LOG4420
TP3
Javascript, AJAX et sécurité

Julien Gascon-Samson
École Polytechnique de Montréal

26 octobre 2009

Contexte

La programmation côté-serveur permet de réaliser des applications complètement dynamiques et personnalisées selon les besoin de chaque utilisateur. Cependant, un problème très important n'est toujours pas résolu : l'interactivité. En effet, les actions posées par l'utilisateur sont toujours ponctuées d'un certain délai : le temps que le client transmette la requête HTTP au serveur, que le serveur prépare le document demandé et le transmette. Pour les applications comprenant de longs formulaires à remplir, il n'y a aucun problème puisque le temps de traitement devient négligeable par rapport au temps pris pour compléter ledit formulaire. Par contre, la tendance est davantage reliée à l'interactivité au sein du "Web 2.0". Il est fort utile de mettre en oeuvre des mécanismes alternatifs permettant d'optimiser l'expérience utilisateur.

Lors du TP précédent, vous avez implémenté une application web complète côté-serveur en utilisant PHP. La charge de travail vous a peut-être paru importante, mais dîtes-vous que le plus gros est fait, les TPs suivants seront passablement moins longs !

Dans le cadre de ce TP, nous utiliserons le langage de programmation Javascript qui permet d'exécuter du code pour valider et interagir localement avec le contenu de la page actuelle. Nous utiliserons également Ajax, qui permet de réaliser des requêtes au serveur en réponse à différents événements et de modifier la page en fonction des données reçues, le tout sans recharger la page. Enfin, il faudra implémenter quelques notions de sécurité afin de protéger le site web des attaques d'injection SQL et de *cross-site scripting* (XSS).

Présentation du travail

Dans ce travail, vous devez dans un premier temps utiliser Javascript pour valider le formulaire d'inscription utilisé par les visiteurs pour s'inscrire au site. Par la suite, vous devrez modifier la façon dont les membres procèdent à l'achat de billets en leur permettant de visualiser graphiquement les sièges de l'aréna lors de l'achat. Pour ce faire, `Ajax` sera utilisé. Enfin, vous devrez protéger certaines pages du site contre les attaques d'injection SQL et XSS. La liste des pages potentielles sur lesquelles je pourrai exécuter des tests de sécurité seront données plus bas.

Validation du formulaire d'inscription

Vous devez valider votre formulaire d'inscription avant de procéder à la soumission au serveur. Autrement dit, l'inscription doit être envoyée au serveur seulement si tous les critères de validation sont respectés. Les critères de validation sont les suivants :

- Tous les champs sont remplis (aucun champ n'est laissé vide)
- Les noms et prénoms doivent contenir uniquement des lettres, espaces, tirets et au minimum 2 caractères. Attention, les accents sont permis !
- L'adresse courriel est la plus compliquée à valider. Elle doit être de la forme standard et est composée de deux parties : usager@domaine.ext. Les caractères autorisés sont les suivants : lettres sans accents, chiffres, tiret, trait de soulignement, point, symbole +.
- Le nom d'utilisateur inclut des lettres sans accents et chiffres uniquement. Minimum 5 caractères.
- L'utilisateur doit être en accord avec les termes de la licence !

– Le nom d'utilisateur choisi ne doit pas déjà avoir été utilisé.

`Javascript` [1] doit être utilisé pour cette partie. L'utilisation des expressions régulières est fortement recommandée. Vous trouverez des informations utiles dans le document *JavaScript Sheet* sur Moodle.

Réservation de sièges

Vous devez fournir une vue graphique de l'aréna dans lequel se déroulera le match. Tel que décrit plus tard, quelques changements seront apportés aux tables du schéma de base de données. Au lieu du nombre de sièges total, les arénas contiendront plutôt une largeur et une profondeur qui serviront à décrire sous forme matricielle la configuration physique d'un aréna.

Ainsi, lorsque l'utilisateur réserve des billets pour un match, il accède à une page qui représente la matrice de sièges de l'aréna, tel que représenté à la figure 1. Vous pourrez représenter la matrice à l'aide d'un tableau `HTML` où chaque cellule aura une couleur définie selon un code de couleurs représentant l'état du siège (disponible, réservé par un autre membre, acheté par un autre membre, réservé par vous, acheté par vous).

