Base de données

Professeur: S. Hiard

Projet - groupe 12

Wanet Sarah, Catteeuw Tristan, Heylen Martin

Université de Liège
Année académique: 2019-2020

1 Introduction

Cette deuxième partie du projet réalisé dans le cadre du cours de "Base de données" consiste à développer un site web permettant d'interagir avec une base de données centralisant les données de visionnage de séries télévisées.
Les différentes requêtes utilisées pour permettre à l'utilisateur d'interagir avec la base de données sont explicitées dans ce rapport.
La première partie du rapport sera centrée sur la description de la structure du site web et sur l'initialisation de la base de données.
Ensuite, les requêtes pour manipuler ces données seront explicitées comme indiqué précédemment.

2 Structure du site web

Cette section est consacrée à la description de la structure du site web ayant pour but d'interagir avec la base de données. Le site web (accessible via le lien suivant
`http://www.student.montefiore.ulg.ac.be/~mheylen/`)
est sécurisé par une page de login où identifiant et mot de passe doivent être insérés.
Dans le cadre de ce projet, l'identifiant et le mot de passe sont:

- Identifiant: group12
- Mot de passe: KS/7yNYP8l

Ensuite, la page va vérifier dans la base de données distante si le couple identifiant-mot de passe est bien valide. Pour ce faire, une requête est envoyée à la base de données *php_users_login* pour vérifier si les champs sont valides.
La requête est
SELECT (*)
FROM `php\_user\_login`
WHERE `username` = "Identifiant"
AND `password` = "Mot de passe"

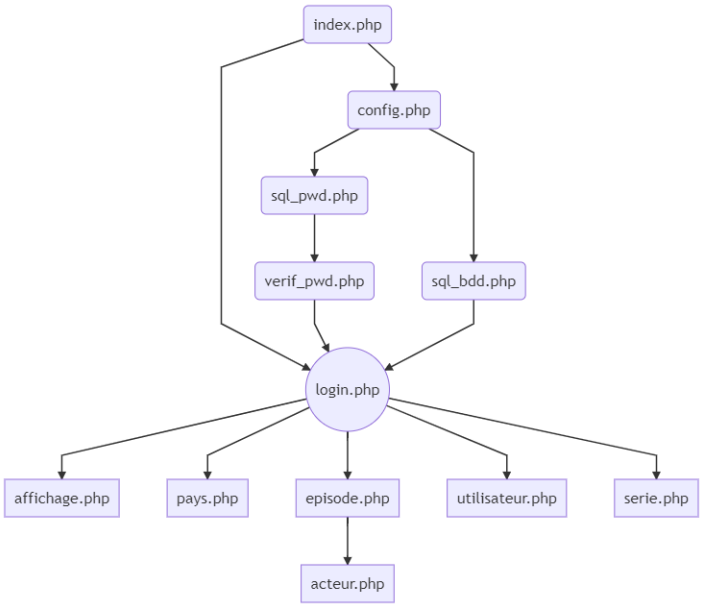

Figure 1: Structure du site web

Une fois la vérification effectuée, l'utilisateur obtient alors le droit d'accéder à une interface permettant d'envoyer diverses requêtes à la base de données centralisant les données de visionnage de séries télévisées.
La structure du site web est représentée par le flow chart présent dans la Figure 1. Comme réprésenté sur le schéma, le noyau central du site web est la page **login.php**. Celle-ci permet de faire le pont entre la page d'accueil **index.php** et les différentes pages permettant de réaliser des manipulations de données.
Les manipulations pouvant être réalisées sont différentes selon la page selectionnée par l'utilisateur:

- **affichage.php**: Après sélection d'une des tables de la base de données, l'utilisateur a accès à la liste des attributs composant la table. Il peut ensuite réaliser une recherche dans la table sur base de ces attributs.

- **pays.php**: Sur base de la plateforme de streaming spécifiée par l'utilisateur, ce dernier peut connaître la liste des pays où celle-ci est disponible.

- **episode.php**: L'utilisateur obtient les droits d'écriture dans la table episode où il peut rajouter un épisode à une série. En spécifiant le nom de la série, le numéro de la saison et de l'épisode, la durée de l'épisode, son synopsis et les acteurs jouant dans l'épisode, une requête sera envoyée à la base de données pour insérer le nouvel épisode.

