Correction du TD 5 - SQL : jointures - partie 2

Base de données formula1

La base de données formula1 est composée des 7 relations suivantes :

- **circuit**(circuitID, cName, cCity, cCountry, cLength, cLapRec, #cDrivRec *réf. driverID*, cYearRec)
- **driver**(driverID, dFirstName, dLastName, dBirthdate, dDeathdate, dCountry, dGender)
- **grandprix**(gpID, gName, #circuitID, gDate, gLaps, gRank)
- **racedriver**(#teamID, #driverID, rSeason, rDriverNb)
- **standings**(#driverID, #gpID, sGrid, sPos, sRes, sPoints, sLaps)
- **team**(teamID, tName, tCountry, #twas *réf. teamID*)
- **tesdriver**(#teamID, #driverID, tSeason)

Cette base est utilisée dans les exercices qui suivent. Les requêtes peuvent être testées sur les sites :

- `http://pedago.uhb.fr/sql/` (à préférer à l'université),
- `http://bdur2m.free.fr/` (avec le login `etudiant` et le mot de passe `rennes2`).

Exercice 1. *Que font ces requêtes ?*

Exprimer en français les informations que renvoient les requêtes suivantes (essayer tout d'abord sans exécuter la requête dans l'interface, puis regarder le résultat d'exécution pour vérifier) :

1. ```
   SELECT old.tName AS 'ancienne équipe', new.tName AS 'nouvelle équipe'
   FROM team AS old JOIN team AS new ON old.teamID = new.tWas
   ```

   > Pour les équipes qui ont changé de nom, l'ancien nom et le nouveau nom.

2. ```
   SELECT d1.dLastName, d1.dBirthdate, d2.dLastName, d2.dBirthdate
   FROM driver AS d1 JOIN driver AS d2 ON d1.dBirthdate <= d2.dBirthdate
                                      AND d1.driverID <> d2.driverID
   ORDER BY d1.dLastName, d2.dLastName
   ```

   > les pilotes (nom, date de naissance) dans l'ordre alphabétique, avec pour chacun les pilotes plus jeunes ou de même âge que lui (dans l'ordre alphabétique).
   >
   > Vérifier dans la liste que Wehrlein (2e plus jeune) n'est associé qu'à Sirotkin (plus jeune) et que Sirotkin n'apparaît pas dans la colonne de gauche.

Exercice 2. *Autojointures, autres jointures*

En utilisant la syntaxe SQL92 (jointures explicites), donner les requêtes SQL qui renvoient les informations suivantes. Donner une seule version, avec l'opérateur de jointure le plus approprié.

1. Les noms et prénoms des pilotes qui ont actuellement un record du tour sur un circuit de leur pays, par ordre alphabétique

   La réponse la plus fréquemment donnée est la suivante :

   ```
   SELECT DISTINCT dLastName, dFirstName
   FROM driver JOIN circuit ON driverID = cDrivRec
   WHERE dCountry = cCountry
   ORDER BY dLastName
   ```

   Elle fournit le bon résultat. . .Cependant, la condition `dCountry = cCountry` limite la correspondance entre un pilote et un circuit pour créer une ligne de résultat de jointure. Il s'agit donc d'une condition de jointure, à écrire de préférence dans le `ON`.

   ```
   SELECT DISTINCT dLastName, dFirstName
   FROM driver JOIN circuit ON driverID = cDrivRec
                           AND dCountry = cCountry
   ORDER BY dLastName
   ```

   Mais il est assez courant d'écrire la condition d'égalité clé étrangère/clé primaire dans le ON et une condition supplémentaire dans le WHERE. . .

2. Les circuits (nom, ville, longueur), plus courts que le « Hungaroring », du plus long au plus court (on suppose que dans l'écriture de la requête on ne connait pas a priori la longueur du Hungaroring, mais seulement son nom)

   ```
   SELECT circ.cName, circ.cCity, circ.cLength
   FROM circuit AS circ JOIN circuit AS hung ON circ.cLength < hung.
      cLength
   WHERE hung.cName = 'Hungaroring'
   ORDER BY circ.cLength DESC
   ```

3. les pilotes qui courent pour une équipe étrangère en 2014 (nom, prénom du pilote, nom de l'équipe)

   Si on veut utiliser des jointures naturelles, la condition de jointure sur les pays ne peut être écrite que dans la clause WHERE. Cf. plus haut, écrire cette condition de jointure supplémentaire dans le WHERE est courant, surtout lorsque la jointure clé étrangère/clé primaire est faite avec une jointure naturelle.

