

飛)カ) や下のふも木 日あさるさ きと 田のかかゑるさ - 本たもかきたも ふひつ木も あや下るさ しるさ しまとあかそるさ きと 行のゑし

Devoir à envoyer via système de transfert de fichier à frederic.mandon@ac-montpellier.fr.

Préciser le nom des ~~coupons~~ participants au projet dans l'en-tête du mail.

Le but de ce projet est de réaliser une interface homme-machine permettant de satisfaire des requêtes simples en SQL. On utilisera la base de données bdfilms construite lors du TD SQL (la version correctement faite, avec les contraintes de clé primaire etc. respectées, pas la version initiale !).

Le minimum à faire : des requêtes qui permettront de sélectionner deux (voire trois) attributs, avec au plus deux (voire trois) conditions, sans ordre. L'utilisateur doit donc pouvoir choisir un ou deux attributs à l'aide de listes déroulantes, éventuellement rajouter une ou deux conditions, à l'aide de listes déroulantes portant sur les attributs et de champs de saisie pour les valeurs portant sur ces conditions. Le programme construit la requête SQL correspondante et renvoie le résultat. Les requêtes les plus compliquées sont donc du type : « SELECT title, release_year, revenue FROM films WHERE title LIKE "%a" AND runtime > 150 ». Comme précisé en cours, ce n'est pas très logique d'avoir une requête sans « ORDER BY ». Vous pouvez l'implémenter par défaut pour que le résultat soit plus « réaliste ». Par exemple sur le premier attribut à afficher, en ordre croissant, pour ne pas avoir à rajouter de choix supplémentaire de la part de l'utilisateur. On peut aussi rajouter « DISTINCT » etc.

Pour aller plus loin version 1 : jointures, avec par exemple le nom des acteurs et du réalisateur d'un film.

Pour aller plus loin version 2 : on peut y rajouter la modification, l'insertion et/ou la suppression de films.

Dans ce cas, c'est pas mal d'avoir un mot de passe, c'est encore un approfondissement possible. Deux solutions pour cela : un mot de passe unique vérifié par un script en JavaScript, éventuellement encodé en base 64 ou autre pour ne pas être lisible directement dans le code. Attention c'est une mesure de « sécurité » totalement illusoire, dans la mesure où il suffit de décoder le mot de passe en base 64 pour le trouver. Vous pouvez sinon faire une table utilisateur avec le mot de passe enregistré dans la table. On supposera que cette table est protégée et inaccessible sauf via des droits administrateur. Là encore, ce que je vous propose n'est pas du tout dans les normes de sécurité même minimales !

Pour ceux qui veulent se renseigner davantage sur la sécurité des identifiants utilisateurs : aller sur <https://stackoverflow.com/questions/1054022/best-way-to-store-password-in-database>, lire la 1^{ère} réponse, tous les liens afférents, et passer quelques mois voire quelques années sur ce sujet super intéressant (aucune ironie dans les propos précédents, aussi bien en ce qui concerne la durée que l'intérêt).

Remarque : si vous mettez un mot de passe/un utilisateur, prière de me mettre un fichier ReadMe avec ces identifiants !

Il y a deux versions pour ce projet.

- La plus riche, assez moderne de plus, est celle en Python.
- Celle en PHP a l'avantage de vous faire découvrir un nouveau langage, qui est encore beaucoup utilisé. La syntaxe du PHP est pour le moins pénible.
- Je vous conseille plutôt la version Python (cf. arguments ci-après).
- Première version avec serveur, en Python. On utilise/crée :
 - Serveur Apache-MySQL-php (avec WAMP par exemple) pour l'accès à la base de données.
 - Bibliothèque flask pour le serveur ;
 - Bibliothèque mysql.connector pour l'accès aux bases de données ;
 - Page html avec formulaire pour la création de la requête ;
 - Page html + langage Jinja (extension de html, simple à utiliser) pour l'affichage des résultats.
 - Cette version a l'avantage de faire travailler sur des outils et conceptions « modernes » : architecture propre et souple, éventuellement monopage (même si ici je ne vous le demanderai pas). Le code est « relativement simple », et a l'avantage d'être en Python en partie. L'autre partie du code est en Jinja, qui est aussi un langage très facile à utiliser.

