Université Ibn Zohr *A.U. 2009-2010*
Faculté des Sciences, Département de Mathématiques et Informatique
Agadir

Cours Programmation Web dynamique (PHP-MySQL)
Filière Sciences Mathématiques et Informatique (SMI5)

1 Introduction

Ce document décrit les grandes lignes du module "Programmation web dynamique". Il est destiné aux étudiants de la filière sciences mathématiques et informatique (SMI5) de la faculté des sciences d'Agadir qui ont suivi le cours "Initiation à la programmation web" et qui ont l'habitude de programmer dans un langage de programmation tel que le langage C.

L'étudiant, y trouvera :

- Une présentation du langage PHP (syntaxe, nouveautés par rapport au langage C…),
- La manipulation de fichiers avec PHP,
- Un résumé sur les fonctions qui permettent de manipuler une base de données MySQL,
- Les fonctions de manipulations de chaine de caractères,
- La gestion de la date et de l'heure.

Le document est accompagné aussi d'un ensemble de travaux pratiques et un mini projet permettant aux étudiants d'attaquer le développement de projet réels.

La dernière partie, décrit d'une manière détaillée les différentes étapes d'installation et de configuration du trio Apache/PHP/MySQL sous le système Windows XP.

2 Présentation du langage PHP

2.1 Historique

Les origines de PHP remontent à la mi-**1993** avec le site de Rasmus Lerdorf : une page personnelle qui permettait de conserver une trace de passage des utilisateurs.

Février **1994** publication de la version **1.0** suite à la demande de la mise à disposition de cet outil par de nombreux utilisateurs.
Eté **1995**, apparition de la version **2.0** (Intégration des structures des structures plus avancées : conditions, boucles, intégration des formulaires …)
La version **2.0** permettra ainsi au développeur d'intégrer des instructions de programmation puissantes directement dans du code HTML.

C'est ainsi l'effort d'une communauté de développeurs qui a ensuite permis de produire les versions ultérieures.

Juin **1998** publication de la version **3.0**
Début **2000** apparition de la version **4.0**
Aujourd'hui: PHP version **5**

2.2 Présentation de PHP

- Langage de scripts coté serveur proche du C,
- Il a été conçu spécifiquement pour la création dynamique des pages HTML,
- Il s'intègre dans une page HTML,
- PHP est un produit Open Source,
- Licence GPL,
- Développé en C,
- Indépendant de la plate-forme utilisée puisqu'il est exécuté côté serveur et non côté client.

Les Principaux atouts de PHP sont:

- La gratuité et la disponibilité du code source (PHP4 est distribué sous licence GNU GPL),
- La simplicité d'écriture de scripts,
- La possibilité d'inclure le script PHP au sein d'une page HTML,
- La simplicité d'interfaçage avec les principaux systèmes de gestion de bases de données,
- L'intégration au sein de nombreux serveurs web (Apache, Microsoft IIS, ...),
- Indépendance vis-à-vis des OS et serveurs même si l'environnement Linux est sa plate forme de prédilection.

2.3 Environnement de développements

Pour vos développements des pages web dynamiques avec PHP, vous aurez besoin :

- Un OS : Windows, Linux
- Un serveur web avec un interpréteur PHP
    - Apache
    - Configurable avec conf/httpd.conf (fichier texte)

- Voir aussi le fichier php.ini pour configurer l'interpréteur
- Un éditeur de texte pour écrire des scripts
  - NotePad par exemple (vous pouvez opter pour d'autres éditeurs plus sophistiqués, Ultraedit, NotePad++…)
- Éventuellement une base de données
  - MySQL par exemple

2.4 Les bases de PHP

Au départ, une page PHP est tout simplement une page HTML. Cela signifie que celle-ci comporte des balises placées au sein d'une structure HTML classique :

```
<HTML>
      <HEAD>
            <TITLE></TITLE>
      </HEAD>
      <BODY>
      < /BODY>
</HTML>
```

Les balises PHP sont incluses à l'intérieur du code HTML, mais elles sont repérées par les mentions suivantes : < ? et ?>

Exemple :

```
<HTML>
 <HEAD>
  <TITLE>   </TITLE>
 </HEAD>
 <BODY>
  <?
      echo "<b>Coucou le monde ! C'est ma première page PHP  </b>";
  ?>
 </BODY>
</HTML>
```

Si nous plaçons une telle page sur un serveur, et veillons à lui donner un nom portant l'extension *.php* (plutôt que *.html*), le serveur va analyser le code avant d'envoyer la page au poste client.

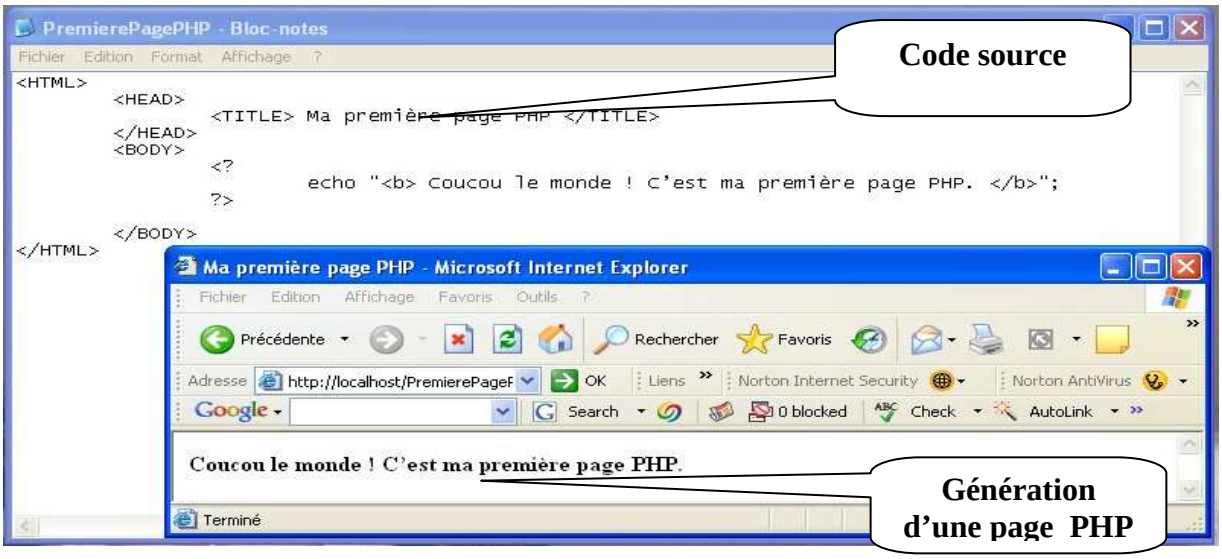


Figure 1 Première page Web en PHP

Pour savoir ce qui s'est passé au niveau du serveur, il suffit d'aller dans le menu **Affichage** puis de l'*Item* **source**

On voit apparaître le code source suivant :

```
<HTML>
      <HEAD>
            <TITLE> Ma première page PHP </TITLE>
      </HEAD>
      <BODY>
            <b> Coucou le monde ! C'est ma première page PHP. </b>
      </BODY>
</HTML>
```

La commande **echo** incluse entre < ? et ?> a disparu. Il ne subsiste que de l'HTML. L'analyseur présent au niveau du serveur a converti la balise PHP de manière à produire du HTML, puis a envoyé cette page au poste client.

Pour mieux prendre la mesure la mesure d'un tel travail au niveau du serveur, considérons la page suivante :

```
<HTML>
      <HEAD>
            <TITLE> Boucle en PHP </TITLE>
      </HEAD>
      <BODY>
            <?
                  for ($i=1; $i<=7; $i++)
                  {
                        echo "<FONT SIZE = $i>";
                        echo "Une boucle en PHP";
                        echo "<BR>";
                  }
            ?>

      </BODY>
</HTML>
```

L'instruction *for* est une boucle. Elle comporte une valeur de départ (1) et une valeur d'arrivée (6). Les instructions entre les accolades sont exécutées aussi longtemps que la variable *$i* est inférieur à 6. L'instruction *$i++* signifie que l'on ajoute 1 à chaque passage.

Figure 2 Exécution d'une boucle avec PHP

La page qui résulte d'une telle exécution affiche la phrase « Une boucle en PHP » avec six tailles de caractères différentes.

