

Le langage PHP et les bases de données

Dans ce chapitre, vous allez poursuivre la découverte du langage PHP. Cette fois-ci PHP va servir à interagir avec une base de données. Le langage PHP est très adapté pour récolter les informations contenues dans les bases de données. L'usage des formulaires va servir à alimenter, modifier, mettre à jour une base de données. Il faudra donc manipuler le langage SQL, propre aux bases de données, que vous avez appris en SI3 et l'encapsuler dans des instructions PHP chargées de la connexion au serveur de bases de données.

Sommaire

A.	L'accès aux bases de données	2
A.1.	Introduction	2
A.1.1.	Interactions avec une base de données.....	2
A.1.2.	Mise en place de la base de données.	2
A.2.	Accéder au contenu d'une base de données.....	4
A.2.1.	Présentation de la classe PDO.....	4
A.2.2.	Création d'un objet PDO	5
A.2.3.	Requête de type SELECT.....	6
A.2.4.	Requête préparée SELECT déclenchée par un formulaire	10
A.3.	Manipulation et insertion de données.....	19
A.3.1.	Insertion des données	19
A.3.2.	Suppression des données.....	22
A.3.1.	Modification des données	26

A. L'accès aux bases de données

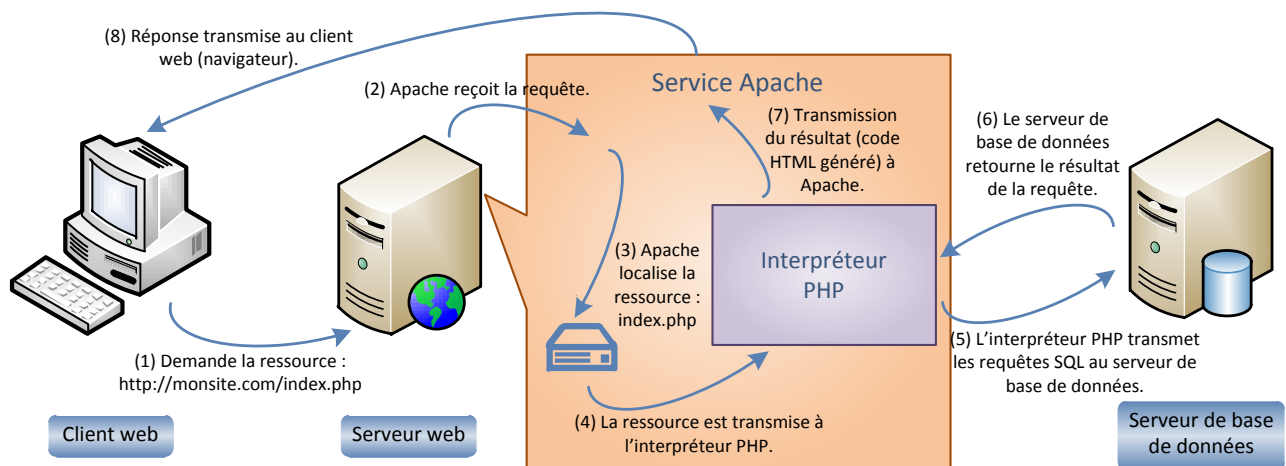
A.1. Introduction

A.1.1. Interactions avec une base de données

L'avantage du langage PHP est qu'il dispose d'imposants jeux d'instructions en liaison avec les bases de données. C'est en partie ce qui lui a permis de se développer et d'étendre le territoire de ses compétences.

PHP dispose de jeux d'instructions qui lui permettent d'interagir avec les bases de données en adressant à celles-ci des requêtes en langage SQL et en récupérant le résultat.

On représente ainsi les interactions :



Le serveur de base de données peut être : PostgreSQL, MySQL, Oracle, SQLServer, etc.

Nous utiliserons MySQL dans nos exemples.

A.1.2. Mise en place de la base de données.

La base de données est souvent installée localement dans les environnements de développement et surtout lorsque l'on utilise les logiciels intégrés : EasyPHP, WAMP, LAMP ou MAMP.

En production nous trouverions certainement la base de données implantée sur un serveur à part permettant d'offrir une charge dédiée au logiciel ainsi qu'un système de redondance et de sauvegarde sur mesure.

Nous le verrons, le langage PHP prévoit la localisation de la base de données dans ses paramètres ce qui permet donc toutes les solutions : base de données locale ou externe.

Exemple : dans notre cas, voici la structure de notre base de données *thefirm* qui contient des informations sur des salariés et le service qui les emploie :

Création de la table *salarie* :

Base de données : thefirm

thefirm (0)
Aucune table n'a été trouvée dans cette base.

Colonne	Type	Taille/Valeurs ¹	Défaut ²	Interclassement	Attributs	Null	Index	A_I
idSalarie	INT		Aucun			<input type="checkbox"/>	PRIMARY	<input checked="" type="checkbox"/>
nomSalarie	VARCHAR	40	Aucun			<input type="checkbox"/>		<input type="checkbox"/>
prenomSalarie	VARCHAR	50	Aucun			<input type="checkbox"/>		<input type="checkbox"/>
sexeSalarie	ENUM	'H', 'F'	Aucun			<input type="checkbox"/>		<input type="checkbox"/>
catSalarie	ENUM	re', 'employé'	Aucun			<input type="checkbox"/>		<input type="checkbox"/>
serviceSalarie	VARCHAR	3	Aucun			<input type="checkbox"/>	INDEX	<input type="checkbox"/>

Commentaires sur la table:

Moteur de stockage: InnoDB

Interclassement:

Création de la table *service* :

Base de données : thefirm (1)

thefirm (1)

salarie

Colonne	Type	Taille/Valeurs ¹	Défaut ²	Interclassement	Attributs	Null	Index	A_I
idService	VARCHAR	3	Aucun			<input type="checkbox"/>	PRIMARY	<input type="checkbox"/>
libService	VARCHAR	40	Aucun			<input type="checkbox"/>		<input type="checkbox"/>

Commentaires sur la table:

Moteur de stockage: InnoDB

Interclassement:

Par défaut la base de données MySQL dispose d'un compte administrateur nommé root sans mot de passe. Il faudra peut-être penser à en mettre un crypté pour s'intéresser à un moment ou un autre aux problèmes de cryptage des informations entre le langage PHP et la base de données.

Dans tous les cas, il est conseillé d'utiliser un compte avec moins de privilèges que celui de root. Par exemple un compte ne disposant que de droits restreints uniquement sur la base de données interrogée.

A.2. Accéder au contenu d'une base de données

A.2.1. Présentation de la classe PDO

PDO (PHP Data Objects) est une interface pour accéder à une base de données apparue avec PHP 5.0 et fournie avec PHP 5.1. PDO permet de faire abstraction du moteur de SGBD utilisé, c'est à dire qu'il n'y a plus de jeux d'instructions spécifiques comme il y avait pour MySQL, PostgreSQL, Oracle, etc. La gestion des erreurs a été facilitée et améliorée de façon à mieux gérer les divers problèmes qui peuvent survenir dans les interactions avec la base de données.

