[IMAC S2] Programmation Back-end

TD 03 : Bases de données - Compléments SQL

Compléments

Pascale Ho - 2018 - Ingénieur IMAC (Mise à jour le 07/03/2018)

Sous-requêtes

L'intérêt de faire des sous-requêtes (une requête à l'intérieur d'une autre requête) est entre autre d'éviter de faire des jointures (qui peuvent être longues dans certains cas), ou de faire une requête sur des résultats partiels (trop de requête dans cette phrase).

Exemples :

Plutôt que de servir d'une jointure pour chercher les emprunts de matériels effectués par les utilisateurs qui habitent à Moisiel :

```
SELECT e.id_emprunt
FROM Emprunt e, Utilisateur u
WHERE e.id_utilisateur = u.id_utilisateur
      AND ville = "Moisiel";
```

on peut faire une sous-requête dans **WHERE** :

```
SELECT id_emprunt
FROM Emprunt
WHERE id_utilisateur = (SELECT id_utilisateur
                        FROM Utilisateur
                        WHERE ville = "Moisiel");
```

Dans cet exemple, il y a de fortes chances qu'on récupère plusieurs utilisateurs. On va plutôt utiliser **IN** pour vérifier si l'id se trouve bien dans LES résultats.

```
SELECT id_emprunt
FROM Emprunt
WHERE id_utilisateur IN (SELECT id_utilisateur
                         FROM Utilisateur
                         WHERE ville = "Moisiel");
```

On peut aussi faire des sous-requêtes dans FROM. Dans ce cas, il faut donner un alias à la sous-requête (sinon SQL va planter). On veut avoir la date du dernier emprunt fait par les utilisateurs qui habitent à Moisiel (il est aussi possible de combiner jonctions et sous-requêtes) :

```
SELECT MAX(date_emprunt)
FROM (
      SELECT e.date_emprunt
      FROM Emprunt e
      INNER JOIN Utilisateur u ON e.id_utilisateur = u.id_utilisateur
      WHERE u.ville = "Moisiel"
) AS dates;
```

À noter qu'il est possible de faire référence dans une sous-requête à une valeur d'une/des table de la requête principale.

Ensemble de requêtes

Il est possible de faire des opérations d'ensemble (comme en mathématiques !!) sur plusieurs requêtes !

Union : l'opération va garder toutes les lignes présentes **dans la requête A et dans la requête B**. Attention à ce que ces deux requêtes renvoient les mêmes champs.

```
SELECT nom, prenom
FROM Utilisateur
WHERE ville = "Moisiel"
UNION
SELECT nom, prenom
FROM Utilisateur
WHERE age >= 25;
```

Intersection : l'opération va garder toutes les lignes présentes dans **les requête A et B en même temps**. Pareil, attention à ce que ces deux requêtes renvoient les mêmes champs.

```
SELECT nom, prenom
FROM Utilisateur
WHERE ville = "Moisiel"
INTERSECT
SELECT nom, prenom
FROM Utilisateur
WHERE age >= 25;
```

Différence : l'opération va garder les lignes qui ne sont présentes **ni dans la requête A ni dans la requête B**. Pareil, attention à ce que ces deux requêtes renvoient les mêmes champs.

```
SELECT nom, prenom
FROM Utilisateur
WHERE ville = "Moisiel"
EXCEPT
SELECT nom, prenom
FROM Utilisateur
WHERE age >= 25;
```