Cette fois, si nous affichons le code source, nous découvrons qu'il est le suivant :

```
<HTML>
      <HEAD>
            <TITLE>Une boucle en PHP </TITLE>
      </HEAD>
      <BODY>
            <FONT SIZE = 1>Une boucle en PHP<BR>
            <FONT SIZE = 2>Une boucle en PHP<BR>
            <FONT SIZE = 3>Une boucle en PHP<BR>
            <FONT SIZE = 4>Une boucle en PHP<BR>
            <FONT SIZE = 5>Une boucle en PHP<BR>
            <FONT SIZE = 6>Une boucle en PHP<BR>
            <FONT SIZE = 7>Une boucle en PHP<BR>
      </BODY>
</HTML>
```

L'analyseur PHP situé au niveau du serveur a généré toute une séquence du code HTML à partir du code PHP.

- Le script PHP est exécuté sur le serveur. Celui-ci n'envoie au poste client que de l'HTML. Cela signifie que le script pourra fonctionner quelque soit le navigateur utilisé.
- Lorsque le langage de script est exécuté sur le client (cas de JavaScript par exemple), il n'est pas toujours sûr qu'il soit exécuté correctement.
- PHP résout ce problème en plaçant l'exécution du script sur le serveur, lequel génère une page HTML normale, généralement compréhensible par tous les navigateurs.

2.5 Identification des commandes PHP

L'analyseur qui se trouve au niveau du serveur ne traite que les commandes PHP. Les balises HTML ne sont pas prises en compte, elles sont envoyées telles quelles au poste client.

Pour utiliser PHP, *Rasmus Lerdorf* a inventé une balise spéciale. Elle commence par **<?** et se termine par **?>**.

Toutes fois, d'autres balises existent :

- **<?php … ?> ou**
- **<script language="php"> … </script>**

2.6 Séparateur d'instructions PHP

A l'instar de langages tels que le C ou Java, PHP requiert un point virgule à la fin de chaque instruction.

2.7 Commentaires

Quelques que soit le langage de programmation utilisé, il est judicieux d'inclure des commentaires qui faciliteront la relecture du code. PHP accepte trois formes de commentaires.

La première forme où le commentaire peut s'étaler sur plusieurs lignes :

```
<?
      echo "Hello Word";            /* commentaire sur une ligne */
      /* ce commentaire
         se prolonge sur
         plusieurs lignes  */
?>
```

La deuxième forme est le commentaire sur une seule ligne :

```
< ?
      echo "Hello Word";      // commentaire sur une ligne
?>
```

La troisième est conforme à Unix :

```
<?
      echo "Hello Word";      # commentaire sur une ligne façon Shell Unix
?>
```

2.8 Afficher du texte

L'instruction PHP **echo** permet d'écrire sur la sortie standard (fichier HTML).

Syntaxe :

```
echo Expression;
echo "String";
```

D'autres fonctions permettent aussi l'affichage :

La fonction print :

```
print(expression);
print("String");
```

La fonction *printf* :

```
printf (chaîne formatée);
```

Exemples :

```
echo (5+9-8)*10;
print (5+9-8)*10);
printf ("L'année %d est bissextile", $Annee);
```

2.9 Les constantes et les variables

2.9.1 Variables

- En PHP les variables sont identifiées avec **$**.
- L'identifiant d'une variable ne doit comporter que des caractères alphanumériques ou le caractère souligné (_).
- Un identificateur ne peut pas commencer par un chiffre.
- Dans le langage PHP, les identificateurs sont sensibles à la casse.
- En PHP, il n'est pas nécessaire de déclarer les variables avant de les utiliser. Une variable est automatiquement créée lorsqu'une valeur lui est affectée pour la première fois.

PHP dispose d'une palette de fonctions qui permettent la manipulation et le test des variables de différentes manières.

2.9.2 Test et définition des types de variables

La plupart des fonctions de variables s'utilisent pour tester le type des variables.
Les deux fonctions de variables les plus générales sont *gettype ()* et *settype ()*.

gettype () s'emploie en lui passant une variable. La fonction détermine alors le type de la variable et retourne une chaîne contenant le nom du type : *boolean, integer, double, string, array*…

Pour utiliser *settype()*, il faut passer à la fonction la variable dont le type doit être modifier, ainsi qu'une chaîne contenant le nom du nouveau type à appliquer à la variable.

Illustration:

```
$var = 15 ;
echo gettype($var).'<br>' ;
settype($var, 'double') ;
echo gettype($var);
```

Lors du premier appel de la fonction *gettype ()*, la variable *$var* est type *integer*. Après l'appel de *settype (), $var* est du type double.

PHP fournit également des fonctions testant un certain type de variable. Chacune de ces fonctions prend une variable comme argument et retourne soit *true*, soit false. Ces fonctions sont :

is_array (), is_double (), is_integer (), is_string ()...

2.9.3 Test de l'état d'une variable

PHP fournit plusieurs fonctions pour tester l'état d'une variable.

isset () : cette fonction prend un nom de variable comme argument et renvoie true si la variable existe et retourne également false si la variable n'existe pas. A ce titre, elle est utilisée pour contrôler l'existence des variables issues des formulaires.

unset () : cette fonction supprime la variable qui lui est passée comme argument.

empty () : cette fonction détermine si une variable existe et contient une valeur non vide. empty () retourne true ou false selon le résultat obtenu.

2.9.4 Constantes

Si l'on désire gérer des valeurs fixes, non modifiables, il faut définir des constantes. C'est l'instruction *define ()* qui permet de définir les constantes :

Exemples :

```
define ("AgeMin", 18) ;
define ("PI", 3.14) ;
echo AgeMin ;          /* affiche le nombre 18 */
```

2.10 Types de données du PHP

Le langage PHP prend en charge les types de données de base suivantes :

- **Entier** : utilisé pour les nombres entiers
- **Float** : (aussi appelé Double). Utilisé pour les nombres réels.
- **Chaîne** : utilisé pour les chaînes de caractères.
- **Booléen** : utilisé pour exprimer les valeurs vraies ou fausses.
- **Tableau** : utilisé pour stocker plusieurs éléments de données de même type

2.11 Transtypage (casting)

A l'instar du langage C, il est possible de modifier le type d'une variable en PHP. Le principe consiste à spécifier le type temporaire entre parenthèses, juste avant la variable concernée.

Syntaxe :

```
$new = (type) $old ;
```

On modifier le type d'une variable en créant une nouvelle variable du type voulu, soit en modifiant le type de la variable elle-même.

En fait, à partir d'une variable donnée *$old*, on peut créer une nouvelle variable *$new* qui aura le type désiré selon la syntaxe :

```
$new = (type désiré) $old ;
```

Dans ce cas la valeur affectée à *$new* est modifiée si cela est nécessaire pour qu'elle soit conforme au type désiré. Le type et la valeur de *$old*, eux, ne sont pas modifiés.

Il existe également une autre méthode qui permet de modifier le type d'une variable, mais sans créer une nouvelle variable ce qui implique que le type initial est perdu et que la valeur initiale peut être perdue. On utilise, pour cela, la fonction *settype()*, qui suit la syntaxe :

```
settype($nom_variable, "nom_du_type_final") ;
```

Il est à noter qu'en plus de la fonction *gettype()* qui permet de retourner le type d'une variable, PHP dispose d'un ensemble de fonctions, qui retournent une valeur booléenne, et qui permettent de vérifier le type d'une variable. Ces fonctions sont : *is_array(), is_bool(), is_double(), is_integer(), is_object(), is_ressource(), is numeric().*

Exemple Quelques cas de transtypage sont présentés ci-après:

```
<?php
 $old = 25.7 ; // $old est de type double
 settype($old, "integer");
 echo "\$old a la valeur ", $old, " et elle est de type ", gettype($old),
 "<br>"; //affiche 25
 $old =  "25.7"; // $old est du type string par typage automatique
 $new =(double) $old ; //$new a la valeur décimale 25.7 et le type double
 $c   =(integer) $old ; //$c a la valeur entière 25 et le type integer
?>
```

2.12 Portée d'une variable

La portée d'une variable désigne les emplacements au sein d'un script où la variable est visible.

