Université Nice Sophia Antipolis
Polytech Nice Sophia

Contrôle continu de Base de données
16 décembre 2015

Durée : 1h45 heure

1.1		3	2.2		6
1.2		6	2.3		4
2.1		6			

1 Normalisation

1.1 Clés

Soit $R_0(A,B,C,D,E,F)$ une relation avec l'ensemble de dépendances suivant :
$DF = \{AC \to B, C \to D, AB \to E, AB \to F, E \to A, E \to B\}$
Donnez la ou les clés de R_0 et justifiez votre réponse?

AC et CE sont les seules clés de R_0:
- C n'est pas conséquence d'autres attributs, et donc il doit être dans toutes les clés
- $\{C\}^+ = \{C,D\}$ et C seul n'est donc pas une clé;
- $\{AC\}^+ = \{CE\}^+ = \{A,B,C,D,E,F\} = Attr(R_0)$ donc AC et CE sont des clés de R_0;
- $\{CD\}^+ = \{CD\}, \{CB\}^+ = \{B,C\}$ et CD et CB ne sont donc pas des clés de R_0

1.2 Forme Normale

Soit $R_1(A,B,C,D,E,F)$ une relation avec l'ensemble de dépendances suivant :
$DF = \{AB \to C, AB \to D, AB \to E, AB \to F, B \to C, D \to E, D \to F\}$
AB est la seule clé de R_1.

- 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 $B \to C$

- On décompose la relation R_1 en R_{11} et R_{12} : $R_{11}(A,B,D,E,F)$ et $R_{12}(B,C)$. Quelles sont (justifiez !) les formes normales des relations R_{11} et R_{12} ? La dépendance fonctionnelle $AB \to C$ est elle perdue ?

R_{11} a comme unique clé AB et vérifie les dépendances fonctionnelles suivantes: $DF = \{AB \to D, AB \to E, AB \to F, D \to E, D \to F\}$
R_{11} est en *2NF*, mais pas en *3NF* (par exemple à cause de $D \to E$).
R_{12} n'ayant que deux attributs est nécessairement en *3NF*) (et même en BCNF).
R_{12} a pour clé B, la dépendance fonctionnelle $AB \to C$ n'est pas perdue car c'est une conséquence de $B \to C$

– Donner une décomposition de R_1 en *3NF*.

Il ne reste qu'à décomposer R_{11}, en $R_{111}(A,B,D)$ (dont la clé est AB et $R_{112}(D,E,F)$ dont la clé est D. On a finalement décomposer R_1 en $R_{111}(\underline{A,B},D)$ $R_{112}(\underline{D},E,F)$ et $R_{12}(\underline{B},C)$. Les seules dépendances fonctionnelles sont celles des clés. Cette décomposition est en *3NF* (et même en $BCNF$)

---

2 SQL

1. Soit la table `employes(id_employe, nom_employe, id_superieur)`.
   – Ecrire une requête qui calcule l'arbre hiérachique de l'employé 3, c'est à dire tous les tuples `(id_employe, nom_employe, id_superieur)` où `id_employe` est l'id d'un des supérieurs de l'employé 3.

```
WITH RECURSIVE hierarchie(id_employe, nom_employe, id_superieur) AS
(
  SELECT id_employe, nom_employe, id_superieur
    FROM employes WHERE id_employe = 3
  UNION ALL
  SELECT e.id_employe, e.nom_employe, e.id_superieur
    FROM hierarchie AS h,employes AS e
    WHERE h.id_superieur = e.id_employe
)
SELECT * FROM hierarchie;  --  where id_employe <>3;
```

   – Ecrire une requête qui calcule le nombre d'employés sous les ordres directs ou indirects du chef de service 18.

```
WITH RECURSIVE souslesordres(id_employe, id_superieur) AS
(
 SELECT id_employe, id_superieur
  FROM employes WHERE id_superieur = 18
UNION ALL
  SELECT e.id_employe, e.id_superieur
  FROM souslesordres  AS ss, employes AS e
  WHERE e.id_superieur = ss.id_employe
)
SELECT count(*) FROM souslesordres;
-ou
WITH RECURSIVE souslesordres(id_employe, id_superieur) AS
(
 SELECT id_employe, id_superieur
  FROM employes WHERE id_employe = 18
UNION ALL
  SELECT e.id_employe,  e.id_superieur
  FROM souslesordres  AS ss, employes AS e
  WHERE e.id_superieur = ss.id_employe
)
SELECT count(*) - 1 FROM souslesordres;
```

---

2. En utilisant les tables en annexe, écrire les requêtes suivantes:
   - Recherche des noms des enseignants (instructors) qui ont le salaire le plus élevé.

```
select ID, name from instructor where
         salary = (select max(salary) from instructor)
```

   - Trouver le nombre d'inscrits pour chaque cours (table 'section') ouvert en automne 2015.

```
select course_id, sec_id, count(student_ID)
      from section natural join takes
      where semester = 'Autumn' and year = 2015
group by course_id, sec_id;
```

   - Rechercher le nombre maximum d'inscrits pour toutes les sections à l'automne 2015.

```
select max(enrollment)
      from (select count(student_ID) as enrollment
                  from section natural join takes
                  where semester = 'Autumn' and year = 2015
                  group by course_id, sec_id);
```

3. En utilisant toujours les tables en annexe, effectuez les mises à jour suivantes:
   - Augmentez le salaire de tous les enseignants en informatique (département 'Comp. Sci.') de 10 %

```
update instructor
    set salary = salary * 1.10
    where dept_name = 'Comp. Sci.'
```

   - Supprimez tous les cours qui n'ont jamais été offerts (c'est à dire qui n'apparaissent pas dans la relation 'section').

```
delete from course
      where course_id not in (select course_id from section)
```

ANNEXE

```
create table course
   (course_id   varchar(8),
    title       varchar(50),
    dept_name   varchar(20),
```

```
    credits     numeric(2,0) check (credits > 0),
    primary key (course_id)
   );

create table instructor
   (ID         varchar(5),
    name       varchar(20) not null,
    dept_name  varchar(20),
    salary     numeric(8,2) check (salary > 29000),
    primary key (ID)
   );

create table section
   (course_id    varchar(8),
    sec_id       varchar(8),
    semester     varchar(6)
      check (semester in ('Fall', 'Winter', 'Spring', 'Summer')),
    year         numeric(4,0) check (year > 1701 and year < 2100),
    time_slot_id varchar(4),
    primary key (course_id, sec_id, semester, year),
    foreign key (course_id) references course
   );


create table takes
   (student_ID         varchar(5),
    course_id  varchar(8),
    sec_id     varchar(8),
    semester   varchar(6),
    year       numeric(4,0),
    grade      varchar(2),
    primary key (student_ID, course_id, sec_id, semester, year),
    foreign key (course_id,sec_id, semester, year) references section
   );
```