Cette extension de PHP est activée par défaut depuis la version 5.1, dans *php.ini* :

```
extension=php_pdo.dll
```

Le driver qui permet à PDO de fonctionner avec la base de données que vous utilisez est activé par défaut pour MySQL (avec EasyPHP, WAMP, LAMP ou MAMP) dans *php.ini* :

```
extension=php_pdo_mysql.dll
```

Si vous utilisez PostgreSQL il faudra activer le driver dédié de la façon suivante dans *php.ini* :

```
extension=php_pdo_pgsql.dll
```

Voici (source *php.net*) le synopsis de la classe :

```
PDO {
    // Création d'une instance PDO qui représente une connexion à la base
    __construct ( string $dsn [, string $username [, string $password [, array $driver_options
]] ] )
    // Démarrage d'une transaction
    bool beginTransaction ( void )
    // Validation d'une transaction
    bool commit ( void )
    // Retour du SQLSTATE associé avec la dernière opération sur la base de données
    mixed errorCode ( void )
    // Retour des info. associées à l'erreur lors de la dernière opération sur la base de don.
    array errorInfo ( void )
    // Exécution d'une requête SQL et retour du nombre de lignes affectées
    int exec ( string $statement )
    // Récupération d'un attribut d'une connexion à une base de données
    mixed getAttribute ( int $attribute )
    // Retour de la liste des pilotes PDO disponibles
    static array getAvailableDrivers ( void )
    // Vérification si nous sommes dans une transaction
    bool inTransaction ( void )
    // Retour de l'identifiant de la dernière ligne insérée ou la valeur d'une séquence
    string lastInsertId ( [ string $name = NULL ] )
    // Préparation d'une requête à l'exécution et retourne un objet
    PDOStatement prepare ( string $statement [, array $driver_options = array() ] )
    // Exécution d'une requête SQL, retour d'un jeu de résultats en tant qu'objet PDOStatement
    PDOStatement query ( string $statement )
    // Protection d'une chaîne pour l'utiliser dans une requête SQL PDO
    string quote ( string $string [, int $parameter_type = PDO::PARAM_STR ] )
    // Annulation d'une transaction
    bool rollBack ( void )
    // Configuration d'un attribut PDO
    bool setAttribute ( int $attribute , mixed $value )
}
```

A.2.2. Création d'un objet PDO

Cette partie donne le code minimal pour se connecter à une base de données tout en gérant les erreurs (exceptions) de liaisons.

Etant donné que les informations de connexion à une base de données sont extrêmement sensibles il convient de les sécuriser au maximum. Pour des raisons de simplification de ce cours, nous n'aborderons pas le cryptage des mots de passe qui peut constituer un supplément non négligeable de sécurité.

Pour nous connecter à la base de données il faudra quatre informations :

- la localisation de la base de données (ici nous travaillons en local donc ce sera *localhost*) ;
- le nom de la base de données (ici il s'agit de *thefirm*) ;
- le nom de l'utilisateur utilisé pour se connecter (dans votre cas vous utiliserez certainement *root*, mais il sera préférable de créer un autre utilisateur) ;
- le mot de passe de l'utilisateur (par défaut, il n'y en a pas).

Il est évident que ces quatre informations et en premier lieu le nom et le mot de passe de l'utilisateur doivent être protégés.

Déclarations de variables dans un fichier externe (ici le fichier *id.php* situé dans un dossier *param*) :

```
<?php
$base='thefirm';
$hote='localhost';
$utilisateur='louzer';
$mdp='mokipasse';
?>
```

Le code des pages incluant les données de connexion :

```
<body>
<?php include_once 'param/id.php'; ?>
<?php include 'includes/entete.inc.php'; ?>
<?php include 'includes/menu.inc.php'; ?>
<div class="centre">
  <?php
    try
    {
      $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
      $bdd = new PDO('mysql:host='.$hote.';dbname='.$base, $utilisateur, $mdp);

      // ici les traitements

    }
    catch (Exception $erreur)
    {
      die('Erreur : ' . $erreur->getMessage());
    }
  ?>
</div>
</body>
```

Remarques :

- a) Afin d'éviter une erreur de l'interpréteur PHP (informations erronées, indisponibilité du serveur, absence de la base de données, etc.) qui conduirait à afficher la ligne erronée de votre script et donc à en révéler le contenu, nous avons prévu de gérer les exceptions et l'affichage des erreurs par nos soins.
C'est le rôle de l'instruction `try{}` qui permet de tester le bon fonctionnement des instructions entre `{}` et en cas d'erreur (captées grâce à la ligne `$pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;`) de récupérer le type d'erreur avec `catch()` et de l'afficher (rôle de `{ die() }`).
- b) C'est l'objet `$bdd` qui servira à l'ensemble des transactions futures avec la base de données.
- c) A la place de l'instruction `include_once` (qui génère un avertissement en cas d'erreur) on pourrait utiliser `require_once` (qui génère une erreur fatale en cas d'erreur et empêche le script de se poursuivre). Ces deux instructions se chargent de vérifier (`_once`) que le script appelé ne le soit qu'une fois en cas de déclarations multiples.
- d) Le dossier `param` contenant le fichier sensible pourra être protégé grâce à un fichier `.htaccess` complété d'un fichier `.htpassword`.

A.2.3. Requête de type SELECT

Nous ne reviendrons pas sur les ordres SQL que vous avez découverts en SI3. Nous allons les utiliser pour nous adresser à notre base de données et nous occuper du traitement et de la présentation des résultats via PHP.

A.2.3.1. Requête SELECT simple

Nous allons passer la requête : `SELECT * FROM salaire;`

Le résultat via l'interface **phpMyAdmin** est :

idSalarie	nomSalarie	prenomSalarie	sexeSalarie	catSalarie	serviceSalarie
1	Lambert	Marc	H	cadre	S01
2	Piot	Stéphanie	F	employé	S02
3	Hermeso	Gérard	H	employé	S01
4	Granola	Marie-Laure	F	cadre	S04
5	Trimoulon	Antoine	H	employé	S02
6	Garnic	Loïc	H	employé	S01
7	Javan	Christine	F	cadre	S02
8	Mirame	Kévin	H	employé	S03

Pour obtenir un résultat similaire à phpMyAdmin nous devons saisir le code suivant :

```
<body>
<?php include_once 'param/id.php'; ?>
<?php include 'includes/entete.inc.php'; ?>
<?php include 'includes/menu.inc.php'; ?>
<div class="centre">
    <?php
    try
    {
        $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
        $bdd = new PDO('mysql:host='.$hote.';dbname='.$base, $utilisateur, $mdp);
        // Spécification de l'encodage (en cas de problème d'affichage :
        $bdd->exec('SET NAMES utf8');
        $reponse = $bdd->query('SELECT * FROM salarie'); // Envoi de la requête
        $nb = $reponse->rowCount(); // Compte du nombre de lignes retournées
        echo '<table>'; // Déclaration d'un tableau et de sa ligne d'en-tête
        echo '<tr><th>ID</th><th>NOM</th><th>PRENOM</th><th>SEXE</th><th>CATEGORIE</th>';
        echo '<th>SERVICE</th></tr>';
        while ( $donnees = $reponse->fetch() ) // Découpage ligne à ligne de $reponse
        {
            echo '<tr>'; // Une ligne appelle les données de $donnees['']
            echo '<td class="c1">'. $donnees['idSalarie']. '</td>';
            echo '<td class="c1">'. $donnees['nomSalarie']. '</td>';
            echo '<td class="c1">'. $donnees['prenomSalarie']. '</td>';
            echo '<td class="c1">'. $donnees['sexeSalarie']. '</td>';
            echo '<td class="c1">'. $donnees['catSalarie']. '</td>';
            echo '<td class="c1">'. $donnees['serviceSalarie']. '</td>';
            echo '</tr>';
        }
        echo '</table>'; // Fin du tableau
        echo '<p>Il y a '.$nb.' salariés.</p>'; // Affichage du compte des lignes
        // On libère la connexion du serveur pour d'autres requêtes :
        $reponse->closeCursor();
    }
    catch (Exception $erreur)
    {
        die('Erreur : ' . $erreur->getMessage());
    }
    ?>
</div>
</body>
```

Remarque : Une fois la connexion établie à la base de données, effectuée grâce à l'instanciation `$bdd = new PDO()`, l'interface PDO est disponible pour passer notre requête et afficher le résultat.

Nous avons procédé en trois étapes :

- L'exécution de la requête :
`$reponse = $bdd->query('SELECT * FROM salarie');`

La variable `$reponse` est un jeu de résultats (de classe `PDOStatement`) qui stocke la réponse. Nous avons demandé toutes les colonnes et toutes les lignes de la table *salarie*.