- Les variables super globales prédéfinies sont visibles à n'importe quel endroit dans un script (exemple *PHP_VERSION, PHP_OS*…).
- Les constantes, une fois déclarées, sont visibles globalement : autrement dit, elles peuvent être utilisées à l'intérieur et à l'extérieur de fonctions.
- Les variables globales déclarées dans un script sont visibles sur tout le script, mais pas à l'intérieur des fonctions.
- Les variables créées à l'intérieur de fonctions et déclarées comme statiques sont invisibles en dehors de la fonction mais conservent leur valeur entre une exécution de la fonction et la suivante.
- Les variables créées à l'intérieur d'une fonction sont locales à la fonction et cessent d'exister lorsque cette dernière se termine.

2.13 Les variables dynamiques

Contrairement à une variable ordinaire dont le nom est fixé une fois pour toute par le programmeur du script, une variable dynamique peut voir son nom défini dynamiquement en cours de script à partir d'une chaîne de caractère qui peut provenir de l'utilisateur ou du contenu d'un champ de base de données.

L'exemple ci-après décrit un script où trois variables sont créées dynamiquement. En fait les variables *$ma_var1*, *$ma_var2*, *$ma_var3* sont définies en ajoutant un numéro *$i* à la valeur de la variable *$chaine* qui est : "*ma_var*". La notation *$$base* désigne la variable dont le nom sera le contenu de la chaîne *$base*. On affecte ensuite des valeurs à ces nouvelles variables. Les variables ainsi créées sont utilisables dans tout le script. Le type de ces variables dépend comme pour les autres variables, de la valeur qui leur est affectée.

```php
<?php
 $chaine = "ma_var" ;
 for ($i=1 ; $i<4 ;$i++){
  $base=$chaine.$i;
  //crée les noms des variables
  $$base=2000+$i ;//affecte les variables
 }
 echo "\$ma_var1 vaut : " , $ma_var1, "<br>";
 echo "\$ma_var2 vaut : " , $ma_var2, "<br>";
 echo "\$ma_var3 vaut : " , $ma_var3, "<br>";
?>
```

2.14 Les tableaux en PHP

Le langage PHP supporte les tableaux indicés numériquement (à l'instar du langage C) et les tableaux associatifs.

- *Les tableaux classiques* : chaque élément du tableau possède un indice numérique qui permet de le repérer.

Exemple :

```php
<?php
      $Articles = array ('Tomates', 'Pommes', 'Carottes', 'Poivrons');
      echo "L'élément d'indice trois est : $Articles[3] " ;
?>
```

Ce script crée un tableau appelé Articles contenant quatre valeurs ('Tomates', 'Pommes', 'Carottes' et 'Poivrons'. La deuxième instruction permet d'afficher le quatrième élément.

- *Les tableaux associatifs:* ils permettent d'utiliser des valeurs plus significatives que des nombres comme l'indice.

L'indice n'est plus nécessairement un nombre mais peut aussi être une chaîne de caractères par laquelle l'élément sera identifié et que l'on appellera la **clé** de l'élément.

```php
<?php
      $LeTablo = array ("Ville" => "Agadir",  "Cité" => "Dakhla");
      echo "La ville choisie est : {$LeTablo["Ville"]}";
?>
```

Remarque :

- PHP permet d'affecter globalement les éléments d'un tableau à un autre.
- On peux enregistrer de manière automatique une série de nombre croissants dans un tableau grâce à la fonction ***range()***

Syntaxe :

```php
$LesEntiersInfACent = range(1,99) ;
```

En PHP, il est possible de redimensionner dynamiquement un tableau.

Illustration :

```
<?php
      $Articles = array ('Tomates', 'Pommes', 'Carottes', 'Poivrons');
      echo "L'élément d'indice trois est : $Articles[3] " ;
      $Articles[4] = 'Oranges' ;
      $Articles[]  = 'Oignons' ;
?>
```

A l'instar du langage C vous pouvez accéder au contenu d'un tableau indicé par une série de nombres en utilisant une boucle for.

```
for ($i = 0; $i < count ($Articles); $i++)
      echo $Articles[$i];
```

PHP utilise une autre boucle spécialement conçue pour être utilisée avec les tableaux c'est la boucle ***foreach***.

Syntaxe :

```
foreach ($Articles as $courant)
      echo $ courant.' ' ;
```

Ce code enregistre tour à tour chacun des éléments de l'ensemble parcouru dans la variable *$courant* et affiche le contenu de celle-ci.

Cas des tableaux associatifs

```
$Prix=array('Tomates'=>7, 'Oignons'=>5, 'Oranges'=> 6, 'Carottes'=>4) ;
```

Cette instruction crée un tableau dont les noms des articles sont les clés et dont les prix sont les valeurs.

Etant donné que dans le tableau Prix, les clés ne sont pas des nombres, il n'est pas possible d'utiliser un compteur simple pour réaliser des itérations sur le tableau. A la place, nous pouvons faire appel à une boucle *foreach* ou aux constructions ***list()*** et ***each()***.

```
foreach ($Prix as $Key => $value)
      echo $Key. ' => ' .$value. '<br>';
```

La fonction *each*() fonction permet de parcourir tous les éléments d'un tableau sans se soucier de ses bornes. Elle retourne la combinaison clé-valeur courante du tableau passé en paramètre, puis se positionne sur l'élément suivant, et cela du premier au dernier indice. Lorsque la fin du tableau est atteinte, *each( )* retourne la valeur faux (false).

La fonction *list()* est très souvent associée à la fonction *each()*, elle permet d'affecter les éléments du tableau dans des valeurs distinctes.

```
$Prix=array('Tomates '=>7, 'Oignons' =>5, 'Oranges' =>6, 'Carottes' =>4);

while($element = each($Prix))
{
      echo $element ['key'];
      echo '-';
      echo $element ['value'];
      echo '<br>';
}
```

Les fonctions de tri

- Tri selon les valeurs
  - La fonction ***sort ()***

    - effectue un tri sur les valeurs des éléments d'un tableau selon un critère alphanumérique : selon les codes ASCII (le caractère a est après Z)
    - Le tableau initial est modifié et non récupérables dans son ordre original
    - Pour les tableaux associatifs les clés seront perdues et remplacées par un indice créé après le tri et commençant à 0

  - La fonction ***rsort()*** effectue la même action mais en ordre inverse des codes ASCII.
  - La fonction ***asort()*** trie également les valeurs selon le critère des codes ASCII, mais en préservant les clés pour les tableaux associatifs
  - La fonction ***arsort()*** la même action mais en ordre inverse des codes ASCII
  - la fonction ***natcasesort()*** effectue un tri dans l'ordre alphabétique non ASCII (le caractère (a est avant z)
- Tri sur les clés
  - La fonction ***ksort()*** trie les clés du tableau selon le critère des codes ASCII, et préserve les associations clé /valeur
  - La fonction ***krsort()*** effectue la même action mais en ordre inverse des codes ASCII

```
<?php
 $tabAuteur=array("1622"=>"Molière","1802"=>"Hugo","1920"=>"Vian",
"1905"=>"Sartre") ;
 ksort ($tabAuteur);
 echo "<h3 >  Tri sur les clés de \$tabAuteur </h3>" ;
 foreach ($tabAuteur as $cle=>$valeur) {
  echo "<b> l'élément a pour clé : $clé; et pour valeur : $valeur</b> <br>";
 }
?>
```

Les fonctions tableaux sont nombreuses et ne seront pas toutes listées dans ce document, on peux tout de même citer quelques unes :

next() => permet de déplacer le pointeur vers l'élément suivant du tableau
end() => retourne le dernier élément du tableau
prev() => positionne le pointeur interne du tableau sur l'élément du tableau situé avant l'élément courant et retourne celui-ci
current() => retourne l'élément courant pointé par le pointeur interne
reset() => replace le pointeur de tableau au premier élément et retourne ce dernier
array_merge() => enchaîne des tableaux entrés en argument afin d'en retourner un seul tableau.

2.15 PHP et les formulaires
2.15.1 Formulaire HTML

Le principal objectif d'un formulaire est la collecte des informations saisies par un utilisateur vers une application serveur. Récupérer les données saisies par un utilisateur dans un tel formulaire se révèle très facile avec PHP mais la méthode employée dépend de la version de PHP que vous utilisez et d'un paramétrage dans votre fichier php.ini.

