Licence Science et Technologies - Devoir surveillé - mars 2014, des solutions

1 A propos de bars

Question 1 (5 points) :
En considérant les relations fournies en annexe, donner le résultat retourné par chacune des requêtes ci-dessous (considérer le système Oracle vu en TP). Il peut s'agir d'une erreur, ou de n-uplets affichés sous forme de tableau.
Requêtes 2 et 4 : sur 1,5. Requêtes 1 et 3 : sur 1

1. select bar
   from Likes natural join Frequents
   where drinker = 'John'
   and beer = 'Pale Ale';

```
BAR
-----------------
Australia Hotel
Coogee Bay Hotel
Lord Nelson
```

2. select bar, drinker from Frequents
   union
   select bar from Sells where price < 10 ;

   La requête est incorrecte car les schémas des sous-requêtes sont incompatibles.

3. select beer, price
   from Bars join Sells on (bar = name)
   where price > 3 ;

```
BEER                          PRICE
--------------------------- ----------
Burragorang Bock                 3.5
Three Sheets                    3.75
Old Admiral                     3.75
```

4. select bar, count(distinct beer) as nbBeers,
   min(price), count(drinker) as nbClients
   from Sells natural join Frequents
   group by bar ;

```
BAR                NBBEERS MIN(PRICE)  NBCLIENTS
------------------ ------- ---------- ----------
Marble Bar               3        2.8          3
Australia Hotel          1        3.5          1
Lord Nelson              2       3.75          4
Coogee Bay Hotel         4        2.3          8
Regent Hotel             2        2.2          2
```

Question 2 (6 points) :
Pour chacune des requêtes ci-dessous, spécifier la relation qu'elle construit, puis la ré-écrire en SQL en respectant à chaque fois les instructions fournies. Les requêtes devront construire des résultats sans répétition de valeurs, la clause distinct ne sera utilisée que lorsque nécessaire.

1. select distinct beer, price from Sells
   where bar in (select name from Bars where addr = 'Toowong')

   <b, p>: la bière b est vendue au prix p dans un bar dont l'adresse est 'Toowong'.

```
select distinct beer, price
from Sells join Bars on (bar = name)
where addr = 'Toowong'
```

2. select bar from Frequents where drinker = 'Adam'
   intersect
   select bar from Frequents where drinker = 'John';

   < b >: b est un bar fréquenté par Adam et par John.

```
select distinct F1.bar
from Frequents F1 join Frequents F2 on (F1.bar = F2.bar)
where F1.drinker = 'Adam' and F2.drinker = 'John'
```

3. select distinct name, manf
   from Beers join Sells on (name=beer)
   where price in (select max(price) from Sells)

   <b, m>: la bière b fabriquée par le brasseur m est la plus chère (parmi toutes les ventes).

```
select distinct name, manf
from Beers join Sells on (name=beer)
     join
     (select max(price) as maxPrice
     from Sells) X on (price = maxPrice)
```

Une autre version :

```
select name, manf
from Beers join Sells on (name=beer)
minus
select B1.name, B1.manf
from Beers B1 join Sells S1 on (B1.name=S1.beer) join Sells S2 on (S1.price<S2.price)
```

2 A propos de programmes de formations

2.1 Expression de requêtes en SQL

Question 3 (9 points) :
Exprimer en SQL les requêtes ci-dessous en respectant rigoureusement les contraintes données.

1. Donner le nom et le prénom des personnes inscrites (comme élève) à un cours dont l'intitulé est patin à roulettes.

```
select distinct P.nom, P.prenom
from LesPersonnes P join LesInscriptions I on (P.noP = I.noEl)
     join LesCours C on (I.noC = C. noC)
where C.intitulé = 'Patin à roulettes'
```

ou bien :

```
select distinct nom, prénom
from lescours natural join lesinscriptions join lespersonnes on (noel=nop)
where intitulé = 'patin à roulettes'
```

2. Pour chaque cours, donner son numéro, son intitulé et sa durée, ainsi que le numéro, l'intitulé et la durée de chacun de ses pré-requis.

```
select C.noC, C.intitule, C.duree, P.noPreRequis,
       CP.intitule as intitulePreRequis, CP.duree as dureePreRequis
from LesCours C join LesPreRequis P on (C.noC = P.noC)
     join LesCours CP on (P.noPreRequis = CP.noC)
```

Une autre version :

```
select C1.noc, C1.intitulé, C1.durée, C2.noc, C2.intitulé, C2.durée
from lescours C1 natural join lesprérequis L1
                 join lescours C2 on (C2.noC=L1.noprerequis)
```

Une autre version :

```
select C.noC, C.intitule, C.duree, P.noPreRequis, intitulePreRequis, dureePreRequis
from LesCours C join
     (select noC, noPreRequis,
             intitule as intitulePreRequis, duree as dureePreRequis
     from LesPreRequis join LesCours on (noPreRequis = noC) ) P
     on (C.noC = P.noP)
     -- <c, p, i, d> ap P <=> le cours de numéro c a le cours p comme pré-requis.
     -- i est l'intitulé de p et d sa durée.
```

3. Pour chaque personne, donner son numéro, son nom et son prénom, ainsi que le nombre de cours auxquels cette personne est inscrite comme étudiant(e) (dans le passé, le présent ou le futur). Ignorer les personnes inscrites, comme étudiant(e), à aucun cours.

```
select noP, nom, prenom, count(distinct noC)
from LesPersonnes join LesInscriptions on (noP = noEl)
group by noP, nom, prenom
```

4. Donner le numéro, le nom et le prénom des personnes qui enseignent (dans le passé ou le futur) dans toutes les villes. Une personne enseigne dans une ville, soit v, lorqu'elle est affectée comme enseignante à au moins un cours programmé dans v.

```
select noP, nom, prénom
from LesPersonnes join LesProgramme on (noP=noC)
group by noP, nom, prénom
having count (distinct ville) in (select count(distinct ville) from LeProgramme)
```

Une autre solution :

```
select noP, nom, prénom
from LesPersonnes natural join
     (select noP, count (distinct ville) as nbVilles from LesProgramme
      group by noP) R1 join
     -- <p, v> dans R1 <=> la personne de nom p a été étudiant dans v villes
     (select count (distinct ville) as nbTotalVilles from LeProgramme) R2
     -- <n> dans R2 <=> des cours ont été programmés dans n villes au total.
     on (nbVilles = nbTotalVilles)
```

5. Donner la requête qui permet de retourner le numéro des personnes qui se sont inscrites comme élève à au moins un cours, sans satisfaire la contrainte sur les pré-requis.

```
-- <e> est retourné par la requête si et seulement si e est le numéro
-- d'un éléve inscrit à un cours à la date d qui a un pré-requis
-- auquel e n'a pas assisté et qui se termine avant d.
select distinct I.noEl
from LesPreRequis P join LesInscriptions I on (P.noC = I.noC)
where (I.noEl, P.noPreRequis, P.noC) not in (
      select I1.noEl, I1.noC as noPre, I2.noC as noC
      from LesCours C join LesInscriptions I1 on (C.noC = I1.noC)
           join LesInscriptions I2
           on (I1.noEl = I2.noEl and I1.dateDebut+C.duree < I2.dateDebut) )
      -- <e, n1, n2> retourné par la sous-requête <=> n1 est le numéro
      -- d'un cours suivi par e et qui a eu lieu avant le cours n2 suivi par e.
```

On rappelle que soit une date d et un entier n, d+n est le jour issu de d et décalé de n jours.