- Le parcours ligne à ligne du jeu de résultats `$reponse` :
`while ($donnees = $reponse->fetch()) { ... }`

La boucle parcourt chaque ligne avec la méthode `fetch()` de `$reponse` et stocke cette ligne dans `$donnees`.

- L'affichage des champs de la base de données contenus dans \$donnees :
\$donnees['idSalarie'], etc.

Chaque donnée est affichée grâce à l'intitulé de sa colonne.

Le site : tout en PHP... ou presque

menu :

- Aller chez [google](#)
- Aller à [ma page](#)
- Aller à [la page](#)
- et puis voilà...

ID NOM PRENOM SEXE CATEGORIE SERVICE

1	Lambert	Marc	H	cadre	S01
2	Piot	Stéphanie	F	employé	S02
3	Hermeso	Gérard	H	employé	S01
4	Granola	Marie-Laure	F	cadre	S04
5	Trimoulon	Antoine	H	employé	S02
6	Garnic	Loïc	H	employé	S01
7	Javan	Christine	F	cadre	S02
8	Mirame	Kévin	H	employé	S03

Il y a 8 salariés.

Remarque : en fin de traitement nous avons affiché le compte des salariés en procédant de la façon suivante :

- Nous avons stocké dans une variable le nombre de lignes retournées par l'exécution de la requête :
\$nb = \$reponse->rowCount();

Le jeu de résultat \$reponse de classe PDOStatement permet d'utiliser la méthode rowCount() qui indique le nombre de lignes.

- Après le tableau, nous avons affiché le contenu de la variable :
`echo '<p>Il y a '.$nb.' salariés.</p>';`

A.2.3.2. Requête préparée avec projection et restriction

Nous pouvons indiquer les champs qui seront projetés ainsi que restreindre le nombre d'enregistrement retournés selon un critère.

Nous allons passer la requête : `SELECT idSalarie, nomSalarie, prenomSalarie FROM salarie WHERE serviceSalarie = 'S01';`

```
<?php
try
{
    $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
    $bdd = new PDO('mysql:host='.$hote.';dbname='.$base, $utilisateur, $mdp);
    $bdd->exec('SET NAMES utf8');
    $reponse = $bdd->prepare('SELECT idSalarie, nomSalarie, prenomSalarie FROM salarie WHERE
    serviceSalarie = ? ');
    $reponse->execute( array('S01') );
    $nb = $reponse->rowCount();
    echo '<table>';
    echo '<tr><th>ID</th><th>NOM</th><th>PRENOM</th></tr>';
    while ( $donnees = $reponse->fetch() )
    {
        echo '<tr>';
        echo '<td class="c1">'.$donnees['idSalarie'].'</td>';
        echo '<td class="c1">'.$donnees['nomSalarie'].'</td>';
        echo '<td class="c1">'.$donnees['prenomSalarie'].'</td>';
        echo '</tr>';
    }
    echo '</table>';
    echo '<p>Il y a '.$nb.' salariés.</p>';
    $reponse->closeCursor();
}
catch (Exception $erreur)
{
    die('Erreur : ' . $erreur->getMessage());
}
?>
```

Le site : tout en PHP... ou presque

menu :

- Aller chez google
- Aller à ma page
- Aller à la page
- et puis voilà...

ID NOM PRENOM

1	Lambert	Marc
3	Hermeso	Gérard
6	Garnic	Loïc

Il y a 3 salariés.

Afin de faire passer un critère de restriction dans notre requête SQL et de l'exécuter nous avons eu recours à une requête préparée qui s'écrit comme suit :

- Le corps de la requête :
`$reponse = $bdd->prepare('SELECT idSalarie, nomSalarie, prenomSalarie FROM salarie WHERE serviceSalarie = ? ');`

La requête est préparée `$bdd->prepare()` avec comme inconnue `?` le service du salarié.

- L'exécution de la requête avec ses paramètres (nos restrictions) :
`$reponse->execute(array('S01'));`

La requête est exécutée `execute()` avec en paramètre un tableau `array()` contenant tous les paramètres (autant que de `?`) de la requête préparée auparavant.

A.2.4. Requête préparée SELECT déclenchée par un formulaire

Dans la plupart des cas ce sera le visiteur qui déclenchera -par ses choix- l'interrogation de la base de données et l'affichage en résultant.

Ce sont les formulaires, que nous avons étudiés très largement qui vont permettre les choix des visiteurs. Nous allons utiliser les informations des formulaires pour faire des restrictions dans les enregistrements de notre base de données.

A.2.4.1. Formulaire avec des champs ouverts

Un formulaire permet de choisir le service à lister :

```
<div class="centre">
  <p>Liste des salariés</p>
  <form id="f1" method="post" action="voirSalaries.php">
    <fieldset>
      <legend>Critère de recherche :</legend>
      Saisir ici le service <input type="input" id="txtChoixService" name="txtChoixService" />
      <input name="brReset" type="reset" value="effacer" /><input name="cmdValider"
        type="submit" value="valider" />
    </fieldset>
  </form>
</div>
```



Le script *voirSalaries.php* va se charger du traitement :

```
<?php
if ( isset($_POST['txtChoixService']) && !empty($_POST['txtChoixService']) ) {
    $serviceChoisi = htmlspecialchars($_POST['txtChoixService']);
    try
    {
        $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
        $bdd = new PDO('mysql:host='.$hote.';dbname='.$base, $utilisateur, $mdp);
        $bdd->exec('SET NAMES utf8');
        $reponse = $bdd->prepare('SELECT idSalarie, nomSalarie, prenomSalarie FROM salarie
        WHERE serviceSalarie = ? ');
        $reponse->execute( array($serviceChoisi) );
        $nb = $reponse->rowCount();
        echo '<h3>Liste nominative des salariés du service '.$serviceChoisi.'</h3>';
        echo '<table>';
        echo '<tr><th>ID</th><th>NOM</th><th>PRENOM</th></tr>';
        while ( $donnees = $reponse->fetch() )
        {
            echo '<tr>';
            echo '<td class="c1">'.$donnees['idSalarie'].'</td>';
            echo '<td class="c1">'.$donnees['nomSalarie'].'</td>';
            echo '<td class="c1">'.$donnees['prenomSalarie'].'</td>';
            echo '</tr>';

        }
        echo '</table>';
        echo '<p>Il y a '.$nb.' salariés.</p>';
        $reponse->closeCursor();
    }
    catch (Exception $erreur)
    {
        die('Erreur : ' . $erreur->getMessage());
    }
}
else {
    echo '<h4 class="error">Erreur, service inconnu !';
}
?>
```

Liste nominative des salariés du service S01

ID NOM PRENOM

1	Lambert	Marc
3	Hermeso	Gérard
6	Garnic	Loïc

Il y a 3 salariés.

Remarque :

- On connaît déjà le contrôle des champs de formulaire avec **isset()** que l'on a complété par la fonction **empty()** afin de vérifier que le champ -en plus d'être envoyé- soit complété. On demande de vérifier un champ non vide : **!empty()**. Un message d'erreur s'affiche si ces deux conditions ne sont pas remplies.
- Il ne faut pas oublier d'utiliser **htmlspecialchars()** pour éviter l'insertion de code.

A.2.4.2. Formulaire avec des choix restreints

Le problème des champs de type texte ouverts est que l'on peut saisir n'importe quoi dedans ou se tromper.