La création d'un formulaire nécessite la connaissance de quelques balises HTML indispensables :

- Structure : un formulaire commence toujours par la balise <form> et se termine par la balise </form>

Entre ces deux balises se situent les balises qui vont créer les différents types de champ que va contenir le formulaire.

- Champ de saisie de text inligne :

```
<input type = "text" name ="nom_du_champ" value="valeur">
```

- Boutons d'envoi et d'effacement :

```
<input type=" submit " value = "Envoyer">
<input type = "reset" name ="efface" value = "Effacer">
```

- Case à cocher et bouton radio :

```
<input type = "checkbox" name ="case1" value="valeur_case">
<input type = "radio" name ="radio1" value ="valeur_radio">
```

- Liste de sélection avec options à choix unique :

```
<select name ="select" size="1">
   <option value = "un"> choix </option>
   <option value ="deux"> choix2 </option>
</select>
```

- Liste de sélection avec options à choix multiples :

```
<select name ="select" size = "1" multiple>
   <option value = "un"> choix1 </option>
   <option value = "deux"> choix2 </option>
</select>
```

La transmission se fait selon une des deux méthodes d'envoi GET ou POST.). Il s'agit de méthodes implémentées dans le protocole HTTP appelées GET et POST

1. **Méthode GET**

   - *La méthode GET place les informations d'un formulaire directement à la suite de l'adresse URL de la page appelée.* Il s'agit du mode de transmission par défaut. Les paramètres passés au moment de la validation sont ajoutés à l'URL du script appelé.
   - inconvénients :

     - rendre visibles les données dans la barre d'adresse du navigateur.
     - de plus, la longueur totale est limitée à 255 caractères, ce qui rend impossible la transmission d'un volume de données important.

Exemple d'utilisation :

Identite.html

```
<html>
 <head><title> Formulaire identité </title></head>
```

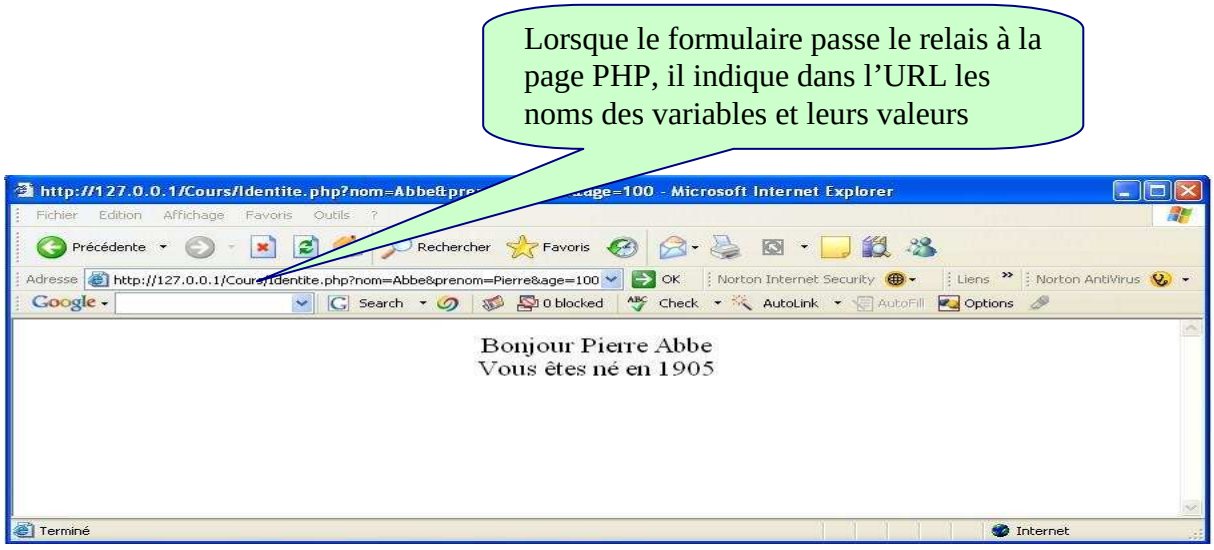
```
  <body>
   <center>
    <FORM action = "Identite.php" method ="get">
     <table>
      <tr>
         <td bgcolor ="#CCCCFF"> Nom </td>
         <td> <INPUT TYPE = "text" NAME = nom  ></td>
      </tr>
      <tr>
         <td bgcolor ="#CCCCFF"> Prénom </td>
         <td><INPUT TYPE = "text" NAME = prenom></td>
      </tr>
         <tr>
         <td bgcolor ="#CCCCFF"> Age </td>
         <td><INPUT TYPE = "text" NAME = age   ></td>
      </tr>
      <tr>
         <td colspan = 2>
    <center>
     <INPUT TYPE = submit VALUE = "Valider">
    </center>
   </td>
      </tr>
     </table>
    </FORM>
   </center>
  </body>
</html>
```

Une fois que le formulaire aura été envoyé, le relais est passé à une autre page Web : Identite.php.

Identite.php

```
<html>
 <?php
   if ($nom <> "" && $prenom <> "" && $age <> ""){
         $LaDateCourante = 2005;
         echo '<center>';
         echo 'Bonjour'.'  ';
         echo "$prenom".'  '."$nom".'<br>';
         echo "Vous êtes né en". "  ";
         echo $LaDateCourante - $age;
         echo '</center>';
   }
```

```
    else{
          echo "<center><b>";
          if($nom == "")
                echo "Le champs nom n'a pas été saisi<BR>\n";
          if ($prenom =="")
                echo "Le champs prénom n'a pas été saisi <BR>\n";
          if ($age == "")
                echo "Le champs age n'a pas été saisi";
    }
 ?>
</html>
```



Lors de la soumission à une page de traitement, chaque élément de saisie est assimilé à une variable PHP dont le nom est constitué par la valeur de l'attribut *name* et son contenu par la valeur de l'attribut value

Remarque :

Quand vous saisissez la chaîne ?nom=Abbe&prenom=Pierre&age=100, vous aurez le même résultat.

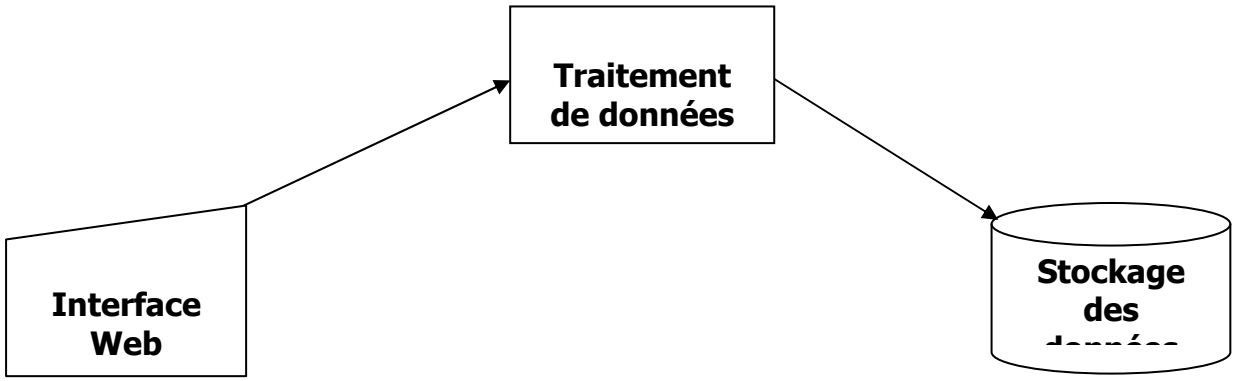
2. Méthode POST

- regroupe les informations dans l'entête d'une requête HTTP
- assure une confidentialité efficace des données

En spécifiant la méthode post pour envoyer les informations, les données sont envoyées dans le corps du message, elles ne sont donc pas visibles dans l'URL. Si vous reprenez les pages Web ci-dessus en utilisant la méthode post, vous aurez le même résultat.

2.16 Stockage et récupération des données

Les données récupérées à partir des formulaires HTML doivent être enregistrées en vue d'un usage ultérieur.



Deux modes de stockage des données sont réellement envisageables : dans des fichiers ''plats'', ou dans une base de données.