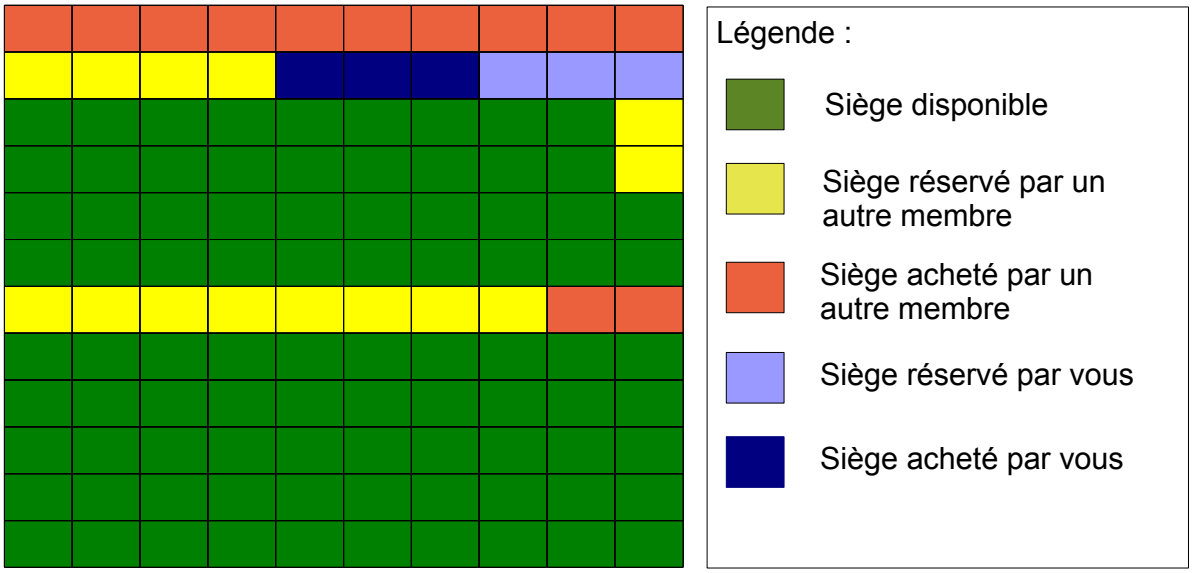

FIGURE 1 – Exemple d'un aréma de largeur de 10 sièges et d'une profondeur de 12 rangées et code de couleurs pour les sièges

Lorsque la page est chargée, la matrice de siège est initialisée selon l'état actuel de l'aréna pour le match concerné. Par la suite, afin d'accomoder la nature hautement dynamique du processus de réservation et d'achat de billets (sait-on jamais, peut-être que des milliers de personnes utiliseront simultanément votre système), il faut actualiser les informations. Vous devez implémenter une fonction qui sera appelée lorsque la souris passera au-dessus des cellules ("mouse over") du tableau ; votre fonction effectuera une requête `Ajax` qui rafraîchira ensuite la cellule en question.

1. Je peux peut-être tolérer aussi l'assembleur ;)

De plus, également lorsque l'utilisateur bouge la souris au dessus des sièges, la position de la place choisie est affichée (numéro de siège et sa rangée). L'utilisateur peut à ce moment effectuer deux actions :

Réserver un billet Valide uniquement pour les billets dont le statut est à *disponible* (vert). L'utilisateur clique sur le siège disponible, une requête `Ajax` est alors envoyée au serveur qui effectue les vérifications pour s'assurer que quelqu'un ne l'aurait pas déjà réservée entre temps. Si la réservation est possible, la transaction est alors enregistrée dans la table des réservations de la base de données et une réponse positive est retournée au fureteur client qui met à jour la couleur de la case dans sa grille. Si la transaction n'est plus possible, le fureteur du client est avisé et affiche un message d'erreur à l'utilisateur.

Annuler une réservation Valide uniquement pour les billets dont le statut est à *réservé* par vous (bleu pâle). L'utilisateur clique sur le siège réservé à annuler, une requête `Ajax` est envoyée au serveur qui retourne le nouveau statut pour le siège [2].

