1 Environnement de travail

Système d'exploitation : Windows

Éditeur de texte : **Sublime text** [1], gedit, notepad++...

Langages : **PHP, SQL, HTML, CSS**

Logiciels nécessaires :

1. **Wamp** (Plateforme de développement web)
2. **Apache** *inclut dans Wamp*
3. **phpMyAdmin** *inclut dans Wamp*
4. **MySql** *inclut dans Wamp*

Notions clés nécessaires à la réalisation du workshop :

- ⋆ Les moteurs de stockage de MySQL
- ⋆ Les transactions
- ⋆ Les procédures stockées
- ⋆ Les requêtes préparées
- ⋆ PDO

Penser à faire des recherches complémentaires avant le workshop.

1. `www.sublimetext.com/docs/3/linux_repositories.htm`

2 Création d'une table produits

Objectifs :
✓ Manipulation de PhpMyAdmin
✓ Création d'une base de données à partir d'un script SQL

1. Lancer **Wamp**
2. Ouvrir **phpMyAdmin**
3. Créer une base de données : WSProsit5
4. La sélectionner, et dans l'onglet SQL copier le script suivant pour créer la table **produits** :

```sql
DROP TABLE IF EXISTS produits; --Supprime la table si elle existe déjà

CREATE TABLE produits -- Crée la table avec les champs suivants
(
    id INT NOT NULL AUTO_INCREMENT,
    nom VARCHAR (30) NOT NULL,
    prix INT UNSIGNED NOT NULL,
    description VARCHAR (200),
    solde BOOLEAN DEFAULT 0,
    urlImage VARCHAR (200) DEFAULT 'Produits/p1.jpg' ,
    PRIMARY KEY (id)
)
ENGINE = InnoDB; -- Change le moteur de stockage

-- Les lignes suivantes permettent d'insérer des enregistrements dans la table

INSERT INTO produits (nom,prix,description,solde,urlImage)
VALUES
('smartX15', '115',"Chassures basses et légère, temps ensoleillé",1,'Produits/p1.jpg'),
('greenLantXY', '159',"Chassures hautes et robustes, saison estivale",1,'Produits/p2.jpg'),
('crazyFrog2056', '174',"Chassures étanches, à l'épreuve des intempéries",0,'Produits/p3.jpg'),
('baroudClimb', '249',"Chassures hautes, tout terrains, toutes saisons",0,'Produits/p4.jpg'),
('rustik2000', '206',"Chassures confortables, terrains peu accidentés",0,'Produits/p5.jpg'),
('flexLightTT', '249',"Chassures légères, souples et tout terrains",0,'Produits/p6.jpg');
```

CreationTableProduits.sql

Ne pas hésiter à réutiliser ce script en cas de manipulation fâcheuse de la table produits.

3 Mise en place des requêtes d'administration

Objectifs :
- ✓ Manipulation des requêtes de base en SQL
- ✓ Mise en place de transactions à partir d'un script
- ✓ Mise en place de procédures stockées à partir d'un script

1. En vérifiant dans la base de données, grâce à l'onglet SQL, mettre en place les **requêtes** permettant de :

   a. Lister l'ensemble des produits disponibles

   b. Lister l'ensemble des produits en solde

   c. Ajouter un nouveau produit

   d. Supprimer un produit

   e. Modifier les caractéristiques d'un produit existant (par exemple mettre un produit en solde)

   f. Modifier l'ensemble des prix des produits en soldes en appliquant une remise de 30%

   Utiliser les commandes pré-enregistrées dans phpMyAdmin.

   Certaines de ces requêtes ont pour effet de modifier la table ***produits***. Il est important de prendre l'habitude de contrôler la bonne exécution de ces instructions par le moyen des **transactions**, d'autant plus lorsque plusieurs requêtes sont exécutées successivement.

   Exemple pour une requête simple qui augmente chaque prix de 5 euros :

```sql
-- on désactive l'autocommit du moteur InnoDB :
SET autocommit = 0;

-- on démarre une nouvelle transaction :
START TRANSACTION;

-- on effectue une requête :
UPDATE produits SET prix = prix + 5;

-- on valide la transaction
COMMIT;
```

TransactionUpdatePrix.sql

2. Déclarer chaque requêtes en tant que **procédures stockées** (avec les **transactions** lorsqu'elles sont nécessaires) **en passant par l'onglet SQL de PhpMyAdmin**.

