), (montant, réel not null (< > 0))
7. PERSONNE ((**numP**, entier), (nomP, text not null), (PrenP, text), (#père, entier), (#mère, entier))
   - NB : les colonnes père et mère font référence à la colonne **nump** de la table PERSONNE.

Travail à faire :

1. Ecrire un script Python permettant de copier les 20 premiers enregistrements de la table Personne dans la table Client.
2. Ecrire un script Python permettant d'ouvrir un « compte courant » (NomTypC) pour chaque client et d'enregistrer un « versement » (nomTypO) de 500 dinars.
3. Ecrire une fonction Python qui reçoit, comme paramètres, 2 numéros de personnes et retourne vrai s'ils sont frères et faux sinon.
4. Un client peut avoir plusieurs comptes. Ecrire un script Python qui affiche, pour chaque client, son numéro, son nom, son prénom et la liste des comptes qu'il possède (NumCpt, DateOuv).