

Exercice 1 (Conception de bases de données et SQL (8 points))

L'exercice porte sur une gestion simplifiée d'une bibliothèque

Le concepteur prévoit 4 tables, Oeuvre, Livre, Adhérent, et Emprunt.

```
CREATE TABLE Adherent (  
    ID int(11) NOT NULL,  
    Nom text NOT NULL,  
    Prenom text NOT NULL,  
    Adresse text NOT NULL,  
    Ville text NOT NULL,  
    'Code postal' int(5) NOT NULL,  
    email text NOT NULL,  
    Tel text NOT NULL,  
    NbreEmprunts int(4) NOT NULL,  
    DureeEmprunt int(4) NOT NULL;  
    PRIMARY KEY (ID));  
  
CREATE TABLE Oeuvre (  
    ISBN text NOT NULL,  
    Titre text NOT NULL,  
    Auteur text NOT NULL,  
    Annee int(4) NOT NULL,  
    PRIMARY KEY (ISBN));  
  
CREATE TABLE Livre (  
    ID int(11) NOT NULL,  
    ISBN text NOT NULL,  
    Date text NOT NULL,  
    PRIMARY KEY (ID),  
    FOREIGN KEY (ISBN) REFERENCES Oeuvre(ISBN));  
  
CREATE TABLE Emprunt (  
    ID int(11) NOT NULL,  
    Livre int(11) NOT NULL,  
    Adherent int(11) NOT NULL,  
    DateSortie date NOT NULL,  
    DateEntree date NOT NULL,  
    PRIMARY KEY (ID),  
    FOREIGN KEY (Livre) REFERENCES Livre(ID),  
    FOREIGN KEY (Adherent) REFERENCES Adherent(ID),  
    UNIQUE (Livre, Adherent));
```

Question 1.1 (0,5 point)

Pour la table **Oeuvre**, donner la seule dépendance fonctionnelle déclarée.

Question 1.2 (1 point)

Pour la table **Emprunt**, donner les seules dépendances fonctionnelles déclarées. Expliquer à quoi correspond la déclaration **UNIQUE(Livre, Adherent)**. Est-ce souhaitable pour la bonne gestion de la bibliothèque ?

Question 1.3 (1 point)

Pour la relation **Emprunt**, dire si elles sont 3NF et/ou BCNF en justifiant la réponse.

Question 1.4 (1 point)

Après en avoir donné une écriture algébrique, écrire une requête SQL qui caractérise le livre emprunté par l'adhérent M. Michel.

Question 1.5 (1,5 point)

On souhaite ajouter une contrainte pour exprimer le fait que la date d'entrée d'un livre ne précède pas celle de sortie. Ecrire cette contrainte en algèbre relationnelle.

Question 1.6 (1,5 point)

On souhaite maintenant ajouter une contrainte pour exprimer le fait qu'un livre ne peut pas être emprunté tant qu'il est sorti. Ecrire cette contrainte en algèbre relationnelle.

Question 1.7 (1,5 points)

Donner une écriture algébrique qui liste les adhérents qui empruntent ou ont emprunté tous les livres de Victor Hugo.

Exercice 2 (Normalisation (9 points))

Pour minimiser le nombre de tables, le concepteur de la base de données réécrit la relation **Emprunt** en y faisant figurer directement les informations sur le livre emprunté et l'adhérent.

Emprunt(Nom, Prenom, Adresse, ISBN, Titre, Auteur, Annee, DateSortie, DateEntree)

où le triplet (Nom, Prenom, ISBN) est unique dans la relation et où l'on a les dépendances fonctionnelles suivantes :

- ISBN $\rightarrow$ Titre, Auteur, Annee
- Nom, Prenom $\rightarrow$ Adresse
- DateSortie $\rightarrow$ DateEntree
- Titre, Auteur $\rightarrow$ Annee

Question 2.1 (2 point)

Ecrire l'ensemble irréductible des dépendances fonctionnelles équivalent.

Question 2.2 (1,5 point)

Donner toutes les clefs candidates de la relation.

Question 2.3 (1,5 point)

Justifier pourquoi la relation n'est pas en troisième forme normale. Donner un exemple où cela peut engendrer un problème de redondance dans la base de données.

Question 2.4 (2 points)

Procéder à une décomposition sans perte d'information de la relation de sorte qu'elle soit en troisième forme normale.

Question 2.5 (2 points)

Les relations obtenues sont-elles en BCNF ? Dans le cas contraire, procéder à une décomposition sans perte d'information de la relation de sorte qu'elle le soit. Est-ce que cette décomposition engendre des pertes de dépendances fonctionnelles ?

Exercice 3 (Reprise après panne. (3 points))

Nous supposons que le SGBD gère les reprises après pannes système.

Pour chaque transaction, il écrit dans un journal (*log*) le début, l'ensemble des effets des requêtes et la fin.

Nous supposons que le journal est intégralement sauvegardé sur disque à tout moment et que le SGBD, à intervalles réguliers (appelés points de synchronisation), sauvegarde la base de données dans son état.

Soit tc le dernier point de synchronisation et $tf > tc$ l'instant où une panne système survient.

Question 3.6 (1 point)

Dire si la base de données est dans un état cohérent et pourquoi.

- À l'instant tf .
- À l'instant tc .

Question 3.7 (1 point)

Soit une transaction $T1$ qui a commencé avant tc et n'a jamais été terminée avant l'instant de la panne tf . Est-ce que $T1$ doit être rejouée (*REDO*) ou annulée (*UNDO*) sur la base de donnée sauvegardée ?

Question 3.8 (1 point)

Même question avec une transaction qui a commencé après tc et s'est terminée correctement avant tf .