IUT de Montpellier - Programmation Web - TD6
Associations entre classes

Rémi Coletta, Romain Lebreton, Bastien Vialla

Semaine du 03 Novembre 2014

Aujourd'hui nous continuons à développer notre site-école de covoiturage. En attendant de pouvoir gérer les sessions d'utilisateur, nous allons développer l'interface "administrateur" du site.

La séance dernière, nous avions un site qui propose une gestion minimale (Create / Read / Update / Delete) des utilisateurs et des trajets. Nous avions rendu le MVC 'Utilisateur' plus générique pour que le développement du MVC 'Trajet' soit plus simple et rapide.

L'objectif de ce TD est d'avoir des interactions entre les utilisateurs et les trajets. Nous allons donc inscrire des utilisateurs à des trajets, puis lister la liste des trajets d'un utilisateur ainsi que la liste des utilisateurs (conducteur + passager) d'un trajet.

1 Mise en route

Pour ce TD, nous vous passons un squelette de site qui sera à compléter et à enrichir.

Q 1. Récupérez sur `http://www.lirmm.fr/~lebreton/teaching.html` l'archive TD6.zip qui vous servira de base pour ce TD. Décompressez cette archive dans votre `public_html`. **Rentrez vos informations de connexion dans `./config/Conf.php`.**

Si vous utilisez NetBeans, ce qui est conseillé, créez un nouveau projet à partir du répertoire TD6 (File → New Project → 'Php application from existing sources' → sélectionnez le répertoire TD6).

2 Association entre utilisateurs et trajets

2.1 Dans la base de donnée

Vous avez couvert dans le cours "Analyse, Conception et Développements d'Applications" les diagrammes de classes. Ce type de diagramme est utile pour penser la base de donnée d'une application web.

Q 2. Si vous deviez dessiner les classes 'Utilisateur' et 'Trajet' et leur lien d'association, quelle multiplicité mettriez-vous ? En fonction de votre multiplicité, comment implémenteriez-vous l'association entre utilisateurs et trajets dans la base de donnée ?

Notre solution Dans ce TD, nous allons développer notre site pour une association entre utilisateurs et trajets de multiplicités non bornées. C'est-à-dire qu'on ne va pas limiter le nombre d'utilisateurs d'un trajet et inversement.

Q 3. Comment implémente-t-on cette association avec des bases de données ?

Réponse : On utilise une table de jointure.

Nous choisissons donc de créer une table `passager` qui contiendra trois champs :
— une clé primaire INT `id` qui numérote les associations,
— l'identifiant INT `trajet_id` d'un trajet et
— l'identifiant VARCHAR(32) `utilisateur_login` d'un utilisateur.

Pour inscrire un utilisateur à un trajet, il suffit d'écrire la ligne correspondante dans la table `passager` avec leur `utilisateur_login` et leur `trajet_id`.

On souhaite que le champ primaire 'id' s'incrémente à chaque nouvelle insertion dans la table. Pour ce faire, sélectionnez pour le champ 'id' la valeur par défaut `NULL` et cochez la case `A_I` (auto-increment). L'interclassement

général de votre table sera toujours `utf8_general_ci` (c'est l'encodage des données, et donc des accents, caractères spéciaux ...).

Q 4. On souhaite que le champ `passager.trajet_id` corresponde à tout moment à un identifiant de trajet `trajet.id`. Quelle est la fonctionnalité des bases de données qui permet ceci ?

Réponse : Il faut utiliser des clés étrangères.

Q 5. Créer la table `passager` en utilisant l'interface de PhpMyAdmin.

Attention : Pour supporter les clés étrangères, il faut que le moteur de stockage de toutes vos tables impliqués soit `InnoDB` . Vous pouvez choisir ce paramètre à la création de la table ou le changer après coup dans l'onglet `Opérations`.

Q 6. À l'aide de l'interface de PhpMyAdmin, faites de `trajet_id` et `login` des *index*.

Aide : Dans l'onglet `Structure` de la table passager, cliquer sur l'icône de l'action `index` en face des champs `trajet_id` et `utilisateur_login`.

Plus de détails : Dire que le champ `trajet_id` est un *index* revient à dire à MySql que l'on veut trouver rapidement les lignes qui ont un `trajet_id` donné. Du coup, MySql va construire une structure de donnée pour permettre cette recherche rapide. Une *clé étrangère* est nécessairement un *index* car on a besoin de ce genre de recherches pour tester rapidement la contrainte de clé étrangère.

Q 7. Rajoutez la contrainte de *clé étrangère* entre `passager.trajet_id` et `trajet.id`, puis entre `passager.utilisateur_login` et `utilisateur.login`. Nous allons utiliser le comportement `ON DELETE CASCADE` pour qu'une association soit supprimé si la clé étrangère est supprimée, et le comportement `ON UPDATE CASCADE` pour qu'une association soit mise à jour si la clé étrangère est mise à jour.