Etant donné que nous connaissons la liste des services (il y a 4 services) nous pouvons limiter les risques en ne proposant que ceux que nous connaissons :

idService	libService
S01	Fabrication
S02	Emballage
S03	Commercial
S04	Administration

Notre formulaire contient une zone de liste déroulante contenant la liste restrictive de nos services :

```
<div class="centre">
  <p>Liste des salariés</p>
  <form id="f1" method="post" action="voirSalaries.php">
    <fieldset>
      <legend>Critère de recherche :</legend>
      Saisir ici le service
      <select id="lstChoixService" name="lstChoixService">
        <option value="S01">Fabrication</option>
        <option value="S02">Emballage</option>
        <option value="S03">Commercial</option>
        <option value="S04">Administration</option>
      </select>
      <input name="brReset" type="reset" value="effacer" /><input name="cmdValider"
        type="submit" value="valider" />
    </fieldset>
  </form>
</div>
```

Liste des salariés

Critère de recherche : _____

Saisir ici le service Fabrication réinitialiser valider

- Fabrication
- Emballage
- Commercial
- Administration

Cependant on peut penser que nos services soient plus nombreux au cours du temps ou ne contiennent aucun salarié (en ce cas le service n'est pas affiché). Il faut donc générer automatiquement (et dynamiquement) la liste des services.

Le formulaire contient une connexion à la base de données pour générer la zone de liste déroulante qui permettra le choix du service :

```
<div class="centre">
  <p>Liste des salariés</p>
  <form id="f1" method="post" action="voirSalaries.php">
    <fieldset>
      <legend>Critère de recherche :</legend>
      Saisir ici le service
      <select id="lstChoixService" name="lstChoixService">
        <?php
          try
          {
            $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
            $bdd = new PDO('mysql:host='.$hote.';dbname='.$base, $utilisateur, $mdp);
            $bdd->exec('SET NAMES utf8');
            $reponse = $bdd->prepare('SELECT distinct service.idService AS "id", libService
            FROM salarie, service WHERE salarie.serviceSalarie = service.idService ');
            $reponse->execute( array() );
            while ( $donnees = $reponse->fetch() )
            {
              echo '<option value="'. $donnees['id']. '>'. $donnees['libService']
              . '</option>';
            }
            $reponse->closeCursor();
          }
          catch (Exception $erreur)
          {
            die('Erreur : ' . $erreur->getMessage());
          }
        ?>
      </select>
      <input name="brReset" type="reset" value="réinitialiser" /><input name="cmdValider"
      type="submit" value="valider" />
    </fieldset>
  </form>
</div>
```

The screenshot shows a web form titled "Liste des salariés". It features a search criterion field labeled "Critère de recherche :". Below this, there is a text input field labeled "Saisir ici le service". To the right of this input is a dropdown menu currently showing "Fabrication". The dropdown menu is open, displaying a list of services: "Fabrication", "Emballage", "Commercial", and "Administration". To the right of the dropdown are two buttons: "réinitialiser" and "valider".

Remarque :

- Nous avons utilisé un alias `service.idService AS "id"` dans notre requête SQL afin de renommer la colonne `idService` en `id`. C'est ce nom de champ qui sera utilisé ensuite avec `$donnees['id']`.
- L'utilisation de la clause `distinct` est nécessaire car notre requête SQL est susceptible d'avoir des doublons à cause de la jointure entre les tables `salarie` et `service`.

A.2.4.3. Formulaire avec plusieurs critères de restriction

Nous pouvons combiner les critères de recherche et les transmettre au script réalisant les traitements.

Notre formulaire propose dorénavant de choisir le service et le sexe d'un salarié :

```
<div class="centre">
  <p>Liste des salariés</p>
  <form id="f1" method="post" action="voirSalaries.php">
    <fieldset>
      <legend>Critères de recherche :</legend>
      Service
      <select id="lstChoixService" name="lstChoixService">
        <?php
          try
          {
            $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
            $bdd = new PDO('mysql:host='.$hote.';dbname='.$base, $utilisateur, $mdp);
            $bdd->exec('SET NAMES utf8');
            $reponse = $bdd->prepare('SELECT distinct service.idService AS "id", libService
            FROM salarie, service WHERE salarie.serviceSalarie = service.idService ');
            $reponse->execute( array() );
            while ( $donnees = $reponse->fetch() )
            {
              echo '<option value="'. $donnees['id']. ">'. $donnees['libService']
              . '</option>';
            }
            $reponse->closeCursor();
          }
          catch (Exception $erreur)
          {
            die('Erreur : ' . $erreur->getMessage());
          }
        ?>
      </select><br/><br/>
      Sexe des salariés recherchés :
      femmes <input type="radio" name="optChoixSexe" id="optChoixSexe" value="F" />
      hommes <input type="radio" name="optChoixSexe" id="optChoixSexe" value="H" />
      les deux <input type="radio" name="optChoixSexe" id="optChoixSexe" value="FH"
      checked="checked" /><br/><br/>
      <input name="brReset" type="reset" value="réinitialiser" /><input name="cmdValider"
      type="submit" value="valider" />
    </fieldset>
  </form>
</div>
```



Remarque : le choix du sexe du salarié comporte trois possibilités :

- ne voir que les femmes,
- ne voir que les hommes,
- voir les femmes et les hommes.

Le script PHP va donc devoir d'une part gérer les deux critères et d'autre part gérer le cas où on ne veut pas voir les personnes d'un sexe en particulier.

```
<?php
if ( isset($_POST['lstChoixService']) && !empty($_POST['lstChoixService'])
    && isset($_POST['optChoixSexe']) && !empty($_POST['optChoixSexe']) ) {
    $serviceChoisi = $_POST['lstChoixService'];
    $sexeChoisi = $_POST['optChoixSexe'];
    try
    {
        $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
        $bdd = new PDO('mysql:host='.$hote.';dbname='.$base, $utilisateur, $mdp);
        $bdd->exec('SET NAMES utf8');
        if ( $sexeChoisi == 'FH' ) {
            $reponse = $bdd->prepare('SELECT idSalarie, nomSalarie, prenomSalarie FROM salarie
            WHERE serviceSalarie = ? ');
            $reponse->execute( array($serviceChoisi) );
        }
        else {
            $reponse = $bdd->prepare('SELECT idSalarie, nomSalarie, prenomSalarie FROM salarie
            WHERE serviceSalarie = ? and sexeSalarie = ? ');
            $reponse->execute( array($serviceChoisi, $sexeChoisi) );
        }
        $nb = $reponse->rowCount();
        if ( $nb > 0 ) {
            echo '<h3>Liste nominative des salariés du service '.$serviceChoisi.'</h3>';
            echo '<table>';
            echo '<tr><th>ID</th><th>NOM</th><th>PRENOM</th></tr>';
            while ( $donnees = $reponse->fetch() )
            {
                echo '<tr>';
                echo '<td class="c1">'.$donnees['idSalarie'].'</td>';
                echo '<td class="c1">'.$donnees['nomSalarie'].'</td>';
                echo '<td class="c1">'.$donnees['prenomSalarie'].'</td>';
                echo '</tr>';
            }
            echo '</table>';
            echo '<p>Il y a '.$nb.' salariés.</p>';
        }
        else {
            echo '<h4>Il n\'y a aucun salarié répondant à vos critères !</h4>';
        }
        $reponse->closeCursor();
    }
    catch (Exception $erreur)
    {
        die('Erreur : ' . $erreur->getMessage());
    }
}
else {
    echo '<h4 class="error">Erreur, service inconnu !';
}
?>
```

Il n'y a aucun salarié répondant à vos critères !

Remarques :

- les contrôles `isset()` ont été étendus au choix du sexe du salarié.
- Une condition examine également si le choix est indifférent `if ($sexeChoisi == 'FH')` : alors la requête ne porte que sur les services (`'SELECT ... WHERE serviceSalarie = ? '`). Dans le cas contraire la requête s'intéresse aussi au sexe demandé (`'SELECT ... WHERE serviceSalarie = ? and sexeSalarie = ? '`)

A.2.4.4. Formulaire et traitements dans une même page

Nous allons marier les connaissances acquises dans la partie précédente à propos des pages qui s'appellent elles-mêmes grâce à un formulaire et la récupération des données de la base.

