Les données du jeu d'enregistrements sont mises en forme avec la même méthode que dans l'exemple de la requête 2 (voir repère 3 de la figure 18-21). Les différentes lignes du résultat sont affichées dans le corps de la page (voir figure 18-23) :

```
while($row_rsListeAdherents = mysql_fetch_assoc($rsListeAdherents)) {
echo "L'adhérent <b>".$row_rsListeAdherents['nom']."</b>";
echo " - <b>".$row_rsListeAdherents['prenom']."</b>";
echo " est né(e) en <b>".$row_rsListeAdherents['anneeNaissance']."</b> <br>";
}
```

Une fois le script terminé, enregistrez-le, puis passez dans le Web local pour accéder au formulaire. Saisissez une année dans le champ (1970, par exemple) puis validez (voir figure 18-22). Si votre application fonctionne correctement, la page du script affiche la liste des adhérents répondant à la condition (voir figure 18-23).

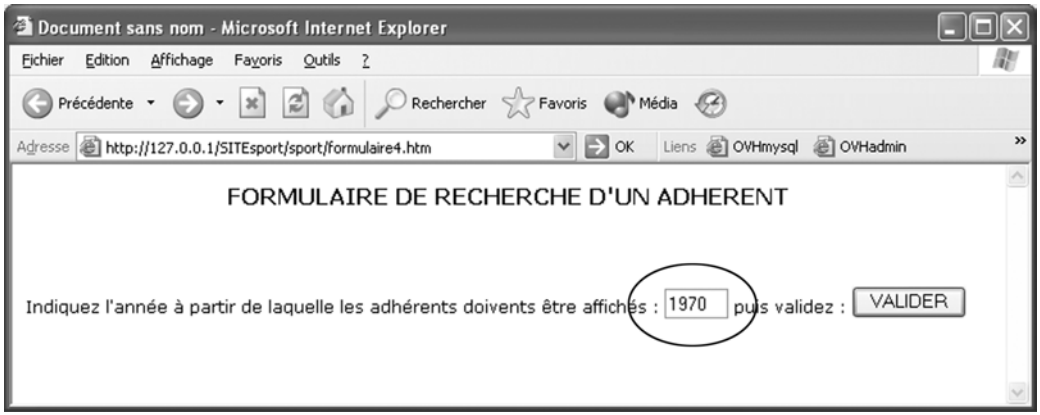


Figure 18-22

Pour tester l'application, affichez le formulaire dans le Web local, saisissez une année dans le champ, puis validez votre choix.

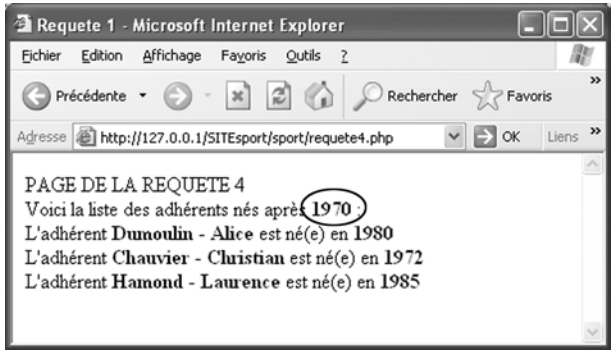


Figure 18-23

Dès réception de la variable HTTP envoyée par le formulaire, le script doit afficher la liste des adhérents dont l'année de naissance est supérieure à la valeur saisie.

Construction d'une requête à partir de variables optionnelles

Ce deuxième exemple sera l'occasion de présenter la technique de construction d'une requête à l'aide d'opérations de concaténation successives conditionnées en fonction de l'existence d'une variable spécifique. Cette fois, les variables utilisées dans les clauses de la requête sont au nombre de trois et peuvent être optionnelles. La table interrogée est la table `adherents` et les trois variables transmises correspondent aux critères suivants :

- sélection selon le cours auquel est inscrit l'adhérent : variable `coursID` ;
- sélection selon la date de naissance de l'adhérent : variable `anneeNaissance` ;
- sélection selon la première lettre du nom de l'adhérent : variable `lettreNom`.

Les variables de sélection ne sont pas transmises par un formulaire mais directement dans l'URL derrière le nom du script (exemple : `requete5.php?coursID=1&anneeNaissance=1972`). Le script `requete5.php` récupère ensuite les variables HTTP (en méthode `GET`) et élabore la requête en fonction de l'existence de chacune d'elles (les variables étant optionnelles).

Pour la création du fichier PHP, utilisez la base du fichier requete4.php précédent, que vous renommerez requete5.php. Cela vous permet de conserver la même structure pour le script et d'apporter uniquement quelques modifications au niveau de la création de la requête et dans la boucle d'affichage des informations.

La ligne d'initialisation du précédent script n'est pas obligatoire car nous allons conditionner l'utilisation des variables lors de la création de la requête.

Les deux lignes destinées à établir la connexion avec le serveur MySQL sont identiques (voir repère 1 de la figure 18-24). En revanche, l'enregistrement de la requête se fait sur quatre lignes (voir repère 2 de la figure 18-24).

La première ligne permet d'initialiser la variable de la requête `$query_rsListeAdherents` avec la partie fixe de la requête SQL. Les trois critères étant optionnels, une première clause `WHERE 1=1` est ajoutée afin que la requête puisse fonctionner si aucune variable n'est passée dans l'URL :

```
$query_rsListeAdherents = "SELECT nom, prenom, anneeNaissance, coursID FROM adherents
➥WHERE 1=1 ";
```

La deuxième ligne permet de conditionner l'ajout de la clause `AND coursID= $_GET['coursID']` à la suite de la requête initiale. Si la variable `$_GET['coursID']` n'est pas passée dans l'URL, cette clause (en gras ci-dessous) n'est pas ajoutée :

```
if(isset($_GET['coursID'])) $query_rsListeAdherents .= " AND coursID=".$_GET['coursID'] ;
```

Les troisième et quatrième lignes fonctionnent sur le même principe mais avec les variables `$_GET['anneeNaissance']` et `$_GET['lettreNom']`.

La ligne qui suit récupère la requête ainsi élaborée et l'envoie au serveur à l'aide de la fonction `mysql_query()` (voir repère 3 de la figure 18-24) :

```
$rsListeAdherents = mysql_query($query_rsListeAdherents, $connexionSport) or die(mysql_error());
```

Si aucun enregistrement ne correspond à la requête envoyée, nous désirons que s'affiche à l'écran le message « AUCUN RESULTAT ». L'affichage de ce message étant conditionné par un nombre de

résultat nul, initialisons dès maintenant une variable à l'aide de la fonction `mysql_num_rows()` afin de préparer la condition du futur test :

```
$nombreResultat=mysql_num_rows($rsListeAdherents);
// Récupère le nombre de résultats
```

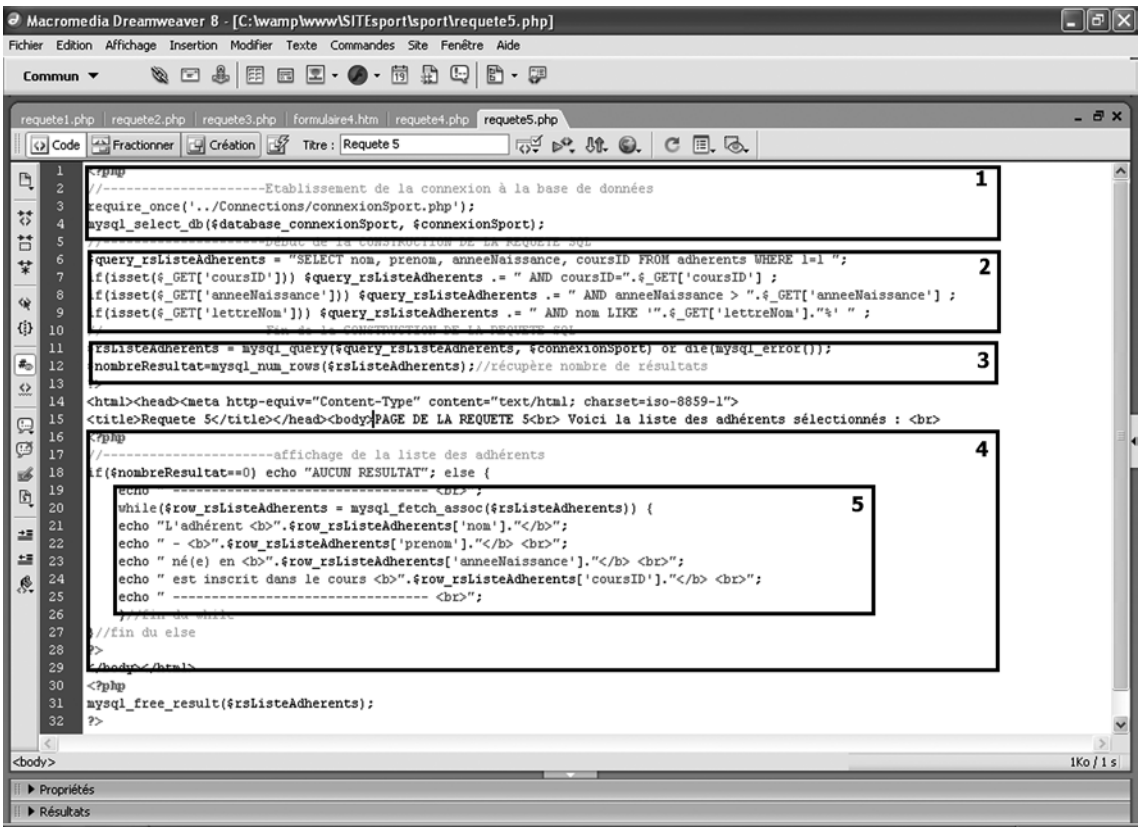


Figure 18-24

Script de la page dynamique requete5.php

À noter

Pour la clause `AND nom LIKE`, la variable et le signe `%` qui la suit doivent être encadrés par des guillemets simples afin de respecter la syntaxe SQL :

```
if(isset($_GET['anneeNaissance'])) $query_rsListeAdherents .=
➥" AND anneeNaissance > ".$_GET['anneeNaissance'] ;
if(isset($_GET['lettreNom'])) $query_rsListeAdherents .=
➥" AND nom LIKE '".$_GET['lettreNom']."%' " ;
```

Le script destiné à afficher les résultats est conditionné afin d'afficher les résultats de la requête uniquement si le nombre d'enregistrement retourné par le serveur est différent de zéro. Dans le cas contraire, un message indique qu'aucun résultat ne correspond à la recherche.

La structure de la boucle `while` est identique à celle de l'application précédente, hormis le fait que nous afficherons le numéro du cours (`coursID`) en complément :

```
//----------------------Affichage de la liste des adhérents
if($nombreResultat==0) echo "AUCUN RESULTAT"; else {
  echo " ---------------------------------- <br>";
  while($row_rsListeAdherents = mysql_fetch_assoc($rsListeAdherents)) {
  echo "L'adhérent <b>".$row_rsListeAdherents['nom']."</b>";
  echo " - <b>".$row_rsListeAdherents['prenom']."</b> <br>";
  echo " né(e) en <b>".$row_rsListeAdherents['anneeNaissance']."</b> <br>";
  echo " est inscrit dans le cours <b>".$row_rsListeAdherents['coursID']."</b> <br>";
  echo " ---------------------------------- <br>";
  } // Fin du while
} // Fin du else
```

Comme dans tous les scripts de requête, ajoutez à la fin une fonction `mysql_free_result()` afin de libérer la mémoire utilisée par le jeu d'enregistrements :

```
mysql_free_result($rsListeAdherents);
```

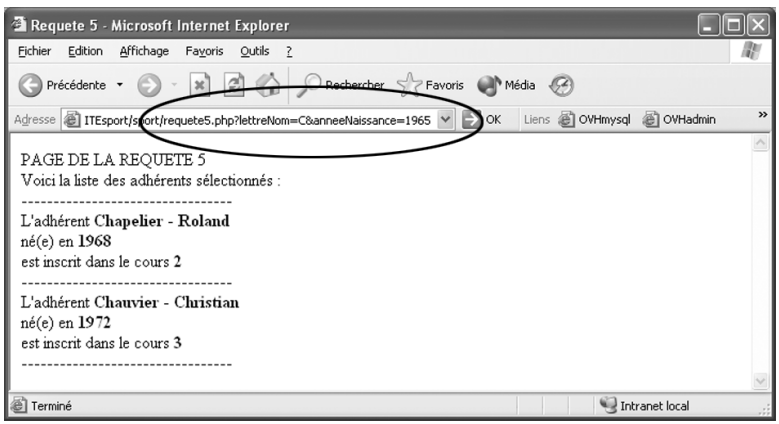
Enregistrez votre script et passez dans le Web local pour le tester. Si vous affichez la page directement sans ajouter de variable dans l'URL, la totalité des adhérents s'affiche. Vous pouvez ensuite saisir différentes combinaisons de variables directement dans l'URL afin de vérifier que le script sélectionne bien les adhérents correspondants (voir figure 18-25).

Exemples de combinaisons de plusieurs critères de sélection à saisir dans l'URL :

```
requete5.php?coursID=3
requete5.php?lettreNom=D
requete5.php?coursID=3&anneeNaissance=1972
requete5.php?coursID=3&lettreNom=C
requete5.php?coursID=1&lettreNom=D&anneeNaissance=1975
```

Figure 18-25

Test de la page dynamique requete5.php



Pages d'administration d'une base de données

Nous vous avons présenté différents exemples utilisant des requêtes SQL (commande `SELECT`). En pratique, vous serez amené à utiliser d'autres commandes SQL pour administrer votre base de données. Voici trois pages dynamiques courantes pour ajouter (commande `INSERT`), modifier (commande `UPDATE`) ou supprimer un enregistrement (commande `DELETE`) (voir figure 18-26). Ces trois pages permettent d'administrer une seule table mais il vous suffit de dupliquer cette structure et de l'adapter aux autres tables si vous souhaitez administrer complètement la base de données.

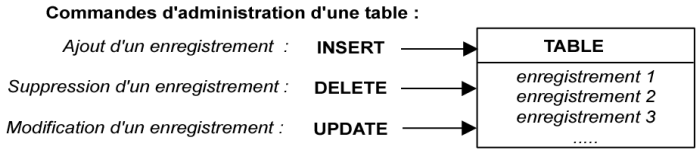


Figure 18-26
Structure d'administration d'une table

Page d'ajout d'un enregistrement

Pour illustrer le fonctionnement de ces trois pages, nous allons administrer la table `adherents`. La page d'ajout d'un nouvel enregistrement est constituée d'un formulaire dont les champs correspondent aux champs de la table et d'un script d'insertion des données dans la table.

