Université de Nice-Sophia Antipolis Ecole Polytech'Nice Sophia

MAM4 SI4

2008–2009

Examen de Bases de données relationnelles

15 Décembre 2008 – Durée : 2 heures

1 Agence Immobilière

On considère une base de données relationnelle d'une agence immobiliere. Cetta base est constituée des tables suivantes (les attributs de la clé primaire de chaque table ont été soulignés):

- *Maison (id, prix_demande, adresse, code_postal, chambres, sdbs, surface, idVendeur)*
  Cette table décrit les maisons à vendre ou ayant été vendues par l'agence :
  - *prix_demande* est le prix initialement demandé par le vendeur ($idVendeur$)
  - $adresse$, *code_postal* localisent la maison
  - le nombre de chambres est *chambres*, le nombre de salles de bains *sdbs*, et la surface $surface$.
- *Agent(id, nom, telephone, email)*
  Cette table décrit les agents immobiliers de l'agence.
- *Vendeur(id, nom, telephone, email, idAgent)*
  Cette table décrit les vendeurs présents, ou passés d'un bien via cette l'agence.
- *Acheteur(id, nom, telephone, email, idAgent)*
  Cette table décrit les acheteurs clients de l'agence
- *Vente(idMaison, idAcheteur, dateVente ,prix_vente)*
  La table Vente renseigne l'identité du bien vendu, celle de l'acheteur, la date de la vente et le prix de vente qui peut différer du prix initialement demandé.

1. Exprimez les requêtes suivantes en **algèbre relationnelle** :
   - Déterminer les identifiants, adresses, prix_demande, prix_vente pour toutes les maisons ayant été vendues pour une somme inférieure au prix initialement demandé

     $\sigma_{prix\_demande>prix\_vente}[\Pi_{id,adresse,prix\_demande}(Maison) \bowtie (\delta_{IdMaison\leftarrow id}\ (\Pi_{idMaison,prix\_vente}\ Vente))]$

   - Déterminer les noms de tous les acheteurs potentiels n'ayant encore rien acheté

     $\Pi_{nom}([\Pi_{id}(Acheteur) \setminus \Pi_{id}(\delta_{IdAcheteur\leftarrow id}(Vente))] \bowtie (\Pi_{id,nom}\ (Acheteur)))$

     ---

2. Exprimez les requêtes suivantes en **SQL**
   - Pour tous les codes postaux pour lesquels au moins trois ventes ont été effectuées, afficher code_postal et prix_vente moyen

     ---

```
SELECT M.code_postal, avg(prix_vente) AS PrixMoyen
     FROM Maison  M, Vente V
```

```
WHERE M.id = V.idMaison
GROUP BY M.code_postal
HAVING COUNT(idMaison) >= 3
```

– Trouver les adresses et les prix_demande de toutes les maisons ayant au moins 3 chambres et deux salles de bains n'ayant pas été vendues. Chaque couple (adresse, prix_demande) doit être affiché une fois seulement.

```
SELECT DISTINCT M.adressse, M.prix_demande,
    FROM Maison M
    WHERE M.chambres > 2 AND M.sdbs > 1
          AND NOT EXISTS (
           SELECT idMaison FROM Vente V WHERE V.idMaison = H.id)
```

---

2 Donnation

Pour stocker les informations concernant des donnations faites par des organisations à des candidats à une élection , la table suivante a été créée:

```
CREATE TABLE donnations
    (nomReceveur CHAR(20) NOT NULL,
     organisationDonnatrice CHAR(20) NOT NULL,
     typeOrganisation CHAR(20),
     montant  REAL,
     PRIMARY KEY (nomReceveur, organisationDonnatrice)
     )
```

Pour chacune des 4 paires de requêtes SQL suivantes, dire si les requêtes sont équivalents, c'est à dire si elle renvoient le même résultat quelle que soit l'instance de la table donnations. Si c'est le cas, un simple oui suffira, mais si ce n'est pas le cas, donner un exemple d'instance de la table donnations pour laquelle les résultats diffèrent.