Notre formulaire et les traitements se suivent. On choisit d'abord le service et ensuite la liste s'affiche.

Première partie : le formulaire propose de choisir un service ou tous (par défaut).

```
<div class="centre">
  <form id="f1" method="post" action="voirSalaries.php">
    <p>
      Service : <select id="lstChoixService" name="lstChoixService">
        <?php
          try
          {
            $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
            $bdd = new PDO('mysql:host='.$hote.';dbname='.$base, $utilisateur, $mdp);
            $bdd->exec('SET NAMES utf8');
            $reponse = $bdd->prepare('SELECT distinct service.idService AS "id", libService
            FROM salarie, service WHERE salarie.serviceSalarie = service.idService ');
            $reponse->execute( array() );
            while ( $donnees = $reponse->fetch() )
            {
              echo '<option value="'. $donnees['id']. '>'. $donnees['libService']. '</option>';
            }
            $reponse->closeCursor();
          }
          catch (Exception $erreur)
          {
            die('Erreur : ' . $erreur->getMessage());
          }
        ?>
        <option value="tous" selected="selected">Tous les services</option>
      </select>
      <input name="cmdService" type="submit" value="valider" />
    </p>
  </form>
```


Deuxième partie : si l'on détecte que le bouton de validation du formulaire a été utilisé `isset($_POST['cmdService'])` et que le service existe `isset($_POST['lstChoixService'])` et qu'il est bien renseigné `!empty($_POST['lstChoixService'])` alors on affiche le résultat.

```
<?php
if ( isset($_POST['cmdService']) && isset($_POST['lstChoixService'])
    && !empty($_POST['lstChoixService']) ) {
    $serviceChoisi = $_POST['lstChoixService'];
    try
    {
        $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
        $bdd = new PDO('mysql:host='.$hote.';dbname='.$base, $utilisateur, $mdp);
        $bdd->exec('SET NAMES utf8');
        if ( $serviceChoisi == 'tous' ) {
            $reponse = $bdd->prepare('SELECT idSalarie, nomSalarie, prenomSalarie
            FROM salarie ');
            $reponse->execute( array() );
            $afficheService = 'de tous les services.';
        }
        else {
            $reponse = $bdd->prepare('SELECT idSalarie, nomSalarie, prenomSalarie
            FROM salarie WHERE serviceSalarie = ? ');
            $reponse->execute( array($serviceChoisi) );
            $afficheService = 'du service '.$serviceChoisi.'.';
        }
        $nb = $reponse->rowCount();
        if ( $nb > 0 ) {
            echo '<h3>Liste nominative des salariés '.$afficheService.'</h3>';
            echo '<table>';
            echo '<tr><th>ID</th><th>NOM</th><th>PRENOM</th></tr>';
            while ( $donnees = $reponse->fetch() )
            {
                echo '<tr>';
                echo '<td class="c1">'.$donnees['idSalarie'].'</td>';
                echo '<td class="c1">'.$donnees['nomSalarie'].'</td>';
                echo '<td class="c1">'.$donnees['prenomSalarie'].'</td>';
                echo '</tr>';
            }
            echo '</table>';
            echo '<p>Il y a '.$nb.' salariés.</p>';
        }
        else {
            echo '<h4>Il n\'y a aucun salarié répondant à vos critères !</h4>';
        }
        $reponse->closeCursor();
    }
    catch (Exception $erreur)
    {
        die('Erreur : ' . $erreur->getMessage());
    }
}
else {
    echo '<h4 class="error">Choisissez un service.';
}
?>
</div>
```

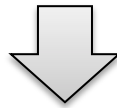
Remarques :

- La requête est différente selon que l'on désire l'affichage d'un service ou de tous les services `if ($serviceChoisi == 'tous') {}`.
- On prévoit également d'adapter le titre du tableau contenant les résultats via la variable : `$afficheService`.

Service :

Choisissez

- Fabrication
- Emballage**
- Commercial
- Administration
- Tous les services



Service :

Liste nominative des salariés du service S01.

ID	NOM	PRENOM
1	Lambert	Marc
3	Hermeso	Gérard
6	Garnic	Loïc

Il y a 3 salariés.

A.3. Manipulation et insertion de données

A.3.1. Insertion des données

La consultation des données de la base de données implique l'existence de données à consulter. Au départ, il faut une interface qui permet d'alimenter la base de données.

Un formulaire est conçu pour permettre l'enregistrement d'un nouveau salarié :

```
<form id="f1" method="post" action="traitementsSalaries.php">
<fieldset>
  <legend>Nouveau salarié :</legend>
  Nom : <input type="text" name="txtNomSalarie" id="txtNomSalarie" />
  Prenom : <input type="text" name="txtPrenomSalarie" id="txtPrenomSalarie" /><br/><br/>
  Sexe : femme <input type="radio" name="optSexeSalarie" id="optSexeSalarie" value="F" />
  homme <input type="radio" name="optSexeSalarie" id="optSexeSalarie" value="H" /><br/><br/>
  Cadre <input type="checkbox" name="chkCatSalarie" id="chkCatSalarie" /><br/><br/>
  Service :
  <select id="lstServiceSalarie" name="lstServiceSalarie">
    <?php
      try
      {
        $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
        $bdd = new PDO('mysql:host='.$hote.';dbname='.$base, $utilisateur, $mdp);
        $bdd->exec('SET NAMES utf8');
        $reponse = $bdd->prepare('SELECT distinct service.idService AS "id", libService
        FROM salarie, service WHERE salarie.serviceSalarie = service.idService ');
        $reponse->execute( array() );
        while ( $donnees = $reponse->fetch() )
        {
          echo '<option value="'. $donnees['id']. '>'. $donnees['libService']. '</option>';
        }
        $reponse->closeCursor();
      }
      catch (Exception $erreur)
      {
        die('Erreur : ' . $erreur->getMessage());
      }
    ?>
  </select><br/><br/>
  <input name="brReset" type="reset" value="réinitialiser" />
  <input name="cmdValider" type="submit" value="valider" />
</fieldset>
</form>
```

Remarques :

- Le formulaire demande autant d'informations que nécessaires, sauf l'identifiant du salarié qui sera généré automatiquement par MySQL (le champ idSalarie est en auto incrémentation).
- Le formulaire ne propose pas le choix entre « cadre » et « employé » mais uniquement un case à cocher « cadre » qui implique que si elle n'est pas cochée le salarié est un employé.

Le script *traitementsSalarie.php* reçoit les données du formulaire et procède à l'enregistrement :

```
<?php
if ( isset($_POST['txtNomSalarie']) && !empty($_POST['txtNomSalarie'])
    && isset($_POST['txtPrenomSalarie']) && !empty($_POST['txtPrenomSalarie'])
    && isset($_POST['optSexeSalarie']) && !empty($_POST['optSexeSalarie'])
    && isset($_POST['lstServiceSalarie']) && !empty($_POST['lstServiceSalarie']) ) {

    $nomSal = htmlspecialchars($_POST['txtNomSalarie']);
    $prenomSal = htmlspecialchars($_POST['txtPrenomSalarie']);
    $sexeSal = $_POST['optSexeSalarie'];
    if ( isset($_POST['chkCatSalarie']) ) {
        $catSal = 'Cadre';
    }
    else {
        $catSal = 'Employé';
    }

    $serviceSal = $_POST['lstServiceSalarie'];
    try
    {
        $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
        $bdd = new PDO('mysql:host='.$hote.';dbname='.$base, $utilisateur, $mdp);
        $bdd->exec('SET NAMES utf8');
        $insertion = $bdd->prepare('INSERT INTO salarie (idSalarie, nomSalarie, prenomSalarie,
        sexeSalarie, catSalarie, serviceSalarie) VALUES (\'', :nom, :prenom, :sexe,
        :categorie, :service)');
        $insertion->execute(array(
            'nom' => $nomSal,
            'prenom' => $prenomSal,
            'sexe' => $sexeSal,
            'categorie' => $catSal,
            'service' => $serviceSal
        ));
        $dernierId = $bdd->lastInsertId();
        echo '<h4 class="good">Le nouveau salarié '.$nomSal.' '.$prenomSal.' a bien été
        enregistré avec l\'identifiant '.$dernierId.'</h4>';
        $insertion->closeCursor();
    }
    catch (Exception $erreur)
    {
        die('Erreur : ' . $erreur->getMessage());
    }
}
else {
    echo '<h4 class="error">Erreur, tous les champs doivent être complétés !';
}
?>
```

Remarques :

- La requête préparée est, cette fois-ci, une requête d'insertion (INSERT) qui intègre toutes les données recensées via le formulaire. Seul l'identifiant, généré automatiquement, reçoit un champ vide (\'). Les autres champs sont associés via des marqueurs (précédés de « : ») et les données que nous avons stocké dans les variables.
- La méthode lastInsertId() de PDO indique l'identifiant du dernier élément inséré (ici il s'agit du 9ème salarie, stocké dans le champ idSalarie).