   ```
   SELECT dLastName, dFirstName, tName
   FROM driver NATURAL JOIN racedriver
               NATURAL JOIN team
   WHERE dCountry <> tCountry
     AND rSeason = 2014
   ORDER BY tName, rDriverNb
   ```

Sinon, on aurait pu écrire (mais la requête est moins lisible)

```
SELECT dLastName, dFirstName, tName
FROM driver NATURAL JOIN racedriver
            JOIN team ON racedriver.teamID = team.teamID
                      AND dCountry <> tCountry
WHERE rSeason = 2014
ORDER BY tName, rDriverNb
```

ou encore (même problème de lisibilité)

```
SELECT dLastName, dFirstName, tName
FROM driver JOIN racedriver ON driver.driverID = racedriver.
   driverID
            JOIN team ON racedriver.teamID = team.teamID
                      AND dCountry <> tCountry
WHERE rSeason = 2014
ORDER BY tName, rDriverNb
```

Par contre, il vaut mieux éviter l'écriture :

```
SELECT dLastName, dFirstName, tName
FROM driver JOIN team ON dCountry <> tCountry
            NATURAL JOIN racedriver
WHERE rSeason = 2014
ORDER BY tName, rDriverNb
```

si l'écriture semble courte et élégante, la première jointure produit « beaucoup » de résultats (nombreuses associations pilote/équipe), supprimés par les conditions implicites de la jointure naturelle qui suit. Cette requête est donc moins efficace en machine (sauf si le SGBD a un excellent optimiseur qui ramène la requête aux précédentes).

4. toutes les associations pilote de course (nom, prénom) / pilote d'essai (nom, prénom) de la même équipe pour la saison 2014

Première version, avec un maximum de `NATURAL JOIN`, qui nécessite des parenthèses pour forcer l'ordre des jointures :

```
SELECT race.dLastName, race.dFirstName, test.dLastName, test.
   dFirstName
FROM (driver AS race NATURAL JOIN racedriver)
       JOIN
     (driver AS test NATURAL JOIN testdriver)
       USING(teamID)
WHERE rSeason = 2014
  AND tSeason = 2014
```

Dans cette première version, la jointure avec `USING(teamID)` ne peut pas être remplacée par `NATURAL JOIN`. Pourquoi ?

Une deuxième version, dans laquelle les 3 jointures sont successives :

```
SELECT race.dLastName, race.dFirstName, test.dLastName, test.
   dFirstName
FROM driver AS race NATURAL JOIN racedriver
          JOIN testdriver USING(teamID)
          JOIN driver AS test ON testdriver.driverID = test.
             driverID
WHERE rSeason = 2014
  AND tSeason = 2014
```

Dans cette version, la 2e jointure peut utiliser `USING` mais pas `NATURAL JOIN`. Pour la 3e, on ne peut même pas utiliser `USING`... assurez-vous d'avoir compris pourquoi !

Une 3e version avec uniquement des `JOIN... ON` :

```
SELECT race.dLastName, race.dFirstName, test.dLastName, test.
   dFirstName
FROM driver AS race
          JOIN racedriver     ON race.driverID = racedriver.
             driverID
          JOIN testdriver     ON racedriver.teamID = testdriver.
             temID
          JOIN driver AS test ON testdriver.driverID = test.
             driverID
WHERE rSeason = 2014
  AND tSeason = 2014
```

Exercice 3. *Requêtes avec jointures externes ; conserver toutes les lignes*

Donner les requêtes SQL qui renvoient les informations suivantes :

1. tous les circuits des grands prix de 2014 (ville, date du grand prix) avec le nom du recordman du tour, lorsqu'il est connu, en présentant les résultats par ordre chronologique

```
SELECT cCity, gDate, dLastName
FROM circuit NATURAL JOIN grandprix
             LEFT JOIN driver ON cDrivRec = driverID
WHERE YEAR(gDate) = 2014
ORDER BY gDate
```

2. tous les grands prix de 2014 (nom, date) avec, lorsqu'il est connu, le nom du vainqueur, dans l'ordre chronologique

La requête suivante (souvent donnée en première intention) est incorrecte :

```
SELECT gName, gDate, dLastName
FROM driver NATURAL JOIN standings
            NATURAL RIGHT JOIN grandprix
WHERE sPos = 1
  AND YEAR(gDate) = 2014
ORDER BY gDate
```

En effet, les grands prix pour lesquels le vainqueur n'est pas connu (les grands prix qui n'ont pas encore été courus à la date du 30 septembre 2014) n'ont pas de résultat avec `sPos = 1`, et sont donc supprimés par le filtrage sur cette condition.

