Travaux pratiques sur base des étudiants
Pour s'entrainer en SQL

Romuald THION

1 Introduction

On considère les tables suivantes :

- `Etudiant (NumEt, NomEt, Adresse)` : le numéro, le nom et l'adresse des étudiants;
- `Enseignant (NumEns, NomEns)` : le nom et le prénom des enseignants;
- `UE (NumUE, Titre, HCours, HTD, HTP)` : le numéro et le titre de l'UE, ainsi que le nombre d'heures de cours magistraux, de TD et de TP par groupe d'étudiants;
- `Enseigne (NumEns, NumUE, NCours, NTD, NTP)` : indique dans quelle UE intervient quel enseignant en présisant le nombre de cours magistraux, de groupes de TD et de groupes de TP pour cet enseignant dans cette UE
- `Inscrit (NumEt, NumUE)` : indique quel étudiant est inscrit dans quelle UE

Exercice 1 : De la langue naturelle à SQL

Charger la base dans `sqlite3`, `sqlitebrowser` ou DBeaver et traduire les requêtes suivantes en SQL.

1. Donner tous les noms des étudiants. *Réponse attendue* :

```
"Armand A."
"Berthe B."
"Cendrine C."
"David D."
"Erwan E."
"Fabien F."
"Gerald G."
"Herbert H."
"Jacques J."
```

2. Donner les titres des UEs dans l'ordre alphabétique *Réponse attendue* :

```
"Algebre"
"Algorithmique"
"Analyse"
"Bases de donnes"
"Programmation"
"Reseaux"
```

3. Donner le titre des UEs dont le nombre d'heures total (cours, td et cm) par groupe est au moins 46. *Réponse attendue* :

```
"Algorithmique"
"Bases de donnes"
```

4. Donner les noms des étudiants qui ont 'Albert A.' comme enseignant. *Réponse attendue* :

```
"David D."
"Gerald G."
"Jacques J."
```

5. Donner les titres des cours ayant au moins un étudiant inscrit et dont le nombre d'heures de TD est au moins 18. *Réponse attendue* :

```
"Bases de donnes"
"Analyse"
"Algebre"
```

6. Donner les noms des enseignants qui enseignent dans une des UEs avec 'Albert A.' (sauf Albert A. lui-même). *Réponse attendue* :

```
"Bertrand B."
```

7. Donner le nombre total d'heures de cours/TD/TP dispensées à l'université. On nommera `TOTAL_HEURES` ce nombre. *Réponse attendue* (la dernière ligne demande un peu d'effort, on peut accepter une première version sans):

```
"1" "70.0"
"2" "70.0"
"3" "75.0"
"4" "95.0"
"5" "126.0"
"6" "0"
```

8. Donner le nombre d'UE n'ayant pas de TP (on appellera `NB_UES` l'attribut donnant ce résultat). *Réponse attendue* :

```
2
```

9. Donner le nombre d'étudiants qui suivent le cours d'Analyse (on appellera `NB_ETUDIANTS` l'attribut donnant ce nombre). *Réponse attendue* :

```
3
```

10. Donner le nombre d'heures moyen de cours, de TD et de TP. On appelera `MOY_COURS` la moyenne des heures, `MOY_TD` celle des TD et `MOY_TP` celle des TPs. *Réponse attendue* :

```
"16.5" "16.3333333333333" "8.33333333333333"
```

11. Donner pour chaque étudiant le nombre total d'heures où il est inscrit. On donnera dans le résultat le numéro de l'étudiant ainsi qu'un attribut `HEURES` qui indiquera son nombre d'heures. *Réponse attendue* (si on veut comme réponses les heures effectivement suivies où il y a un prof, c'est plus compliqué):

```
"1111" "149.0"
"1112" "104.0"
"1114" "143.0"
"1115" "50.0"
"1116" "62.0"
"1117" "185.0"
"1118" "62.0"
"1119" "135.0"
```

12. Donner les numéros des enseignants qui effectuent plus de 17 heures de cours magistraux. Attention, ici une clause `HAVING` est nécessaire. *Réponse attendue* :

```
"111"
"112"
"114"
"115"
```

