Correction du TD 9 - SQL : sous-requêtes simples, op. ensemblistes

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. *Sous-requêtes dans la clause SELECT*

Donner les requêtes SQL qui renvoient les informations suivantes :

1. pour chaque grand prix de 2014, le nom du grand prix, le nombre de tours à parcourir, et l'écart par rapport au nombre de tours moyens des grands prix (de 2014)

```
SELECT gName, gLaps, (SELECT AVG(gLaps)
                      FROM grandprix
                      WHERE YEAR(gDate) = 2014) - gLaps
FROM grandprix
WHERE YEAR(gDate) = 2014
```

2. pour chaque pilote de la saison 2014, le nombre de points qu'il a marqué après 14 grand prix, et le nombre de points de retard par rapport au premier, classés du premier au dernier

```
SELECT dFirstname, dLastname, SUM(sPoints) AS points, SUM(sPoints) -
     (SELECT SUM(sPoints)
      FROM standings
      WHERE gpID BETWEEN 201401 AND 201414
      GROUP BY driverID
      HAVING SUM(sPoints) >= ALL (SELECT SUM(sPoints)
                                  FROM standings
                                  WHERE gpID BETWEEN 201401 AND 201414
                                  GROUP BY driverID)) AS 'late pts'
FROM driver NATURAL JOIN standings
WHERE gpID LIKE '2014%'
```

```
  AND SUBSTRING(gpID,5,2) <= 14
GROUP BY driverID
ORDER BY points DESC
```

Exercice 2. *Sous-requêtes dans la clause FROM*

Donner les requêtes SQL qui renvoient les informations suivantes :

1. pour chaque pilote son nom et le nombre de fois qu'il a terminé une course au moins aussi bien classé qu'il ne l'a commencée (ie. la place à l'arrivée est meilleure que la place sur la grille), en 2014, par ordre alphabétique (sans sous-requête)

```
SELECT dLastName, COUNT(*) AS nombre
FROM driver NATURAL JOIN standings
WHERE sPos <= sGrid
  AND gpID LIKE '2014%'
GROUP BY driverID
ORDER BY dLastName
```

2. le nombre moyen de fois qu'un pilote a terminé une course au moins aussi bien classé qu'il ne l'a commencée, en 2014

```
SELECT AVG(nombre)
FROM (SELECT COUNT(*) AS nombre
      FROM standings
      WHERE sPos <= sGrid
        AND gpID LIKE '2014%'
      GROUP BY driverID) as tabNbProg
```

3. même chose, en affichant les résultats par équipe, puis de façon globale

```
SELECT tName, AVG(nombre)
FROM (SELECT tName, COUNT(*) AS nombre
      FROM standings NATURAL JOIN racedriver
                     NATURAL JOIN team
      WHERE sPos <= sGrid
        AND gpID LIKE '2014%'
      GROUP BY driverID) as tabNbProg
GROUP BY tName WITH ROLLUP
```

```
SELECT tName, AVG(nombre)
FROM (SELECT driverID, COUNT(*) AS nombre
      FROM standings
      WHERE sPos <= sGrid
        AND gpID LIKE '2014%'
      GROUP BY driverID) as NbMeilleur
                     NATURAL JOIN racedriver
                     NATURAL JOIN team
GROUP BY tName WITH ROLLUP
```

Dans la première version, on associe à chaque *résultat en progression* l'équipe du pilote qui l'a réalisé, puis on groupe par pilote et on compte combien chacun en a réalisé. La requête principale n'a plus qu'à grouper selon l'équipe (avec ROLLUP) et faire une moyenne.

Dans la deuxième version, la sous-requête porte seulement sur la table standings et calcule pour chaque identifiant de pilote le nombre de *résultats en progression* ; l'association à l'équipe (par jointure) se fait seulement ensuite, dans la requête principale, donc sur une table plus petite. Cela n'est pas forcément plus efficace : la table est plus petite (ce qui accélère la jointure en général), mais comme il s'agit d'une table intermédiaire, sur laquelle aucun index n'est construit, l'algorithme de jointure ne peut bénéficier de l'indexation des attributs de jointure (ce qui ralentit les calculs).

Exercice 3. *Opérateurs ensemblistes*

Donner les requêtes SQL qui renvoient les informations ci-dessous.
Donner d'abord la requête avec l'opérateur ensembliste puis, pour `INTERSECT` et `EXCEPT`, la version équivalente avec une sous-requête (pour pouvoir la tester avec le SGBD MySQL).

Remarque : les requêtes utilisant les opérateurs `INTERSECT` et `EXCEPT` peuvent être testées sur un SGBD PostgreSQL qui gère ces opérateurs ; la BD formula1, sur un SGBD PostgreSQL, est disponible à l'adresse `http://bdur2p.free.fr/` (connexion avec le login `etudiant` et le mot de passe `rennes2`).