Afin d'encourager les clients à sortir leur carte de crédit le plus rapidement possible, il faut raccourcir le délai d'expiration des réservations effectuées par rapport au TP précédent. Il faudrait qu'au bout de 5 minutes, tout billet réservé (peu importe le match et l'utilisateur) qui n'aurait pas été acheté expire.

Sécurité de l'application web

Dans un monde où le prix du billet du canadien est sans cesse à la hausse, il pourrait être tentant de s'introduire dans votre application web pour obtenir certaines réductions de prix par exemple [3]. Dans cette optique, on vous demande d'ajouter des notions élémentaires de sécurité à votre site, tel que vu en cours. Cependant, sécuriser le site en entier est une tâche relativement longue alors nous vous demanderons de sécuriser uniquement certaines pages. Voici les pages qui seront vérifiées :

- Inejction SQL : connexion au site (formulaire de connexion), affichage des détails d'un match
- Attaque XSS : ajout/modification d'un aréna, ajout/modification d'un match

2. Remarque : il pourrait théoriquement être possible que le siège aie expiré immédiatement avant que l'utilisateur ne l'aie sélectionné et qu'un autre utilisateur l'aie réservé à son tour - il faut prévoir cette situation et mettre à jour correctement la couleur, c'est-à-dire ne pas prendre pour acquis que le siège redeviendra automatiquement disponible.

3. Il fallait un prétexte...

Modèle de la base de données

Nous vous fournissons un schéma (liste des tables) légèrement modifié par rapport à la base de données du TP2. Nous vous fournissons également un fichier “backup” qui contient du code SQL permettant de préparer le schéma de votre base de données avec MySQL Administrator, tel que réalisé au TP2 (le contenu de ce fichier se trouve en annexe).

Création du schéma

Voici un rappel des instructions pour importer votre schéma de base de données. Lancer l'utilitaire `MySQL Administrator` situé dans le menu `Programmation`. Connectez-vous en suivant les étapes illustrées dans la figure 2. Procédez ensuite à la restauration du backup en suivant les étapes décrites dans la figure 3.