Corrections

Solution de l'exercice 1

1. Notez que l'ordre des tuples *n'est pas garanti* sans clause `ORDER BY`

```
SELECT NomEt
FROM Etudiant;
```

2.
```
SELECT Titre
FROM UE
ORDER BY Titre;
```

3.
```
SELECT titre
FROM UE
WHERE hCours + hTD + hTP >= 46;
```

4. Première requête avec des jointures, ici avec la syntaxe `USING` que je préfère à `NATURAL JOIN`

```
SELECT DISTINCT NomEt
FROM   Etudiant
       JOIN Inscrit USING (numET)
       JOIN Enseigne USING (numUE)
       JOIN Enseignant USING (numEns)
WHERE NomEns = 'Albert A.';
```

5.
```
SELECT DISTINCT Titre
FROM UE JOIN Inscrit USING (numUE)
WHERE HTD >= 18;
```

6. C'est une des requêtes les plus difficiles, car il faut joindre plusieurs fois les mêmes tables (ce qu'on appelle l'*auto jointure*, mais qui n'a en fait rien de particulier fondamentelement). Voir la vidéo `https://youtu.be/XxJRKd7WMvA` pour l'explication du cheminement.

```
SELECT DISTINCT E2.NomEns
FROM (Enseignant E1 JOIN Enseigne C1 USING (numEns))
       JOIN
      (Enseignant E2 JOIN Enseigne C2 USING (numEns))
      ON C1.NumUE = C2.NumUE
WHERE E1.NomEns = 'Albert A.' AND E2.NomEns <> 'Albert A.';
```

7. Techniquement, la seconde réponse n'est *pas au programme* mais la question sera très probablement posée car elle est très classique.

```
-- la version de base, sans la dernière ligne
--
SELECT NumUE, SUM(NCours*HCours + NTD*HTD + NTP*HTP) AS TOTAL_HEURES
FROM Enseigne JOIN UE USING (NumUE)
GROUP BY NumUE;

-- pour avoir toutes les UE, y compris à 0 heures, il faut
--  * faire une jointure ouverte à gauche (LEFT JOIN) qui va assurer que
--    TOUTES les UEs seront bien dans le résultat, mëme si personne ne
--    les enseigne effectivement
--  * transformer la valeurs NULL du total pour ces tuples là avec COALESCE
--
SELECT UE.NumUE, COALESCE(SUM(NCours*HCours + NTD*HTD + NTP*HTP), 0) AS
    TOTAL_HEURES
FROM UE LEFT OUTER JOIN Enseigne USING (NumUE)
GROUP BY UE.NumUE;
```

8. ```
   SELECT COUNT(*) AS NB_UES
   FROM UE
   WHERE HTP = 0;
   ```

9. ```
   SELECT COUNT(*) AS NB_ETUDIANTS
   FROM UE JOIN Inscrit USING (numUE)
   WHERE Titre = 'Analyse';
   ```

10. ```
    SELECT AVG(HCOURS) AS MOY_COURS, AVG(HTD) AS MOY_TD, AVG(HTP) AS MOY_TP
    FROM UE;
    ```

11. ```
    -- l'interprétation la plus simple
    SELECT NumEt, SUM(HCOURS + HTD + HTP) AS HEURES
    FROM Inscrit I JOIN UE USING (numUE)
    GROUP BY NumEt;

    -- variante ou on garde bien les étudiants inscrits à rien
    SELECT E.NumEt, coalesce(SUM(HCOURS + HTD + HTP),0) AS HEURES
    FROM Etudiant E LEFT OUTER JOIN (Inscrit I NATURAL JOIN UE) I on E.NumEt = I.
        NumET
    GROUP BY E.NumEt;

    -- variante ou on est sur qu'un prof enseigne (mais sans vérifier s'il
        enseigne bien CM TD et TP)
    SELECT NumEt, SUM(HCOURS + HTD + HTP) AS HEURES
    FROM Inscrit I JOIN UE USING (numUE) JOIN Enseigne USING (numUE)
    GROUP BY NumEt;

    -- un peu compliqué si on doit vérifier que qu'un enseignant enseigne bien CM
        TD et TP dans la matière!
    ```

12. ```
    SELECT NumEns
    FROM Enseigne JOIN UE USING(NumUE)
    GROUP BY NumEns
    HAVING SUM(NCours*HCours) > 17;

    -- attention, la solution suivante ne marche pas dans le cas général
    -- un prof pourrait faire 17h mais en cumulant sur plusieurs UE, ci
    -- dessous c'est avec une seule !
    SELECT NumEns
    FROM Enseigne NATURAL JOIN UE
    WHERE NCours*HCours >17
    ```
    ```