  Si un tuple déjà existant dans la table est identifié par le même nom de la série et les mêmes numéros de la saison et de l'épisode que ceux insérés par l'utilisateur, la requête sera refusée.

  De même, si l'acteur choisi joue déjà dans l'épisode ou n'existe pas, la requête sera refusée également.

- **utilisateur.php**: Cette page liste les abonnés ayant regardé tous les épisodes de la série Black Mirror.

- **serie.php**: Cette page liste les épisodes des séries triés par le nombre d'utilisateurs l'ayant regardé, par ordre décroissant.

Toutes les requêtes utilisées sont explicitées dans une section ultérieure.

3 Mise en place de la base de données

Pour permettre à l'utilisateur d'accéder à la base de données, il faut dans un premier temps l'initialiser. Comme il a été mentionné précédemment, il y a d'une part la table *php_users_login* (qui sert à la vérification de l'identifiant et du mot de passe), et d'autre part les tables appartenant à la base de données des visionnages.
Ces différentes tables sont initialisées en lisant de manière séquentielle les scripts "sql_pwd.sql" et "sql_bdd.sql". Ensuite, concernant les tables liées à la base de données des visionnages, les différents tuples sont chargés depuis des fichiers texte au format .csv.
Il faut noter que l'ordre de création et de remplissage des tables a son importance. En effet, si on suppose que la table B dépend de la table A, cette dernière doit être créée avant.

Un exemple de création et de remplissage de la table episodes dépendant de la table serie est indiqué ci-dessous.

```
CREATE TABLE IF NOT EXISTS episodes
(
n_saison int not null,
n_episode int not null,
duree int,
synopsys varchar (1024),
nom_serie varchar (32) not null,
primary key (n_saison, n_episode, nom_serie),
foreign key (nom_serie) references serie (nom_serie)
) ENGINE=InnoDB;
LOAD DATA LOCAL INFILE './contenu_tables/episodes.csv'
INTO TABLE episodes
FIELDS
TERMINATED BY ';'
ENCLOSED BY '"'
ESCAPED BY '\'
LINES
TERMINATED BY '\r\n';
```

La lecture séquentielle des fichiers "sql_pwd.sql" et "sql_bdd.sql" est réalisée dans les scripts "sql_pwd.php" et "sql_bdd.php".

4 Liste des requêtes SQL

Dans chacune des pages accessibles depuis la page **login.php**, des requêtes SQL sont générées pour permettre d'interagir avec les tables de la base de données. Pour chacune des pages, les requêtes réalisées vont être décrites.

4.1 Page affichage.php

Cette page réalise de manière séquentielle plusieurs requêtes SQL. La première requête consiste à demander au serveur de fournir les différents attributs composant la table sélectionnée par l'utilisateur.

SELECT (*)
FROM `$_SESSION["liste"]`
où `$_SESSION["liste"]` est remplacé par le nom la table sélectionnée.

Suite à cette requête, la liste des attributs constituant la table sélectionnée sont affichés. L'utilisateur a alors la possibilité de remplir certains champs pour filtrer les données présentes dans la table. La requête SQL associée à cette opération est indiquée ci-dessous.

SELECT (*)
FROM `$_SESSION["liste"]`
WHERE `$condition`
Le contenu de `$condition` depend du nombre de champs remplis par l'utilisateur. Pour chaque champ rempli, une condition est générée. Ces conditions sont ensuite liées via un opérateur AND . De plus, l'écriture de la condition dépend de la nature de l'attribut.
En effet, considérons le champ n°$i .

- Si le champ demande un entier ou une date, la condition est
  `$condition = "$fields[$i] = $_POST[fields[$i]]"`
- Si le champ demande une chaîne de caractères, la condition est
  `$condition = "$fields[$i]`
  LIKE '%`$_POST[fields[$i]]`%'"

avec `$fields` étant une variable comportant le nom des différents champs de la table.

4.2 Page pays.php

Cette page web permet à l'utilisateur de savoir dans quels pays est disponible une plateforme donnée. Tout d'abord, un menu déroulant reprenant la liste des plateformes de streaming enregistrées dans la base de données est générée sur base de la requête suivante.
SELECT `nom_platf`
FROM `plateforme_streaming`
Ensuite, une fois la plateforme sélectionnée, une requête est envoyée pour obtenir la liste des pays où la plateforme est disponible.
SELECT (*)
FROM `pays`
WHERE `nom_platf=$plateforme`
Avec `$plateforme` étant le nom de la plateforme choisie.