Un fichier plat peut être enregistré sous de nombreux formats. Le terme ''fichier plat'' est employé pour désigner un simple fichier texte.

Cette partie est consacrée au stockage des données dans des fichiers sous la forme d'un enregistrement par ligne. Ce mode de stockage est très simple à mettre en œuvre, mais se révèle assez limité (dès que le nombre d'informations à traiter devient assez conséquent, l'usage d'une base de données est recommandé).

Le problème est le suivant :

On veut à partir d'une interface Web stocker les informations saisies dans un fichier de données.

2.17 Présentation des fonctions de traitement des fichiers

L'écriture dans un fichier s'effectue en trois étapes :

1. Ouverture du fichier.
2. Ecriture dans le fichier.
3. Fermeture du fichier.

De la même manière, la lecture des données d'un fichier s'effectue en trois étapes :

1. Ouverture du fichier. Si l'ouverture du fichier se révèle impossible, on doit le signaler par un message adéquat.
2. Lecture des données dans le fichier.
3. Fermeture du fichier.

Ouverture d'un fichier

En PHP, l'ouverture d'un fichier s'effectue au moyen de la fonction *fopen ()*. Lors de l'ouverture d'un fichier, vous devez spécifier le mode de fichier, c'est-à-dire la manière dont vous voulez utiliser le fichier.

Lorsque vous ouvrez un fichier :

1. Vous avez la possibilité de l'utiliser en lecture seule, en écriture seule, ou bien en lecture et en écriture.
2. Pour écrire des données dans le fichier, vous pouvez soit remplacer le contenu existant par vos nouvelles données ("écraser" le contenu), soit ajouter les nouvelles données à la suite du contenu existant.

Utilisation de fopen () pour ouvrir un fichier

Syntaxe :

```
$file = fopen ("identite.txt", 'w') ;
```

Le premier paramètre est le nom du fichier à ouvrir, avec éventuellement le chemin d'accès correspondant.

Le deuxième paramètre à passer à la fonction *fopen ()* est le mode du fichier, lequel doit être spécifié sous la forme d'une chaîne. Celui-ci indique l'usage prévu pour le fichier. Dans l'exemple considéré ici, nous avons donné la valeur 'w' à ce paramètre, ce qui signifie que le fichier doit être ouvert en vue d'y écrire des données. Le tableau ci-dessous récapitule les divers modes de fichier disponibles.

Mode	Nom du module	Signification
r	Lecture	Le fichier est ouvert en lecture, à partir de son début.
r+	Lecture	Le fichier est ouvert en lecture et en écriture, à partir de son début.
w	Ecriture	Le fichier est ouvert en écriture, à partir de son début. Si le fichier existe déjà, son contenu est écrasé. Dans le cas contraire, le fichier est crée.
w+	Ecriture	Le fichier est ouvert en écriture et en lecture, à partir de son début. Si le fichier existe déjà, son contenu est écrasé. Dans le cas contraire, le fichier est crée.
x	Ecriture prudente	Le fichier est ouvert pour écriture, en commençant au début du fichier. Si le fichier existe déjà, il n'est pas ouvert, fopen () retourne false et PHP génère un avertissement.
x+	Ecriture prudente	Le fichier est ouvert pour écriture et lecture, en commençant au début du fichier. Si le fichier existe déjà, il n'est pas ouvert, fopen () retourne false et PHP génère un avertissement.
a	Ajout	Le fichier est ouvert pour ajout (écriture) uniquement, en commençant à la fin du contenu existant. Si le fichier n'existe pas PHP tente de le créer.
a+	Ajout	Le fichier est ouvert pour ajout (écriture) et lecture, en commençant à la fin du contenu existant. Si le fichier n'existe pas PHP tente de le créer.

Lorsque l'ouverture d'un fichier avec la fonction *fopen ()* réussit, celle-ci retourne un pointeur sur le fichier, lequel doit être enregistré dans une variable. Cette variable permet ensuite d'accéder au fichier pour y lire ou y écrire des données.

Lorsque l'appel de la fonction *fopen ()* échoue, celle-ci retourne la valeur false. Vous pouvez traiter l'erreur survenue avec plus de convivialité en supprimant le message d'erreur PHP et en générant votre propre message d'erreur :

```
@$file = fopen ("identite.txt", 'w') ;
if( !$file)
{
 echo '<p><strong>Le fichier ne peut pas être ouvert  en ce moment.'
 . '</strong></p>';
 exit ;
}
```

La présence du symbole @ avant l'appel de la fonction *fopen ()* informe l'interpréteur PHP qu'il doit supprimer toute erreur résultant de l'appel de la fonction.

L'instruction if teste la variable $file pour déterminer si un pointeur de fichier valide a été retourné par la fonction *fopen ()*. Si ce n'est pas le cas, elle affiche un message d'erreur et interrompe l'exécution du script.

Ecriture dans un fichier

En PHP, l'écriture des données dans un fichier peut se faire par la fonction *fwrite*

```
fwrite ($File, $ChaineDeSortie, strlen($ChaineDeSortie)) ;
```

Cette instruction demande à l'interpréteur PHP d'écrire la chaîne stockée dans la variable $ChaineDeSortie dans le fichier sur lequel pointe $File.

Formats de fichiers

Lors de la création d'un fichier de données, le choix du format de stockage des donnés vous appartient (bien sûr, si vous prévoyez d'utiliser le fichier de données avec une autre application, vous devez en tenir compte dans votre choix).

Pour construire une chaîne représentant un enregistrement du fichier de données identite.txt, nous pouvons procéder comme ceci :

```
$lidentite = $vnom ."\t" .$vprenom ."\t" .$vage . "\n";
```

Fermeture d'un fichier

Une fois que vous avez fini avec un fichier, vous devez le fermer au moyen de la fonction *fclose ()*, comme suit :

```
fclose ($File) ;
```

Lecture dans un fichier

Il est possible de lire le contenu d'un fichier grâce à la fonction *fgets ()* qui permet de récupérer une ligne du fichier. Elle renvoie 0 en cas d'échec, 1 dans le cas contraire

Syntaxe :

```
fgets (entier Etat_du_fichier, entier Longueur);
```

Le paramètre Longueur désigne le nombre de caractères maximum que la fonction est sensée récupérer sur la ligne

Pour récupérer l'intégralité du contenu d'un fichier, il faut insérer la fonction fgets () dans une boucle *while*. On utilise la fonction *feof (),* fonction testant la fin du fichier.

Lecture de l'intégralité du contenu d'un fichier : readfile (), fpassthru () et file ()

L'appel de la fonction *readfile ()* ouvre le fichier, affiche son contenu sur la sortie standard, puis ferme le fichier

```
readfile ("identite.txt") ;
```

La fonction **fpassthru ()** permet d'envoyer le contenu d'un fichier dans la fenêtre du navigateur.

```
fpassthru ($file);
```

Elle permet d'envoyer le contenu du fichier à partir de la position courante dans le fichier. Elle n'ouvre pas automatiquement un fichier. Il faut donc l'utiliser avec *fopen ()*.

La fonction *file ()* permet de retourner dans un tableau l'intégralité d'un fichier en mettant chacune de ces lignes dans un élément du tableau

Autres fonctions utiles pour la manipulation des fichiers

Vérification de l'existence d'un fichier : *file_exists ()*. Cette fonction permet de déterminer si un fichier existe, sans même l'ouvrir

```
if (file_exists ("identite.txt"))
      "On peut manipuler le fichier";
else
      "Le fichier identité n'existe pas" ;
```

filesize ("identite.txt") : cette instruction détermine la taille en octets d'un fichier grâce à la fonction.

unlink ("identite.txt") : cette instruction détruit le fichier identité.

Navigation dans un fichier : rewind (), fseek () et ftell ()

rewind () : déplace le pointeur de fichier au début du fichier.

ftell () : retourne la position du pointeur dans le fichier, en nombre d'octets comptés depuis le début du fichier.