1. Créez une nouvelle page dynamique dans Dreamweaver (Fichier>Nouveau>Page dynamique>PHP). Enregistrez cette page sous le nom `adherentsAjout.php` (par la suite, vous pourrez appliquer la même convention de nommage pour les autres tables).
2. Passez en mode création. Ajoutez un titre en haut de la page (PAGE D'AJOUT, par exemple) puis cliquez sur le bouton Formulaire du panneau Formulaire de la barre d'outils Insérer (voir repère 1 de la figure 18-27).
3. Créez à l'intérieur du formulaire un tableau HTML de cinq lignes et deux colonnes (utilisez pour cela le bouton Tableau du panneau Commun de la barre d'outils Insérer). Dans la colonne de gauche, saisissez les libellés correspondant à chaque champ de saisie (voir figure 18-27). Dans la colonne de droite, insérez un champ texte de saisie (utilisez le bouton Champ de texte du panneau Formulaire de la barre d'outils Insérer ; voir repère 2 de la figure 18-27) dans les trois premières lignes et nommez-les respectivement `nom`, `prenom` et `anneeNaissance`. Sélectionnez le champ de la date de naissance et configurez sa largeur et son nombre maximal de caractères à 4 dans le panneau des propriétés.
4. Dans la cellule de la quatrième ligne de la colonne de droite, insérez un élément Liste/Menu (utilisez le bouton du repère 4 de la figure 18-27), puis nommez-le `coursID`. Renseignez ensuite

les valeurs et les étiquettes des options (valeur `1` pour l'étiquette `Débutant`, valeur `2` pour l'étiquette `Intermédiaire` et valeur `3` pour l'étiquette `Perfectionnement`).

5. Dans la dernière ligne, ajoutez un bouton de soumission (utilisez le bouton du repère 5 de la figure 18-27) dans la cellule de droite et un champ caché dans la cellule de gauche (utilisez le bouton du repère 3 de la figure 18-27). Nommez le champ caché `action` et attribuez-lui la valeur `ajout`.
6. Sélectionnez le formulaire (placez votre curseur dans une des cellules du formulaire puis cliquez sur la balise `<form>` dans le sélecteur de balise ; voir repère 6 de la figure 18-27). Une fois le formulaire sélectionné, renseignez le champ `Action` du panneau de propriétés avec le nom du fichier actuel adherentsAjout.php (voir repère 7 de la figure 18-27). Assurez-vous que la méthode `POST` est bien sélectionnée dans le panneau Propriétés.

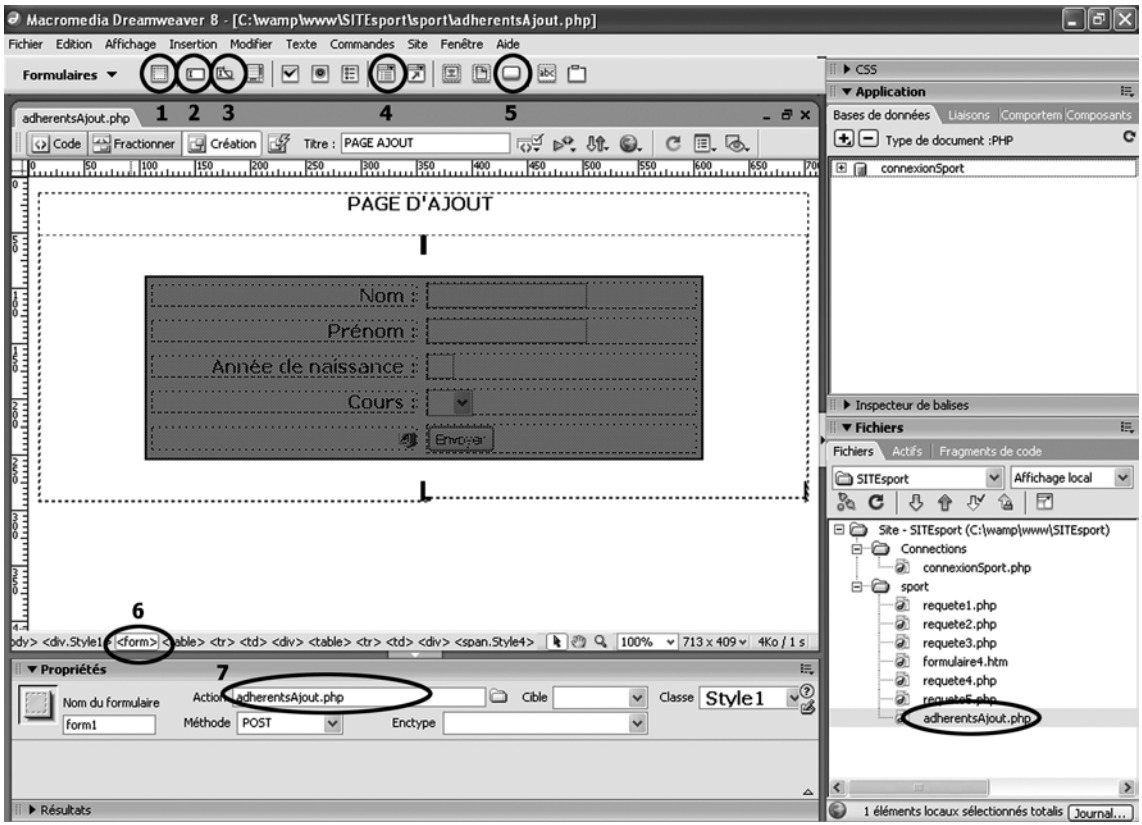


Figure 18-27

Création du formulaire d'ajout dans Dreamweaver (en mode création)

7. Passez maintenant en mode code (en cliquant sur le bouton Code situé en haut et à gauche de la zone de travail ; voir repère 1 de la figure 18-28).

8. Saisissez autant de lignes d'initialisation de variable qu'il y a de champs dans le formulaire sans oublier le champ caché (voir repère 2 de la figure 18-28) :

```
//--------INITIALISATION DES VARIABLES---------
if(isset($_POST['nom']))  $nom=$_POST['nom']; else $nom="";
if(isset($_POST['prenom']))  $prenom=$_POST['prenom']; else $prenom="";
if(isset($_POST['anneeNaissance']))  $anneeNaissance=$_POST['anneeNaissance'];
➥else $anneeNaissance="";
if(isset($_POST['coursID']))  $coursID=$_POST['coursID']; else $coursID="";
if(isset($_POST['action']))  $action=$_POST['action']; else $action="";
```

9. Saisissez les deux lignes qui permettent de vous connecter au serveur et de sélectionner la base (voir repère 3 de la figure 18-28) :

```
//--------CONNEXION ET SÉLECTION DE LA BASE ------------
require_once('../Connections/connexionSport.php');
mysql_select_db($database_connexionSport, $connexionSport);
```

10. Construisez une structure de test conditionnée par l'égalité `action=="ajout"` (information du champ caché du formulaire) afin de déterminer si la page est appelée suite à l'envoi du formulaire (voir repère 4 de la figure 18-28) :

```
//--------TESTE SI ENVOI DEPUIS FORMULAIRE
if($action=="ajout") {
// Ici le bloc conditionné
}
```

11. Dans le bloc conditionné par la structure de test, saisissez l'expression d'élaboration de la requête SQL (voir repère 5 de la figure 18-28), suivie de l'instruction de soumission de la requête :

```
//------------------------REQUÊTE SQL
      $insertAdherents = "INSERT  INTO adherents SET
      nom='".$nom."',
      prenom='".$prenom."',
      anneeNaissance='".$anneeNaissance."',
      coursID=".$coursID."" ;
      //-----------------------SOUMISSION REQUÊTE

      mysql_query($insertAdherents, $connexionSport) or die(mysql_error());
```

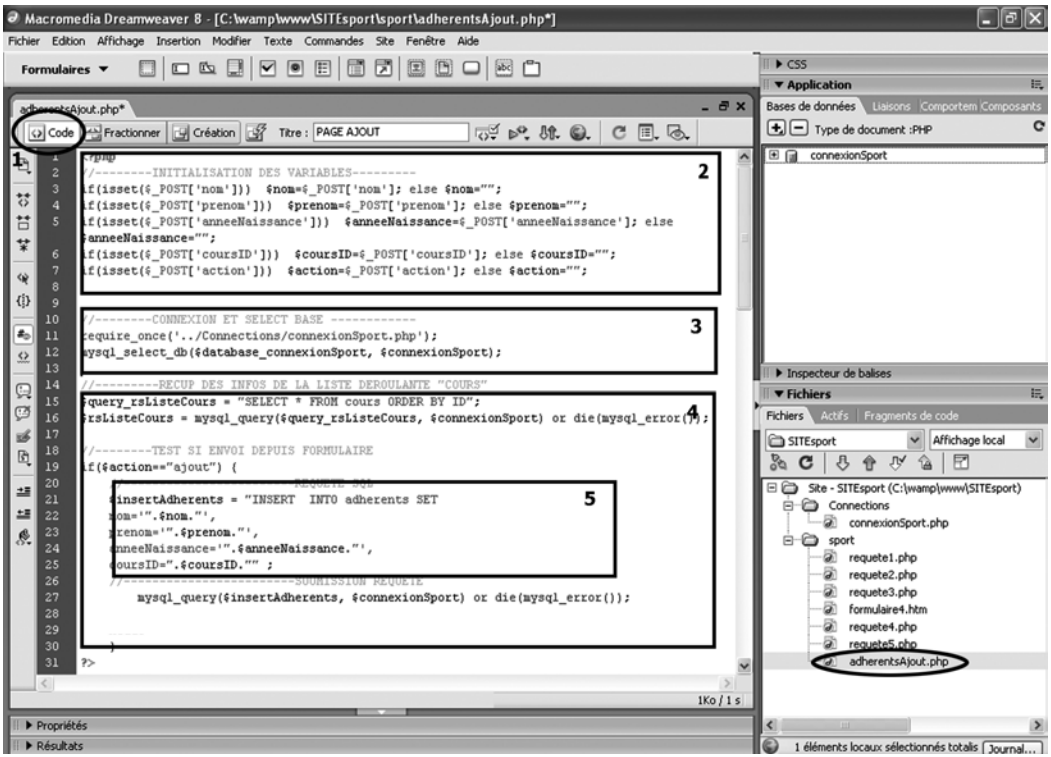


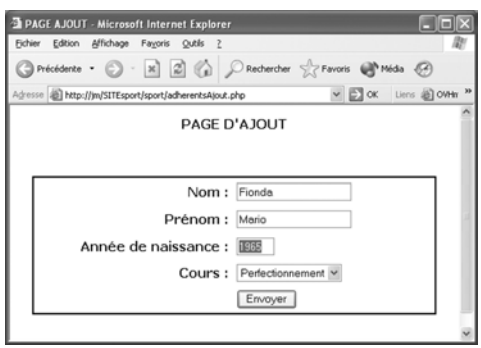
Figure 18-28

Modification du code de la page adherentsAjout.php

12. Enregistrez le fichier et passez en Web local pour tester son fonctionnement (voir figure 18-29).
13. Remplissez le formulaire en saisissant des informations plausibles (notamment en ce qui concerne l'année de naissance) et cliquez sur le bouton pour valider votre saisie (voir figure 18-29). Après validation, vous pouvez vérifier avec phpMyAdmin que les données ont bien été enregistrées dans la table `adherents` (voir figure 18-30).

Figure 18-29

Test du formulaire d'ajout d'un adhérent depuis le Web local



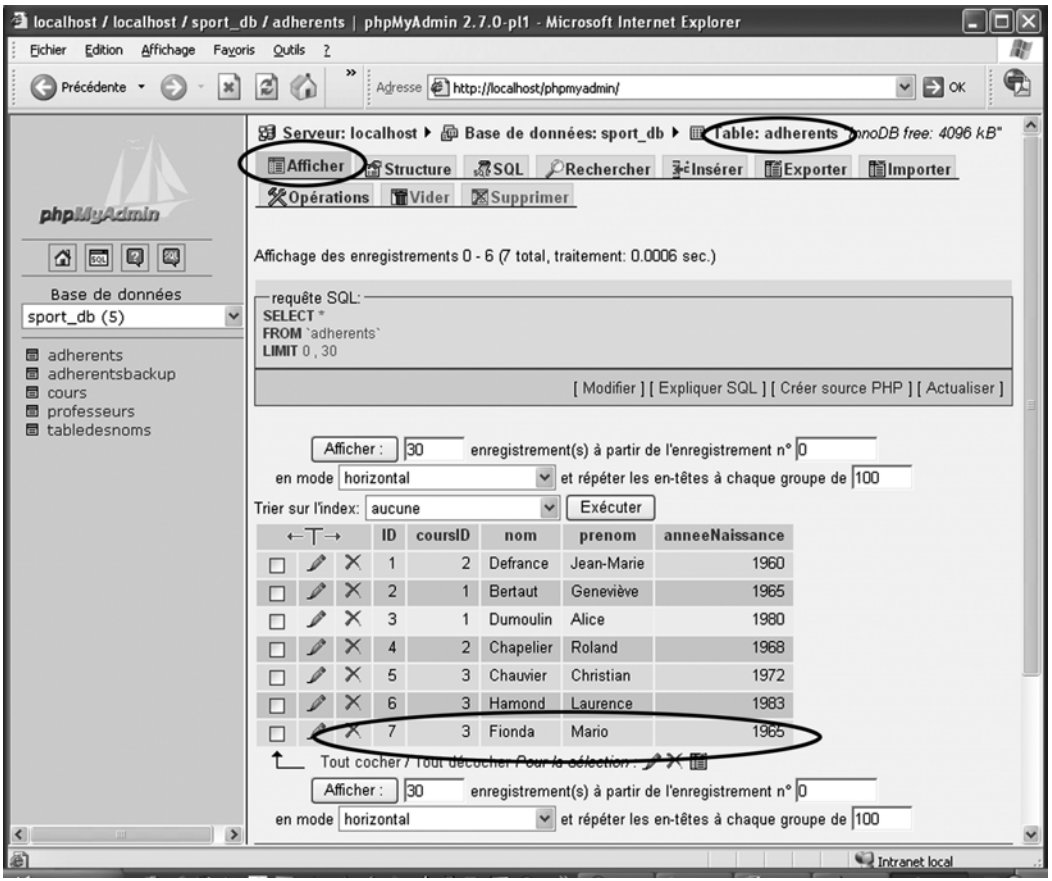


Figure 18-30

Après un ajout, vous pouvez vérifier que le nouvel adhérent a bien été ajouté dans la table adherents avec phpMyAdmin

Affichage dynamique du menu (option)

Actuellement, le menu déroulant (correspondant aux niveaux de cours) est configuré directement dans le code HTML de la page. Cependant, si l'on ajoute un quatrième niveau de cours, il faudra intervenir sur cette page pour modifier les options de l'objet de formulaire `select`. En pratique, il est préférable que ce menu soit dynamique afin que les options qu'il affiche s'actualisent automatiquement dès qu'un nouvel enregistrement est ajouté dans la table `cours`. Voici les modifications à effectuer pour ce faire :

1. Ouvrez le fichier `adherentsAjout.php` et ajoutez les instructions en gras dans le script ci-dessous après les deux lignes qui établissent la connexion et sélectionnent la base. Ces instructions complémentaires permettent de créer un nouveau jeu d'enregistrements `$rsListeCours` qui contient les colonnes `ID` et `niveau` de la table `cours` :

```
//--------CONNEXION ET SÉLECTION DE LA BASE ------------
require_once('../Connections/connexionSport.php');
```

```
mysql_select_db($database_connexionSport, $connexionSport);

//---------RÉCUPÉRATION DES INFORMATIONS DE LA LISTE DÉROULANTE "COURS"
$query_rsListeCours = "SELECT ID,niveau FROM cours ORDER BY ID";
$rsListeCours = mysql_query($query_rsListeCours, $connexionSport) or die(mysql_error());
```