Gestion des salariés

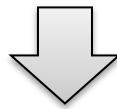
Nouveau salarié :

Nom : Prenom :

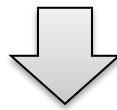
Sexe : femme ☒ homme ☐

Cadre ☐

Service :



Le nouveau salarié Lamotte Julie a bien été enregistré avec l'identifiant 9



idSalarie	nomSalarie	prenomSalarie	sexeSalarie	catSalarie	serviceSalarie
1	Lambert	Marc	H	cadre	S01
2	Piot	Stéphanie	F	employé	S02
3	Hermeso	Gérard	H	employé	S01
4	Granola	Marie-Laure	F	cadre	S04
5	Trimoulon	Antoine	H	employé	S02
6	Garnic	Loïc	H	employé	S01
7	Javan	Christine	F	cadre	S02
8	Mirame	Kévin	H	employé	S03
9	Lamotte	Julie	F	employé	S02

Remarque : il faudra ne pas oublier, avec Javascript, de faire des contrôles sur les champs de formulaire avant de les envoyer au formulaire. Ces contrôles soulageront d'autant plus le serveur qu'ils seront précis.

A.3.2. Suppression des données

L'effacement des données de la base de données doit être rigoureusement surveillé !

- D'une part une erreur peut rapidement survenir et sans confirmation de l'effacement cela peut conduire à effacer accidentellement des informations ;
- D'autre part un visiteur malveillant peut en profiter pour vider votre base de données de tout son contenu.

Nous allons adjoindre à notre tableau la possibilité d'effacer les salariés (sans confirmation) :

```
<?php
if ( isset($_POST['cmdService']) && isset($_POST['lstChoixService'])
    && !empty($_POST['lstChoixService']) ) {
    $serviceChoisi = $_POST['lstChoixService'];
    try
    {
        $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
        $bdd = new PDO('mysql:host='.$hote.';dbname='.$base, $utilisateur, $mdp);
        $bdd->exec('SET NAMES utf8');
        if ( $serviceChoisi == 'tous' ) {
            $reponse = $bdd->prepare('SELECT idSalarie, nomSalarie, prenomSalarie
FROM salarie ');
            $reponse->execute( array() );
            $afficheService = 'de tous les services.';
        }
        else {
            $reponse = $bdd->prepare('SELECT idSalarie, nomSalarie, prenomSalarie
FROM salarie WHERE serviceSalarie = ? ');
            $reponse->execute( array($serviceChoisi) );
            $afficheService = 'du service '.$serviceChoisi.'.';
        }
        $nb = $reponse->rowCount();
        if ( $nb > 0 ) {
            echo '<h3>Liste nominative des salariés '.$afficheService.'</h3>';
            echo '<table>';
            echo '<tr><th>Id</th><th>Nom</th><th>Prénom</th><th>Action</th></tr>';
            while ( $donnees = $reponse->fetch() )
            {
                echo '<tr>';
                echo '<td class="c1">'.$donnees['idSalarie'].'</td>';
                echo '<td class="c1">'.$donnees['nomSalarie'].'</td>';
                echo '<td class="c1">'.$donnees['prenomSalarie'].'</td>';
                echo '<td class="c1">
                    <a href="traitementsSalaries.php?suppSalarie='.$donnees['idSalarie'].'">
                    </a></td>';
                echo '</tr>';
            }
            echo '</table>';
            echo '<p>Il y a '.$nb.' salariés.</p>';
        }
        else {
            echo '<h4>Il n\'y a aucun salarié répondant à vos critères !</h4>';
        }
        $reponse->closeCursor();
    }
    catch (Exception $erreur)
    {
        die('Erreur : ' . $erreur->getMessage());
    }
}
else {
    echo '<h4 class="error">Choisissez un service.';
}
?>
```

Remarques :

- Une colonne « Action » a été ajoutée au tableau `<th>Action</th>`.
- La colonne propose un lien imagé pour effacer le salarié courant en passant le paramètre (l'identifiant du salarié) par l'URL. Le paramètre s'appelle `suppSalarie` et contient l'identifiant du salarié `$donnees['idSalarie']` :






```
<a href="traitementsSalaries.php?suppSalarie='.$donnees['idSalarie'].'">  
  
</a>
```

Valeur du
paramètre.

Paramètre.

Service :

Liste nominative des salariés du service S02.

Id	Nom	Prénom	Action
2	Piot	Stéphanie	
5	Trimoulon	Antoine	
7	Javan	Christine	
9	Lamotte	Julie	
10	Tzam	François	

Il y a 5 salariés.

Ce salarié va être sélectionné
pour l'effacement.

Le script *traitementsSalarie.php* (qui sert également à l'enregistrement des nouveaux salariés) reçoit les données transmises dans l'URL :

```
if ( isset($_GET['suppSalarie']) && !empty($_GET['suppSalarie']) ) {  
    // Ici on supprime un salarié  
    $suppSal = $_GET['suppSalarie'];  
    try  
    {  
        $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;  
        $bdd = new PDO('mysql:host='.$hote.';dbname='.$base, $utilisateur, $mdp);  
        $modification = $bdd->prepare('DELETE FROM salarie WHERE idSalarie=:id');  
        $modification->execute(array(  
            'id' => $suppSal  
        ));  
        $modification->closeCursor();  
        echo '<h4 class="good">Le salarié <i>numéro ' . $suppSal . '</i> a été supprimé.</h4>';  
    }  
    catch (Exception $e)  
    {  
        die('Erreur : ' . $e->getMessage());  
    }  
}  
elseif ( !isset($_POST['cmdValider']) ) {  
    echo '<h4 class="error">Erreur, le salarié n\'a pas pu être effacé !';  
}
```

Le salarié numéro 10 a été supprimé.

Remarques :

- On commence par contrôler si l'on a bien reçu le paramètre avec `isset()` et qu'il dispose bien d'un contenu `!empty()`.
- Sinon on teste `elseif ()` que les données ne nous parviennent pas d'un autre formulaire, en l'occurrence celui d'enregistrement d'un nouvel utilisateur. Si l'on avait utilisé le formulaire d'enregistrement le bouton `cmdValider` aurait été pressé. Autrement dit le message d'erreur lié à l'effacement ne s'affiche que si on est sûr que le traitement demandé n'est pas pour une autre fonctionnalité de notre script (ici celle d'ajout d'un nouvel utilisateur située après et non présentée).