- Personnellement, je suis bluffé par la puissance, avec autant de simplicité, de flask + templates Jinja... Pour ceux qui souhaitent approfondir un peu, on peut même mettre des templates à l'intérieur de templates (par exemple un menu que l'on retrouve sur plusieurs pages)

- Deuxième version avec serveur, en php + html. On utilise/crée :
 - Serveur Apache (avec WAMP par exemple) pour l'accès à la base de données, et pour que le php puisse être interprété.
 - Page html avec formulaire pour la création de la requête ;
 - Page php pour l'affichage des résultats.
 - Cette version a l'avantage de faire travailler sur des outils très répandus, même si l'architecture induite par le php commence à avoir bien vieilli. Cela reste une version riche. Le code est « assez simple », même pour ceux qui n'ont jamais fait de php. Votre honoré professeur n'aime vraiment pas le php, mais il avoue son manque total d'objectivité sur ce point !

1. Exemple d'interface.

C'est presque un contre-exemple vu comment c'est moche. L'idée est là, à vous de faire mieux. Vous pourrez voir dans les exemples que le html est plus que minimaliste... vous avez les connaissances de 1^{ère} pour en faire quelque chose de correct ! Ces captures d'écran montrent des versions de travail, avec les requêtes affichées pour vérification. Les résultats sont affichés de manière incomplète volontairement. Les codes html/php/Python-flask sont donnés dans le dossier compressé du devoir.

<p>Recherche des attributs de l'année 1975 ou 1976</p> <p>Choix de la catégorie : <input type="text" value="toutes"/></p> <p>Pour 1975 tapez oui (sinon c'est 1976):</p> <p><input type="text" value="oui"/></p> <p><input type="button" value="VALIDEZ"/></p>	<p>Liste des attributs 1975 ...ou pas</p> <p>Traduction de l'attribut à afficher : *</p> <p>début de la requête : SELECT * FROM films</p> <p>requête finale : SELECT * FROM films WHERE release_year = 1975 ;</p> <p>SELECT * FROM films WHERE release_year = 1975 ;</p> <p>Rowcount : 6</p> <ul style="list-style-type: none"> • One Flew Over the Cuckoo's Nest • Jaws • Monty Python and the Holy Grai • Barry Lyndon • The Return of the Pink Panther • Death Race 2000
Page de sélection des attributs et conditions	Page d'affichage des résultats

Bien sûr, vous ferez quelque chose de bien plus joli, avec une section <head> propre, avec du css bien rangé dans son dossier (idem s'il y a des images et/ou du JavaScript). Et vous ferez passer tout ce code dans les validateurs html/css histoire qu'il soit à peu près correct.

Vous pouvez (re)prendre l'exemple de 1^{ère}, notamment pour la section <head> : http://www.maths-info-lycee.fr/programmes/exemple_site.zip.

2. Installation du serveur Apache-MySQL-PHP

Deux versions : UWAMP pour un logiciel d'apprentissage simple, mais manquant de puissance. WAMP pour un vrai serveur.

Les serveurs utilisent des ports : ces ports doivent être différents des ports pour les usages standards (80 pour http, 443 pour https, etc.). Le port est un numéro logique associé à toute application communiquant via Internet.

