Université Nice Sophia Antipolis
Polytech Nice Sophia
SI3
2016–2017

Contrôle continu de Base de données
13 Janvier 2017

Durée : 2 heures

1.1		3
1.2		4
2.1		4
2.2		9

Seul les support de cours et de TD sont autorisés.

1 Normalisation

1.1 Clés

Soit $R_0(A,B,C,D,E,F)$ une relation avec l'ensemble de dépendances suivant :
$DF = \{AB \rightarrow C,\ C \rightarrow D,\ BD \rightarrow F,\ E \rightarrow F\}$
Est ce que $\{A,B,C\}$ est une super-clé de R_0 ? Pourquoi ?

Non, car $\{A,B,C\}^+ = \{A,B,C,D,F\} \neq R_0$

Donnez la ou les clés de R_0 et justifiez votre réponse?

$\{A,B,E\}$ est la seule clé de R_0:
- A, B et E ne sont déterminés par aucun élément et sont donc nécessairement éléments de la clé;
- $\{ABE\}^+ = \{A,B,C,D,E,F\} = Attr(R_0)$

1.2 Forme Normale

Soit $R_1(\underline{TitreLivre,NomAuteur},PrixLivre,DatePublication,AdresseAuteur)$ une relation qui porte sur une bibliothèque avec l'ensemble de dépendances suivant :
$TitreLivre,NomAuteur \rightarrow PrixLivre,DatePublication,AdresseAuteur$
$NomAuteur \rightarrow AdresseAuteur$

- Quelle est la forme normale de R_1 ? Justifiez votre réponse.

  R_1 est en *1NF*, mais pas en *2NF* à cause de la dépendance fonctionnelle $NomAuteur \rightarrow AdresseAuteur$

- On décompose la relation R_1 en R_{11} et R_{12} :
  $R_{11}(\underline{TitreLivre,NomAuteur},PrixLivre,DatePublication)$ et
  $R_{12}(\underline{NomAuteur},AdresseAuteur)$

Quelles sont les formes normales des relations R_{11}, et R_{12} ? Justifiez votre réponse.

- R_{11} a pour clé ($TitreLivre$,$NomAuteur$) et est en *2NF* et en *3NF*
- R_{12} a pour clé $NomAuteur$. Comme R_{12} n'a que deux attributs, elle est nécessairement en *3NF*.

---

2 SQL

2.1 Modélisation

On veut modéliser une bibliothèque dont les composants sont les livres, les copies des livres et les auteurs. Les relations suivantes sont fournies par l'utilisateur:

- `livre(IdLivre,Titre, DatePubli, Prix, NombreCopies)` où `IdLivre` détermine tous les autres attributs;
- `copie(IdCopie,IdLivre, Statut)` où `IdCopie` détermine tous les autres attributs; le statut est soit libre (valeur l), emprunté (valeur e), indisponible (valeur i);
- `auteur(IdAuteur,Nom, Prenom, Adresse)` où `IdAuteur` détermine tous les autres attributs.

Un livre peut avoir *plusieurs* auteurs et un auteur peut avoir écrit *plusieurs* livres.
Donnez les tables SQL nécessaires pour stocker ces objets et *leurs relations*, en explicitant les clés primaires, les clés externes et les contraintes sur le statut.

```
DROP TABLE IF EXISTS auteurlivre, copie, livre, auteur ;

CREATE TABLE livre (
       IdLivre INT PRIMARY KEY ,
       Titre CHAR(2) NOT NULL,
       DatePubli DATE NOT NULL,
       Prid INT  NOT NULL
       );

CREATE TABLE copie (
       Idcopie  INT PRIMARY KEY,
       IdLivre INT  NOT NULL REFERENCES livre,
       Statut CHAR(1)  NOT NULL CHECK (Statut in ('l', 'e', 'i'))
       );
CREATE TABLE auteur (
       IdAuteur INT PRIMARY KEY,
       Nom CHAR(20) NOT NULL,
       Prenom  CHAR(20),
       Adresse CHAR(200) NOT NULL
       );
CREATE TABLE auteurlivre (
       IdAuteur INT  REFERENCES auteur,
       IdLivre INT   REFERENCES livre,
       PRIMARY KEY (IdAuteur,IdLivre)
       );
```