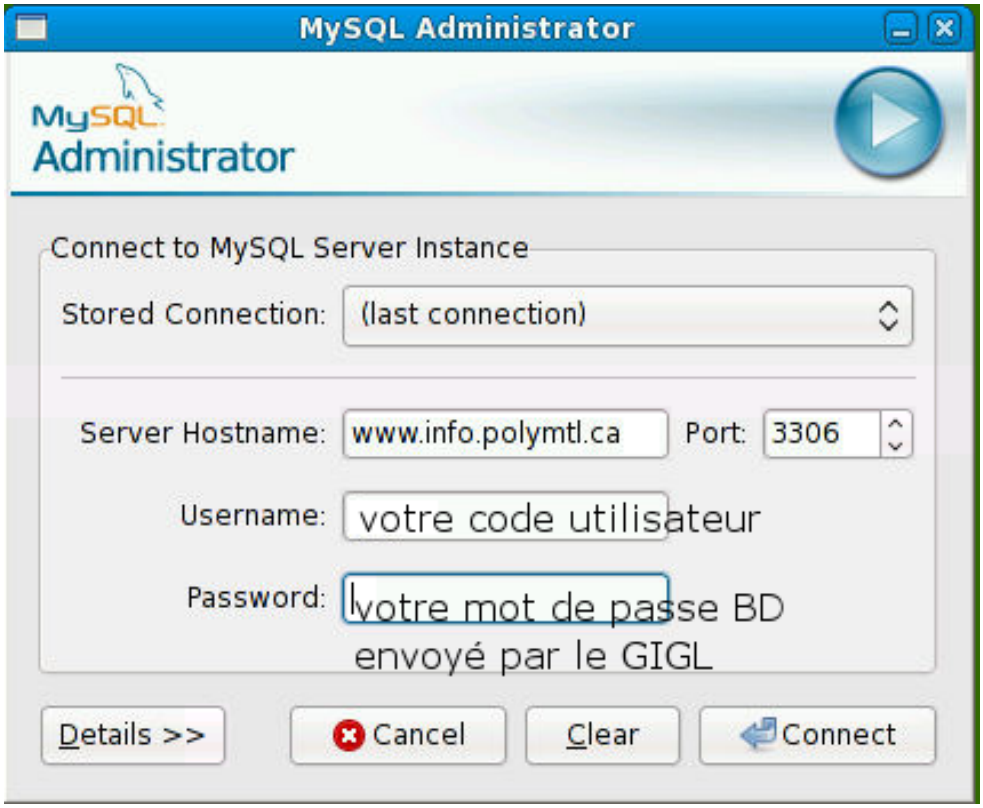

FIGURE 2 – Connexion à `MySQL Administrator`

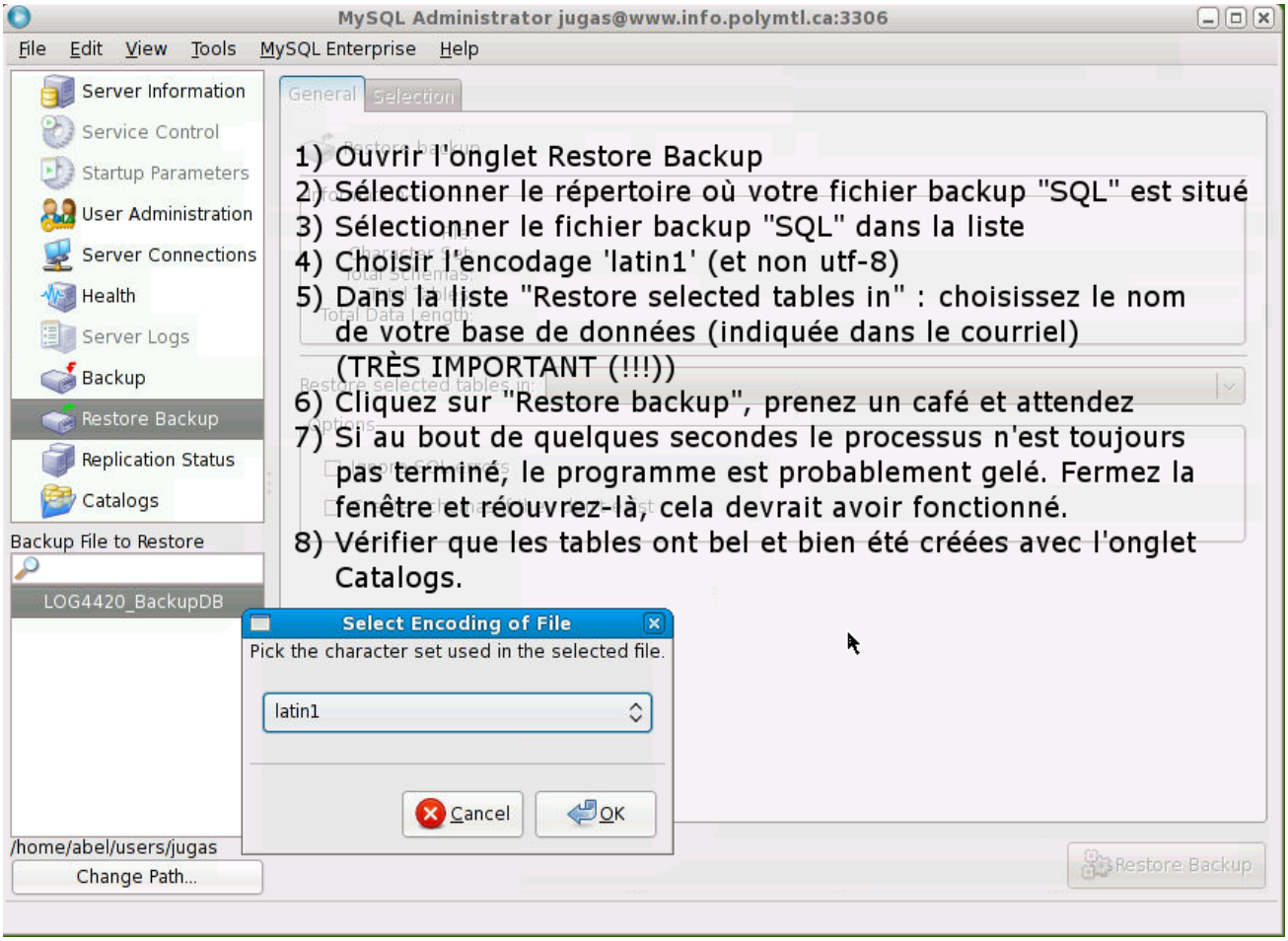

FIGURE 3 – Restauration du "backup" avec `MySQL Administrator`

Pour ne pas mélanger vos anciennes tables avec vos nouvelles tables [4], le nom de chacune des tables est précédé de *tp3_*.