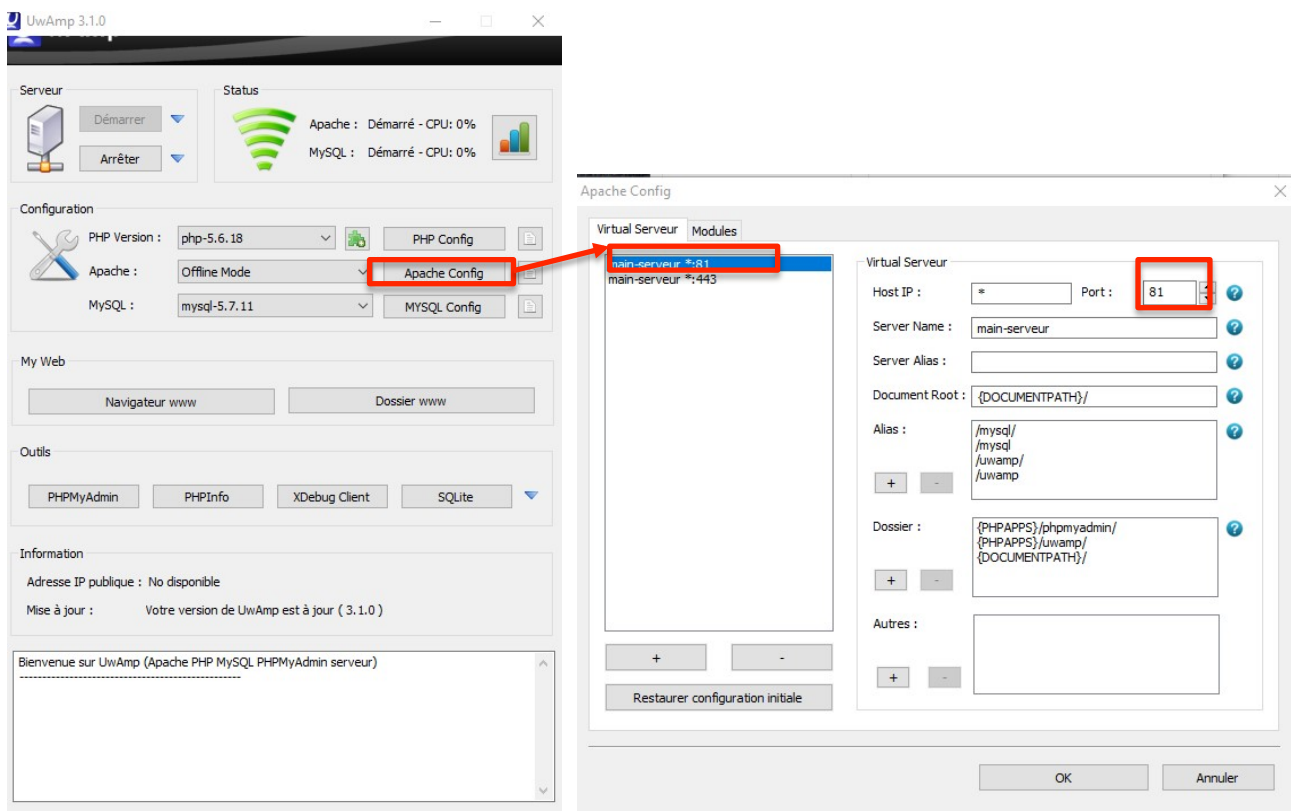
Utilisez svp les ports 3306 pour MySQL et 8888 pour Apache (cf. juste ci-dessous pour changer les ports). 5000 pour flask.

a. UWAMP

UWAMP est un logiciel limité d'apprentissage, qui suffit pour ce projet. Si vous voulez un vrai logiciel, passez au b. (WAMP).

Il faut **changer le port** utilisé par Apache, cf. captures d'écrans ci-dessous. En effet par défaut il est réglé sur 80, qui est le port utilisé par le protocole http. En théorie ça doit rester sur 80, en pratique chez moi ça plante. Prenez 8888.

Le port par défaut pour MySQL est 3306. Modifiez si nécessaire pour avoir cette valeur.