Aide : Dans l'onglet `Structure` de la table `passager`, il faut cliquer sur `Gestion des relations` pour accéder à la gestion des clés étrangères.

Q 8. À l'aide de l'interface de PhpMyAdmin, insérer quelques associations pour que la table `passager` ne soit pas vide.

2.2 Au niveau du PHP

2.2.1 Liste des utilisateurs d'un trajet et inversement

Nous allons maintenant pouvoir compléter le code PHP de notre site pour gérer l'association. Commençons par rajouter des fonctions à nos modèles 'Utilisateur' et 'Trajet'.

Avant toute chose, vous souvenez-vous comment faire une jointure en SQL ? Si vous n'êtes pas tout à fait au point sur les différents `JOIN` de SQL, vous pouvez vous rafraîchir la mémoire en lisant `http://www.w3schools.com/sql/sql_join.asp`.

Q 9. Créer une `public static function findUtilisateurs($data)` dans `ModelTrajet.php` qui prendra en entrée un tableau associatif `$data` avec un champ `$data['id']`. Cette fonction devra retourner la liste des utilisateurs inscrit aux trajet d'identifiant `$data['id']` en faisant la requête adéquate.

Indice : Utiliser une requête à base d'`INNER JOIN`. Une bonne stratégie pour développer la bonne requête est d'essayer des requêtes dans l'onglet SQL de PhpMyAdmin jusqu'à tenir la bonne.

Q 10. De la même manière, créer une `public static function findTrajets($data)` dans `ModelUtilisateur.php` qui prendra en entrée un tableau associatif `$data` avec un champ `$data['login']`.

Nous allons maintenant compléter la vue et le contrôleur de notre site. Comme nous avons déjà fait ensemble ce type exact d'exercice, nous allons juste vous donner le résultat voulu et vous laisser vous débrouiller pour y parvenir.

Q 11. Nous souhaitons avoir un lien `Liste des trajets` dans la vue de détail (action `read`) d'un utilisateur. Ce lien amènera vers une nouvelle vue associée à l'action `readAllTrajets` qui listera les trajets de l'utilisateur. La vue associée ressemblera à la vue de listage des trajets classique (vue `List`) mais avec un titre `Liste des trajets de l'utilisateur ...` : au lieu de `Liste des trajets :`.

Q 12. Faire la même chose mais avec la liste des utilisateurs d'un trajet.

2.2.2 Désinscrire un utilisateur d'un trajet et inversement

Rajoutons une dernière fonctionnalité : dans la vue qui liste les trajets d'un utilisateur, nous voudrions avoir un lien 'Désinscrire' qui enlèverait l'utilisateur courant du trajet sélectionné.

Q 13. Créer une `public static function deleteUtilisateur($data)` dans `ModelTrajet.php` qui prendra en entrée un tableau associatif `$data` avec deux champs `$data['trajet_id']` et `$data['utilisateur_login']`. Cette fonction devra désinscrire l'utilisateur `utilisateur_login` du trajet `trajet_id`.

Comme précédemment, nous allons vous donner le 'cahier des charges' de la fonctionnalité 'Désinscription' et nous vous laissons le soin de l'implémenter.

Q 14. Nous désirons avoir un lien 'Désinscription' dans la vue de liste des trajets d'un utilisateur (action `readAllTrajets`). Ce lien mènera vers une nouvelle vue associée à l'action `deleteUtilisateur` qui écrira `L'utilisateur .. a été désinscrit du trajet n°..` puis réaffichera la liste mise à jour des utilisateurs du trajet.

Q 15. Faire de même pour désinscrire un trajet d'un utilisateur.

Amélioration possible : En fait, les fonctions `deleteUtilisateur($data)` de `ModelTrajet.php` et `deleteTrajet($data)` de `ModelUtilisateur.php` sont identique. On peut donc faire une unique fonction `deletePassager($data)` dans `Model.php`. Ainsi `ModelUtilisateur` et `ModelUtilisateur` hériteront de cette fonction.

2.3 Et si le temps le permet

Voici une liste d'idée pour compléter notre site :

1. Notre liste des trajets d'un utilisateur est incomplète : il manque les trajets dont il est conducteur (et non passager). La page qui liste les trajets d'un utilisateur pourrait donner les deux listes comme conducteur et comme passager.
2. Similairement, nous avons oublié le conducteur de la liste des passagers d'un trajet. Le rajouter avec un statut à part.
3. Vous pouvez aussi éventuellement mettre en place des `trigger` dans votre SQL pour gérer le nombre de passager par véhicule, le fait qu'un passager ne soit pas inscrit deux fois à un trajet ...