1. a
```
SELECT DISTINCT nomReceveur FROM donnations A
    WHERE NOT EXISTS
      (SELECT B.organisationDonnatrice FROM donnations B
              WHERE nomReceveur = 'Campbell'
              AND A.organisationDonnatrice <> B.organisationDonnatrice)
```
b
```
SELECT DISTINCT nomReceveur FROM donnations A
    WHERE NOT EXISTS
    ( (SELECT organisationDonnatrice FROM donnations
              WHERE nomReceveur = 'Campbell' )
      EXCEPT
      (SELECT organisationDonnatrice FROM donnations B
              WHERE B.nomReceveur = A.nomReceveur))
```

Non.
On peut considérer l'instance :

nomReceveur	organisationDonnatrice
John	A
John	B
Campbell	A
Campbell	B

Le résultat de requête 1.a est vide, la requête 1.b retourne John et Campbell.

---

2. a
```
(SELECT DISTINCT nomReceveur FROM donnations
          WHERE montant>= 500)
  UNION
 (SELECT DISTINCT nomReceveur FROM donnations
          WHERE montant < 500)
```
b
```
SELECT DISTINCT nomReceveur FROM donnations
```

Non. On peut considérer l'instance :

nomReceveur	montant
John	100
Campbell	2000
Mary	null

La requête 2.a retourne John and Campbell, la requête 2.b retourne les trois noms.

---

3. a
```
SELECT DISTINCT A.nomReceveur FROM donnations A, donnations B
         WHERE A.montant>= 1000
               AND A.nomReceveur = B.nomReceveur
               AND A.organisationDonnatrice <> B.organisationDonnatrice
```
b
```
SELECT DISTINCT nomReceveur FROM donnations
         WHERE montant >= 1000
         GROUP BY nomReceveur
         HAVING COUNT(organisationDonnatrice) >=2
```

Non.
On peut considérer l'instance :

nomReceveur	organisationDonnatrice	montant
Campbell	CUPE	100
Campbell	Canucks	2000

La requête 3.a retourne Campbell, la requête 3.b ne retourne rien.

---

4. a
```
SELECT DISTINCT nomReceveur FROM donnations
        WHERE nomReceveur NOT IN
             (SELECT nomReceveur FROM donnations
                     WHERE typeOrganisation = 'tabac')
```
   b
```
SELECT DISTINCT nomReceveur FROM donnations A
        WHERE EXISTS
              (SELECT * FROM donnations B
                     WHERE B.nomReceveur = A.nomReceveur
                           AND typeOrganisation <> 'tabac')
```

Non.
On peut considérer l'instance :

nomReceveur	typeOrganisation
John	tabac
John	entertainment
Mary	sports

La requête 4.a retourne les personnes n'ayant reçu aucune donation de la part d'une compagnie tabacière (Mary pour l'eexemple), la requête 4.b retourne les personnes ayant reçu au moins une donation de la part d'une compagnie non tabacière, (John et Mary pour l'exemple).

---

3 Formes normales

On donne la relation : R(A, B, C, D, E, F) et l'ensemble de dépendances fonctionnelles :
$DF1 = (A \rightarrow B; DE \rightarrow F; B \rightarrow C)$

1. Determiner l'ensemble des clés de R (en justifiant)

A,D et E n'apparaisent dans aucune partie droite des dépendances fonctionnelles. Ils doivent donc être inclus dans toutes les clés. Comme ils forment une clé, il forment en fait la seule clé

---

2. Si R n'est pas en 3NF, la décomposer en 3NF

   (a) à cause $DE \rightarrow F$ R viole la 2NF, un attribut non clé dépend d'une sous clé. On décompose donc la table en deux tables R_1(D,E,F) et R_2(A,B,C,D,E)
   R_1 a pour clé DE et ne satisfait aucune dépendance fonctionnelle non triviale; R_1est donc en $3NF$
   R_2 a pour clé ADE, et satisfait les dépendances $A \rightarrow B; B \rightarrow C$

   (b) R_2 n'est pas en $2NF$, à cause de la dépendance $A \rightarrow B$, on décompose R_2 en deux tables R_3(A,C,D,E) et R_4(A,B). R_4 n'ayant que deux attributs est en $3NF$. R_3 a pour clé ADE et vérifie la dépendance $A \rightarrow C$, elle n'est donc pas en $3NF$

(c) On décompose R_3 en $R_5(A,C)$ et $R_6(ADE)$ toutes deux en 3NF.

