Jointures avancées avec MySQL

Hyacinthe MENIET

3 août 2019

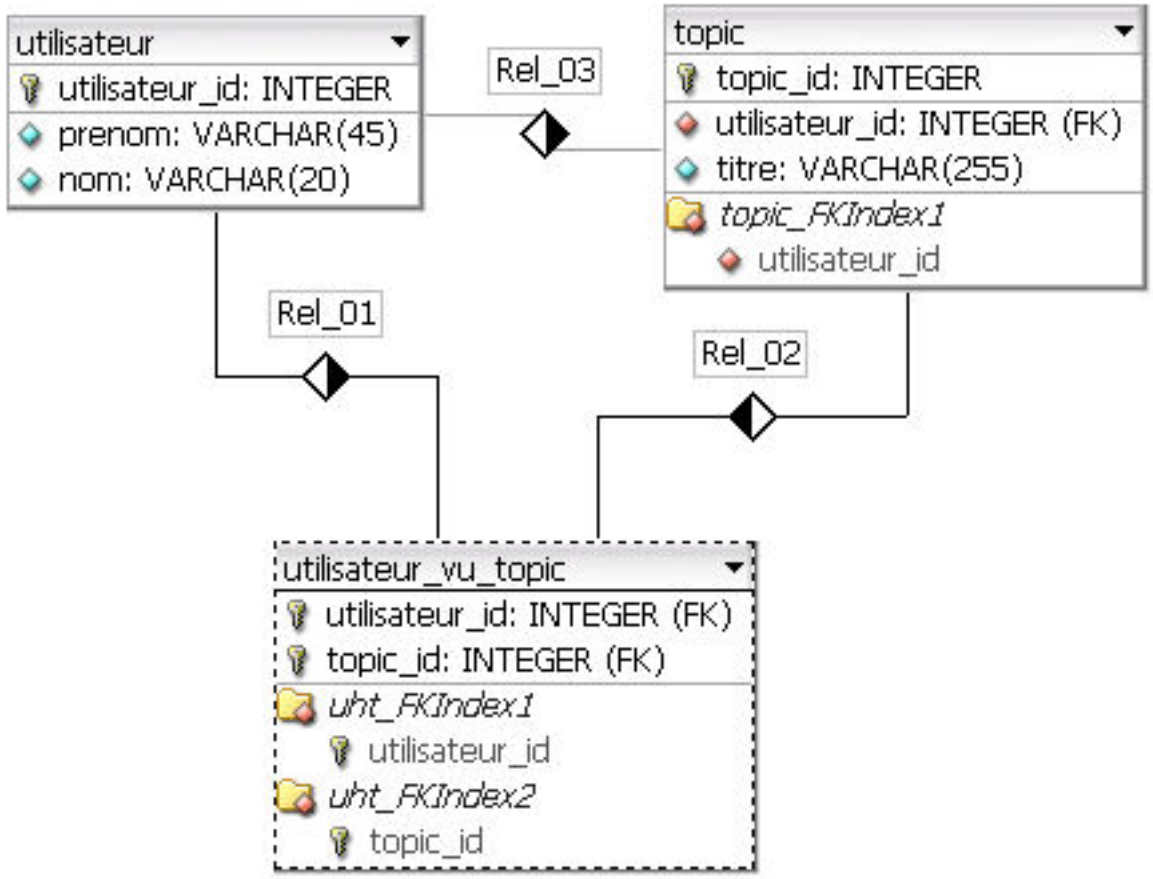
Dans ce document je vais traiter quelques exemples de jointure. Une jointure vous permet de sélectionner des enregistrements dans plusieurs tables grâce aux relations qui existent entre ces tables. C'est ce que vous faîtes avec la commande *« WHERE »*. Simplement ici vous utiliserez *« JOIN »* qui offre plus de souplesse.