2. Descendez plus bas dans la page au niveau du code de l'objet `select`. Nous désirons remplacer les valeurs et les étiquettes des `options` par les données issues du jeu d'enregistrements créé précédemment. Afin que le nombre d'option corresponde au nombre d'enregistrements dans la table `cours`, intégrez les balises de l'option dans une boucle `while` comme nous l'avons fait dans un précédent exemple de requête.
3. Remplacez le code de l'objet rappelé ci-dessous :

```
<select name="coursID" id="coursID">
<option value="1">D&eacute;butant</option>
<option value="2">Interm&eacute;diaire</option>
<option value="3">Perfectionnement</option>
</select>
```

par celui-ci :

```
<select name="coursID" id="coursID">
<?php while($row_rsListeCours = mysql_fetch_assoc($rsListeCours)) { ?>
<option value="<?php echo $row_rsListeCours['ID']; ?>"><?php echo $row_rsListeCours
➥['niveau']; ?></option>
<?php } ?>
</select>
```

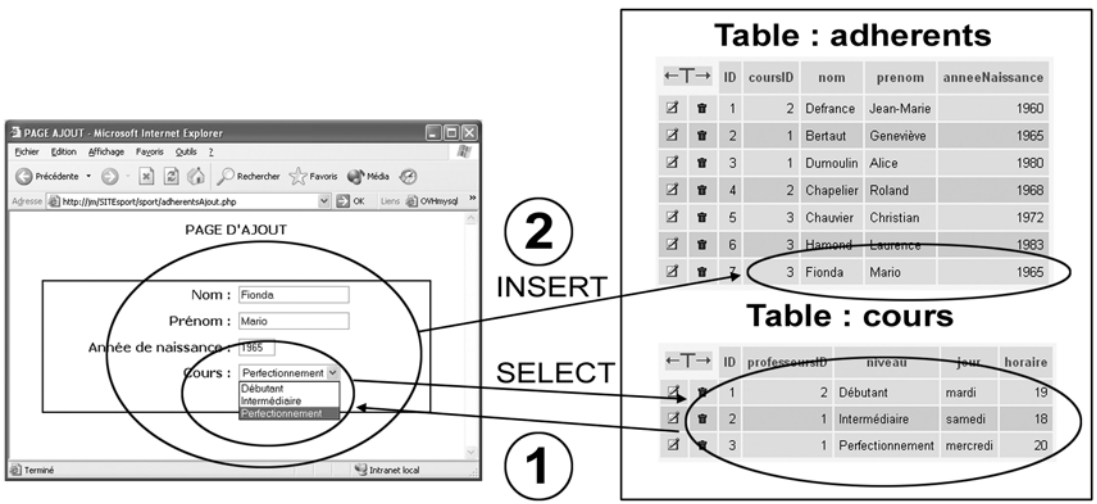


Figure 18-31

Schéma de principe illustrant la requête SELECT destinée à configurer dynamiquement le menu déroulant à partir de la table cours (repère 1) et la commande INSERT qui permet d'ajouter un nouvel enregistrement dans la table adherents (repère 2).

4. Enregistrez votre fichier et testez-le dans le Web local. Le fonctionnement du menu doit être identique à celui des tests précédents. Cependant, si vous ajoutez un enregistrement dans la table `cours` (avec phpMyAdmin, par exemple), il doit apparaître automatiquement dans la liste des options du menu.

Enregistrement dynamique d'un fichier (option)

Il faut souvent gérer des images ou des fichiers PDF dynamiquement. Dans ce cas, il faut mémoriser l'URL du document dans la base de données et enregistrer le fichier lors de la procédure d'ajout d'un enregistrement. Pour illustrer cette technique, nous vous proposons de modifier le formulaire précédent afin d'inclure la photo de l'adhérent :

1. À l'aide du gestionnaire phpMyAdmin, ajoutez un champ supplémentaire nommé `photo` dans la table `adherents` après le champ `prenom` (type `VARCHAR` de 200 caractères au maximum) (revoir si besoin le chapitre 16).
2. Dans le fichier `adherentsAjout.php`, ajoutez les lignes de code suivantes afin d'enregistrer le document dans un répertoire dédié (par exemple `/photos/`). Attention ! si vous utilisez ce script sur votre serveur distant, modifiez au préalable les droits du répertoire avec CHMOD. Dans ce script, l'action de copier le fichier est conditionnée par une structure de choix `if` qui teste la sélection préalable d'un fichier dans le formulaire (à l'aide du bouton Parcourir). Si aucun fichier n'est sélectionné, le nom d'une photo par défaut sera affecté à la variable `$nomphoto` :

```
if($action=="ajout") {
      if ($_FILES['photo']['name']!="")
      {
      $nomphoto='photo_'.$nom.'.jpg';
      copy($_FILES['photo']['tmp_name'] ,'../photos/'.$nomphoto);
      }else{
      $nomphoto='photo_defaut.jpg';
      }
```

3. À la suite de ce premier script, modifiez la requête afin d'ajouter le couple champ/variable correspondant au nom de la photo (voir code en gras ci-dessous). Attention ! dans la requête SQL, n'oubliez pas d'encadrer la variable `"$nomphoto"` avec des guillemets simples car l'URL de la photo est considérée comme un champ texte :

```
//------------------------REQUÊTE SQL
      $insertAdherents = "INSERT INTO adherents SET
      nom='".$nom."',
      prenom='".$prenom."',
      anneeNaissance='".$anneeNaissance."',
      coursID=".$coursID.",
      photo='".$nomphoto."' ";
      //------------------------SOUMISSION REQUETE
      mysql_query($insertAdherents, $connexionSport) or die(mysql_error());
      header("Location:adherentsGestion.php");
      } // Fin du if($action)
```

4. Dans le formulaire de cette même page, ajoutez un champ de fichier et nommez-le `photo` :

```
<input name="photo" type="file" id="photo">
```

5. Enregistrez votre page et testez son fonctionnement depuis le Web local.

Page de suppression d'un enregistrement

Pour gérer les suppressions d'enregistrements dans une table, on peut afficher un premier formulaire destiné à sélectionner l'enregistrement à supprimer puis envoyer une commande de suppression à la base de données. Une autre solution consiste à afficher la liste de tous les enregistrements de la table concernée dans un tableau HTML. Au bout de chaque ligne d'enregistrement, un lien de suppression paramétré dynamiquement permet d'appeler un script générant une commande `DELETE`. Ainsi, si l'on clique sur le lien d'un enregistrement, une commande de suppression est envoyée à la base de données et supprime l'enregistrement concerné. Voici la procédure pour créer une page de suppression selon cette deuxième méthode :

1. Ouvrez un nouveau document PHP et enregistrez-le sous le nom `adherentsGestion.php`.

> **À noter**
>
> Utilisez le suffixe `Gestion` et non `Suppression` pour nommer ce fichier car cette page sera également utilisée pour appeler le formulaire de modification présenté ci-dessous.

2. Passez en mode Création afin de créer la structure du tableau HTML utilisé pour afficher dynamiquement les différents enregistrements de la table `adherents`. Créez un tableau de deux lignes et cinq colonnes (utilisez le bouton Tableau du panneau Commun de la barre d'outils Insérer ; voir le repère 1 de la figure 18-32). Une fois le tableau créé, nommez les têtes de quatre colonnes : `Nom`, `Prénom`, `Supp.` et `Modif.` (la colonne Modif. sera utilisée dans l'exemple de page suivant).

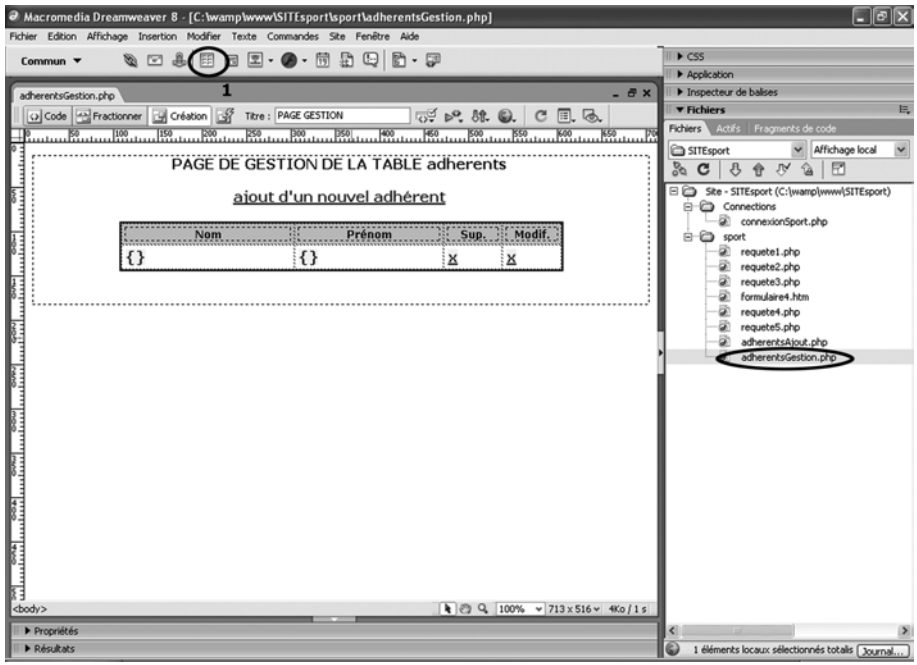


Figure 18-32

Création du tableau HTML pour la page adherentsGestion.php avec Dreamweaver en mode création

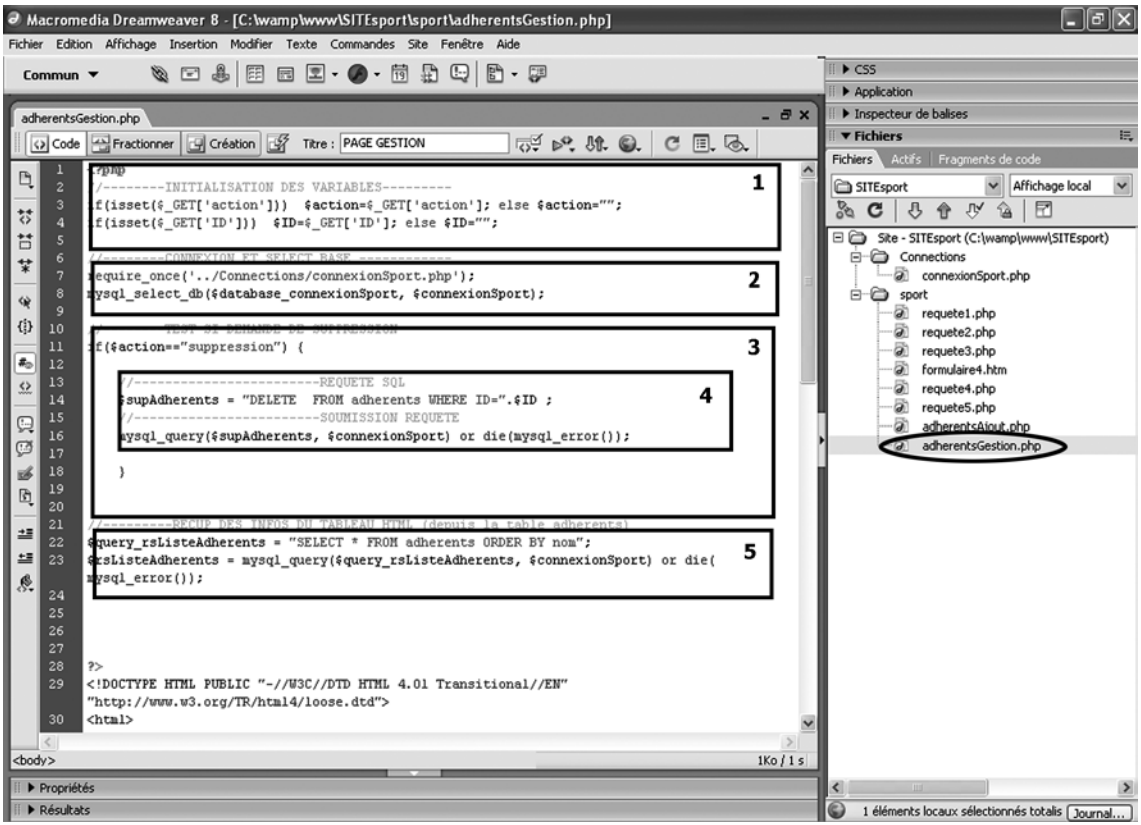


Figure 18-33

Script du haut de la page adherentsGestion.php

Au-dessus du tableau, ajoutez un lien hypertexte afin de pouvoir appeler le formulaire d'ajout d'un nouvel adhérent depuis cette page. Enfin, en haut de la page, placez le titre `PAGE DE GESTION DE LA TABLE adherents` (voir figure 18-32).

1. Passez en mode code et positionnez votre curseur en haut de la page (au-dessus de la balise `<html>`). Ajoutez une balise d'ouverture de script PHP suivie des deux lignes de code (voir repère 1 de la figure 18-33) destinées à initialiser les variables `ID` et `action` transmises par l'URL (méthode `GET`) :

```
<?php
//--------INITIALISATION DES VARIABLES---------
if(isset($_GET['action']))  $action=$_GET['action']; else $action="";
if(isset($_GET['ID']))  $ID=$_GET['ID']; else $ID="";
```

2. Ajoutez les deux instructions qui permettent d'établir une connexion avec le serveur MySQL et de sélectionner la base de données `sport_db` (voir repère 2 de la figure 18-33) :

```
//--------CONNEXION ET SÉLECTION DE LA BASE ------------
require_once('../Connections/connexionSport.php');
mysql_select_db($database_connexionSport, $connexionSport);
```

3. Créez une structure de choix conditionnée par l'expression `$action=="suppression"` (voir repère 3 de la figure 18-33). Vous pourrez ainsi déterminer si la page est appelée suite à un clic sur un lien de suppression (`adherentsGestion.php?action=suppression&ID=3` par exemple) :

```
if($action=="suppression") {
//bloc conditionné
}
```

4. Entre les accolades du bloc conditionné, placez les deux lignes de code qui permettent de construire la requête SQL de suppression (paramétrée avec le `ID` de l'enregistrement à supprimer) suivies de l'instruction de soumission à la base de données (voir repère 4 de la figure 18-33) :

```
//------------------------REQUÊTE SQL
      $supAdherents = "DELETE  FROM adherents WHERE ID=".$ID ;
      //------------------------SOUMISSION REQUÊTE
      mysql_query($supAdherents, $connexionSport) or die(mysql_error());
```

5. Après l'accolade de fermeture du bloc conditionné, ajoutez les deux lignes de code qui permettent de créer le jeu d'enregistrements destiné à récupérer toutes les données de la table `adherents` (voir repère 5 de la figure 18-33). Ce jeu d'enregistrements sera ensuite utilisé dans la construction dynamique du tableau HTML affichant la liste des adhérents :

```
//---------RECUPÉRATION DES INFORMATIONS DU TABLEAU HTML (depuis la table adherents)
$query_rsListeAdherents = "SELECT * FROM adherents ORDER BY nom";
$rsListeAdherents = mysql_query($query_rsListeAdherents, $connexionSport) or
die(mysql_error());
```

6. Fermez cette première zone de script PHP en ajoutant la balise `?>` puis descendez dans le code de la page et placez votre curseur sur la deuxième ligne du tableau.