Voir en annexe pour pouvoir utiliser une base de données de taille assez importante dans UWAMP (comme celle donnée des films en version complète).

b. Installation de WAMP (sous Windows)

C'est plus compliqué que UWAMP. C'est aussi plus généraliste, ce n'est pas juste un logiciel d'apprentissage.

Il y a d'autres types de serveurs Apache-MySQL-PHP, vous pouvez utiliser ce que vous préférez.

Installation :

- suivre avec attention les instructions lors de l'installation ; pour une fois il faut les lire !
- Notamment :
 - Présence des différentes versions de Visual C++. Lien pour l'outil disponible dans le premier écran d'installation (logiciel check_vc_redist.exe). Le lancer. S'il manque des logiciels, les installer **avant** wampserver. Les liens sont disponibles là aussi dans le premier écran d'installation, la totalité étant là : http://wampserver.aviatechno.net/files/vcpackages/all_vc_redist_x86_x64.zip.
 - Le répertoire d'installation doit être à la racine du disque, et le nom ne doit contenir ni espaces ni caractères diacritiques.

Usage de WAMP pour le serveur de bases de données :

- Lancer wampserver. Une icône « wampmanager » rouge puis orange puis verte apparaît dans le menu accessible par la petite flèche. Si elle ne passe pas au vert, un clic gauche dessus permet de « démarrer les services ».

Remarque : sous OSX, installer MAMP, sous Linux XAMP (les procédures sont beaucoup plus simples... Windows reste objectivement le plus mauvais système. D'ailleurs Windows était autrefois développé sous OSX)

c. Créer/Accéder à la base de données

L'outil qui permet d'accéder aux bases de données est **phpMyAdmin**, ceci quel que soit le type de serveur utilisé.

Il est possible que vous ayez des modifications à faire dans les fichiers de requêtes de créations de la base de données. En effet, j'ai la fâcheuse impression que chaque version que je donne de ces requêtes n'est plus valable l'année suivante... il y a à chaque fois des petites variantes.

Dans WAMP, sélectionner phpMyAdmin, qui ouvre une page dans le navigateur. Se connecter sous le nom `root`, sans mot de passe (ou bien mot de passe `root`), serveur MySQL.

Deux options : importer la base de données ou la créer étape par étape.

Import direct :

Importer la base de données `bdfilms.db.sql` , à télécharger sur <http://www.maths-info-lycee.fr/programmes/bdfilms.db.sql> (c'est « fichier de requêtes SQL de construction de la base de données `bdfilms` » sur l'accueil ou la page de terminale)

Vous pouvez ensuite vérifier dans phpMyAdmin la présence de la base de données, ainsi que les noms des attributs etc.

Etape par étape : voir l'annexe 2 pour une construction par étapes de la base de données.

Vous pouvez aussi utiliser la base de données en version allégée, si vous craquez sur l'usage de WAMP (fichier unique `bdfilmslightdb.sql` pour l'import direct, ou bien requêtes de création/remplissage par étapes dans les différents fichier `*filmslight*.sql`).

Il est probable que vous deviez rajouter la requête `USE bdfilms;` ou `USE bdfilmslight;` avant les requêtes de recherche usuelles.

3. Version flask

Il faudra ici aussi préciser un port d'utilisation pour flask (différent du port SQL, http, etc.). Utilisez svp le port 5000.

