Devoir Maison Terminale NSI - A rendre le lundi 30 septembre 2024

EXERCICE 1

Cet exercice porte sur la notion de base de données relationnelle et le langage SQL.

On pourra utiliser les mots-clés `SQL` suivants : `AND`, `FROM`, `INSERT`, `INTO`, `JOIN`, `ON`, `SELECT`, `SET`, `UPDATE`, `VALUES`, `WHERE`.

Un grand magasin de meubles propose à ses clients un large choix de meubles. Les informations correspondantes sont rangées dans une base de données composée de trois relations.

Voici le schéma de deux de ces relations :

- `Clients (id, nom, prenom, adresse, ville)`
- `Commandes (id, #idClient, #idMeuble, quantite, date)`

Dans ce schéma :

- la clé primaire de chaque relation est définie par les attributs soulignés ;
- les attributs précédés de `#` sont les clés étrangères.

La troisième relation est appelée `Meubles` et concerne les meubles du magasin.
Le tableau de la figure 1 ci-dessous en présente un extrait :

id	intitule	prix	stock	description
62	'skap'	69.99	2	'Armoire blanche 3 portes'
63	'skap'	69.99	3	'Armoire noire 3 portes'
74	'stol'	39.99	10	'Chaise en bois avec tissu bleu'
98	'hylla'	99.99	0	'Bibliothèque 5 étages blanche'

Figure 1 – Extrait de la relation `Meubles`

1. Dans cette question, on s'intéresse au modèle relationnel.

a. Donner la caractéristique qu'un attribut doit avoir pour être choisi comme clé primaire.

b. Expliquer le rôle des deux clés étrangères de la relation `Commandes`.

c. Donner le schéma relationnel de la relation `Meubles` en précisant la clé primaire et les éventuelles clés étrangères.

2. En vous basant uniquement sur les données du tableau de la figure 1, donner le résultat de la requête suivante :

```
SELECT id, stock, description
FROM Meubles
WHERE intitule = 'skap';
```

3. Donner la requête SQL permettant d'afficher les noms et prénoms des clients habitant à Paris.

4. Le magasin vient de recevoir des meubles dont l'intitulé est `'hylla'` et dont l'attribut `id` dans la relation `Meubles` vaut `98`. Le stock de ces meubles est alors de `50`.
Recopier et compléter la requête SQL ci-dessous qui permet de mettre à jour la base de données.

```
UPDATE ...
SET ...
WHERE ...
```

5. Le magasin vient d'ajouter à son catalogue un nouveau meuble dont les caractéristiques sont les suivantes :

id	intitule	prix	stock	description
65	'matta'	95.99	25	'Tapis vert à pois rouges'

Donner la requête SQL qui permet d'ajouter cet article à la relation `Meubles`.

6. Donner la requête SQL permettant de récupérer le nom et le prénom des différents clients qui ont passé une commande le 30 avril 2021.
On précise que, dans la relation `Commandes`, les dates sont des chaînes de caractères, par exemple `'21/08/2002'`.

EXERCICE 2

Soit `tab` un tableau non vide d'entiers triés dans l'ordre croissant et `n` un entier.

La fonction `chercher` ci-dessous doit renvoyer un indice où la valeur `n` apparaît dans `tab` si cette valeur y figure et `None` sinon.

Les paramètres de la fonction sont :

- `tab`, le tableau dans lequel s'effectue la recherche ;
- `x`, l'entier à chercher dans le tableau ;
- `i`, l'indice de début de la partie du tableau où s'effectue la recherche ;
- `j`, l'indice de fin de la partie du tableau où s'effectue la recherche.

L'algorithme demandé est une recherche dichotomique récursive.

Recopier et compléter le code de la fonction `chercher` suivante :

```
def chercher(tab, x, i, j):
    '''Renvoie l'indice de x dans tab, si x est dans tab,
    None sinon.
    On suppose que tab est trié dans l'ordre croissant.'''
    if i > j:
        return None
    m = (i + j) // ...
    if ... < x:
        return chercher(tab, x, ... , ...)
    elif tab[m] > x:
        return chercher(tab, x, ... , ...)
    else:
        return ...
```

Exemples :

```
>>> chercher([1, 5, 6, 6, 9, 12], 7, 0, 10)
>>> chercher([1, 5, 6, 6, 9, 12], 7, 0, 5)
>>> chercher([1, 5, 6, 6, 9, 12], 9, 0, 5)
4
>>> chercher([1, 5, 6, 6, 9, 12], 6, 0, 5)
2
```