1. Pré-requis

— Vous êtes familier de MySQL et de sa syntaxe SQL
— Vous disposez de MySQL en version 4.x

2. Initialisation des données

Dans la suite du document vous exécuterez des requêtes, ces requêtes supposent que vous ayez les tables suivantes :



La table *« utilisateur »* contient des individus. Ces individus peuvent créer des topics dans la table *« topic »*. La table *« utilisateur_vu_topic »* contient des couples (utilisateur,topic), chaque couple indique que l'utilisateur a consulté le topic associé.
Vous pouvez obtenir ce résultat avec les requêtes suivantes :

```
CREATE TABLE topic (
topic_id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
utilisateur_id INTEGER UNSIGNED NOT NULL,
titre VARCHAR(255) NULL,
PRIMARY KEY(topic_id),
INDEX topic_FKIndex1(utilisateur_id)
);
CREATE TABLE utilisateur (
utilisateur_id INTEGER UNSIGNED NOT NULL AUTO_INCREMENT,
prenom VARCHAR(45) NULL,
nom VARCHAR(20) NULL,
PRIMARY KEY(utilisateur_id)
);
CREATE TABLE utilisateur_vu_topic (
utilisateur_id INTEGER UNSIGNED NOT NULL,
topic_id INTEGER UNSIGNED NOT NULL,
PRIMARY KEY(utilisateur_id, topic_id),
INDEX uht_FKIndex1(utilisateur_id),
```

```
INDEX uht_FKIndex2(topic_id)
);
```

Peuplez les tables ci-dessus avec les données suivantes :

```
INSERT INTO `topic` (`topic_id`, `utilisateur_id`, `titre`)
VALUES (1, 3, 'La cuisine des pas riches: Recettes faciles pour
    fins de mois difficiles');
INSERT INTO `topic` (`topic_id`, `utilisateur_id`, `titre`)
VALUES (2, 8, 'Cuisine pour les nuls');
INSERT INTO `topic` (`topic_id`, `utilisateur_id`, `titre`)
VALUES (3, 1, 'Fabrication en tis: Les bases de la couture');
INSERT INTO `topic` (`topic_id`, `utilisateur_id`, `titre`)
VALUES (4, 2, 'Sacs et cabas: On les aime tous');
INSERT INTO `topic` (`topic_id`, `utilisateur_id`, `titre`)
VALUES (5, 6, 'Le dessin de mode: Techniques et illustration');
INSERT INTO `topic` (`topic_id`, `utilisateur_id`, `titre`)
VALUES (6, 4, 'Le tricot pour les nulls');
INSERT INTO `topic` (`topic_id`, `utilisateur_id`, `titre`)
VALUES (7, 4, 'Le vendeur qui voulait une bonne lessive');
INSERT INTO `topic` (`topic_id`, `utilisateur_id`, `titre`)
VALUES (8, 1, 'Ajax: mangez tout');
INSERT INTO `utilisateur` (`utilisateur_id`, `prenom`, `nom`)
    VALUES (1, 'Hyacinthe', 'MENIET');
INSERT INTO `utilisateur` (`utilisateur_id`, `prenom`, `nom`)
    VALUES (2, 'Richard', 'STALLMAN');
INSERT INTO `utilisateur` (`utilisateur_id`, `prenom`, `nom`)
    VALUES (3, 'Linus', 'TORVALDS');
INSERT INTO `utilisateur` (`utilisateur_id`, `prenom`, `nom`)
    VALUES (4, 'Bill', 'GATES');
INSERT INTO `utilisateur` (`utilisateur_id`, `prenom`, `nom`)
    VALUES (5, 'Steve', 'JOBS');
INSERT INTO `utilisateur` (`utilisateur_id`, `prenom`, `nom`)
    VALUES (6, 'Michael', 'DELL');
INSERT INTO `utilisateur_vu_topic` (`utilisateur_id`, `topic_id`)
     VALUES (1, 2);
INSERT INTO `utilisateur_vu_topic` (`utilisateur_id`, `topic_id`)
     VALUES (1, 5);
INSERT INTO `utilisateur_vu_topic` (`utilisateur_id`, `topic_id`)
     VALUES (1, 8);
INSERT INTO `utilisateur_vu_topic` (`utilisateur_id`, `topic_id`)
     VALUES (2, 6);
INSERT INTO `utilisateur_vu_topic` (`utilisateur_id`, `topic_id`)
     VALUES (3, 4);
INSERT INTO `utilisateur_vu_topic` (`utilisateur_id`, `topic_id`)
     VALUES (5, 7);
```