7. Dans les deux premières cellules de la seconde ligne, ajoutez deux inclusions PHP afin d'afficher le nom et le prénom de l'adhérent dans chacune des cellules (voir repère 2 de la figure 18-34). Dans la cellule suivante, ajoutez un lien hypertexte dont le paramètre d'URL `ID` sera configuré avec l'élément du jeu d'enregistrements correspondant :

```
<tr>
<td><?php echo $row_rsListeAdherents['nom']; ?></td>
<td><?php echo $row_rsListeAdherents['prenom']; ?></td>
<td>
<a href="adherentsGestion.php?action=suppression&ID=<?php echo $row_rsListeAdherents
➥['ID']; ?>">x</a>
```

```
</td>
<td>x</td>
</tr>
```

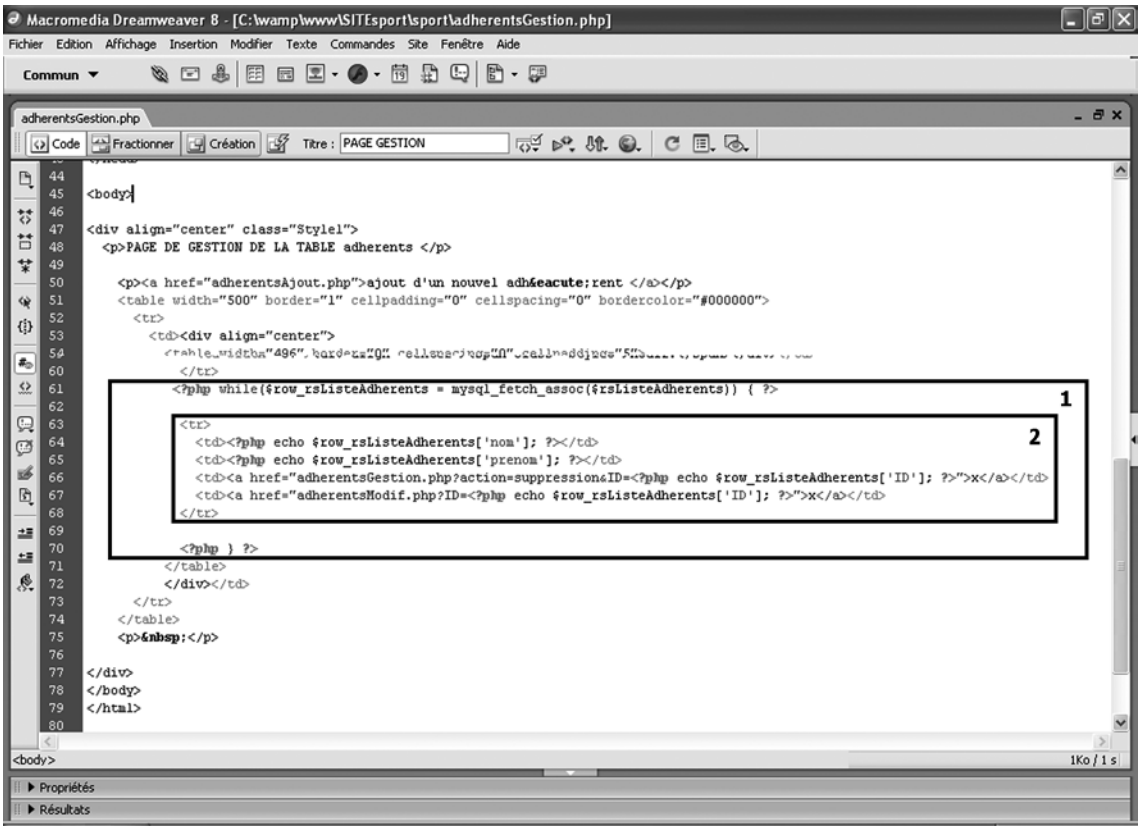


Figure 18-34

Script inséré dans la structure du tableau HTML de la page adherentsGestion.php

8. Placez votre curseur au-dessus de la balise de cette deuxième ligne `<tr>` et ajoutez le script de la boucle `while` afin de générer autant de lignes qu'il y aura de résultats retournés dans le jeu d'enregistrements `$row_rsListeAdherents`. Placez ensuite votre curseur en dessous de la balise de fin de cette ligne (`</tr>`) et ajoutez un deuxième script afin d'indiquer, par la présence d'une accolade fermante, la fin du bloc à dupliquer (voir repère 1 de la figure 18-34).
9. Enregistrez la page `adherentsGestion.php`. Ouvrez la page `adherentsAjout.php` afin d'ajouter un script qui permette de retourner automatiquement à la page de gestion dès qu'un nouvel adhérent est ajouté dans la table. Pour cela, placez votre curseur à la fin du bloc conditionné par l'expression

`$action=="ajout"` et ajoutez une instruction de redirection (`header("Location:adherentsGestion.php")`) à la fin du bloc (voir la ligne en gras dans le script ci-dessous). Enregistrez ensuite les modifications effectuées dans cette page :

```
//--------TESTE SI ENVOI DEPUIS FORMULAIRE
if($action=="ajout") {
      //------------------------REQUÊTE SQL
      $insertAdherents = "INSERT  INTO adherents SET
      nom='".$nom."',
      prenom='".$prenom."',
      anneeNaissance='".$anneeNaissance."',
      coursID=".$coursID."" ;
      //------------------------SOUMISSION REQUÊTE
            mysql_query($insertAdherents, $connexionSport) or die(mysql_error());
      //------------------------RETOUR À LA PAGE DE GESTION APRÈS L'AJOUT
      header("Location:adherentsGestion.php");
      }
```

10. Passez dans le Web local (voir figure 18-35). Cliquez sur le lien d'ajout d'un nouvel adhérent (au-dessus du tableau HTML) et saisissez les informations du formulaire afin de créer un adhérent. Après validation du formulaire d'ajout, vous devez être redirigé vers la page de gestion. Le nouvel adhérent doit apparaître dans la liste de la page de gestion de la table `adherents`. Cliquez sur le lien Supp. correspondant à ce nouvel adhérent (voir figure 18-35) et assurez-vous qu'il disparaît bien de la liste des adhérents lors de l'actualisation de la page `adherentsGestion.php`.

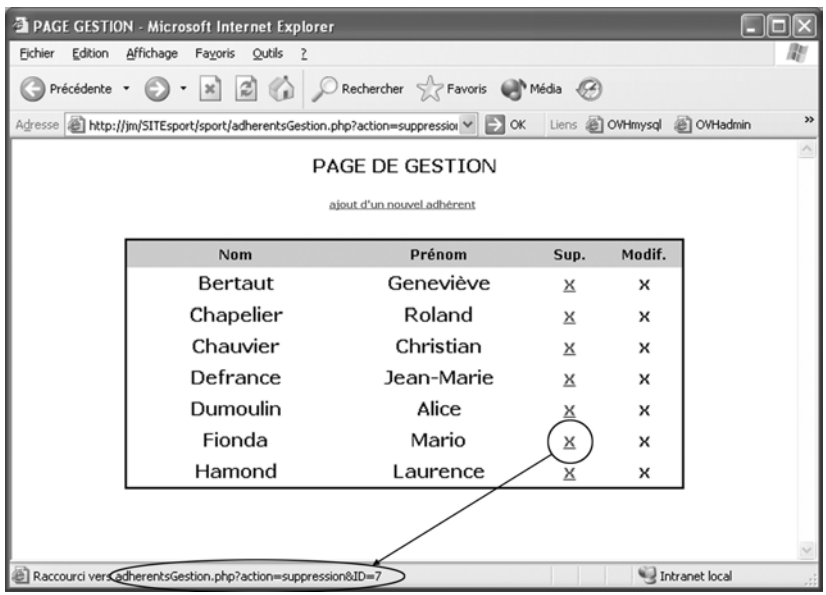


Figure 18-35

Test de la fonction de suppression de la page adherentsGestion.php

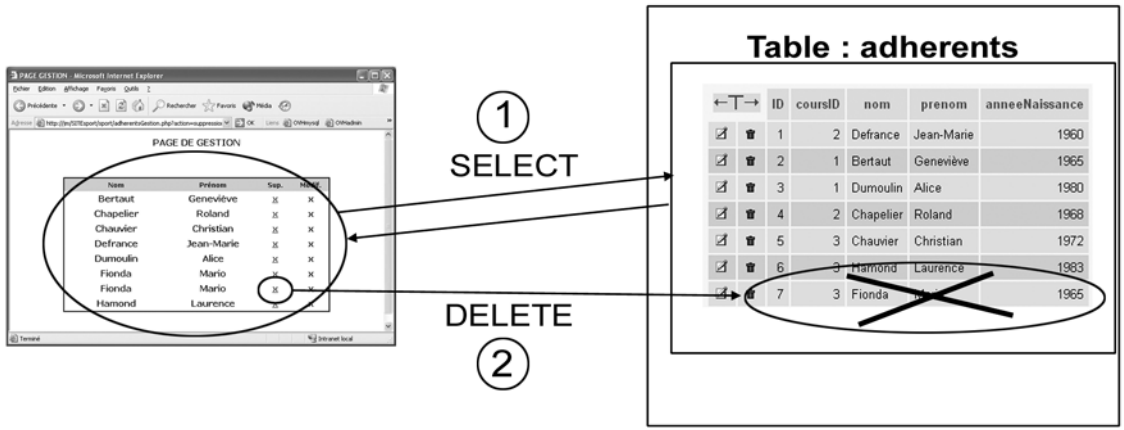


Figure 18-36

Schéma de principe illustrant la suppression d'un enregistrement à partir du clic sur un lien intégré dans un tableau HTML

Page de modification d'un enregistrement

La troisième page est destinée à modifier les enregistrements d'une table. La structure du formulaire utilisé est identique à celle du formulaire de la page d'ajout, hormis le fait que chaque champ doit être initialisé avec la valeur de la colonne correspondante issue de la table et que la requête envoyée n'est pas une commande d'insertion (`INSERT`) mais une commande de modification (`UPDATE`).

1. Pour éviter une saisie de code inutile, nous vous proposons d'ouvrir la page `adherentsAjout.php` et de l'enregistrer sous le nom `adherentsModif.php`.

2 Passez en mode Création et ajoutez un second champ masqué nommé `IDform` (voir figure 18-37) à côté du premier champ masqué dans le formulaire (utilisez le bouton Champ masqué du panneau Formulaire de la barre d'outils Insérer). La valeur initiale de ce champ sera configurée dynamiquement par la suite. Cliquez sur le champ masqué `action` et remplacez sa valeur par `modif`.

3. Passez en mode Code et placez-vous en haut du code de la page. Ajoutez dans la page les deux lignes en gras du script ci-dessous (voir le repère 1 de la figure 18-38). La première ligne permet de récupérer la valeur de l'identifiant `ID` envoyée dans l'URL depuis la page d'administration (en méthode `GET`) lorsque l'utilisateur clique sur le lien Modif. La seconde ligne permet de récupérer ce même identifiant lors de la soumission du formulaire de modification (méthode `POST` depuis le champ caché `IDform`).

À noter

Pour éviter tout conflit entre ces deux variables, nous avons utilisé un nom différent pour chacune d'elles.

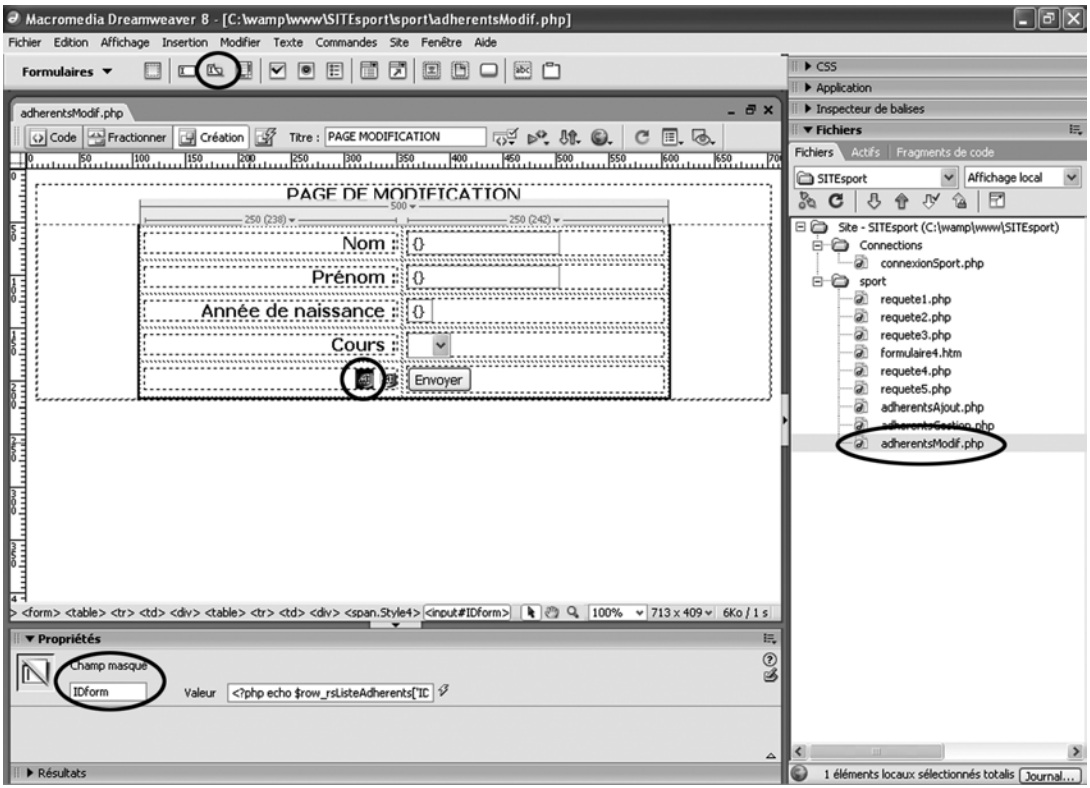


Figure 18-37

Ajout du champ masqué IDform dans le formulaire de modification d'un enregistrement

La variable de cet identifiant est ensuite intégrée dans la clause `WHERE` de la requête de mise à jour afin de spécifier l'enregistrement qui doit être modifié.

```
//--------INITIALISATION DES VARIABLES---------
//---------INFORMATION ENVOYÉE PAR URL (GET)
if(isset($_GET['ID']))  $ID=$_GET['ID']; else $ID=0;
//---------INFORMATION ENVOYÉE PAR LE FORMULAIRE (POST)
if(isset($_POST['nom']))  $nom=$_POST['nom']; else $nom="";
if(isset($_POST['prenom']))  $prenom=$_POST['prenom']; else $prenom="";
if(isset($_POST['anneeNaissance']))  $anneeNaissance=$_POST['anneeNaissance'];
➥else $anneeNaissance="";
if(isset($_POST['coursID']))  $coursID=$_POST['coursID']; else $coursID="";
if(isset($_POST['action']))  $action=$_POST['action']; else $action="";
if(isset($_POST['IDform']))  $IDform=$_POST['IDform']; else $IDform=0;
```

4. Les deux lignes destinées à établir la connexion avec le serveur MySQL et à sélectionner la base de données sont conservées à l'identique. Il en est de même pour la requête du jeu d'enregistrements `$rsListeCours` utilisée pour afficher les différents niveaux de cours dans le menu déroulant `Cours` (voir le repère 2 de la figure 18-38) :

```
//--------CONNEXION ET SÉLECTION DE LA BASE ------------
require_once('../Connections/connexionSport.php');
mysql_select_db($database_connexionSport, $connexionSport);
//---------RECUPÉRATION DES INFORMATIONS DE LA LISTE DÉROULANTE "COURS"
$query_rsListeCours = "SELECT * FROM cours ORDER BY ID";
$rsListeCours = mysql_query($query_rsListeCours, $connexionSport) or
➥die(mysql_error());
```

