Université Paris 7
Master 1 Informatique, Bases de données avancées.
13 janvier 2011
Durée : 3 heures. Documents manuscrits, notes de cours, notes de TD/TP autorisés. Livres et ordinateurs interdits.

Le sujet comporte 5 pages.

Dépendances

Exercice 1 On considère une relation $R(A, B, C, D)$ qui satisfait les dépendances fonctionnelles suivantes :

$$A \to B$$
$$B \to C$$
$$A \to C$$
$$B, D \to A$$

Dans la suite on note cette ensemble de dépendances F.

Question 1: Calculer la fermeture $(A)^+$.

Question 2: Donner toutes les clés candidates de la relation R. Justifier. (Justifier ne veut pas dire qu'il faut calculer systématiquement toutes les clôtures possibles.)

Question 3: Est-ce que dans F il y a une dépendance (ou des dépendances) qu'on peut enlever sans que cela change la clôture ? C'est-à-dire trouver le plus petit ensemble G de DF tel que $G^+ = F^+$.

Question 4: Est-ce que R est en 3NF ? Justifier.
Si R n'est pas 3NF alors décomposer pour obtenir les relations 3NF.
Est-ce votre décomposition est sans perte d'information ? Est-ce qu'elle préserve les dépendances fonctionnelles ?

Exercice 2 Nous avons une relation $R(A, B, C, D, E)$ avec les dépendances :

$$A \to B,$$
$$B, C \to D$$
$$B, C \to E$$

Question 1: Trouver les clés candidates.

Question 2: Est-ce que R est en BCNF ? Si elle ne est pas BCNF décomposer la en relations BCNF.

Modélisation

Exercice 3 Votre tâche consiste à modéliser [1] un Système d'information pour un grand port de marchandises. Le port accueille uniquement des containers. Pour simplifier on suppose que

1. partiellement, sinon cela devient trop complexe

la position exacte de container dans le port est donnée par un couple d'entiers non négatifs. Le premier entier x identifie la position de container le deuxième entier h donne la position en hauteur (on empile de containers un sur l'autre).
Pour chaque container on veut garder en plus des informations suivantes :
- expéditeur et destinataire,
- le transporteur qui a acheminé le container le port, ça peut être soitun transporteur terrestre si le container est arrivé sur un camion soit un bateau si le container est arrivé par la mer,
- la date d'arrivée,
- le port de depart pour les containers arrivés par la mer,
- le transporteur qui prendra le container pour le sortir de port, encore une fois cela peut être un transporteur terrestre ou un bateau,
- le poids (important pour rééquilibrer le bateau),
- type de marchandise stocké dans le container (plusieurs types possibles pour le même container (par exemple la moitié de container contient des télés et l'autre moitié de vêtements).

La liste ci-dessus n'est pas forcement exhaustive et elle mélange pêle-mêle des attributs et des entités. Vous pouvez ajouter à ce Système d'information d'autres informations qui vous paraissent pertinentes. Vous pouvez aussi le modifier, j'ai donné juste de suggestions.

CONTRAINTE Terre-Terre :
Notez qu'une de contraintes du système est que aucun container ne peut pas venir par la voie terrestre pour ensuite partir par la voie terrestre. Un container arrivé par terre doit partir par la mer et l'inverse [2]. Donc il convient bien de distinguer le transporteurs terrestres et des bateaux.

Question 1: Faire un modèle qui convient pour représenter les activités du port N'oubliez pas de mettre quelques attributs pour chaque entité (entité sans attributs ne sert à rien). Marquez de façon lisible et non ambiguë les cardinalités pour chaque relation.

Question 2: Quelles tables on obtient à partir de votre modèle ? (S'il a trop de tables à faire il suffit de donner quelques tables pertinentes et indiquer au moins les clés et les clés étrangères pour ces tables. Cette question sert à vérifier si vous savez passer de modèle graphique vers les tables.)
Remarque : C'est un exercice qui apporte des points s'il est bien fait. Par contre si mal fait cela ne rapportera pas grand chose. Et pour bien le faire il faut du temps. Peut-être à faire à la fin si le temps reste ?
En tout cas il vaut mieux bien modéliser sans faire de tables que mal modéliser et faire mal les tables par la suite (moins mais bien fait c'est mieux que plus mais mal fait).

Triggers - déclencheurs

Exercice 4
Un système de gestion d'une bibliothèque contient deux tables [3] :

```
lecteur( id_lecteur int primary key,
         nom varchar(50) not null,
         interdit DATE
```

2. C'est un port, les containers qui arrivent et partent par terre n'ont rien à faire ici. Par contre un container arrivé par la mer peut aussi partir par la mer.

3. Le système d'information de la bibliothèque contient d'autres tables mais dans cet exercice uniquement deux tables nous intéressent.

```
)

livre(   id_livre int primary key,
         id_lecteur int foreign key references lecteur,
         emprunt DATE,
         retour  DATE,
         empruntable boolean
)
```

lecteur La table `lecteur` sert, on le devine, à recenser les lecteurs. Les attributs `id_lecteur` et `nom` de la table `lecteur` ont la signification qu'on attend. L'attribut `interdit` de `lecteur`, s'il n'est pas NULL, indique jusqu'à quel jour le lecteur est interdit d'emprunt suite à un retour en retard. Les explications détaillées concernant cet attribut suivent plus loin.

livre La table `livre` sert à enregistrer tous les livres et leurs emprunts. Ici "livre" signifie "exemplaire de livre", chaque exemplaire a son propre numéro `id_livre` qui permet de l'identifier.
Si `id_lecteur` est non NULL alors l'attribut `id_lecteur` donne l'identifiant du lecteur qui a emprunté le livre, par contre si `id_lecteur` est NULL alors le livre se trouve dans le rayon.
Si `id_lecteur` est non NULL (le livre est emprunté) alors l'attribut `emprunt` donne la date d'emprunt et l'attribut `retour` donne la date avant laquelle le livre doit être rendu.
Si `id_lecteur` est NULL alors les valeurs de `emprunt` et `retour` non pas d'intérêt et peuvent être quelconque.
Si `empruntable` est `TRUE` alors le livre est empruntable sinon il ne l'est pas.