4. Mais surtout pour faciliter ma correction du TP2 ;)

Table `tp3_arenas`

Champ	Type	Description
id	entier	Index de la salle [clé primaire]
nom	texte	Nom de l'aréna
~~sieges~~	~~entier~~	~~Nombre de sièges au total dans cet aréna~~
largeur	**entier**	**Nombre de sièges en largeur dans cet aréna**
profondeur	**entier**	**Nombre de sièges en profondeur dans cet aréna**

Table `tp3_matchs`

Champ	Type	Description
id	entier	Index du match [clé primaire]
description	texte	`Equipe1` vs `Equipe2`
arena	entier	**Id** de l'aréna dans lequel le match se déroulera
date	date	Date du match
prix	entier	Prix des billets
~~places~~	~~entier~~	~~Nombre de places restantes pour ce match~~

Table `tp3_reservations`

Champ	Type	Description
id	entier	Index de la réservation [clé primaire]
utilisateur	entier	**Id** de l'utilisateur qui a effectué la réservation
id_match	entier	**Id** du match pour lequel porte la réservation
~~qte~~	~~entier~~	~~Quantité de billets réservés~~
siege	**entier**	**Index du siège dans une rangée donnée ("colonne")**
rangee	**entier**	**Index de la rangée de l'aréna**
expiration	date/heure	Moment où la réservation expirera

Attention : désormais, il n'y a qu'un seul billet par réservation puisque le billet est automatiquement réservé dès que l'utilisateur le sélectionne dans la matrice des sièges pour les matchs !

Noter qu'une entrée est ajoutée à la table réservation lorsqu'un membre place un match dans son panier. Tel que mentionné précédemment, pour éviter la situation où nous pourrions excéder accidentellement la capacité de l'aréna, il faut décrémenter le nombre de places restantes pour le match dès que la réservation est effectuée. Toutefois, si l'utilisateur décide de vider son panier ou si le délai de validité de la réservation expire, le nombre de places restantes sera "restauré" (donc incrémenté). Il pourrait être judicieux de définir une procédure de "nettoyage" des réservations expirées. Cette procédure pourrait être appelée dès qu'une nouvelle réservation survient ou est libérée, par exemple.

Une fois le panier d'achat confirmé, les réservations seront converties en achats.

Table `tp3_achats`

Champ	Type	Description
id	entier	Index de l'achat [clé primaire]
utilisateur	entier	**Id** de l'utilisateur qui a effectué l'achat
id_match	entier	**Id** du match pour lequel porte la l'achat
~~qte~~	~~entier~~	~~Quantité de billets achetés~~
siege	**entier**	**Index du siège dans une rangée donnée ("colonne")**
rangee	**entier**	**Index de la rangée de l'aréna**
date	date/heure	Moment où l'achat a été effectué

Attention : désormais, il n'y a qu'un seul billet par achat également puisque l'ancien format ne permettait pas de spécifier les sièges. Il y aura donc un achat pour chaque réservation confirmée.

Table `tp3_utilisateurs`

Champ	Type	Description
id	entier	Index de l'utilisateur [clé primaire]
role	entier	Rôle de l'utilisateur (1=Membre,2=Admin)
utilisateur	texte	Nom d'utilisateur (login)
motdepasse	texte	Mot de passe (password)
prenom	texte	Prénom de l'utilisateur
nom	texte	Nom de l'utilisateur
courriel	texte	Courriel de l'utilisateur
jour	entier	Jour de naissance de l'utilisateur
mois	entier	Mois de naissance de l'utilisateur
annee	entier	Année de naissance (entre 1900 et 2010 :P)
sexe	entier	Sexe de l'utilisateur (1=Homme,2=Femme,3=Génie logiciel)
theme	texte	Thème choisi par l'utilisateur