2.2 Requêtes

On veut modéliser une bibliothèque dont les composants sont: les livres et les auteurs. Contrairement à la question précédante, on suppose ici *qu'un livre n'a qu'un seul auteur*. On utilisera donc les tables ci-dessous pour répondre aux différentes questions.

```
CREATE TABLE auteur (
       IdAuteur INT PRIMARY KEY,
       Nom CHAR(20) NOT NULL,
       DateNaissance DATE NOT NULL,
       DateDeces DATE
       );
CREATE TABLE livre (
       IdLivre INT PRIMARY KEY ,
       Titre CHAR(20) NOT NULL,
       IdAuteur INT  REFERENCES auteur,
       Prix INT NOT NULL,
       DatePubli DATE NOT NULL,
       NbEmprunts INT
       );
```

Exprimez les requêtes suivantes en SQL :

1. Affichez le titre et le prix des livres dont le prix est inférieur à 10€.

```
SELECT L.Titre, L.Prix FROM livre L WHERE L.Prix < 10;
```

---

2. Affichez les noms des auteurs n'ayant publiés aucun livre.

```
SELECT nom FROM auteur
    WHERE IdAuteur NOT IN (SELECT IdAuteur FROM livre);
```

---

3. Affichez les noms des auteurs et le prix moyens de leurs livres pour les auteurs qui ont publiés au moins quatre livres.

```
SELECT A.Nom, AVG(L.Prix) AS PrixMoyen
       FROM auteur   A, livre L
       WHERE A.IdAuteur = L.IdAuteur
       GROUP BY A.IdAuteu, A.Nom
       HAVING COUNT(*) >= 4;
```

---

4. Affichez le titre et la date de publication pour les livres publiés avant l'an 2000 et dont l'auteur est toujours vivant.

```
SELECT L.Titre, L.DatePubli  FROM livre L
       WHERE L.DatePubli < '01/01/2000'
       AND IdAuteur IN
         (SELECT IdAuteur FROM auteur A WHERE  A.DateDeces IS NULL);
```

---

5. Remarquant que la date de naissance de certains auteurs est aussi la date de la mort d'un ou de plusieurs autres auteurs, lesquels à leur tour peuvent être nés le jour de la mort d'un ou plusieurs auteurs on désire afficher pour chaque auteur combien de fois on peut remonter

dans le temps sur ces coïncidences de dates. Par exemple si a est né le jour de la mort de b et de c (et de personne d'autre) et que b est né le jour de la mort de d , que ni c ni d ne sont né le jour de la mort d'un auteur pour l'auteur a on peut remonter deux fois dans le temps.

```
WITH RECURSIVE PL(Idauteur, Idancetre, Generation) AS (
SELECT A1.IdAuteur, A1.IdAuteur, 0  FROM auteur A1
union
Select PL.Idauteur, A2.Idauteur, Generation+1 from PL, auteur A2 , auteur A3
       WHERE A3.IdAuteur=PL.IdAncetre and A3.Datenaissance=A2.DateDeces)

       SELECT Idauteur, Max(Generation) FROM PL group by Idauteur;

ou
WITH RECURSIVE PL(Idauteur, Idancetre, Datepivot, Generation) AS (
SELECT A1.IdAuteur, A1.IdAuteur, A1.DateNaissance, 0  FROM auteur A1
union
SELECT PL.Idauteur, A2.Idauteur, A2.DateNaissance, Generation+1 from PL, auteur A2
       WHERE A2.DateDeces=PL.Datepivot)

       SELECT Idauteur, Max(generation) FROM PL group by Idauteur;
```