Voici synthétiquement la structure des conditions :

```
<?php
if ( isset($_GET['suppSalarie']) && !empty($_GET['suppSalarie']) ) {

    // Ici on supprime un salarié

}
elseif ( !isset($_POST['cmdValider']) ) {
    echo '<h4 class="error">Erreur, le salarié n\'a pas pu être effacé !';
}

if ( isset($_POST['txtNomSalarie']) && !empty($_POST['txtNomSalarie'])
    && isset($_POST['txtPrenomSalarie']) && !empty($_POST['txtPrenomSalarie'])
    && isset($_POST['optSexeSalarie']) && !empty($_POST['optSexeSalarie'])
    && isset($_POST['lstServiceSalarie']) && !empty($_POST['lstServiceSalarie']) ) {

    // Ici on ajoute un salarié

}
elseif ( !isset($_GET['suppSalarie']) ) {
    echo '<h4 class="error">Erreur, tous les champs doivent être complétés !';
}
?>
```

Pour « sécuriser » l'effacement il faudra prévoir :

- une confirmation avant de soumettre le formulaire ;
- la restriction de l'accès au formulaire d'effacement a un public restreint (par exemple avec la gestion des sessions) ;
- la gestion des transactions SQL qui permet de revenir en arrière en cas d'erreur.

Par exemple, la confirmation de l'effacement avec fonction Javascript :

```
function attention(idEffacer, prenom, nom) {
    if( confirm('Etes-vous certain de vouloir effacer '+prenom+ ' '+nom+ ' ? ') )
    {
        location.href='traitementsSalaries.php?suppSalarie='+idEffacer;
    }
}
```


L'effacement d'un salarié avec un lien <a> est remplacé par un bouton image et associé à l'événement onclick placé en attribut.






```
echo '<td class="c1"><input type="image" id="suppSal" name="suppSal" src="img/supp.png"
    onclick="attention('.$donnees['idSalarie'].' , \''. $donnees['prenomSalarie'].'\' ,
    \''. $donnees['nomSalarie'].'\'');" /></td>';
```

Le code source obtenu dans Firefox permet d'y voir plus clair (ici pour le salarié sur lequel nous allons cliquer) :

```
<input type="image" id="suppSal" name="suppSal" src="img/supp.png" onclick="attention(10,
'François', 'Tzam');" />
```

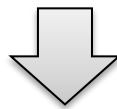
Service :

Liste nominative des salariés du service S02.

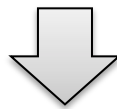
Id	Nom	Prénom	Action
2	Piot	Stéphanie	
5	Trimoulon	Antoine	
7	Javan	Christine	
9	Lamotte	Julie	
10	Tzam	François	

Il y a 5 salariés.

Ce salarié va être sélectionné pour l'effacement.



Etes-vous certain de vouloir effacer François Tzam ?



Le salarié numéro 10 a été supprimé.

A.3.1. Modification des données

On peut éditer l'enregistrement d'un salarié pour modifier tout ou partie des informations le concernant. Seul son identifiant, contrainte d'intégrité de la base de données ne sera pas modifiable.

Nous avons ajouté dans une nouvelle colonne la possibilité d'éditer la fiche d'un salarié :

```
<?php
if ( isset($_POST['cmdService']) && isset($_POST['lstChoixService']) &&
!empty($_POST['lstChoixService']) ) {
    $serviceChoisi = $_POST['lstChoixService'];
    try
    {
        $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
        $bdd = new PDO('mysql:host='.$hote.';dbname='.$base, $utilisateur, $mdp);
        $bdd->exec('SET NAMES utf8');
        if ( $serviceChoisi == 'tous' ) {
            $reponse = $bdd->prepare('SELECT idSalarie, nomSalarie, prenomSalarie
FROM salarie ');
            $reponse->execute( array() );
            $afficheService = 'de tous les services.';
        }
        else {
            $reponse = $bdd->prepare('SELECT idSalarie, nomSalarie, prenomSalarie
FROM salarie WHERE serviceSalarie = ? ');
            $reponse->execute( array($serviceChoisi) );
            $afficheService = 'du service '.$serviceChoisi.'.';
        }
        $nb = $reponse->rowCount();
        if ( $nb > 0 ) {
            echo '<h3>Liste nominative des salariés '.$afficheService.'</h3>';
            echo '<table>';
            echo '<tr><th>Id</th><th>Nom</th><th>Prénom</th><th colspan="2">Actions</th></tr>';
            while ( $donnees = $reponse->fetch() )
            {
                echo '<tr>';
                echo '<td class="c1">'.$donnees['idSalarie'].'</td>';
                echo '<td class="c1">'.$donnees['nomSalarie'].'</td>';
                echo '<td class="c1">'.$donnees['prenomSalarie'].'</td>';
                echo '<td class="c1"><input type="image" id="editSal" name="editSal"
src="img/edit.png"
onclick="location.href=\''.traitementsSalaries.php?editSalarie=
'.$donnees['idSalarie'].'\'"/></td>';
                echo '<td class="c1"><input type="image" id="suppSal" name="suppSal"
src="img/supp.png" onclick="attention(\''.$donnees['idSalarie'].'',
\''.$donnees['prenomSalarie'].'', \''.$donnees['nomSalarie'].'\')"/></td>';
                echo '</tr>';
            }
            echo '</table>';
            echo '<p>Il y a '.$nb.' salariés.</p>';
        }
        else {
            echo '<h4>Il n\'y a aucun salarié répondant à vos critères !</h4>';
        }
        $reponse->closeCursor();
    }
    catch (Exception $erreur)
    {
        die('Erreur : ' . $erreur->getMessage());
    }
}
else {
    echo '<h4 class="error">Choisissez un service.';
}
?>
```

Remarque : un bouton image est encore utilisé et l'événement onclick permet de changer l'URL de la page (vers *traitementsSalarie.php*) en spécifiant le paramètre d'édition et l'identifiant du salarié.

Valeur du paramètre.

```
location.href=\"traitementsSalaries.php?editSalarie='.$donnees['idSalarie'].'\"
```

Paramètre.

Service :

Liste nominative des salariés du service S02.

Id	Nom	Prénom	Actions	
2	Piot	Stéphanie		
5	Trimoulon	Antoine		
7	Javan	Christine		
9	Lamotte	Julie		

Il y a 4 salariés.

Il est à noter que l'on aurait pu garder un lien `<a>` dans les deux cas.

- Pour l'effacement avec confirmation :

```
<a id="suppSal" name="suppSal" href="Javascript:attention('.$donnees['idSalarie'].',\n\n'.$donnees['prenomSalarie'].'\n\n'.$donnees['nomSalarie'].'\n\n');"/>\n\n</a>
```
- Pour la modification :

```
<a id="editSal" name="editSal" href="Javascript:\nlocation.href=\"traitementsSalaries.php?editSalarie='.$donnees['idSalarie'].'\""/>\n\n</a>
```