Évaluation

Remettez votre travail avant le 51 octobre 2009 [5] à 23h55. Vous devez compresser votre arborescence (qui incluera tous les fichiers apprioriés) sous un fichier ZIP. Nommez votre fichier ZIP selon la convention suivante : `TP3_Matricule1_Matricule2.zip`. Téléversez votre fichier ZIP sur Moodle. Mettez également votre site web en ligne sur le serveur www.info.polymtl.ca/... afin que je puisse le corriger facilement et assurez-vous qu'il soit pleinement fonctionnel sur ce serveur avant de remettre puisque votre TP sera hébergé et évalué par ce serveur pour la correction. Placez un fichier texte (Readme.txt) contenant l'adresse URL exacte pour accéder à votre site dans votre fichier ZIP. Une pénalité de 10% par jour de retard s'applique. Le barème est le suivant :

- La validation suivante est effectuée : tous les champs doivent être remplis (1pt)
- La validation suivante est effectuée : le prénom correspond aux critères (0.5pt)
- La validation suivante est effectuée : le nom correspond aux critères (0.5pt)
- La validation suivante est effectuée : l'adresse courriel correspond aux critères (1pt)
- La validation suivante est effectuée : le nom utilisateur correspond aux critères (0.5pt)
- La validation suivante est effectuée : le case d'acception des termes est cochée (0.5pt)
- La validation suivante est effectuée : le nom d'utilisateur choisi est inexistant (1pt)
- Réservation de sièges : la matrice est initialisée au début et rafraîchie tel que spécifié ("mouse over") et affiche les bonnes informations (2.5pt)
- Réservation de sièges : la réservation fonctionne tel que spécifié (1.5pt)
- Réservation de sièges : l'annulation de réservations fonctionne tel que spécifié (1pt)
- Réservation de sièges : les réservations sont converties en achats selon le nouveau mode (0.5pt)
- Réservation de sièges : les réservations expirent correctement selon le nouveau délai (1pt)
- Réservation de sièges : les informations sur le siège sont affichées lorsque la souris passe au-dessus (1pt)
- Sécurité (injection SQL) : le formulaire de connexion n'est pas vulnérable (1.5pt)
- Sécurité (injection SQL) : la page d'affichage des détails de matchs n'est pas vulnérable (1.5pt)
- Sécurité (attaque XSS) : la page d'ajout et modification d'aréna n'est pas vulnérable (1.5pt)
- Sécurité (attaque XSS) : la page d'ajout et modification de match n'est pas vulnérable (1.5pt)
- Compatibilité : les pages existantes de votre site ont été ajustées pour prendre en compte le nouveau schéma de BD (1.5pt)

Total : 20 points.

Bon deux semaines [6] !
Julien Gascon-Samson, chargé de travaux dirigés

5. Ce qui correspond au 20 novembre 2009

6. Ceux et celles qui jouent à Starcraft et qui veulent partir du bon pied peuvent écouter le vidéo suivant : http ://www.youtube.com/watch ?v=DzpRuWkAjxg