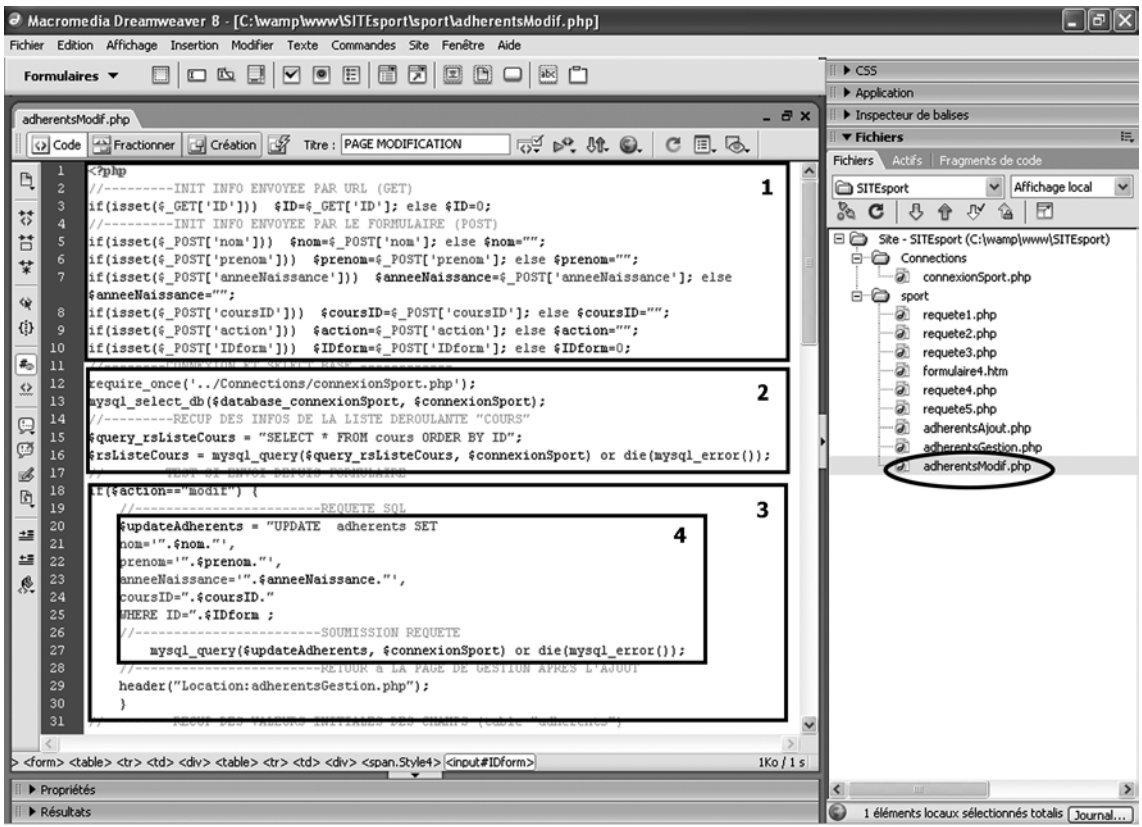


Figure 18-38

Début du script de la page adherentsModif.php

5. Sous ces lignes, la condition de la structure de choix est remplacée par `$action=="modif"` (la variable utilisée est issue du champ masqué `action` du formulaire ; voir le repère 3 de la figure 18-38) afin de détecter si le formulaire de modification a bien été envoyé. De même, la requête d'insertion est transformée en requête de mise à jour comme le montre le script ci-dessous. L'instruction de soumission doit être adaptée au nouveau nom de la requête (soit `$updateAdherents` ; voir repère 4 de la figure 18-38). Enfin, l'instruction de redirection après l'envoi de la commande vers la page `adherentsGestion.php` est conservée à l'identique (voir le repère 3 de la figure 18-38) :

```
//--------TESTE SI ENVOI DEPUIS FORMULAIRE
if($action=="modif") {
      //------------------------REQUETE SQL
      $updateAdherents = "UPDATE  adherents SET
      nom='".$nom."',
      prenom='".$prenom."',
      anneeNaissance='".$anneeNaissance."',
      coursID=".$coursID."
      WHERE ID=".$IDform ;
      //------------------------SOUMISSION REQUÊTE
            mysql_query($updateAdherents, $connexionSport) or die(mysql_error());
      //-------------------RETOUR À LA PAGE DE GESTION APRÈS L'AJOUT
      header("Location:adherentsGestion.php");
      }
```

6. Ajoutez à la suite de la structure de choix les lignes de code suivantes afin de disposer du jeu d'enregistrements `$rsListeAdherents`. Celui-ci sera ensuite utilisé pour initialiser les différents champs du formulaire :

```
//---------RECUPÉRATION DES VALEURS INITIALES DES CHAMPS
$query_rsListeAdherents = "SELECT * FROM adherents WHERE ID=".$ID." ORDER BY ID";
$rsListeAdherents = mysql_query($query_rsListeAdherents, $connexionSport)
➥or die(mysql_error());
$row_rsListeAdherents = mysql_fetch_assoc($rsListeAdherents);
```

7. Descendez dans le code et placez votre curseur au niveau du formulaire de modification (voir les repères 1, 2, 3, 4 et 5 correspondant aux cinq lignes du formulaire dans la figure 18-39). Insérez dans chaque balise de champ un attribut `value` dont la valeur est initialisée dynamiquement à l'aide du jeu d'enregistrements `$row_rsListeAdherents` (voir les zones en gras dans le script ci-dessous). L'initialisation du menu déroulant est particulier car il faut générer le mot-clé `selected` dans la balise option si la valeur de l'option correspond à celle qui est mémorisée dans la base de données (voir le repère 4 de la figure 18-39) :

```
<form action="adherentsModif.php" method="post" name="form1">
<table width="500" border="1" cellpadding="0" cellspacing="0" bordercolor="#000000">
```

```
<tr><td><div align="center">
<table width="500" border="0" cellspacing="0" cellpadding="5">
<tr><td width="250"><div align="right"><span class="Style4">Nom : </span></div></td>
<td width="250"><div align="left">
<input name="nom" type="text" id="nom"
value="<?php echo $row_rsListeAdherents['nom']; ?>">
</div></td></tr>
<tr><td><div align="right"><span class="Style4">Pr&eacute;nom : </span></div></td>
<td><div align="left">
<input name="prenom" type="text" id="prenom"
value="<?php echo $row_rsListeAdherents['prenom']; ?>">
</div></td></tr>
<tr><td><div align="right"><span class="Style4">Ann&eacute;e de naissance : </span></
div></td>
<td><div align="left">
<input name="anneeNaissance" type="text" id="anneeNaissance" size="4" maxlength="4"
value="<?php echo $row_rsListeAdherents['anneeNaissance']; ?>">
</div></td></tr>
<tr><td><div align="right"><span class="Style4">Cours : </span></div></td>
<td><div align="left">
<select name="coursID" id="coursID">
<?php while($row_rsListeCours = mysql_fetch_assoc($rsListeCours)) { ?>
<option value="<?php echo $row_rsListeCours['ID']; ?>"
<?php if($row_rsListeCours['ID']==$row_rsListeAdherents['coursID']) echo "selected";
?> >
<?php echo $row_rsListeCours['niveau']; ?></option>
<?php } ?>
</select>
</div></td></tr>
<tr><td><div align="right"><span class="Style4">
<input name="IDform" type="hidden" id="IDform"
value="<?php echo $row_rsListeAdherents['ID']; ?>">
<input name="action" type="hidden" id="action" value="modif">
</span></div></td>
<td><div align="left">
<input type="submit" name="Submit" value="Envoyer">
</div></td></tr>
</table>
</div></td></tr>
</table>
</form>
```

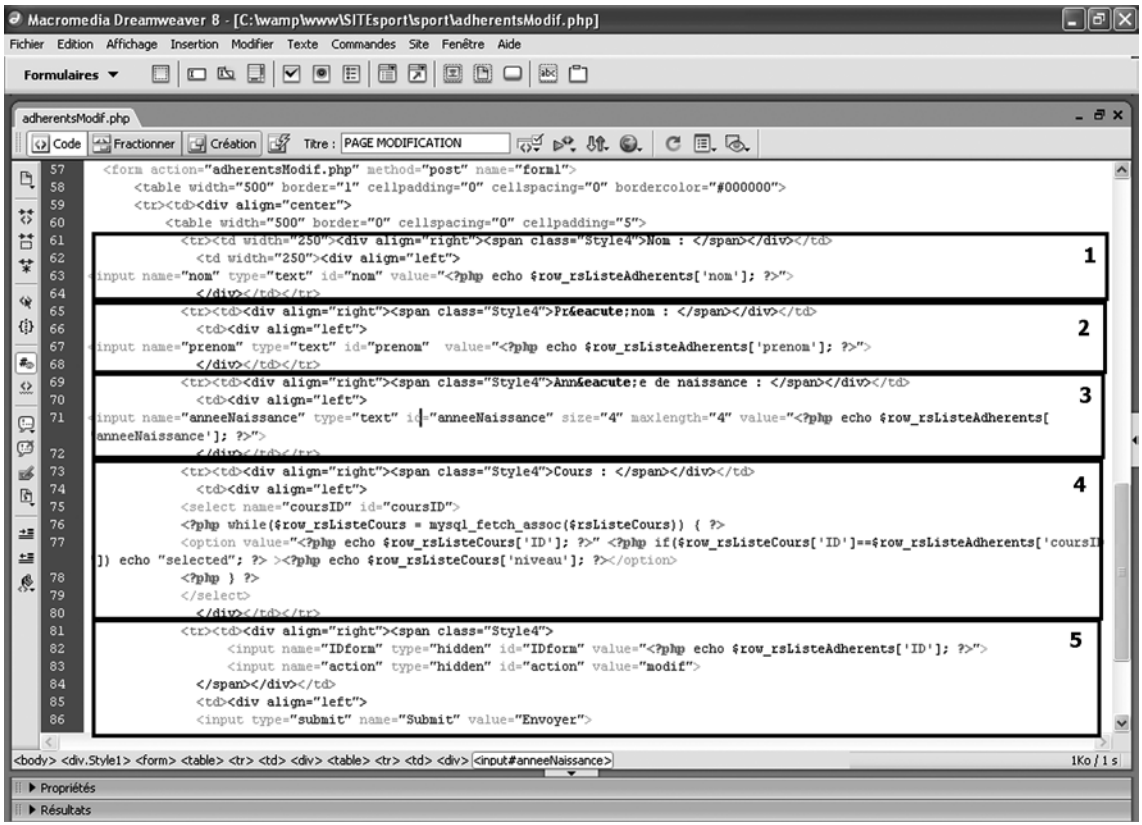


Figure 18-39

Code source du formulaire de modification

8. Enregistrez la page et passez dans le Web local. Affichez la page `adherentsGestion.php` puis cliquez sur un lien de la colonne Modif. Le formulaire de modification s'affiche. Ses champs sont initialisés selon les valeurs actuellement mémorisées dans la base de données (voir figure 18-40). Modifiez le champ de votre choix et cliquez sur le bouton de validation. Après l'envoi de la requête, vous devez être redirigé vers la page `adherentsGestion.php`. Cliquez de nouveau sur le lien Modif. du même adhérent pour vous assurer que la modification a bien été effectuée.

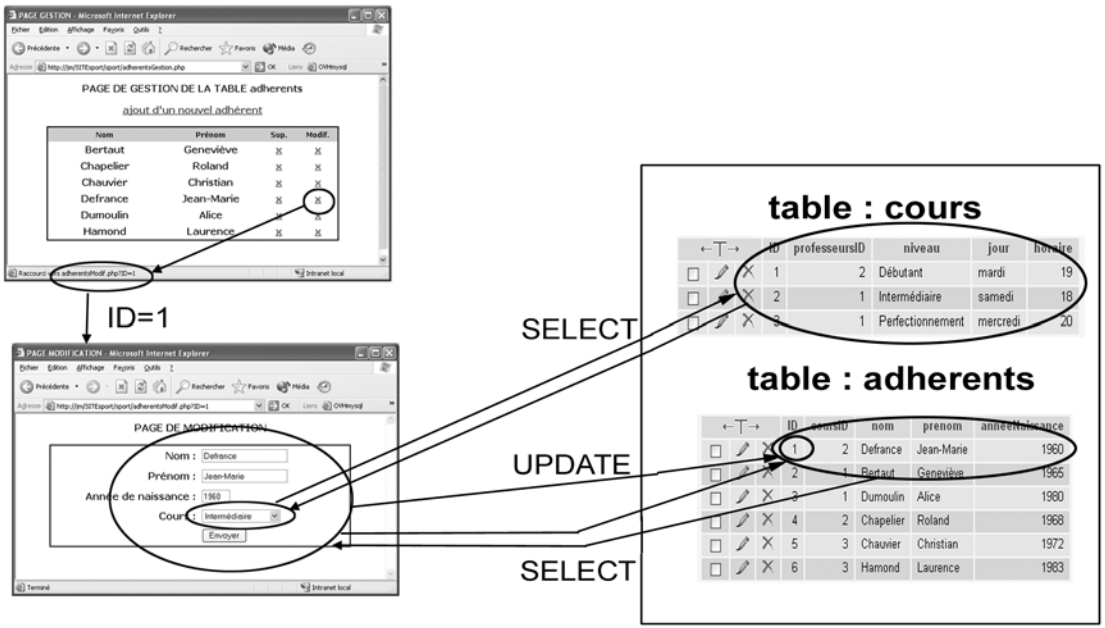


Figure 18-40

Schéma de principe du fonctionnement du système de modification d'un enregistrement

Modification dynamique d'un fichier (option)

Dans le formulaire d'ajout, nous vous avons proposé d'intégrer un champ `photo` dans la table afin de mémoriser le fichier de la photo de l'adhérent. Voici les modifications à apporter à votre formulaire de modification si vous souhaitez modifier ce fichier. Nous supposons que la structure de la base de données a déjà été modifiée et que le champ `photo` est présent dans la table `adherents`.