2.18 Chaînes de caractères

Ci-dessous les principales fonctions de manipulations de chaine de caractères sont décrites :

- ***strpos* (chaine, sous chaine)** : retourne la position de la sous chaîne dans la chaîne. Dans le cas où la chaîne existe en plusieurs exemplaires, c'est la position de la première occurrence qui est retournée. *strrpos* retourne quand à elle la position de la dernière occurrence.
- ***strstr* (chaine, sous chaine)** retourne la portion de la chaîne à partir de la première occurrence de la sous chaîne.
- ***strtolower*|*strtoupper* (chaine)** : retourne la chaîne passée en paramètres an minuscules (resp. majuscules).
- **str_replace (car d'origine, car de destination, chaine)** : remplace le caractère d'origine par le caractère de destination dans la chaîne.
- **strcmp (chaine1, chaine2)** : compare les chaînes 1 et 2. Elle retourne 0 si les deux chaînes sont égales. Si chaine1 vient après (ou supérieure à) chaine2 dans l'ordre lexicographique, la fonction *strcmp* retourne un nombre supérieur à zéro ; si c'est l'inverse, elle retourne un nombre inférieur à zéro. Cette fonction est sensible à la casse.
- ***strlen* (chaine)** : retourne la taille de la chaîne.
- ***strcasecmp*** : elle identique à *strcmp*, si ce n'est qu'elle est insensible à la casse.
- ***substr ()*** : permet d'accéder à une sous chaîne d'une chaîne, en spécifiant les points de début et de fin de la sous chaîne.

  Soit la chaîne *$Politesse = "Bonjour tous le monde";*

  substr ($politesse, 0) retourne ***Bonjour tous le monde***. Les positions dans les chaînes sont numérotées à partir de zéro.

  ($politesse, -8) retourne ***le monde***.

  ($politesse, 0, 5) retourne ***Bonjo***.

  Exo à partir de la chaîne '21/11/2005' construisez la chaîne '21/11/05'

  - ***explode ()*** : découpe une chaîne en fragments à l'aide d'un délimiteur et retourne un tableau.

  $email_array = explode('@', $email) ;

  L'exécution de cette instruction conduit à la division de l'adresse de courrier électronique en deux parties : le nom de l'utilisateur, qui est enregistré dans $email_array [0], et le nom de domaine, qui est enregistré dans $email_array [1].

L'effet produit par la fonction *explode ()* peut être annulé avec les fonctions ***implode ()*** ou ***join ()***. Ces deux fonctions sont identiques.

$new_array = implode('@', $email_array) ;

2.19 Gestion de la date et de l'heure

- ***getdate ()*** : cette fonction retourne les éléments de date et d'heure courante dans un tableau associatif. Ci-dessous, les clés et les valeurs du tableau associatif.

Clé	Valeur
seconds	Secondes, numérique
Minutes	Minutes, numérique
Hours	Heures, numérique
Mday	Jour du mois, numérique
Wday	Jour de la semaine, numérique
Mon	Mois, numérique
Year	Année, numérique
Yday	Jour de l'année, numérique
Weekday	Jour de la semaine, format texte
Month	Mois, format texte

- ***checkdate ()*** : cette fonction s'emploie pour s'assurer de la validation de d'une date. Elle se révèle particulièrement utile pour vérifier les dates saisies par l'utilisateur. Elle prend en paramètre le mois, le jour et l'année et retourne true ou false selon la validité ou non de la date passée en argument.
- ***date ()*** : est une autre fonction PHP de détermination de date, elle retourne une chaîne de caractères date/heure selon le format.

Les codes de format reconnus par la fonction date () sont donnés dans le tableau ci-dessous.

Code	Description
a	Matin ou après-midi, représenté sous la forme de deux caractères en minuscules : respectivement "am" et "pm"
A	Matin ou après-midi, représenté sous la forme de deux caractères en majuscules : respectivement "AM" et "PM"
d	Jour du mois, sous la forme d'un nombre à deux chiffres, éventuellement préfixé par un zéro. La plage autorisée s'étend de "01" à "31".
D	Jour de la semaine, abrégé en trois lettres : exemple "Fri" pour (vendredi)
F	Mois de l'année en anglais, au format texte, version longue.
h	Heure du jour, exprimée dans le système à 12 h.
H	Heure du jour, exprimée dans le système à 24 h.
i	Minutes, si nécessaires préfixées avec un zéro. La plage autorisée s'étend de "00" à "59".
j	Jour du mois au format numérique, sans zéro en préfixe. La plage autorisée s'étend de "1" à "31".
l	Jour de la semaine en anglais, au format texte, version longue.
m	Mois de l'année, sous la forme d'un nombre à deux chiffres, éventuellement préfixés par un zéro. La plage autorisée s'étend de "01" à "12".
M	Mois de l'année en anglais, dans un format texte abrégé en trois lettres.
n	Mois de l'année, sous la forme d'un nombre à deux chiffres, sans zéro en préfixe. La plage autorisée s'étend de "1" à "12".
s	Secondes avec si nécessaire un zéro en préfixe. La plage autorisée s'étend de "00" à "59".
t	Nombres total du jours dans le mois donné. La plage autorisée s'étend de "28" à "31".
T	Fuseau horaire du serveur
U	Nombres total de secondes qui se sont écoulées depuis le 1er janvier 1970 jusqu'au moment considéré.

w	Jour de la semaine sous la forme d'un seul chiffre. La plage autorisée s'étend de "0" (dimanche) à "6" (samedi).
W	Numéro de la semaine dans l'année.
y	Année exprimée en deux chiffres, par exemple "04"
Y	Année exprimée en quatre chiffres, par exemple "2004"
z	Jour de l'année sous forme d'un nombre. La plage autorisée s'étend de "0" à "365"

- ***mktime ()*** : convertit une date et une heure en un entier contenant le nombre de secondes écoulées depuis le 1 Janvier 1970 (époque UNIX).

entier mktime*(entier heure, entier minute, entier seconde, entier mois, entier jour, entier année).*

Si l'heure n'a pas d'importance dans votre le contexte de votre application, vous pouvez passer la valeur zéro pour chacun des trois premiers paramètres, auquel cas l'interpréteur PHP les définira automatiquement sur les valeurs courantes.

```
<?php
   $date=date('d/m/y', mktime(0,0,0,date('m'),date('d')-1,date('y')));
   echo $date;
?>
```

ce code affiche la date d'hier au format d/m/r '21/11/05'.

3 Interfaçage de PHP et MySQL

3.1 Introduction

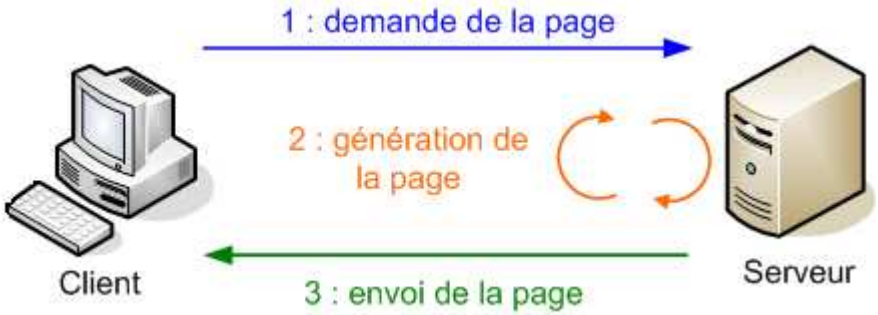
Ce document décrit les éléments indispensables pour travailler avec une base de données Web. Le système de gestion de base de données que l'on va utiliser pour cette partie est MySQL.

Pour mieux tirer profit des fonctionnalités qu'offrent les bases de données Web, il est nécessaire de connaître le langage PHP qui permet la connexion à MySQL et le langage SQL qui permet la manipulation de vos données.

3.2 Architecture d'une base de données Web

Ce paragraphe présente l'architecture externe des systèmes de bases de données Web et présente la méthodologie permettant de développer ces systèmes.

Le fonctionnement fondamental d'un serveur Web est présenté à la figure 3. Ce système est composé de deux objets : un navigateur Web et un serveur Web. Un lien de communication doit exister entre ces deux objets. Le navigateur effectue des requêtes auprès du serveur et le serveur renvoie des réponses. Cette architecture est parfaitement adaptée à un serveur qui fournit des pages statiques. L'architecture qui permet de servir des sites Web faisant intervenir des bases de données est un peu plus complexe.