Préliminaires :

- Importer flask suivant votre environnement :
 - `pip install flask`, dans une invite de commande Windows. Il est possible que vous deviez auparavant rajouter une variable d'environnement Python, si `pip` n'est pas reconnu comme une commande. Les adresses suivantes donnent la méthode (visez les réponses utilisant les interfaces graphiques) :
<https://stackoverflow.com/questions/7054424/python-not-recognized-as-a-command>
(1^{ère} réponse)
<https://stackoverflow.com/questions/47539201/python-is-not-recognized-windows-10>
(3^{ème} réponse)
Pour trouver les variables d'environnement, tapez simplement « variables d'environnement » dans la barre de recherche, puis dans la fenêtre « propriétés systèmes », le bouton « variables d'environnement » et « nouvelle ». Suivez ensuite les instructions sur une des adresses précédentes. Il y a d'autres pages sur ce sujet, vous pouvez simplement googler « variable d'environnement Python » ou « Python not recognized », « Python non reconnu » etc. Ce n'est pas compliqué, contrairement à ce que vous pourriez croire.
 - `conda install flask` , dans une invite de commande Anaconda
 - Doc (si nécessaire uniquement, notamment en cas de problème d'installation. Il peut être nécessaire de créer un environnement de travail virtuel) :
<https://flask.palletsprojects.com/en/1.1.x/installation/>
- Créez un dossier de travail dans lequel vous mettez :
 - Le fichier `exemples_flask.py`
 - Un sous-dossier `templates`, avec `index.html` et `resultats.html`. Si vous avez des images, du css, du javascript etc., créez un dossier `static` et des sous dossiers `css`, `images` etc. pour que l'architecture du site soit propre
 - Dans ce sous-dossier, le fichier `index.html`. Ouvrez ce fichier dans un éditeur, vous verrez qu'il y a du code de langage de programmation avec une boucle `for`, ce qui n'existe pas en html.
 - Ce code est du jinja : un vrai langage de programmation, intégré au html. Doc :
<https://jinja.palletsprojects.com/en/2.11.x/>

- Et surtout : <https://jinja.palletsprojects.com/en/2.11.x/templates/> qui est la doc pour la construction d'un document html « template ».
- Une fois le serveur flask lancé, en exécutant le programme flask_intro/exemple_flask2.py, allez dans le navigateur et saisissez comme adresse localhost:5000/index ou localhost:5000. La page index.html du dossier template s'ouvre. On voit bien que l'on utilise le port 5000 pour flask.
- Le serveur flask s'arrête avec ctrl-C (dans le shell Python)
- Importer un connecteur SQL.
 - Comme ci-dessus, suivant votre environnement :

```
pip install mysql-connector-python
```

```
conda install anaconda::mysql-connector-python
```

 (ou `conda install mysql-connector-python` qui est la version ancienne de cette commande)
Remarque : j'ai du réinstaller tout ça récemment, et ça a mis au moins 30 minutes... sans raison valable.
 - Utiliser ensuite l'exemple donné dans le fichier `connecteur_sql.py` dans un IDE Python (même si ça parle de SQL, ça reste du Python). Utiliser le port SQL choisi lors de la configuration du serveur AMP (normalement 3306). Le port MySQL est différent du port flask : deux outils utilisent deux ports différents.
- Vous pouvez ensuite voir le fonctionnement SQL + flask avec `flask_sql/exemple_flask2.py`.
- On insiste : les pages html + Jinja doivent être dans le dossier `templates`, situé dans le même dossier que le programme Python/flask.

4. Version php

Usage de WAMP pour le php

- Créer un projet, en ajoutant un dossier dans le dossier `www` de wampserver (sans espaces ni tiret bas). Y mettre les fichiers html, css, php etc.
- Un clic gauche sur l'icône wampmanager permet d'accéder au menu « virtualhost » puis « gestion virtualhost ». Ceci lance une page php, suivre les indications pour remplir le formulaire. Une autre possibilité est de taper localhost dans le navigateur.
- Comme indiqué après complétion du formulaire, redémarrer le serveur DNS ; accès par clic droit sur l'icône wampmanager.
- Pour que les pages PHP fonctionnent correctement, il est indispensable de passer par le menu « virtualhost » de wampmanager. Un clic sur l'hôte virtuel désiré ouvre une page html donnant accès aux différentes pages du projet, et permet de les ouvrir ainsi dans le navigateur.
- Ouvrir les pages directement depuis le système de fichiers ne lance pas l'interpréteur PHP ; ça n'est pas fonctionnel.