Ces grands prix non courus ont été ajoutés par la jointure externe, et la valeur de `sPos` qui leur est associée est `NULL`. Pour corriger, on peut donc être tenté d'écrire :

```
SELECT gName, gDate, dLastName
FROM driver NATURAL JOIN standings
            NATURAL RIGHT JOIN grandprix
WHERE (sPos = 1 OR sPos IS NULL)
  AND YEAR(gDate) = 2014
ORDER BY gDate
```

Cette requête est aussi incorrecte ! Pour en comprendre la raison, il suffit d'afficher toutes les colonnes par :

```
SELECT *
FROM driver NATURAL JOIN standings
            NATURAL RIGHT JOIN grandprix
WHERE (sPos = 1 OR sPos IS NULL)
  AND YEAR(gDate) = 2014
ORDER BY gDate
```

On peut alors constater que `sPos` est aussi `NULL` lorsque le pilote a fini une course sans être classé (e.g. abandon). Pour détecter les grands prix ajoutés par la jointure externe qu'il ne faut pas supprimer par la condition `sPos = 1`, il faut donc tester sur une colonne qui ne peut pas être `NULL`, comme par exemple la clé `driverID`, ce qui donne :

```
SELECT gName, gDate, dLastName
FROM driver NATURAL JOIN standings
            NATURAL RIGHT JOIN grandprix
WHERE (sPos = 1 OR driverID IS NULL)
  AND YEAR(gDate) = 2014
ORDER BY gDate
```

3. pour chaque pilote (prénom, nom, date de naissance), les grands prix (nom, date) qui ont eu lieu le jour de son anniversaire, et NULL s'il n'y en a pas, dans l'ordre alphabétique des pilotes

   Remarque : la fonction `MONTH(<date>)` (resp. `DAY(<date>)`) permet d'extraire le mois (resp. le jour) d'une date.

```
SELECT dFirstName, dLastName, dBirthdate, gName, gDate
FROM driver LEFT JOIN grandprix
                 ON MONTH(dBirthdate) = MONTH(gDate)
                AND DAY(dBirthdate) = DAY(gDate)
ORDER BY dLastName, dFirstName
```

Exercice 4. *Requêtes avec jointures externes ; les lignes sans correspondant*

Donner les requêtes SQL qui renvoient les informations suivantes :

1. les équipes (nom) qui n'ont jamais eu de pilote d'essai ;

```
SELECT tName
FROM team NATURAL LEFT JOIN testdriver
WHERE driverID IS NULL
ORDER BY tName
```

2. les pilotes (nom, prénom) qui n'ont jamais été pilote de course, dans l'ordre alphabétique

```
SELECT dLastName, dFirstName
FROM driver NATURAL LEFT JOIN racedriver
WHERE teamID IS NULL
ORDER BY dLastName
```

Exercice 5. *Que renvoient ces requêtes avec jointure externe ?*

Exprimer en français les informations que renvoient les requêtes suivantes (essayer tout d'abord sans exécuter la requête dans l'interface, puis regarder le résultat d'exécution pour vérifier) :

1.
```
SELECT dFirstName, dLastName, gName, gDate
FROM grandprix NATURAL JOIN standings
               NATURAL RIGHT JOIN driver
ORDER BY dLastName, dFirstName, gDate
```

   Tous les pilotes (nom, prénom) dans l'ordre alphabétique, associés aux grands prix qu'ils ont courus (1 par ligne) dans l'ordre chronologique, s'ils en ont couru, et associés à NULL sinon.

2.
```
SELECT dFirstName, dLastName
FROM driver NATURAL LEFT JOIN standings
WHERE gpID IS NULL
ORDER BY dLastName, dFirstName
```

   Les pilotes qui n'ont pas de résultat de course dans la BD (= qui n'ont jamais fait de course, où dont les courses, trop anciennes, ne sont pas répertoriées)