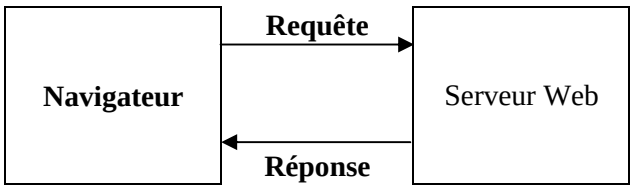


Figure 3 La relation client/serveur entre un navigateur Web et un serveur Web nécessite un lien de communication.

Les applications de bases de données Web ont la structure représentée par la figure 4 ci-dessous.

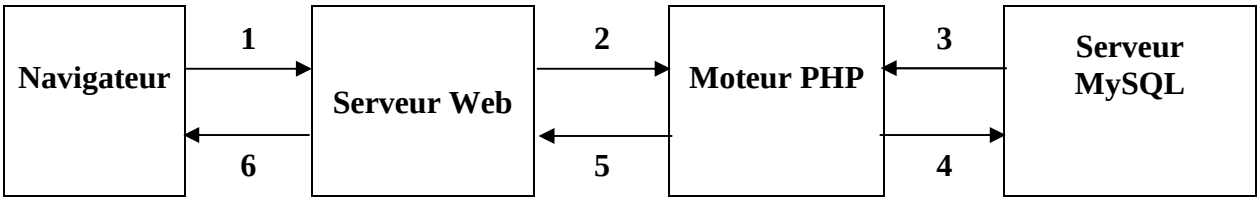


Figure 4 L'architecture fondamentale des bases de données Web est composée d'un navigateur Web, d'un serveur Web, d'un moteur de scripts et d'un serveur de bases de données.

Une transaction de base de données Web typique est composée des étapes suivantes, qui sont numérotées à la figure 4.

1. Le navigateur Web d'un utilisateur envoie une requête http pour une page particulière. Par exemple, cette requête peut concerner tous les étudiants de SMI5 et être envoyée à partir d'un formulaire HTML. La page de recherche des résultats est appelée ***results.php***.
2. Le serveur Web reçoit la requête pour ***results.php***, récupère le fichier et le passe au moteur PHP afin qu'il soit traitée.
3. Le moteur PHP commence à analyser le script. A l'intérieur de ce script se trouve une commande permettant de se connecter à la base de données et d'exécuter une requête. PHP ouvre une connexion vers le serveur MySQL et transmet la requête appropriée.
4. Le serveur MySQL reçoit la requête de base de données et la traite, puis renvoie les résultats au moteur PHP.
5. Le moteur PHP termine l'exécution du script, ce qui consiste généralement en un formatage des résultats de la requête en HTML. Il renvoie ensuite le fichier HTML obtenu au serveur Web.
6. Le serveur Web transmet la page HTML au navigateur, pour que l'utilisateur puisse voir la liste des étudiants de SMI5.

3.3 Création de la base de données Web

Pour pouvoir créer une base de données Web, vous devez avoir accès à MySQL. Cela signifie normalement que :

Vous avez effectué l'installation de la base de MySQL sur votre serveur Web. Pour cela, il faut notamment :

- installer les fichiers
- configurer un utilisateur sous le nom duquel MySQL sera exécuté ;
- configurer votre chemin d'accès ;
- définir le mot de passe root ;
- supprimer l'utilisateur anonyme ;
- …

3.4 Ouverture d'une session MySQL

Pour démarrer une session MySQL, ouvrez une interface de type ligne de commande sur votre ordinateur et saisissez la ligne suivante :

```
mysql/bin > mysql –h hostname –u username –p
```

La commande mysql invoque le moniteur MySQL. Il s'agit d'un client en ligne de commande qui vous connecte au serveur MySQL.

L'option –h est utilisée pour spécifier l'hôte auquel vous souhaitiez vous connecter.

L'option –u sert à spécifier le nom de l'utilisateur sous lequel vous souhaitez vous connecter.

L'option –p indique au serveur que vous souhaitez vous connecter en utilisant un mot de passe.

3.5 Création des bases de données

Le système de base de données de MySQL peut prendre en charge différentes bases de données. Généralement, il existe une base de données par application.

Une base de données peut être créée en ligne de commande par la commande suivante :

```
mysql > create database comptes;
```

3.6 Choix de la base de données

Tout d'abord vous devriez avoir ouvert une session sous un compte d'utilisateur MySQL, soit parce que vous avez configuré cet utilisateur, soit parce que votre administrateur de serveur Web l'a créé pour vous.

L'étape suivante consiste à spécifier la base de données avec laquelle vous souhaitez travailler. Pour ce faire, saisissez la commande suivante :

```
mysql > use comptes;
```

Ou bien lors de l'ouverture de votre session, saisissez la commande suivante :

```
mysql/bin > mysql -D comptes –h hostname –u username –p
```

Remarque

1. Il est important de configurer un utilisateur pour que vos scripts PHP puissent se connecter à MySQL.
2. L'utilisateur root doit généralement servir uniquement pour l'administration du système, à cause de certains problèmes de sécurité.
3. Pour chaque utilisateur qui doit avoir accès au système, vous devez définir un compte et un mot de passe.
4. Vous devez définir un système de privilèges pour les utilisateurs qui accèdent au système.
5. Un utilisateur (ou un processus) doit posséder le niveau de privilège le plus bas possible pour pouvoir effectuer correctement sa tâche.

La commande ***grant*** est utilisée pour accorder des droits d'accès aux utilisateurs de MySQL.

L'inverse de *grant* est ***revoke***, cette commande est utilisée pour supprimer les privilèges des utilisateurs.

Exemples d'utilisation de grant et revoke

Pour configurer le compte d'un administrateur, vous pouvez saisir la commande suivante :

```
mysql > grant all
     -> on *
     -> to V1 identified by 'V1'
     -> with grant option;
```

Cette commande accorde tous les privilèges sur toutes les bases de données à un utilisateur appelé *V1*, avec le mot de passe *V1* et l'autorise à transmettre ces privilèges.

Cet utilisateur n'a vraiment pas la raison d'être, votre système à besoin d'un seul super utilisateur (*root*), c'est pourquoi vous pouvez le supprimer immédiatement :

```
mysql > revoke all privileges, grant
   -> on *
   -> from V1;
```

Pour configurer le compte d'un utilisateur classique, sans aucun privilèges :

```
mysql > grant usage
     -> on comptes.*
     -> to sgully identified by 'Lly';
```

Une fois, sgully définit ce qu'il souhaite réellement faire, nous pouvons lui fournir les privilèges correspondant :

```
mysql > grant select, insert, update, create, alter, drop
     -> on comptes.*
     -> to sgully;
```

Si l'on se rend compte que sgully abuse de ces privilèges, on peut les réduire :

```
mysql > revoke alter, drop, create
     -> on comptes.*
     -> to sgully;
```

S'il s'avère, que l'utilisateur sgully n'a plus besoin d'utiliser la base de données, on peut supprimer tous ces droits :

```
mysql > revoke all
     -> on comptes.*
     -> to sgully;
```

3.7 Création des tables de la base de données

Pour créer une table, vous pouvez vous servir de la commande SQL create table. Voici le format général de l'instruction create table :

```
mysql > create table comptes (colonnes) ;
```