4.3 Page episode.php

Cette page permet à l'utilisateur de créer un nouvel épisode et de l'insérer dans une série déjà existante. Comme indiqué dans la section 2, différents champs doivent être complétés par l'utilisateur. En ce qui concerne le champ `nom_serie` , un menu déroulant listant les différentes séries disponibles est généré sur base de la requête suivante.
SELECT `nom_serie`
FROM `serie`
Une fois les champs complétés, le formulaire est envoyé via la requête SQL suivante.
INSERT INTO `episodes`
VALUES (`$num_saison`, `$num_episode`, `$duree`, `$synopsis`, `$nom_serie`)
Dans le cas où la requête est accepté, ce nouveau tuple est intégré dans la table 'episodes'.

Une fois l'épisode créé, la page **acteur.php** est affichée à l'utilisateur pour ajouter des acteurs à l'épisode. Deux listes déroulantes affichent les noms et les prénoms des acteurs existants.
Pour ce faire, un INNER JOIN entre la table acteur et la table personne est réalisé pour extraire les noms et prénoms des acteurs sur base de leur identifiant.
Les requêtes SQL liées à chacune des listes s'écrivent
SELECT `nom`
FROM `acteur`
INNER JOIN `personne`
ON `personne.numero` = `acteur.numero`
et
SELECT `prenom`
FROM `acteur`
INNER JOIN `personne`
ON `personne.numero` = `acteur.numero`
Une fois la combinaison nom + prénom sélectionnée, le numéro de l'acteur est extrait en utilisant la requête
SELECT `numero`
FROM `personne`
WHERE `nom=$nom_acteur`
AND `prenom=$prenom_acteur`
Dans le cas où l'acteur existe, celui-ci est ajouté dans la table `joue_dans` via la requête SQL
INSERT INTO `joue_dans`
VALUES (`$numero_acteur`, `$num_saison`, `$num_episode`, `$nom_serie`)

4.4 Page utilisateur.php

Cette page affiche les abonnés ayant regardé tous les épisodes de la série Black Mirror (toutes plateformes confondues).
Tout d'abord, le nombre total de personnes enregistrées dans la base de données est extrait via la requête suivante
SELECT COUNT(`numero`)
AS `nombre_de_personne`
FROM `personne`
Ensuite, pour chacune des personnes, la requête
SELECT (*)
FROM `regarde`
WHERE `nom_serie`='Black Mirror' AND `numero`=`$num`
est effectuée pour ne conserver que les tuples corresponds aux épisodes de Black Mirror vu par la personne n°`$num` .
Une fois ce filtre effectué, une simple condition "if" vérifie si le nombre de tuples restant est égal à 5 (nombre total d'épisode de Black Mirror). Si la condition est remplie, cela signifie que la personne a bien visionné tous les épisodes de la série.
Dans ce cas, son nom et prénom sont obtenus via la requête ci-dessous.
SELECT (*)
FROM `persone`
WHERE `numero`=`$num`

4.5 Page serie.php

Cette dernière page liste les épisodes des séries triés par le nombre d'utilisateurs l'ayant regardé, par ordre décroissant. Pour ce faire, la requête SQL indiqué ci-joint est envoyée au serveur.
SELECT `regarde.n_saison`, `regarde.n_episode`, `regarde.nom_serie`,
COUNT(*) AS `nombre_de_spectateurs`
FROM `regarde`
INNER JOIN `episodes`
ON `episodes.n_saison` = `regarde.n_saison`
AND `episodes.n_episode` = `regarde.n_episode`
AND `episodes.nom_serie` = `regarde.nom_serie`
GROUP BY `regarde.nom_serie`, `regarde.n_saison`, `regarde.n_episode`
ORDER BY `nombre_de_spectateurs` DESC, `regarde.nom_serie`, `regarde.n_saison`, `regarde.n_episode`

5 Conclusion

Pour conclure, le site internet (disponible via le lien `http://www.student.montefiore.ulg.ac.be/~mheylen/`) a été développé dans l'optique d'intéragir avec une base de données MySQL.