Le script *traitementsSalaries.php* s'enrichi d'un nouveau traitement détecté par la présence et l'affectation du paramètre **editSalarie**.

```
if ( isset($_GET['editSalarie']) && !empty($_GET['editSalarie']) ) {
    // Ici on édite la fiche d'un salarié
    $idSal = $_GET['editSalarie'];
    try
    {
        $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
        $bdd = new PDO('mysql:host='.$hote.';dbname='.$base, $utilisateur, $mdp);
        $bdd->exec('SET NAMES utf8');
        $reponse = $bdd->prepare('SELECT * FROM salarie WHERE idSalarie = ? ');
        $reponse->execute( array($idSal) );
        $donnees = $reponse->fetch();
        $reponse->closeCursor();
    }
    ?>

    <h4>Gestion des salariés</h4>
    <form id="f1" method="post" action="traitementsSalaries.php">
    <fieldset>
    <legend>Modification du salarié numéro <b><?php echo $donnees['idSalarie']; ?></b>
    </legend>
    Nom : <input type="text" name="txtNomSalarie" id="txtNomSalarie"
    value="<?php echo $donnees['nomSalarie']; ?>" />
    Prenom : <input type="text" name="txtPrenomSalarie" id="txtPrenomSalarie"
    value="<?php echo $donnees['prenomSalarie']; ?>" /><br/><br/>
    Sexe :
    <?php
    if ( $donnees['sexeSalarie'] == 'F' ) {
        echo 'femme <input type="radio" name="optSexeSalarie" id="optSexeSalarie" value="F"
        checked="checked" />';
        echo 'homme <input type="radio" name="optSexeSalarie" id="optSexeSalarie" value="H"
        /><br/><br/>';
    }
    else {
        echo 'femme <input type="radio" name="optSexeSalarie" id="optSexeSalarie" value="F"
        />';
        echo 'homme <input type="radio" name="optSexeSalarie" id="optSexeSalarie" value="H"
        checked="checked" /><br/><br/>';
    }
    if ( $donnees['catSalarie'] == 'cadre' ) {
        echo 'Cadre <input type="checkbox" name="chkCatSalarie" id="chkCatSalarie"
        checked="checked" /><br/><br/>';
    }
    else {
        echo 'Cadre <input type="checkbox" name="chkCatSalarie" id="chkCatSalarie"
        /><br/><br/>';
    }
    echo 'Service : ';
    echo '<select id="lstServiceSalarie" name="lstServiceSalarie">';
    try
    {
        $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
        $bdd = new PDO('mysql:host='.$hote.';dbname='.$base, $utilisateur, $mdp);
        $bdd->exec('SET NAMES utf8');
        $reponse = $bdd->prepare('SELECT distinct service.idService AS "id", libService
        FROM salarie, service WHERE salarie.serviceSalarie = service.idService ');
        $reponse->execute( array() );
        while ( $donnS = $reponse->fetch() )
        {
            if ( $donnees['serviceSalarie'] == $donnS['id'] ) {
                echo '<option value="'.$donnS['id'].'" selected="selected">'
                . $donnS['libService'] . '</option>';
            }
            else {
                echo '<option value="'.$donnS['id'].'">' . $donnS['libService'] . '</option>';
            }
        }
    }
    $reponse->closeCursor();
}
```

```

    }
    catch (Exception $erreur)
    {
        die('Erreur : ' . $erreur->getMessage());
    }
    echo '</select><br/><br/>';
?>
<input type="hidden" name="hdIdSalarie" id="hdIdSalarie"
value="<?php echo $donnees['idSalarie']; ?>" />
<input name="brReset" type="reset" value="réinitialiser" />
<input name="cmdModifier" type="submit" value="valider" />
</fieldset>
</form>

<?php
}
catch (Exception $erreur)
{
    die('Erreur : ' . $erreur->getMessage());
}
}
elseif ( !isset($_POST['cmdValider']) && !isset($_GET['suppSalarie']) ) {
    echo '<h4 class="error">Erreur, le salarié demandé ne peut être édité !';
}

```

Remarques :

- Le script propose un nouveau formulaire après avoir été rechercher dans la base de données toutes les données concernant le salarié (première requête). Le formulaire est comparable à celui que nous avons déjà utilisé pour enregistrer un nouveau salarié. Il contient comme valeurs par défaut toutes les informations relatives au salarié sélectionné.
- Le formulaire marie code HTML incluant du PHP et PHP incluant du HTML. En effet, l'identifiant du salarié comme les champs contenant son nom et son prénom sont en HTML :

```
<input type="text" name="txtNomSalarie" id="txtNomSalarie" value="<?php echo $donnees['nomSalarie']; ?>" />
```

 Ici la valeur par défaut du champ sera affectée de \$donnees['nomSalarie'] qui est une variable PHP accessibles en délimitant le code et en utilisant l'instruction d'affichage **echo**.
- Toutes les listes (bouton radio, zone de liste déroulante) et la case à cocher sont traitées au travers de conditions permettant d'adapter l'élément par défaut selon ce qui enregistré dans la base de données pour le salarié.

- Pour pouvoir également transmettre l'identifiant du salarié, car il s'agit d'une information incontournable pour mettre à jour les données qui le concernent, nous utilisons un champs de type caché :

```
<input type="hidden" name="hdIdSalarie" id="hdIdSalarie" value="<?php echo $donnees['idSalarie']; ?>" />
```

Ce champ sera donc envoyé comme les autres champs du formulaire à l'insu du visiteur et nous pourrons récupérer l'information de la même manière que les autres.

- Il faut être vigilant sur les variables qui sont utilisées lors des interrogations de la base de données. En effet \$donnees aurait été écrasée si l'on n'avait pas utilisée \$donnS dans le traitement de la requête suivante. Il a fallu également « libérer » \$reponse->closeCursor(); avant de pouvoir l'utiliser dans la requête suivante.

Le script *traitementsSalarie.php* peut maintenant enregistrer les modifications demandées :

```
if (isset($_POST['cmdModifier']) && isset($_POST['txtNomSalarie']) &&
!empty($_POST['txtNomSalarie'])
    && isset($_POST['txtPrenomSalarie']) && !empty($_POST['txtPrenomSalarie'])
    && isset($_POST['optSexeSalarie']) && !empty($_POST['optSexeSalarie'])
    && isset($_POST['lstServiceSalarie']) && !empty($_POST['lstServiceSalarie']) ) {

    // Ici on modifie un salarié
    $nomSal = htmlspecialchars($_POST['txtNomSalarie']);
    $prenomSal = htmlspecialchars($_POST['txtPrenomSalarie']);
    $sexeSal = $_POST['optSexeSalarie'];
    if ( isset($_POST['chkCatSalarie']) ) {
        $catSal = 'Cadre';
    }
    else {
        $catSal = 'Employé';
    }

    $serviceSal = $_POST['lstServiceSalarie'];
    $idSal = $_POST['hdIdSalarie'];

    try
    {
        $pdo_options[PDO::ATTR_ERRMODE] = PDO::ERRMODE_EXCEPTION;
        $bdd = new PDO('mysql:host='.$hote.';dbname='.$base, $utilisateur, $mdp);
        $bdd->exec('SET NAMES utf8');
        $modification = $bdd->prepare('UPDATE salarie SET nomSalarie = :nom,
        prenomSalarie = :prenom, sexeSalarie = :sexe, catSalarie = :categorie,
        serviceSalarie = :service WHERE idSalarie = :id');
        $modification->execute(array(
            'nom' => $nomSal,
            'prenom' => $prenomSal,
            'sexe' => $sexeSal,
            'categorie' => $catSal,
            'service' => $serviceSal,
            'id' => $idSal
        ));
        echo '<h4 class="good">Les données à propos du salarié '.$nomSal.' '.$prenomSal.' ont
        bien été mises à jour</h4>';
        $modification->closeCursor();
    }
    catch (Exception $erreur)
    {
        die('Erreur : ' . $erreur->getMessage());
    }
}

elseif ( !isset($_POST['cmdValider']) && !isset($_GET['suppSalarie']) &&
!isset($_GET['editSalarie']) ) {
    echo '<h4 class="error">Erreur, tous les champs doivent être complétés !';
}
```

Remarque : peu de grandes nouveautés ici hormis la requête SQL de mise à jour.

Gestion des salariés

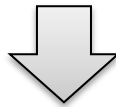
Modification du salarié numéro 2

Nom Prenom :

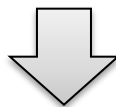
Sexe : femme ☐ homme ☒

Cadre ☒

Service :



Les données à propos du salarié Pliot Stéphanie ont bien été mises à jour



IdSalarie	nomSalarie	prenomSalarie	sexeSalarie	catSalarie	serviceSalarie
1	Lambert	Marc	H	cadre	S01
2	Pliot	Stéphanie	F	cadre	S02
3	Hermeso	Gérard	H	employé	S01
4	Granola	Marie-Laure	F	cadre	S04
5	Trimoulon	Antoine	H	employé	S02
6	Garnic	Loïc	H	employé	S01
7	Javan	Christine	F	cadre	S02
8	Mirame	Kévin	H	employé	S03
9	Lamotte	Julie	F	employé	S02