1. Comme dans le formulaire d'ajout, il faut ajouter une structure de choix afin d'enregistrer le fichier sur le serveur uniquement si un nouveau fichier photo est sélectionné par l'utilisateur :

```
if($action=="modif") {
    if ($_FILES['photo']['name']!="")
            {
            $nomphoto='photo_'.$nom.'.jpg';
            copy($_FILES['photo']['tmp_name'] ,'../photos/'.$nomphoto);
            }
```

2. Modifiez ensuite de la même manière la requête SQL afin de mettre à jour le champ `photo` uniquement si un fichier photo est sélectionné par l'utilisateur :

```
//-----------------------REQUÊTE SQL
      $updateAdherents = "UPDATE  adherents SET
      nom='".$nom."',
      prenom='".$prenom."',
      anneeNaissance='".$anneeNaissance."',
```

```
coursID=".$coursID." ";
if ($_FILES['photo']['name']!="")
{
$updateAdherents .=", photo='".$nomphoto."'";
}
$updateAdherents .=" WHERE ID=".$IDform ;
```

3. Ajoutez un champ de fichier nommé `photo` dans le formulaire de modification :

```
<input name="photo" type="file" id="photo">
```

4. Contrairement aux autres champs qui rappellent dans l'élément du formulaire la valeur actuelle de la variable concernée (à l'aide de l'attribut `value=`, par exemple, pour un élément `INPUT` de type `text`), l'objet « champ de fichier » (élément `INPUT` de type `file`) ne peut pas être initialisé avec une valeur par défaut. Cependant, vous pouvez rappeler la photo actuellement mémorisée dans la base en l'affichant sous le champ à l'aide du code suivant :

```
<img  src="../photos/<?php echo $row_rsListeAdherents['photo']; ?>" >
```

5. Enregistrez puis testez votre formulaire depuis le Web local.

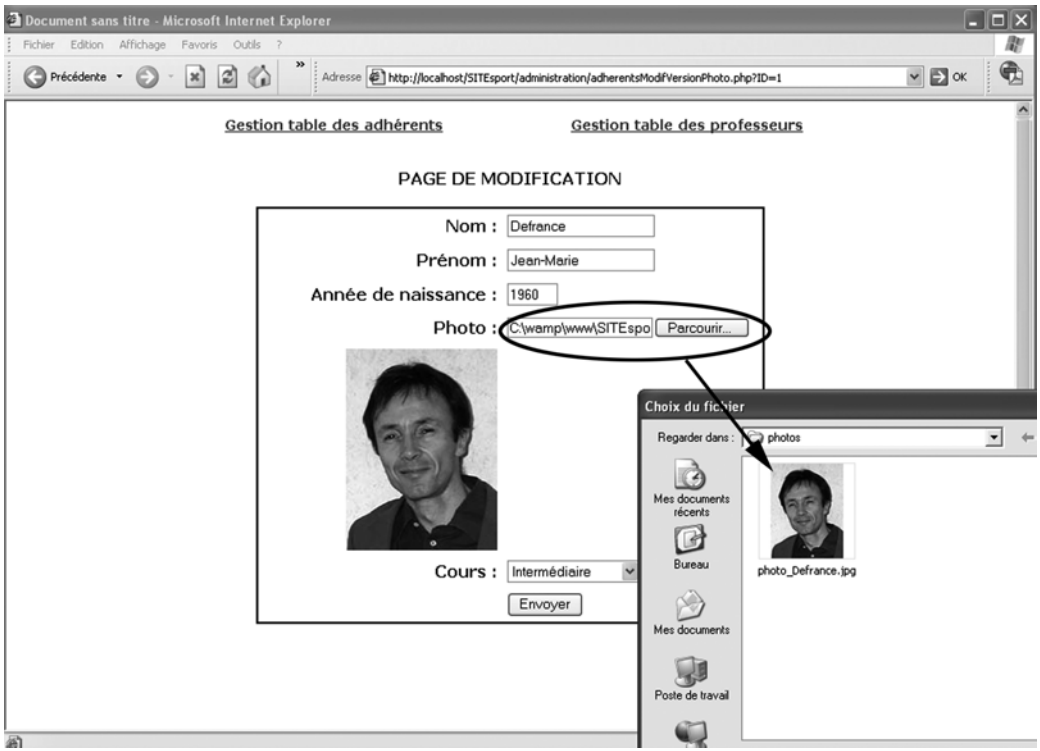


Figure 18-41

Page de modification d'un enregistrement de la base adherents avec gestion d'une photo

Structure d'un espace d'administration multitable

Vous disposez désormais de scripts pour gérer l'ajout, la suppression et la modification d'un enregistrement dans une table. Cependant, en pratique il vous faut créer des espaces d'administration afin de gérer plusieurs tables de la base de données. Un espace d'administration est évolutif (ajout ou suppression d'une table, par exemple) ; il est donc judicieux de créer des pages d'administration dont le menu transversal (qui permet d'accéder à chaque formulaire de gestion d'une table) pourra être actualisé simplement et rapidement.

Pour réaliser ces pages d'administration, vous pouvez soit créer avec Dreamweaver un modèle de page d'administration comportant une barre de navigation (menu transversal permettant d'accéder aux pages de gestion de chaque table), soit intégrer dynamiquement cette barre de navigation dans toutes les pages à l'aide d'une instruction `include()`.

Création d'un modèle de page d'administration

Cette première solution consiste à créer avec Dreamweaver une page modèle dans laquelle sera intégré un menu dont les différents liens permettent d'accéder à chaque page de gestion de table.

Si vous utilisez ce modèle pour créer toutes les pages de l'espace d'administration, lorsque vous modifiez son menu et y ajoutez un nouveau lien, toutes les pages issues du même modèle seront automatiquement actualisées.

Pour illustrer ce principe, nous vous proposons de créer une page modèle destinée à gérer les tables `adherents` et `professeurs`. Par la suite, vous pourrez transposer la structure de ce modèle en l'adaptant à la base de données de votre futur projet et gérer autant de table que vous le souhaitez :

1. Ouvrez une nouvelle page dynamique dans Dreamweaver. Créez un tableau d'une ligne et de deux colonnes. Saisissez les noms des tables à gérer dans les cellules du tableau. Convertissez ces noms en liens hypertextes afin qu'ils appellent respectivement les pages de gestion de chaque table (soit dans notre exemple : `adherentsGestion.php` et `professeursGestion.php`).
2. Créez un autre tableau d'une seule cellule en dessous. Sélectionnez ce nouveau tableau et paramétrez sa largeur et sa hauteur à 100 % afin qu'il s'affiche en plein page.
3. Assurez-vous que le tableau est toujours sélectionné et déroulez le menu Insertion>Objets modèle>Région modifiable. Validez puis donnez le nom de votre choix à cette zone modifiable dans la boîte de dialogue Nouvelle région modifiable.
4. Enregistrez cette page comme modèle (menu Fichier>Enregistrez comme modèle).
5. Créez ensuite dans le même répertoire `/administration/` toutes les pages de gestion des tables `adherents` et `professeurs` à partir de ce même modèle (menu Nouveau, onglet Modèle, sélectionnez le site Sport puis le modèle précédemment créé).

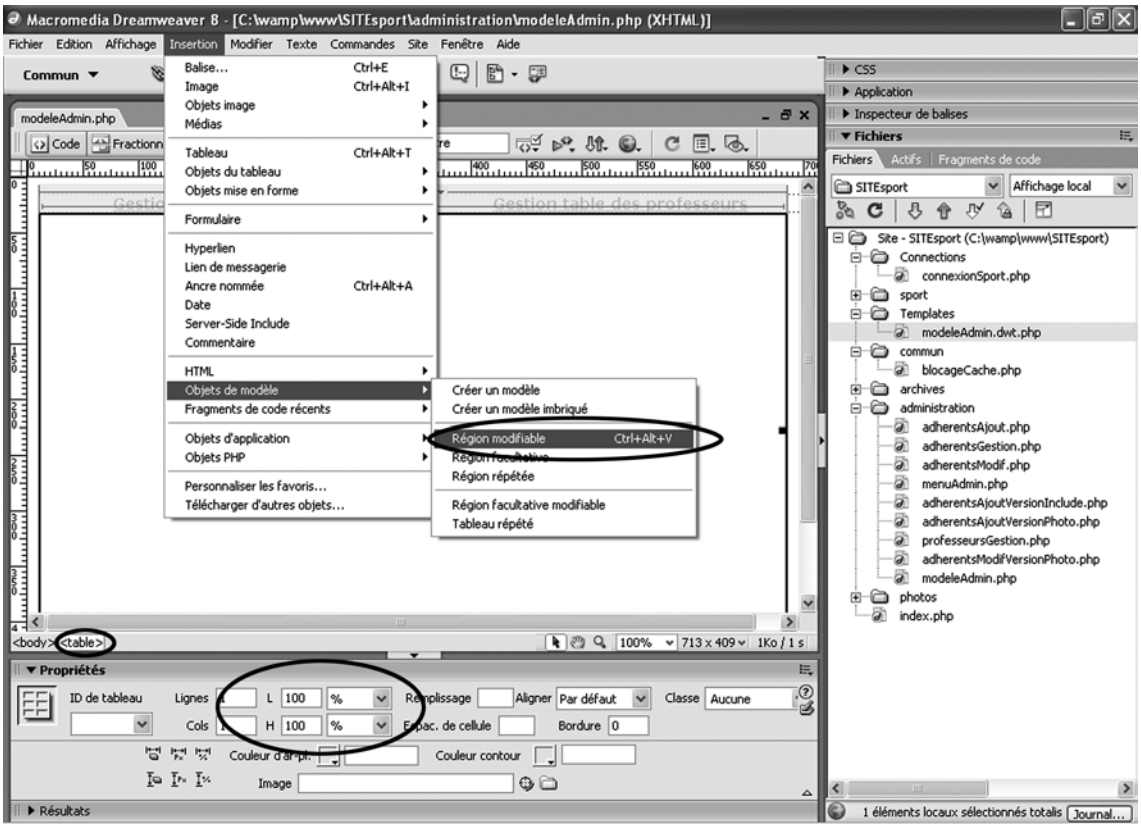


Figure 18-42

Création d'une région modifiable pour le futur modèle de la page d'administration.

6. Votre espace d'administration dispose des six pages suivantes, qui vous permettent d'administrer les enregistrements des deux tables de l'exemple :
   - `adherentsGestion.php`
   - `adherentsAjout.php`
   - `adherentsModif.php`
   - `professeursGestion.php`
   - `professeursAjout.php`
   - `professeursModif.php`
7. Par la suite, si vous désirez modifier la barre de navigation (afin de pouvoir gérer une nouvelle table, par exemple), il vous suffit d'ouvrir et de modifier le modèle d'administration (dans Dreamweaver, les modèles sont regroupés dans un répertoire `/Templates/` placé à la racine du site). Lors de l'enregistrement du modèle, une boîte de dialogue vous demande si vous désirez mettre à jour tous les documents issus du modèle. Si vous acceptez, la barre de navigation de toutes les pages de l'espace d'administration sera actualisée automatiquement.

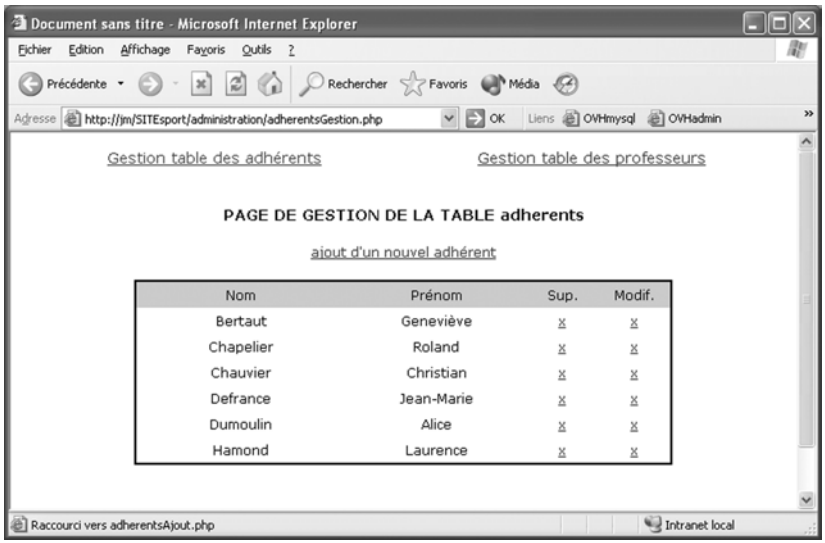


Figure 18-43

Page de gestion de la table adherents créée à partir du modèle d'administration

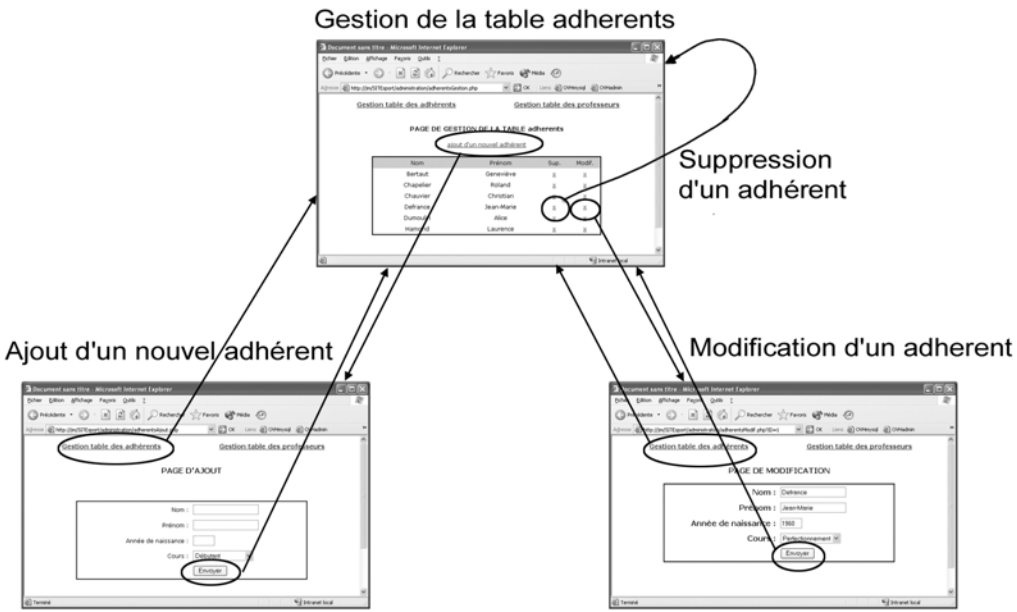


Figure 18-44

Organisation des pages de gestion de la table adherents

Intégration dynamique du menu d'administration

La seconde solution pour intégrer dynamiquement un menu transversal dans chaque page de l'espace d'administration consiste à utiliser l'instruction `include()`. Pour illustrer cette méthode, reprenons le même exemple que ci-dessus (deux tables à gérer) :

1. Un premier fichier contenant le menu doit d'abord être créé (fichier `menuAdmin.php`) :

```
<table width="700" border="0" align="center" cellpadding="0" cellspacing="0">
  <tr><td width="350"><div align="center" class="Style3">
<a href="adherentsGestion.php">Gestion table des adh&eacute;rents </a>
</div></td><td width="350"><div align="center" class="Style3">
<a href="professeursGestion.php">Gestion table des professeurs </a>
</div></td></tr>
</table>
```

2. Ensuite, chaque page d'administration doit comporter une instruction `include()` afin d'insérer tout le code du menu dans le haut des pages (après la balise `<body>`, par exemple) comme le montre le fragment de code ci-dessous. Ainsi, si vous désirez ajouter un lien supplémentaire au menu, il vous suffit de modifier le code du fichier `menuAdmin.php` et le nouveau menu s'actualise automatiquement dans toutes les pages comportant l'instruction `include()` présentée ci-dessus :

```
…
<body>
<?php include("menuAdmin.php"); ?>
…
```

Gestion des erreurs MySQL

PHP met à votre disposition la fonction `mysql_error()`, qui retourne le type d'erreur sous forme de chaîne (s'il y en une, sinon la fonction retourne une chaîne vide) et la fonction `mysql_errrorno()`, qui retourne le numéro de l'erreur (s'il y en a une, sinon la fonction retourne la valeur `0`). Pour utiliser ces deux fonctions, vous pouvez passer l'identificateur de connexion au serveur MySQL en argument de ces deux fonctions. Si vous utilisez ces fonctions sans argument, c'est la dernière connexion active qui sera utilisée.