1. la liste ordonnée des pays impliqués en F1 en 2014 (ceux pour lesquel il y a des pilotes de course, des équipes, ou des grands prix)

```
SELECT dCountry AS pays
FROM driver NATURAL JOIN racedriver
WHERE rSeason = 2014

UNION DISTINCT

SELECT tCountry AS pays
FROM team NATURAL JOIN racedriver
WHERE rSeason = 2014

UNION DISTINCT

SELECT cCountry AS pays
FROM circuit NATURAL JOIN grandprix
WHERE gdate >= '2014-01-01'
  AND gdate <= '2014-12-31'

ORDER BY pays
```

```
SELECT dCountry AS pays
FROM driver NATURAL JOIN racedriver
WHERE rSeason = 2014

UNION DISTINCT

SELECT tCountry AS pays
```

```
FROM team NATURAL JOIN racedriver
WHERE rSeason = 2014

UNION DISTINCT

SELECT cCountry AS pays
FROM circuit NATURAL JOIN grandprix
WHERE gdate >= '2014-01-01'
  AND gdate <= '2014-12-31'

ORDER BY 1
```

Remarquez qu'il n'est pas possible d'ordonner selon `dCountry`, `tCountry` ou `cCountry`, car ces noms de colonne n'existent que dans une partie de la requête, et plus au moment d'ordonner l'union. Deux solutions peuvent être utilisées : i) renommer la colonne produite, et utiliser ce nouveau nom ou ii) utiliser le numéro d'ordre de la colonne dans le résultat.

Remarquez aussi que le SGBD PostgreSQL est moins permissif que MySQL sur certains points de la norme, et que les syntaxes diffèrent parfois :

- Les chaînes doivent être entre apostrophe et non entre guillemets (MySQL accepte les 2).
- L'opérateur `LIKE` ne fonctionne que sur des chaînes de caractères et PostgreSQL ne fait pas de transtypage implicite. Ainsi `gdate LIKE '2014%'` est refusé car `gdate` n'est pas une chaîne. Il faut écrire `CAST(gdate as text) LIKE '2014%'` qui transforme d'abord la date en chaîne.
- La fonction `YEAR()` n'est pas prévue par la norme SQL[a]. Il s'agit d'une fonction *propriétaire* utilisée par MySQL, Oracle... PostgreSQL utilise plutôt la fonction `DATE_PART()`[b] (qui elle aussi est propriétaire !). Pour extraire l'année d'une date, il faut donc remplacer `YEAR(gDate)` par `DATE_PART('year', gdate)`.

a. Voir par exemple : `http://sqlpro.developpez.com/cours/sqlaz/fonctions/#L1.7`
b. Voir : `http://www.postgresql.org/docs/9.4/static/functions-datetime.html`

2. les pays qui ont à la fois un pilote de course, une écurie et un grand prix en 2104

```
SELECT dCountry AS pays
FROM driver NATURAL JOIN racedriver
WHERE rSeason = 2014

INTERSECT

SELECT tCountry AS pays
FROM team NATURAL JOIN racedriver
WHERE rSeason = 2014

INTERSECT

SELECT cCountry AS pays
FROM circuit NATURAL JOIN grandprix
WHERE gdate >= '2014-01-01'
  AND gdate <= '2014-12-31'
```

```
ORDER BY pays
```

```
SELECT DISTINCT dCountry AS pays
FROM driver NATURAL JOIN racedriver
WHERE rSeason = 2014
  AND dCountry IN
        (SELECT DISTINCT tCountry AS pays
         FROM team NATURAL JOIN racedriver
         WHERE rSeason = 2014
           AND tCountry IN
                 (SELECT DISTINCT cCountry AS pays
                  FROM circuit NATURAL JOIN grandprix
                  WHERE gdate >= '2014-01-01'
                    AND gdate <= '2014-12-31'))
ORDER BY pays
```

On peut aussi mettre les 2 sous-requêtes au même niveau dans la requête principale.

```
SELECT DISTINCT dCountry AS pays
FROM driver NATURAL JOIN racedriver
WHERE rSeason = 2014
  AND dCountry IN
        (SELECT DISTINCT tCountry AS pays
         FROM team NATURAL JOIN racedriver
         WHERE rSeason = 2014)
  AND dCountry IN
        (SELECT DISTINCT cCountry AS pays
        FROM circuit NATURAL JOIN grandprix
        WHERE gdate >= '2014-01-01'
        AND gdate <= '2014-12-31')
ORDER BY pays
```

3. les pays qui ont un grand prix, mais pas de pilote de course en 2014

```
SELECT cCountry AS pays
FROM circuit NATURAL JOIN grandprix
WHERE gdate >= '2014-01-01'
  AND gdate <= '2014-12-31'

EXCEPT

SELECT dCountry AS pays
FROM driver NATURAL JOIN racedriver
WHERE rSeason = 2014

ORDER BY pays
```

```
SELECT DISTINCT cCountry AS pays
FROM circuit NATURAL JOIN grandprix
WHERE gdate >= '2014-01-01'
  AND gdate <= '2014-12-31'
```

```
  AND cCountry NOT IN
        (SELECT DISTINCT dCountry AS pays
         FROM driver NATURAL JOIN racedriver
         WHERE rSeason = 2014)
ORDER BY pays
```