---

3. Même questions avec l'ensemble de dépendances :
   $DF2 = (AB \rightarrow C; CD \rightarrow A; C \rightarrow EB; D \rightarrow F)$

Toute clé doit contenir D qui n'apparait dans aucune partie droite de Dépendance Fonctionnelle.
Aucune clé ne peut contenir F. D n'est pas une clé. DC est une clé (on vérifie que $\{D,C\}^+ = \{A,B,C,D,E,F\}$. On a $\{D,A\}^+ = \{A,D,F\}$, $\{D,B\}^+ = \{B,D,F\}$, $\{D,E\}^+ = \{E,D,F\}$, $\{D,F\}^+ = \{D,F\}$, il n'y a donc pas d'autre clé ayant deux attributs.
On a $\{A,B,D\}^+ = \{A,B,C,D,E,F\}$, et donc $\{A,B,D\}$ constitue une clé.
On vérifie que $\{D,X,Y\}$ n'est pas une clé lorsque X = A ou B et Y = E ou F. On vérifie aussi que $\{B,X,E,F\}$,'est pas une clé lorsque X = A ou B.
Il y a donc deux clé CD et ABD.
La table n'est pas en 3NF , il faut l'y mettre

---

4 Division

R est une relation définie sur {X, Y,Z, T}, S est une relation définie sur {X, Y }. Les instances respectives de R et S sont :

X	Y	Z	T
x1	y1	z1	t1
x1	y1	z1	t2
x1	y2	z1	t1
x2	y2	z1	t1
x2	y1	z1	t1
x1	y1	z2	t1
x1	y1	z2	t2
x2	y2	z2	t1

X	Y
x1	y1
x1	y2
x2	y2

1. Ecrivez en SQL la requête $\Pi_{X,Y,Z}(R) \div S$

```
SELECT DISTINCT Z FROM R AS R1
      WHERE NOT EXISTS
            (SELECT X,Y FROM S
                  WHERE NOT EXISTS
                        (SELECT X,Y,Z FROM R AS R2
                              WHERE R1.Z=R2.Z AND R2.X = S.X
                                          AND R2.Y= S.Y ) )
```

---

2. Avec les instances ci-dessus quelles sont les valeurs renvoyées par les requêtes :
   (a) $\Pi_{X,Y,Z}(R) \div S$

---

z_1

(b) $\Pi_{X,Y,Z}(R) \div \Pi_X(S)$

(y_1,z_1) et (y_2,z_1)

---

3. Que renvoient les requêtes suivantes lorsqu'elles sont exécutées sur l'instance ci dessus.

(a) CREATE VIEW V1 AS SELECT X,Y,Z FROM R;
SELECT COUNT(*) , Z FROM V1 GROUP BY Z ;

(5,z_1) et (3,z_2)

(b) CREATE VIEW V2 AS SELECT DISTINCT X,Y,Z FROM R;
SELECT COUNT(*) , Z FROM V2 GROUP BY Z ;

(4,z_1) et (2,z_2)

(c) CREATE VIEW V3 AS SELECT DISTINCT X,Y FROM S;
SELECT Z FROM V1 GROUP BY Z
HAVING COUNT(*) = (SELECT COUNT(*) FROM V3) ;

---

z_2

(d) SELECT Y,Z FROM V2 GROUP BY Y,Z
HAVING COUNT(*) = (SELECT COUNT(DISTINCT X) FROM S);

---

y_1,z_1 et y_2,z_1

(e) la requête 3.c est supposée traduite en SQL la division 2.a, la requête3.d est supposée traduire la division 2.b. L'une de ces traductions est correcte, laquelle? Quelle est la raison qui fait échouer l'autre.

La requête 3.d est une bonne traduction, l'autre échoue car on utilise $V1$ et non $V3$