Lors de la soumission d'une requête, il suffit de lier l'une de ces deux fonctions à l'instruction `mysql_query()` à l'aide d'une fonction `or die()` ou `or trigger_error()` pour afficher un message d'erreur explicite en cas de refus de la requête SQL :

```
mysql_query($updateAdherents, $connexionSport) or die(mysql_error());
```

En phase de développement, ces messages vous seront précieux pour déboguer vos scripts. Cependant, si votre application passe en production, il faudra contrôler ces erreurs et éviter l'affichage de ces messages peu esthétiques. PHP vous permet de neutraliser l'affichage des messages d'erreur grâce à l'ajout du caractère `@` devant la fonction :

```
@mysql_query($updateAdherents, $connexionSport)
```

Rappelons que la fonction `mysql_query()` renvoie un pointeur de résultat (qui doit ensuite être interprété à l'aide d'une fonction comme `mysql_fetch_assoc()`) dans le cas d'une commande `SELECT`. Pour les autres commandes SQL comme `DELETE, INSERT, UPDATE`, la fonction renvoie une valeur `TRUE` en cas de succès ou une valeur `FALSE` en cas de problème. Il peut être intéressant de récupérer cette valeur afin de traiter l'erreur :

```
$res=mysql_query($updateAdherents, $connexionSport)
if(!$res) {
echo "ERREUR MYSQL".mysql_error()." N° ". mysql_errorno();
}
```

De même, il est intéressant d'exploiter cette structure pour rediriger le visiteur vers un message plus rassurant et adapté à la charte graphique du site en cas de problème comme le montre ce deuxième exemple (vous remarquerez que la fonction est précédée du caractère `@` afin de neutraliser l'affichage du message d'erreur généré par défaut) :

```
$res=@mysql_query($updateAdherents, $connexionSport)
if(!$res) {
header("Location:erreurMysql.php");
}
```

Des erreurs peuvent aussi être générées par la fonction `mysql_connect()` si la connexion au serveur MySQL est impossible. Si vous désirez contrôler les messages et rediriger le visiteur vers une page adaptée, modifiez la fonction `mysql_connect()` dans le fichier `connexionSport.php` comme indiqué ci-dessous (suppression de l'affichage du type d'erreur et ajout d'un `@` devant la fonction) :

```
$connexionSport = @mysql_connect($hostname_connexionSport, $username_connexionSport,
➥$password_connexionSport);
```

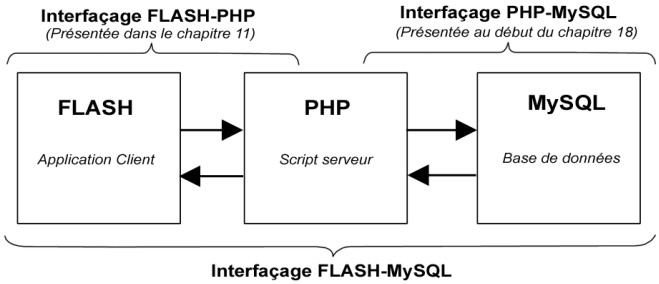
Il faudra évidemment intégrer un test au début de chaque page utilisant cette fonction comme l'illustre le script ci-dessous :

```
require_once('../Connections/connexionSport.php');
if(!$connexionSport) {
header("Location:erreurMysql.php");
}
```

Interfaçage Flash-MySQL

L'interfaçage Flash-MySQL permet à une application Flash d'enregistrer ou d'exploiter des informations provenant d'une base de données MySQL. Une application Flash ne peut pas interagir directement sur une base de données MySQL. Il faut donc utiliser des scripts PHP adaptés afin de réaliser une passerelle entre Flash et MySQL. En pratique, l'interfaçage Flash-MySQL est constitué d'une interface Flash-PHP couplée à une interface PHP-MySQL (voir figure 18-45). Comme nous venons de présenter les solutions d'interfaçage PHP-MySQL et que l'interfaçage entre une application Flash et un script PHP a été présenté dans le chapitre 11, nous allons maintenant étudier l'exploitation conjointe de ces deux types d'interfaçage dans le cadre d'une application concrète de contrôle d'accès par mot de passe.

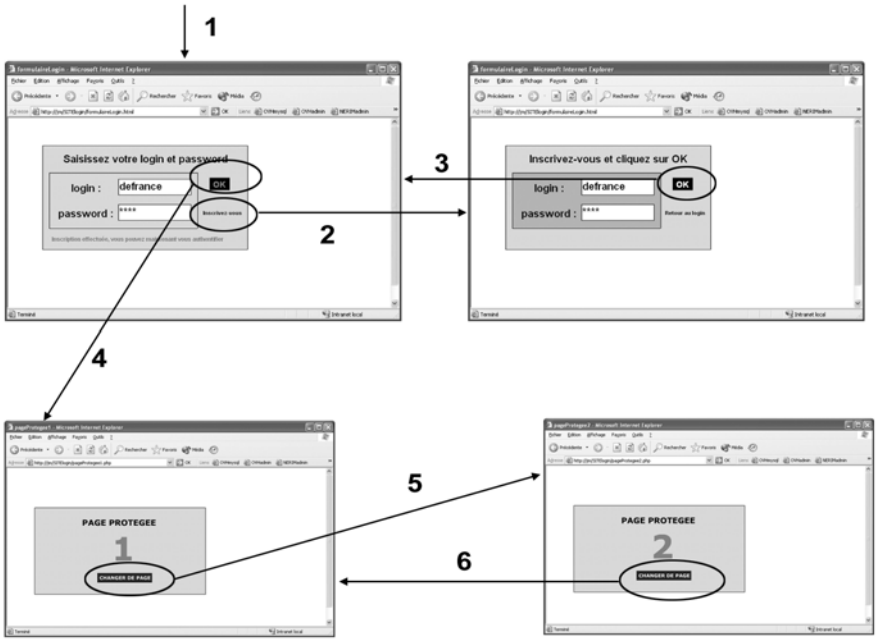
Figure 18-45
L'interfaçage Flash-MySQL est composé d'une interface Flash-PHP couplée avec une interface PHP-MySQL.



Contrôle d'accès par mot de passe

Cette application Flash dynamique permet de contrôler l'accès d'un utilisateur à des pages protégées (`pageProtegee1.php` et `pageProtegee2.php`). La page d'accès est constituée d'un formulaire Flash qui invite l'utilisateur à saisir son login et son mot de passe. Sur cette même page, un lien permet à l'utilisateur d'afficher un second formulaire lui permettant de s'inscrire. Les informations recueillies lors de l'inscription sont mémorisées dans une base de données via un script PHP adapté (`inscription.php`).

Figure 18-46
Exemple de navigation dans l'application de contrôle d'accès par mot de passe : 1 – accès au formulaire de login, 2 – passage sur le formulaire d'inscription, 3 – inscription et retour au formulaire de login, 4 – saisie du login et du mot de passe puis, si l'authentification est positive, redirection vers la première page protégée, 5 et 6 – possibilité de passer d'une page protégée à l'autre grâce à la mémorisation de l'authentification durant la session.



La procédure d'authentification est elle aussi assurée par un script PHP (`authentification.php`) qui a pour fonction de récupérer les paramètres de connexion fournis par l'utilisateur et d'interroger la base de données afin de vérifier la présence et la concordance du login et du mot de passe fournis. Si la réponse de la base est positive, l'utilisateur est dirigé automatiquement vers la première page protégée (voir figure 18-46). Dans le cas contraire, un message d'erreur l'invite à renouveler sa saisie

après avoir vérifié l'exactitude de ses paramètres. Si un utilisateur tente d'accéder directement aux pages protégées, il sera redirigé automatiquement vers le formulaire d'authentification car chaque page protégée comporte un script qui teste l'authentification de l'utilisateur (grâce à l'inclusion du code contenu dans le fichier `controle.php`).

Ce projet comporte un document Flash (`formulaireLogin.fla`), une base de données MySQL nommée `login_db` (constituée d'une seule table `utilisateurs`) et plusieurs scripts PHP afin d'assurer les interactions entre l'application Flash et la base de données ainsi que la mémorisation de l'authentification d'un utilisateur durant la session.

La base de données

Les paramètres de chaque utilisateur sont mémorisés dans la base de données. Il faut donc commencer le développement en créant la base et sa table. Pour cela, ouvrez le gestionnaire phpMyAdmin (déroulez le menu du manager de Wamp 5 puis sélectionnez phpMyAdmin). Une fois le gestionnaire ouvert, saisissez le nom de la nouvelle base dans le champ de droite (voir figure 18-47) et validez en cliquant sur le bouton Créer.

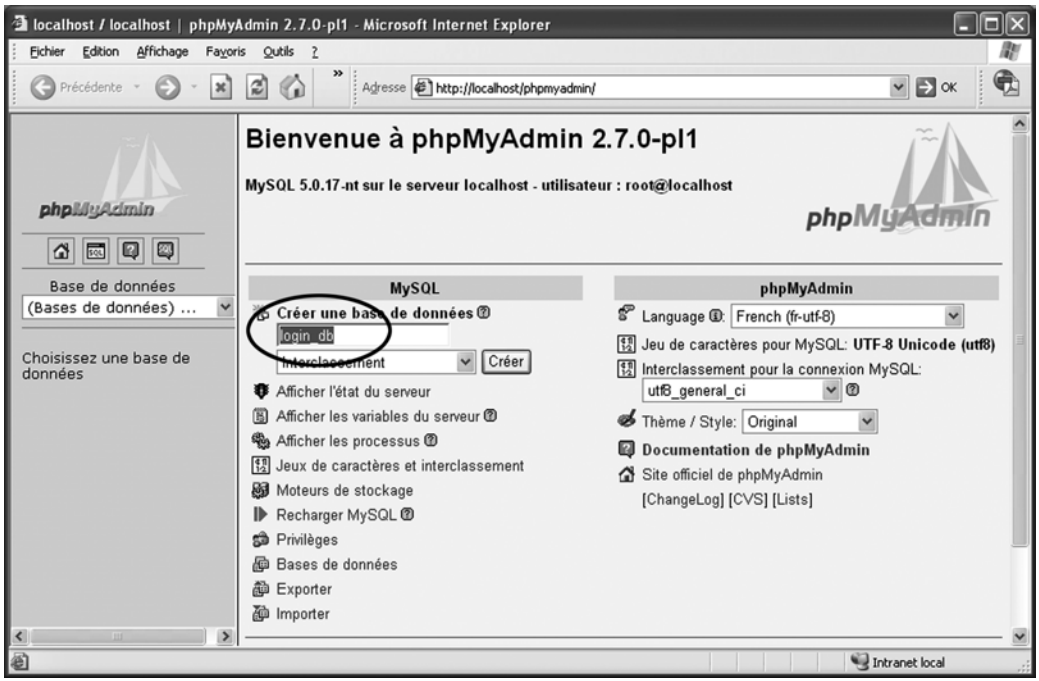


Figure 18-47

Création de la nouvelle base de données login_db

Une fois la base créée, deux solutions s'offrent à vous pour construire la structure de la table : vous pouvez soit suivre les étapes de création avec le gestionnaire (ces étapes sont détaillées ci-dessous, voir figure 18-49), soit utiliser les instructions de restauration SQL que vous trouverez dans le

dossier `/archives/sql/` du projet `SITElogin` (disponible en téléchargement sur le site `www.editions-eyrolles.com`). Ce fichier, nommé `login_db.sql`, doit être appelé depuis le champ Emplacement du fichier texte de l'onglet Importer du gestionnaire (voir figure 18-48). Après validation grâce au bouton Exécuter, le fichier est téléchargé sur le serveur et la base de données est restaurée automatiquement.

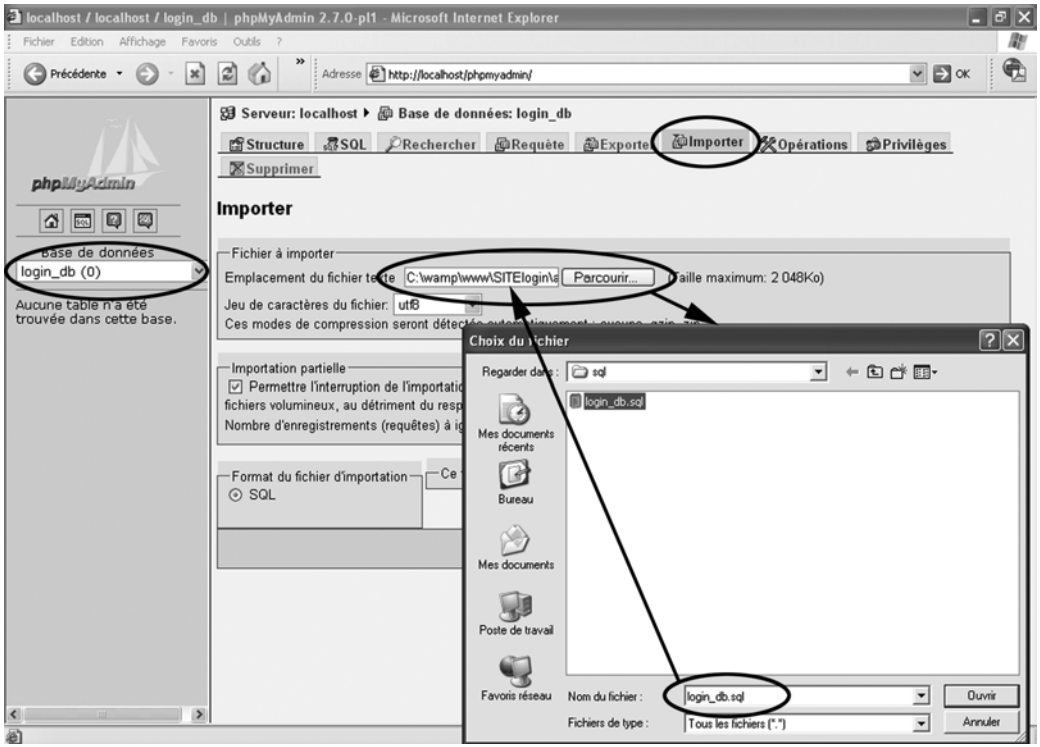


Figure 18-48

Chargement du fichier de sauvegarde de la base de données (login_db.sql)

Voici les différentes instructions SQL contenues dans le fichier de restauration `login_db.sql` qui permettent de créer la structure de la table `utilisateurs` et d'y insérer un premier enregistrement (`login=defrance` et `secret=1234`) :

```
DROP TABLE IF EXISTS `utilisateurs`;
CREATE TABLE `utilisateurs` (
  `ID` smallint(6) NOT NULL auto_increment,
  `login` varchar(20) NOT NULL default '',
  `secret` varchar(20) NOT NULL default '',
  PRIMARY KEY  (`ID`)
) TYPE=MyISAM AUTO_INCREMENT=2 ;
INSERT INTO `utilisateurs` VALUES (1, 'defrance', '1234');
```

Voici les étapes à suivre si vous désirez construire la table `utilisateurs` vous-même à l'aide du gestionnaire phpMyAdmin :

1. Saisissez le nom de la table à créer (soit `utilisateurs`) dans le champ Créer une nouvelle table et indiquez qu'elle comportera trois champs puis validez.
2. Dans le formulaire de configuration des champs de la table (voir figure 18-49), saisissez `ID` et sélectionnez `SMALLINT` pour le premier champ. À droite sur la même ligne, sélectionnez l'option `auto_increment` dans la colonne Extra et cochez la case Primaire.
3. Pour les deux autres champs, saisissez les noms `login` et `secret`. Sélectionnez `VARCHAR` pour leur type et `20` pour leur taille. Cliquez ensuite sur Sauvegarder pour confirmer vos choix et créer la table.