   Exemple de procédures permettant d'afficher tous les produits de la table :

```sql
DROP  PROCEDURE IF EXISTS listeProduits;

DELIMITER | -- Détermine un nouveau délimiteur pour la requête globale

CREATE PROCEDURE listeProduits() -- Procédure sans paramètre
BEGIN
    SELECT *
    FROM produits;
END |
```

ProcedureStockeeListeProduits.sql

3. Vérifier le comportement de la procédure grâce à l'instruction **CALL** :

```sql
CALL ListeProduits;
```

+ Options

id	nom	prix	description	solde	urlImage
1	smartX15	60	Chassures basses et légère, temps ensoleillé	1	Produits/p1.jpg
2	greenLantXY	81	Chassures hautes et robustes, saison estivale	1	Produits/p2.jpg
3	crazyFrog2056	174	Chassures étanches, à l'épreuve des intempéries	0	Produits/p3.jpg
4	baroudClimb	249	Chassures hautes, tout terrains, toutes saisons	0	Produits/p4.jpg
5	rustik2000	206	Chassures confortables, terrains peu accidentés	0	Produits/p5.jpg
6	flexLightTT	249	Chassures légères, souples et tout terrains	0	Produits/p6.jpg
8	NewProd	150	Nouvelles chaussures	0	Produits/p7.jpg

FIGURE 1 – Résultat à l'exécution de la procédure stockée dans phpMyAdmin

4 Une table utilisateurs

Objectif :
✓ Application des connaissances de création et remplissage de table

1. A partir du script de création de la table ***produits***, proposer un script de création d'une table ***utilisateurs*** disposant des attributs suivants :
   - id int qui s'auto-incrémente + clé primaire
   - pseudo varchar(15)
   - motDePasse varchar(10)
   - statutAdmin bool

2. Ajouter la création de 5 utilisateurs dont un administrateur.

```sql
INSERT INTO utilisateurs (pseudo,motDePasse,statutAdmin)
VALUES
('Katare', 'toppot',0),
('Chap', 'elgnuj',0),
('Melon', 'middim',0),
('Wakz', 'adda',1),
('Caëlan', 'troppus',0);
```

CreationUtilisateurs.sql

5 De nouveaux utilisateurs depuis le site web

Objectifs :
- ✓ Lier le site web avec la base de données
- ✓ Créer et manipuler un objet PDO
- ✓ Mettre en place une requête préparée d'écriture dans la base de données
- ✓ Mettre en place une requête préparée de lecture de la base de données

L'objectif de cette partie est de proposer un script, *scriptInscription.php*, permettant de mettre à jour la base de données après l'inscription d'un nouvel utilisateur via le formulaire du site web.

1. **Vérifier que PDO est activé** Wamp > PHP > Extensions PHP > ✓php_pdo_mysql
2. Enregistrer l'ensemble des fichiers de l'archive *Templace Code Source* fournie dans WAMP (C :\wamp64\www\Site) et vérifier que le site est bien accessible en local à l'adresse *http ://localhost/Site*
3. Ouvrir le fichier **scriptInscription.php** fourni dans le code source du site.

Les requêtes préparées sont à privilégier lorsque que l'on souhaite exécuter des requêtes utilisant des paramètres saisis directement par l'utilisateur. On crée alors un modèle de requête en lui indiquant les différents champs à remplacer grâce au : et on associe ensuite ces champs avec des variables utilisateur en spécifiant bien le type "chaîne de caractère".

4. Récupérer les commandes de création d'un **objet PDO** de connexion à la base de données et de construction d'une **requête préparée** permettant d'ajouter un utilisateur à la base de données à partir de la page web d'inscription.

```php
<?php
// /!\ Adapter dbname et mot de passe si besoin
$bdd = new PDO('mysql:host=localhost;dbname=wsprosit5;charset=utf8', 'root', '');

// Récupération des données utilisateurs
$pseudo = $_POST['pseudo'];
$motDePasse = $_POST['motDePasse'];

// Requête préparée pour empêcher les injections SQL
$requete = $bdd->prepare("INSERT INTO utilisateurs (pseudo, motDePasse) VALUES( :pseudo,:motDePasse)");

// Liaison des variables de la requête préparée aux variables php
$requete->bindValue(':pseudo', $pseudo, PDO::PARAM_STR);
$requete->bindValue(':motDePasse', $motDePasse, PDO::PARAM_STR);

// Exécution de la requête préparée avec les données liées
$requete->execute();
?>
```

scriptInscription.php

5. Accéder à la page **Inscription** du site via l'onglet du menu. Renseigner un pseudo et un mot de passe et cliquer sur "S'inscrire". Vérifier dans la base de données que la table utilisateurs s'est bien mise à jour.
6. Ajouter une requête qui permet de vérifier que le pseudo choisit n'est pas déjà utilisé par un autre utilisateur en utilisant la méthode **fetch()** de la classe PDOStatement.
7. Sur le même modèle, compléter le fichier *scriptConnexion.php*, vérifiant les informations *Pseudo* et *MotDePasse* dans la base de données lorsqu'un utilisateur essaye de se connecter depuis la page **Connexion** du site web.

6 Administration web de la base de données

Objectif :
✓ Application des connaissances
✓ Mettre en place une interface d'administration

1. Mettre en place une interface d'administration entre la page admin.php fournie (accessible depuis le site http ://localhost/Site/admin) et la table ***produits*** pour pouvoir utiliser des procédures stockées directement depuis le site web **tout en contrôlant le statut d'administrateur de l'utilisateur connecté**.

2. Ajouter à cette étape les réalisations demandées en prosit.