Question 1: L'emprunt de livre s'effectue en modifiant l'enregistrement qui correspond à ce livre dans la table `livre` à l'aide de la requête suivante :

```
UPDATE TABLE livre
SET  id_lecteur=id_de_lecteur_qui_emprunt
WHERE id_livre=id_de_livre_a_emprunter;
```

If faut écrire un trigger déclenché par cet UPDATE (en fait par chaque UPDATE sur `livre`). Le trigger doit vérifier s'il s'agit d'emprunt ou d'une autre action et vérifier si les conditions autorisant emprunt sont toutes réunies. Vous devez écrire aussi bien la fonction qui réalise le trigger que la requête `CREATE TRIGGER` associée.
On considère qu'un `UPDATE` est un UPDATE correspondant à un emprunt si cet UPDATE change la valeur de `id_lecteur` de NULL vers une valeur non NULL.
Si UPDATE n'est pas un UPDATE correspondant à un emprunt alors le trigger permet d'exécuter UPDATE et se termine tout de suite.
Si UPDATE est un UPDATE réalisant un emprunt alors le trigger doit effectuer les vérifications suivantes :

- si le livre n'est pas empruntable annuler UPDATE,
- si `id_lecteur` pour ce livre est non NULL alors il y a un problème, on essaie d'emprunter un livre déjà marqué comme emprunté. On annule UPDATE puisque il faut d'abord vérifier la fiabilité de notre système.
- si le lecteur est frappé d'interdiction d'emprunter, c'est-à-dire l'attribut `interdit` est non NULL et `interdit > current_date`, alors annuler l'emprunt,
- si le lecteur a emprunté déjà au moins 5 livres alors annuler l'emprunt (5 est le nombre maximal de livres qu'on peut emprunter),

- si le lecteur a un livre emprunté pour lequel il est en retard pour le retour alors annuler l'emprunt.

Si UPDATE qui réalise l'emprunt a passé positivement tous ces tests alors le trigger permet d'exécuter UPDATE en modifiant en plus les valeurs des attributs `emprunt` (on y met la date courante comme la date d'emprunt) et `retour` (on y mettant la date courante +21 jours, on suppose qu'on emprunt au maximum pour trois semaines).

Question 2: Le retour d'un livre est réalisé par une requête de forme

```
UPDATE TABLE livre
SET    id_lecteur=NULL
WHERE  id_livre=id_du_livre_retourne
```

Il faut associé à UPDATE un deuxième trigger qui vérifie si le lecteur est ou non en retard pour le livre qu'il retourne. Donc le trigger vérifie d'abord si c'est un UPDATE qui correspond à un retour.

On suppose qu'un UPDATE est un UPDATE de retour si cet UPDATE met à NULL l'attribut `id_lecteur` dans la table `livre` et si cet attribut n'était pas NULL auparavant.

Si UPDATE n'est pas un UPDATE de retour alors le trigger laisse exécuter UPDATE.

Si le trigger constate que c'est un UPDATE de retour de livre alors

- le trigger vérifie si le retour est en retard ou non, si le retour est sans retard le trigger se termine en permettant exécuter UPDATE,
- pour le retour en retard on calcule de nombre de jours d'interdiction :

$$\text{durée d'interdiction} = 2\times \text{ durée de retard},$$

- on calcule la date `interdit` comme date courante + durée d'interdiction,
- on met dans la date `interdit` pour le lecteur concerné la date calculé précédemment **sauf si le lecteur est déjà frappé par une interdiction plus sévère**[4].

Et autres

Exercice 5 Dans une base de données avec une très grande table

```
achat(
   id_achat int primary key,
   id_client   int,
   autres attributs utiles
)
```

on observe on grand nombre de requêtes de forme

```
select truc
from achat
where id_client = numero
```

où `numero` est une constante.

Pour améliorer les performances l'administrateur de BD crée un index de type B-arbre sur l'attribut `id_client`.

On observe que le temps d'exécution de requêtes avec la condition `id_client = numero` est divisé par 100.

4. Il faut éviter la situation où un lecteur retourne un livre qui donne interdiction de 100 jours et le jour suivant il rend un livre qui donnerai une interdiction de 5 jours que cette interdiction moins pénalisant efface l'interdiction plus sévère.

Question 1: Dans cette base on lance aussi beaucoup de requêtes de forme

```
select truc
from achat
where id_client <> numero
```

où `numero` est une constante, c'est-à-dire avec `<>` à la place de `=`. Est-ce que après la création de l'index le temps moyen d'exécution de ces requêtes

1. ne change pas,
2. diminue un peu,
3. diminue beaucoup,
4. augmente.

Justifier (très brièvement).

Question 2: Est-ce que après la création d'index le temps d'insertion (INSERT) dans la table `achat` (1) diminue, (2) ne change pas, (3) augmente.
Justifier la réponse (en une phrase).

Exercice 6 Soit $T_1 = r_1(X)w_1(Y)r_1(Z)w_1(X)w_1(Y)$ et $T_2 = r_2(X)r_2(Y)$ deux transactions, r - lecture, w - écriture. Donner un exemple d'histoire h où les deux transactions s'exécutent en parallèle et tel que h n'est pas sérializable.