3. Les bases de la jointure

3.1 Jointure interne

Lorsque vous utilisez une jointure interne, ne sont inclus dans le résultat final que les lignes qui se correspondent dans les deux tables. Pour avoir uniquement les personnes qui ont créé un topic et les topics associés, vous faîtes une *« JOIN »* :

```
SELECT u.utilisateur_id, u.prenom, u.nom, t.topic_id, t.titre
FROM utilisateur u
JOIN topic t
ON u.utilisateur_id = t.utilisateur_id
```

3.2 Jointure externe gauche

Lorsque vous utilisez une jointure externe gauche, vous conservez l'intégralité des enregistrements de la table de gauche, y compris lorsqu'il n'existe pas d'enregistrements correspondants dans la seconde table.
Pour avoir tous les utilisateurs y compris ceux qui n'ont pas créés de topic, ainsi que les topics associés, quand il y en a, vous faîtes une *« LEFT JOIN »* :

```
SELECT u.utilisateur_id, u.prenom, u.nom, t.topic_id, t.titre
FROM utilisateur u
LEFT JOIN topic t
ON u.utilisateur_id = t.utilisateur_id
```

Pour avoir les utilisateurs qui n'ont pas créé de Topic c'est :

```
SELECT u.utilisateur_id, u.prenom, u.nom, t.topic_id, t.titre
FROM utilisateur u
LEFT JOIN topic t
ON u.utilisateur_id = t.utilisateur_id
WHERE t.topic_id is null
```

3.3 Jointure externe droite

Lorsque vous utilisez une jointure externe droite, vous conservez l'intégralité des enregistrements de la table de droite, y compris lorsqu'il n'existe pas d'enregistrements correspondants dans la première table.
Pour avoir tous les topics y compris ceux qui n'ont pas d'auteur valide, ainsi que les auteurs associés, quand il y en a, vous faîtes une *« RIGHT JOIN »* :

```
SELECT u.utilisateur_id, u.prenom, u.nom, t.topic_id, t.titre
FROM utilisateur u
RIGHT JOIN topic t
ON u.utilisateur_id = t.utilisateur_id
```

Pour avoir les topics qui ont été créés par un auteur inexistant :

```
SELECT u.utilisateur_id, u.prenom, u.nom, t.topic_id, t.titre
FROM utilisateur u
RIGHT JOIN topic t
ON u.utilisateur_id = t.utilisateur_id
WHERE u.utilisateur_id is NULL
```

3.4 Remarques

Dans les exemples ci-dessus, vous pouvez remplacer JOIN par INNER JOIN, LEFT JOIN par LEFT OUTER JOIN et RIGHT JOIN par RIGHT OUTER JOIN. Le résultat sera le même, ce sera juste plus long à écrire.

4. Pour aller plus loin

Maintenant je veux les livres que l'utilisateur *« 1 »* a lu. Pour cela je vais faire une jointure sur trois tables. Une fois qu'on a compris le principe, ça fonctionne avec quatre tables, cinq tables etc :

```
SELECT u.utilisateur_id, u.prenom, u.nom, t.topic_id, t.titre
FROM (utilisateur u JOIN utilisateur_vu_topic uht ON u.
    utilisateur_id = uht.utilisateur_id)
JOIN topic t ON t.topic_id = uht.topic_id
WHERE u.utilisateur_id = 1
```

De la même manière pour avoir les livres que l'utilisateur *« 1 »* n'a pas lu :

```
SELECT u.utilisateur_id, u.prenom, u.nom, t.topic_id, t.titre
FROM (utilisateur u JOIN utilisateur_vu_topic uht ON u.
    utilisateur_id = uht.utilisateur_id)
RIGHT JOIN topic t ON t.topic_id = uht.topic_id
WHERE u.utilisateur_id = 1 AND uht.utilisateur_id is NULL
```