Annexe

Voici le contenu du fichier "backup" (`LOG4420_BackupDB_TP3.sql`) à restaurer à l'aide de `MySQL Administrator` :

```
-- phpMyAdmin SQL Dump
-- version 3.1.2deb1ubuntu0.1
-- http://www.phpmyadmin.net
--
-- Serveur: localhost
-- Généré le : Lun 26 Octobre 2009 à 17:41
-- Version du serveur: 5.0.75
-- Version de PHP: 5.2.6-3ubuntu4.2

SET SQL_MODE="NO_AUTO_VALUE_ON_ZERO";

--
-- Base de données: `log4420a2009`
--

-- --------------------------------------------------------

--
-- Structure de la table `tp3_achats`
--

CREATE TABLE IF NOT EXISTS `tp3_achats` (
  `id` int(11) NOT NULL auto_increment,
  `utilisateur` int(11) NOT NULL,
  `id_match` int(11) NOT NULL,
  `siege` int(11) NOT NULL,
  `rangee` int(11) NOT NULL,
  `date` date NOT NULL,
  PRIMARY KEY  (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;

--
-- Contenu de la table `achats`
--


-- --------------------------------------------------------

--
-- Structure de la table `tp3_arenas`
--
```

```
CREATE TABLE IF NOT EXISTS `tp3_arenas` (
  `id` int(11) NOT NULL auto_increment,
  `nom` varchar(50) NOT NULL,
  `largeur` int(11) NOT NULL,
  `profondeur` int(11) NOT NULL,
  PRIMARY KEY  (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=7 ;

--
-- Contenu de la table `tp3_arenas`
--

INSERT INTO `tp3_arenas` (`id`, `nom`, `largeur`, `profondeur`) VALUES
(1, 'Air Canada Centre', 10, 12),
(2, 'Centre Bell', 11, 8),
(3, 'Scotiabank Place', 12, 12),
(4, 'Verizon Center', 6, 20),
(5, 'HSBC Arena', 8, 14),
(6, 'Madison Square Garden', 6, 6);

-- --------------------------------------------------------

--
-- Structure de la table `tp3_matchs`
--

CREATE TABLE IF NOT EXISTS `tp3_matchs` (
  `id` int(11) NOT NULL auto_increment,
  `description` varchar(50) NOT NULL,
  `arena` int(11) NOT NULL,
  `date` date NOT NULL,
  `prix` int(11) NOT NULL,
  PRIMARY KEY  (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=11 ;

--
-- Contenu de la table `tp3_matchs`
--

INSERT INTO `tp3_matchs` (`id`, `description`, `arena`, `date`, `prix`) VALUES
(1, 'Canadiens de Montreal vs New Jersey Devils', 1, '2009-12-07', 100),
(2, 'Canadiens de Montreal vs New York Islanders', 3, '2009-12-08', 1850),
(3, 'Canadiens de Montreal vs Pittsburgh Penguins', 5, '2009-12-09', 5),
(4, 'Canadiens de Montreal vs Philadelphia Flyers', 3, '2009-12-10', 70),
```

```
(5, 'Canadiens de Montreal vs New York Rangers', 2, '2009-12-11', 450),
(6, 'Canadiens de Montreal vs New York Islanders', 4, '2009-12-12', 222),
(7, 'Canadiens de Montreal vs New Jersey Devils', 6, '2009-12-13', 199),
(8, 'Canadiens de Montreal vs New Jersey Devils', 6, '2009-12-14', 110),
(9, 'Canadiens de Montreal vs Pittsburgh Penguins', 1, '2009-12-15', 514),
(10, 'Canadiens de Montreal vs New York Rangers', 5, '2009-12-16', 300);

-- --------------------------------------------------------

--
-- Structure de la table `tp3_reservations`
--

CREATE TABLE IF NOT EXISTS `tp3_reservations` (
  `id` int(11) NOT NULL auto_increment,
  `utilisateur` int(11) NOT NULL,
  `id_match` int(11) NOT NULL,
  `siege` int(11) NOT NULL,
  `rangee` int(11) NOT NULL,
  `expiration` datetime NOT NULL,
  PRIMARY KEY  (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO_INCREMENT=1 ;

--
-- Contenu de la table `tp3_reservations`
--


-- --------------------------------------------------------

--
-- Structure de la table `tp3_utilisateurs`
--

CREATE TABLE IF NOT EXISTS `tp3_utilisateurs` (
  `id` int(11) NOT NULL auto_increment,
  `role` int(11) NOT NULL,
  `utilisateur` varchar(50) NOT NULL,
  `motdepasse` varchar(50) NOT NULL,
  `prenom` varchar(50) NOT NULL,
  `nom` varchar(50) NOT NULL,
  `courriel` varchar(50) NOT NULL,
  `jour` int(11) NOT NULL,
  `mois` int(11) NOT NULL,
  `annee` int(11) NOT NULL,
```

```
  `sexe` int(11) NOT NULL,
  `theme` varchar(50) NOT NULL,
  PRIMARY KEY  (`id`)
) ENGINE=InnoDB  DEFAULT CHARSET=latin1 AUTO_INCREMENT=2 ;

--
-- Contenu de la table `tp3_utilisateurs`
--

INSERT INTO `tp3_utilisateurs` (`id`, `role`, `utilisateur`,
`motdepasse`, `prenom`, `nom`, `courriel`, `jour`, `mois`,
`annee`, `sexe`, `theme`) VALUES
(1, 2, 'admin', 'admin', 'Yejamais', 'Troptard',
'luc-rozon.smith-cambell@caramail.fr', 1, 1, 1901, 1, 'Standard');
```