Fichiers d'exemples pour cette version: `rechercheFilms.html` et `listeAttributs1975.php`

Vous trouverez aussi quelques scripts élémentaires en PHP dans le dossier compressé pour le devoir.

ANNEXE 1 : augmenter la taille maximale des fichiers importés dans phpmyadmin

Ouvrir les fichiers de configuration PHP de démarrage suivants :

`UwAmp/bin/php/php_[*]/php_uwamp.ini` avec le bloc-notes ou avec Notepad++. Attention il peut y avoir plusieurs fichiers, d'où le `php_[*]` : ceci désigne la version de php. Suivant la version utilisée (de préférence la dernière), modifier un ou les deux fichiers. L'emplacement exact du(es) fichier(s) dépend de où vous avez mis le dossier UwAmp.

Rechercher (ctrl-F) les éléments suivants et les changer. Il faut parfois faire une recherche vers le haut, et d'autres fois vers le bas.

Trouver :	Changer en :
<code>post_max_size = 8M</code>	<code>post_max_size = 750M</code>
<code>upload_max_filesize = 2M</code>	<code>upload_max_filesize = 750M</code>
<code>max_execution_time = 30</code>	<code>max_execution_time = 500</code>

max_input_time = 60	max_input_time = 500
memory_limit = 8M	memory_limit = 1000M

Posts d'origine :

- <http://forum.wampserver.com/read.php?2,45000>
- <https://www.uwamp.com/fr/?page=doc>.

Une fois ces modifications faites, on peut importer une base de données plus lourde, éventuellement en plusieurs fois. En effet, l'importation s'arrête après quelques minutes, suivant une valeur de temps qui ne correspond pas aux durées indiquées ci-dessus. Il suffit de relancer l'import avec le même fichier. Après test, il manque quand même 70000 enregistrements dans la table des acteurs... cela n'empêche pas de faire le projet.

ANNEXE 2 : construire la base de données par étapes

Deux versions : bases de données standard ou allégée (Light)

Quatre fichiers de requêtes permettent de :

- créer la base de données puis les tables (`creationBdEtTables.sql`);
- remplir les tables une par une. Respecter l'ordre : les films en premier pour respecter les contraintes de référence (`insertionBdFilms.sql`, `insertionBdDistribution.sql`, `insertionBdRealisateurs.sql`)

Ouvrez le fichier de création ainsi que celui d'insertion des films : vous pouvez constater que ce sont des requêtes SQL standard.

Il est ensuite plus simple de les importer dans WAMP.

ANNEXE 3 : pour information, les requêtes (avec les détails pour arriver à la requête finale) qui ont permis d'alléger la base de données à partir de la base originale.

```
USE bdfilmslight;
```

Trouver les acteurs ayant joué dans plus de 20 films

```
SELECT COUNT(*), distribution.actor FROM films
INNER JOIN distribution ON films.movie_id = distribution.movie_id
GROUP BY distribution.actor
HAVING COUNT(*) > 20;
```

Trouver le nombre d'enregistrements dans la table distribution, des acteurs ayant joué dans plus de 20 films

```
SELECT SUM(p) FROM (
SELECT COUNT(*) as p FROM films
INNER JOIN distribution ON films.movie_id = distribution.movie_id
GROUP BY distribution.actor
HAVING COUNT(*) > 20) as dd;
```

La suppression des données (acteurs ayant joué dans moins de 20 films) : on crée une table temporaire appelée tt.

```
CREATE TEMPORARY TABLE tt
SELECT distribution.actor FROM distribution INNER JOIN films
ON distribution.movie_id = films.movie_id
GROUP BY distribution.actor
HAVING COUNT(*) > 20;
```

```
DELETE FROM distribution WHERE
distribution.actor NOT IN (
SELECT * FROM tt);
```