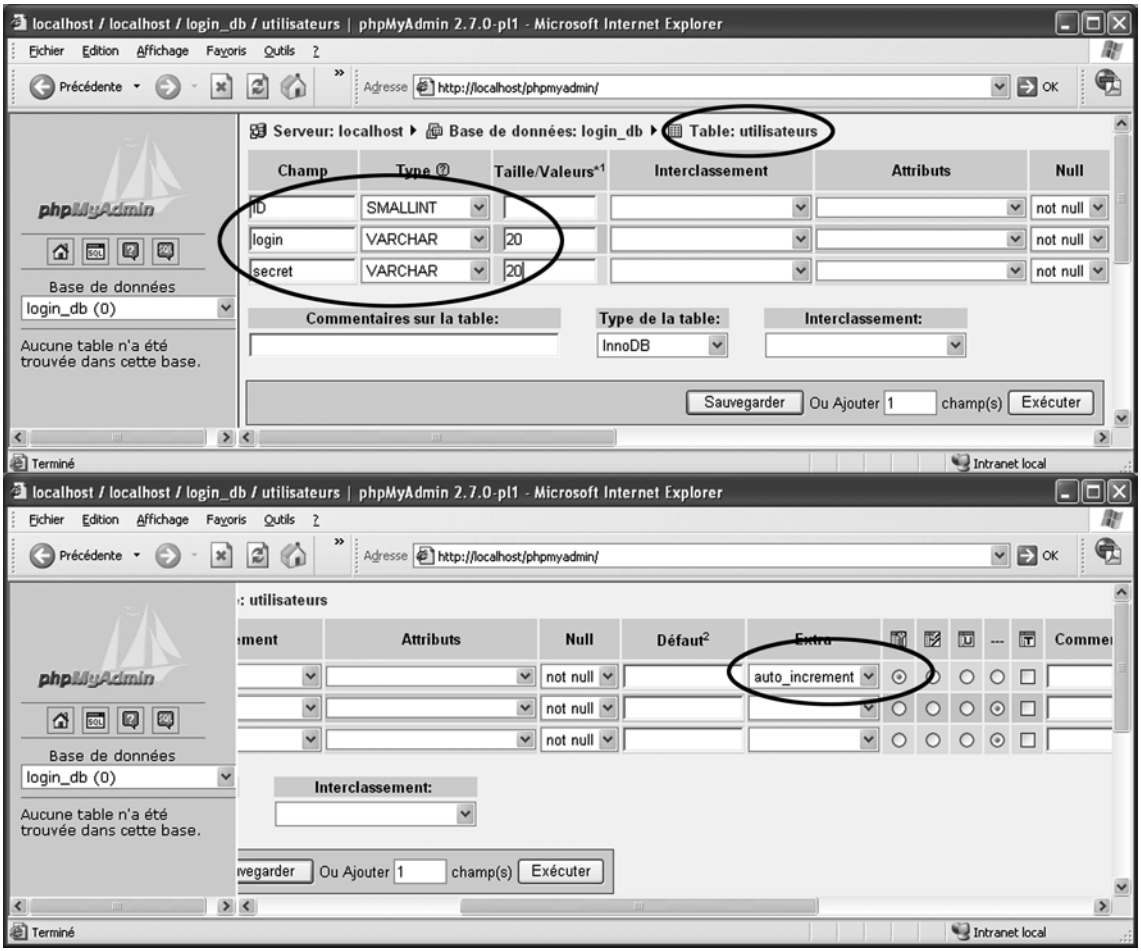


Figure 18-49

Création de la table utilisateurs à l'aide du gestionnaire phpMyAdmin

Votre table est désormais créée. Vous pouvez maintenant commencer à développer vos scripts PHP.

À noter

Actuellement, la table est vide mais cela n'est pas gênant car vous pourrez bientôt ajouter des enregistrements depuis le formulaire d'inscription. D'autre part, nous n'avons pas créé de `user` spécifique pour accéder à cette table car nous utiliserons le compte `root` par défaut (sans mot de passe) pour nos tests en local. Cependant, si vous devez mettre cette application en ligne, il vous faudra créer et configurer un compte `user` avec son mot de passe (revoir si besoin la figure 16-30 et les commentaires correspondants).

Les fichiers PHP

Nous devons d'abord créer deux scripts PHP (`authentification.php` et `inscription.php`) qui serviront de passerelle entre la base de données que nous venons de créer et l'application Flash que nous développerons ci-dessous. Un troisième script PHP, nommé `controle.php`, sera intégré dans toutes les pages à protéger à l'aide d'une simple instruction `require_once()`.

Configuration d'un nouveau site dynamique

Avant de créer les pages PHP, il faut ouvrir et configurer un nouveau site dans Dreamweaver. Comme pour les précédentes applications de ce chapitre, nous utiliserons un fichier de connexion à la base de données MySQL compatible avec l'usage des comportements serveur de Dreamweaver. La procédure de création de ce fichier ayant déjà était présentée, nous n'en détaillerons pas toutes les étapes et vous invitons à vous reporter à la figure 18-2 et aux commentaires en rapport si nécessaire.

Configuration du site :

1. Ouvrez Dreamweaver. Dans le menu, cliquez sur la rubrique Sites puis sélectionnez Gérer les sites.
2. Dans la boîte de dialogue Gérer les sites, cliquez sur le bouton Nouveau.
3. Dans la boîte de dialogue Définition du site, cliquez sur l'onglet Avancé s'il n'est pas déjà sélectionné.
4. Cliquez sur la catégorie Infos Locales et renseignez les champs suivants :
   - Nom du site : `SITElogin`.
   - Dossier racine : cliquez sur le petit dossier situé à droite du champ afin d'ouvrir l'explorateur de fichier. Ensuite, créez puis sélectionnez un nouveau dossier `/SITElogin/` placé dans le répertoire racine `www` de la suite Wamp 5.
5. Cliquez sur la catégorie Serveur d'évaluation et renseignez les champs suivants :
   - Modèle de serveur : sélectionnez PHP-MySQL dans le menu déroulant ;
   - Accès : sélectionnez Local/Réseau dans le menu déroulant ;
   - Dossier du serveur d'évaluation : théoriquement, ce champ doit être préconfiguré avec le chemin menant au répertoire du site `www/SITElogin/` ;
   - Préfixe de l'URL : ajoutez `SITElogin/` à la suite de `http://localhost/`.
6. Cliquez sur le bouton OK de la boîte de dialogue puis sur le bouton Terminer de la boîte de dialogue Gérer les sites.

Création du fichier de connexion :

1. Ouvrez une page PHP dans Dreamweaver (menu Fichier>Nouveau, sélectionnez Page dynamique et PHP puis cliquez sur Créer).
2. Déroulez le panneau Application et sélectionnez l'onglet Base de données.
3. Cliquez sur le bouton + et sélectionnez la rubrique Connexion MySQL qui s'affiche dans le menu contextuel.
4. La fenêtre Connexion MySQL s'affiche.
5. Saisissez un nom pour la connexion que vous allez créer. Ce nom doit être explicite et ne pas comporter d'espace. Dans le cadre de notre application, nous utiliserons `connexionLogin`.
6. Dans le champ `Serveur MySQL`, saisissez `localhost`.
7. Saisissez ensuite les paramètres de l'utilisateur `root` (utilisateur par défaut), soit nom d'utilisateur = `root` sans mot de passe.
8. Cliquez sur le bouton Sélectionner pour afficher toutes les bases de données disponibles. Sélectionnez la base `login_db` et validez en cliquant sur le bouton OK.
9. La base sélectionnée doit être affichée dans le champ Base de données. Vous pouvez maintenant vérifier si la connexion est valide en cliquant sur le bouton Tester.
10. Fermez la fenêtre du message et confirmez la création de la connexion en cliquant sur le bouton OK.
11. La connexion à la base de données est désormais établie. Une icône représentant la base `login_db` apparaît dans la fenêtre Base de données.

Si vous désirez créer manuellement le fichier de connexion à la base, saisissez les lignes de code suivantes dans votre éditeur et enregistrez-les sous le nom `connexionLogin.php` dans le répertoire `SITElogin/Connection/` :

```
<?php
$hostname_connexionLogin = "localhost";
$database_connexionLogin = "login_db";
$username_connexionLogin = "root";
$password_connexionLogin = "";
$connexionLogin = mysql_connect($hostname_connexionLogin, $username_connexionLogin,
➥$password_connexionLogin) or trigger_error(mysql_error(),E_USER_ERROR);
?>
```

Création de la page d'authentification

La page `authentification.php` a pour fonction de récupérer le login et le mot de passe envoyés par le formulaire de login de Flash, de construire une requête SQL afin d'interroger la base sur l'existence et la concordance de ces paramètres, de réceptionner la réponse de la base et de la mettre en forme avant de la retourner à l'application Flash. Si la réponse est positive, le login de l'utilisateur est mémorisé tant qu'il visite des pages protégées.

1. Ouvrez un nouveau document dynamique PHP et enregistrez-le sous le nom `authentification.php` à la racine du site (répertoire `/SITElogin/`).

2. Saisissez une balise ouvrante PHP puis la fonction `session_start()` afin d'activer l'usage des sessions dans cette page (voir figure 18-50) :

```
<?php
// Active les sessions
session_start();
```

3. Saisissez deux lignes d'instruction destinées à récupérer (et à convertir au bon format si elles existent) les variables `login` et `secret` envoyées par l'application Flash :

```
// Récupération des variables envoyées par Flash
if(isset($_POST['login'])) $login= utf8_decode($_POST['login']); else $login="inconnu";
if(isset($_POST['secret'])) $secret= utf8_decode($_POST['secret']); else
$secret="inconnu";
```

4. Déclarez la fonction destinée à mettre en forme les variables retournées à l'application Flash :

```
//--------------FONCTIONS
function envoi($var, $val){
echo "&".$var."=".utf8_encode($val);

}
```

5. Saisissez les deux lignes suivantes afin d'établir une connexion à la base de données (appel du fichier de connexion créé précédemment) et de sélectionner la base concernée :

```
//--------CONNEXION ET SÉLECTION DE LA BASE ------------
require_once('Connections/connexionLogin.php');
mysql_select_db($database_connexionLogin, $connexionLogin);
```

6. Créez et envoyez la requête SQL à la base de données. La clause `WHERE` de cette requête est personnalisée avec la variable `login` envoyée par le formulaire Flash. Le jeu d'enregistrements retourné par la requête (valeur du champ `secret` correspondant au `login`) est ensuite converti en tableau associatif avec la fonction `mysql_fetch_assoc()`. La dernière ligne permet de connaître le nombre d'enregistrements retournés dans le résultat (afin de détecter si le login n'existe pas) :

```
//---------RECUPÉRATION DES INFORMATIONS (depuis la table adherents)
$query_rsVerifLogin = "SELECT secret FROM utilisateurs WHERE login='$login' ";
$rsVerifLogin = mysql_query($query_rsVerifLogin, $connexionLogin) or die(mysql_error());
$row_rsVerifLogin = mysql_fetch_assoc($rsVerifLogin);
$total_rsVerifLogin=mysql_num_rows($rsVerifLogin);
```

7. Une structure de choix `if` conditionnée par l'existence et la concordance du login et du mot de passe permet de retourner le couple `retour=ok` à l'application Flash puis de mémoriser la valeur du login dans la session si le test est positif. Dans le cas contraire, le couple `retour=pb` est envoyé à l'application Flash :

```
/---------TEST DE L'EXISTENCE ET DE CONCORDANCE LOGIN/PWD
if(($secret == $row_rsVerifLogin['secret'])&&($total_rsVerifLogin!=0)){
//test si le controle du mot de passe est ok
$_SESSION['login']=$login;
//mémorise le login de l'utilisateur en session
envoi("retour","ok");
```

```
//envoie la confirmation de l'authentification à l'application Flash
}
else
{
envoi("retour","pb");
//dans le cas contraire, un message d'erreur est renvoyé à l'application Flash
}
```

8. Enregistrez votre fichier sous le nom `authentification.php`.

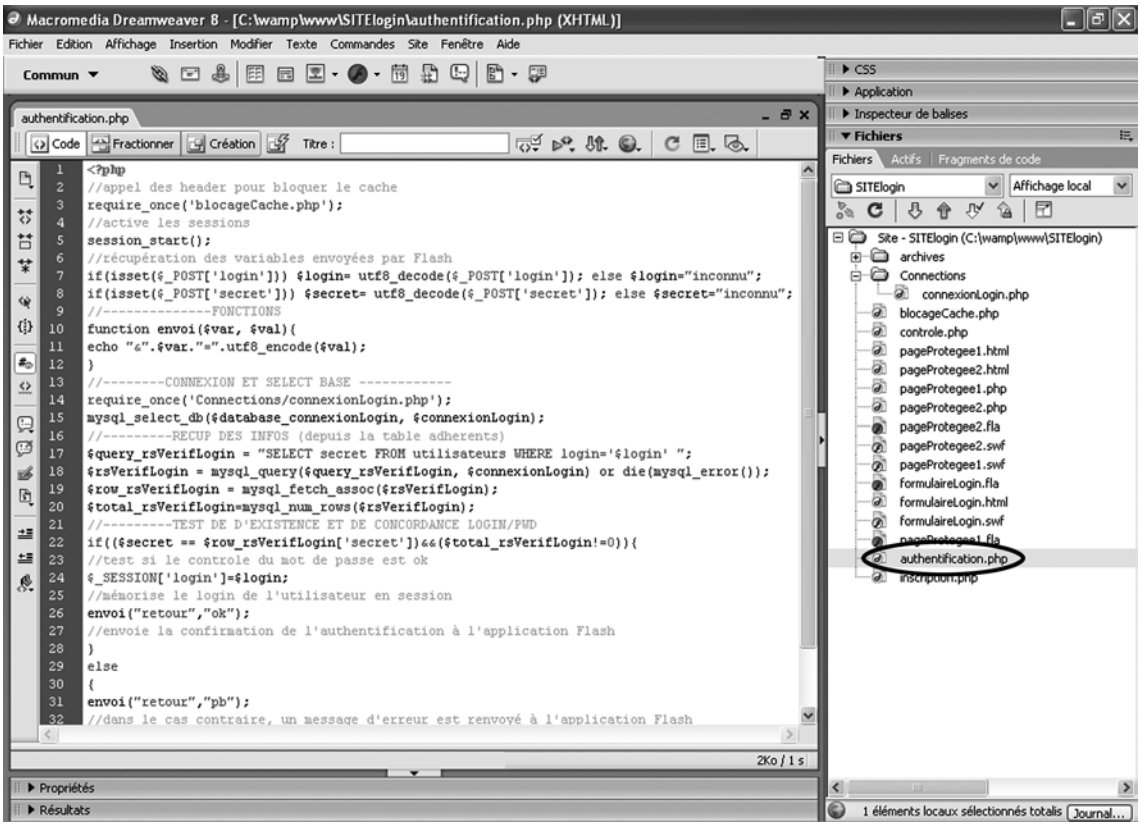


Figure 18-50

Début du programme de la page authentification.php

Création de la page d'inscription

La page `inscription.php` a pour fonction de récupérer le login et le mot de passe envoyés par le formulaire d'inscription de Flash et de construire la commande `INSERT` nécessaire pour ajouter un nouvel utilisateur dans la base de données. Afin d'éviter les doublons lors de la création de login, une première requête `SELECT` est envoyée à la base de données avant la commande `INSERT` afin de s'assurer que le login proposé par l'utilisateur n'est pas déjà attribué. Si le nombre d'enregistrements retourné