Programmation Web Dynamique 2

9. Requêtes avec jointures

Requêtes sur plusieurs tables

- La clause FROM dans la requête SELECT permet d'effectuer une requête sur plusieurs tables en même temps, ce qui permet effectivement d'obtenir des rangées constituées d'informations contenues dans plusieurs tables différentes.
- Pour se faire, il suffit de mettre une virgule entre le nom des tables que vous voulez utilisez.
- Par la suite, si des noms de colonnes sont identiques dans plusieurs tables différentes (ce qui ne devrait techniquement pas arriver, mais vous n'aurez pas toujours créé vous-mêmes les tables que vous utiliserez), il est possible de spécifier de quelle colonne on veut retirer de l'information en utilisant la syntaxe nom_table.nom_colonne.
- Pour obtenir des infos pertinentes, on répète généralement le lien exprimé par une clé étrangère dans la clause WHERE.
- Ex : Pour faire afficher le nom d'un film suivi du nom du réalisateur de ce film, informations contenues dans deux tables différentes (films et realisateurs), on utiliserait la clé étrangère idRealisateur située dans la table films :
  - SELECT films.nom AS nomDuFilm, realisateurs.nom AS nomDuRealisateur from films, realisateurs where films.realisateur = realisateurs. idRealisateur;
- Il est aussi possible d'attribuer des noms aux tables pour en raccourcir la référence en mettant le nouveau nom tout de suite après le nom réel de la table.
  - SELECT f.nom AS nomDuFilm, r.nom AS nomDuRealisateur from films f, realisateurs r where f.realisateur = r.idRealisateur ;
- Rien ne vous empêche, par la suite, d'ajouter des conditions de filtrage dans votre clause where par la suite!
  - SELECT f.nom AS nomDuFilm, r.nom AS nomDuRealisateur from films f, realisateurs r where f.realisateur = r.idRealisateur and f.annee = 1992;
- Il est aussi possible d'utiliser 3 tables, ou plus!

Requêtes avec jointure

- Les puristes du SQL disent que les requêtes effectuées sur plusieurs tables en utilisant la clause WHERE pour y faire les liens ne sont pas idéales :
  - En effet, la clause WHERE devrait techniquement être utilisée pour *filtrer* des données, et non pas élargir les données recherchées.
  - La lisibilité des requêtes est meilleure si la clause WHERE contient le filtrage, et que les jointures entre les tables sont exprimées par une autre clause.
  - Ils ont donc inventé la clause JOIN pour le faire.

Types de jointure

- Il existe plusieurs types de jointure :
  - La jointure interne
  - La jointure externe
  - La jointure croisée
  - La jointure d'union
  - La jointure naturelle (pas toujours supportée)

La jointure interne

- La jointure interne fait le lien entre deux tables, mais ne retournera que les lignes ayant une correspondance entre les deux tables jointes.
  - On la spécifie en utilisant INNER JOIN, ou tout simplement JOIN, puisque c'est le type de jointure par défaut.
  - Ex : SELECT films.nom AS nomDuFilm, realisateurs.nom AS nomDuRealisateur from films JOIN realisateurs ON films.realisateur = realisateurs.idRealisateur WHERE annee = 1998;
    - Dans le cas de la requête avec jointure interne, les films n'ayant pas de réalisateurs et les réalisateurs n'ayant pas de films ne sont pas affichés.

La jointure externe

- Les jointures externes sont pratiques pour rapatrier le maximum d'informations possibles. En effet, elles vont ramener aussi les rangées d'une des tables jointes même si celle-ci ne possède pas de rangée liée vers elle dans l'autre table.
  - Ex : Vous voulez obtenir tous les noms des réalisateurs et des films qu'ils ont fait, même s'ils n'ont pas réalisé de films!
- Elles se font généralement en spécifiant OUTER JOIN, accompagné de LEFT ou RIGHT pour spécifier le « côté » que l'on veut garder en cas de non-correspondance. Le côté fait référence à la position du nom de la table autour du mot JOIN.
- Ex :
  - SELECT films.nom, realisateur.nom FROM films LEFT OUTER JOIN realisateurs ON films.idRealisateur = realisateurs.id
    - Retournera tous les noms de films ainsi que les noms des réalisateurs de ces films, mais retournera aussi les noms de tous les films qui n'ont pas de réalisateurs!
  - SELECT films.nom, realisateur.nom FROM films RIGHT OUTER JOIN realisateurs ON films.idRealisateur = realisateurs.id
    - Retournera tous les noms de films ainsi que les noms des réalisateurs de ces films, mais retournera aussi les noms de tous les réalisateurs qui n'ont pas fait de films!
- Certains SGBD permettent l'utilisation de FULL OUTER JOIN, mais pas MySQL.