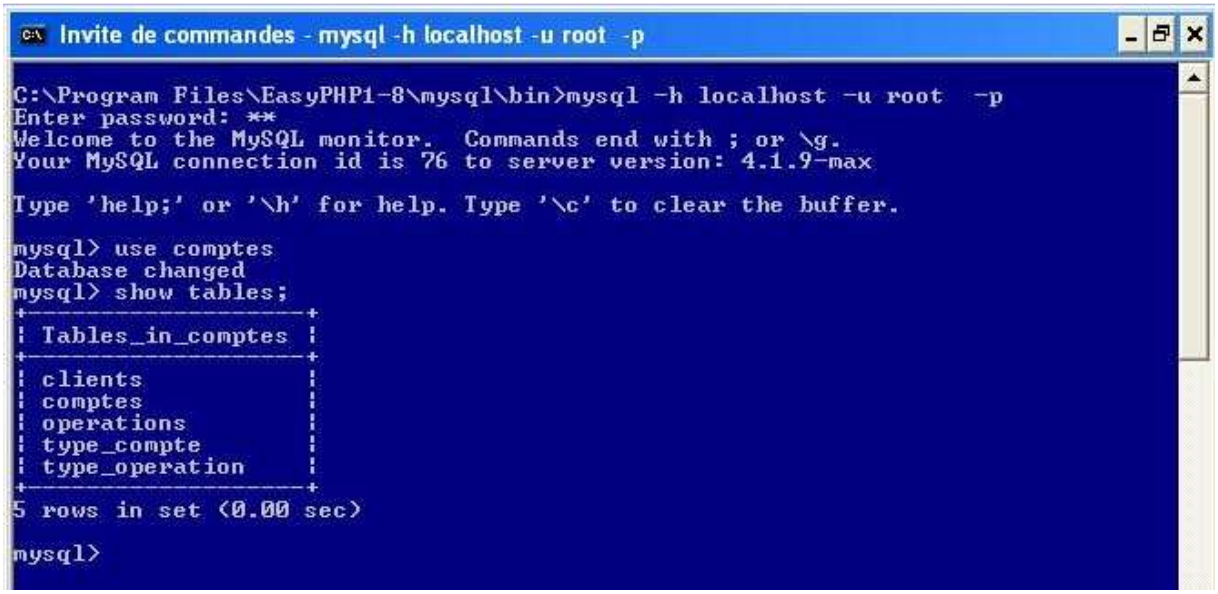
Pour créer la base de données *comptes*, vous allez exécuter le script *create_DB_Comptes.sql* (me demander le script) soit en ligne de commande ou bien en utilisant l'interface phpMyAdmin.

Vous pouvez visualisez le contenu de votre base de données en utilisant la commande *show* ou *describe.*

Ouvrez une session et sélectionner la base de données *comptes*. Vous pouvez afficher les tables de cette base en saisissant la commande suivante :

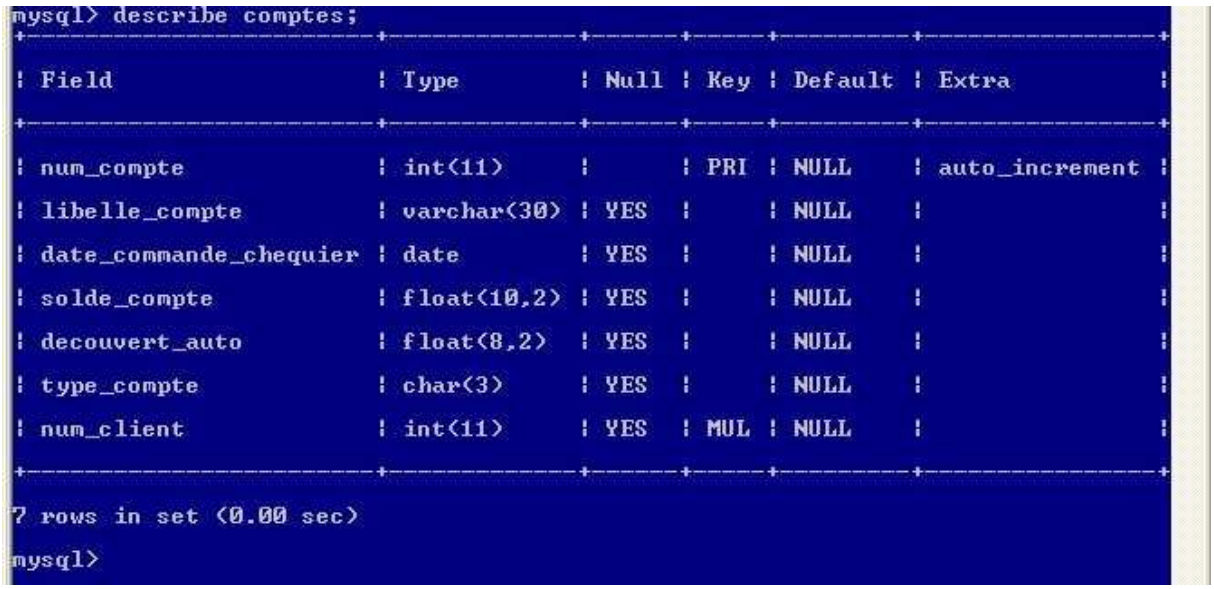
```
mysql > show tables;
```

MySQL affiche alors la liste de toutes les tables de cette base de données :



Pour afficher plus d'informations sur une table particulière, par exemple la table *comptes*, servez vous de la commande *describe* :

```
mysql > describe comptes;
```



3.8 Types de données manipulés par MySQL

MySQL reconnaît trois principaux types de données dans les colonnes : des valeurs numériques, des dates et des heures, et des chaînes de caractères. Dans chacune de ces catégories il existe plusieurs types différents représentés dans les tableaux ci-dessous.

Type	***Taille (octets)***	***Description***
TINYINT [(M)]	1	Entiers très courts
BIT		Synonyme de TINYINT
BOOL		Synonyme de TINYINT
SMALLINT [(M)]	2	Entiers courts

MEDIUMINT [(M)]	3	Entiers de taille moyenne
INT [(M)]	4	Entiers classiques
INTEGER [(M)]		Synonyme de INT
BIGINT [(M)]	8	Entiers larges

Tableau 1 : Types de données entiers

Type	***Taille (octets)***	***Description***
FLOAT [(M, D)]	4	Nombres à virgule flottante en simple précision.
DOUBLE [(M, D)]	8	Nombres à virgule flottante en double précision.
REAL [(M, D)]		Synonyme de DOUBLE [(M, D)]

Tableau 2 : Types de données à virgule flottante

MySQL prend en charge plusieurs types de dates et d'heures, qui sont présentés dans le tableau ci-dessous

Type	***Affichage***
DATE	YYYY-MM-DD
TIME	HH:MM:SS
DATETIME	YYYY-MM-DDHH:MM:SS
TIMESTAMP	YYYYMMDDHHMMSS
TIMESTAMP (14)	YYYYMMDDHHMMSS
TIMESTAMP (12)	YYMMDDHHMMSS
TIMESTAMP (10)	YYMMDDHHMM
TIMESTAMP (8)	YYYYMMDD
TIMESTAMP (6)	YYMMDD
TIMESTAMP (4)	YYMM
TIMESTAMP (2)	YY
YEAR [(2|4)]	

Tableau 3 : Les types de dates et heures

Les types de chaînes de caractères manipulés par MySQL sont :

- **CHAR** : chaîne de caractère de taille fixe avec une longueur comprise entre 0 et 255.
- **VARCHAR** : chaîne de caractère de taille variable.
- **BLOB** : « Binary Large OBjects », ce type correspond à des données binaires qui peut contenir des images ou des sons.
- **TEXT** : pour représenter un champ text de taille normal.

3.9 Accès à votre base de données MySQL à partir du Web, avec PHP

Après avoir crée votre base de données, vous pouvez écrire le code PHP qui permet d'exécuter les étapes décrites au paragraphe 2.

PHP sert en effet d'intermédiaire entre vous et MySQL. Problème : PHP ne peut pas dire à MySQL dès le début "Récupère-moi ces valeurs". En effet, MySQL demande d'abord un nom d'utilisateur et un mot de passe. S'il ne faisait pas ça, tout le monde pourrait accéder à votre BDD et lire les informations qu'il y a dedans (parfois confidentielles !).

Il va donc falloir que PHP s'authentifie, on dit qu'il établit une connexion avec MySQL. Une fois que la connexion sera établie, vous pourrez faire n'importe quelle opération sur votre base de données.

PHP fournit un grand choix de fonctions permettant de manipuler les bases de données. Toutefois, parmi celles-ci quatre fonctions sont essentielles :

- La fonction de connexion au serveur
- La fonction de choix de la base de données
- La fonction de requête
- La fonction de déconnexion

Par exemple, avec le SGBD *MySQL*, ces fonctions sont les suivantes:

- ***mysql_connect***
- ***mysql_select_db***
- ***mysql_query***
- ***mysql_close***

1. La connexion à MySQL

Pour vous connecter, vous utiliserez une fonction PHP : *mysql_connect*. Cette fonction nécessite trois paramètres qu'il vous faudra renseigner :

- **$host** : Le nom de l'ordinateur l'hôte (ordinateur sur lequel le SGBD est installé) ;
- **$user** : Le nom de l'utilisateur ;
- **$passwd** : Le mot de passe.

Aussi longtemps que vous testez vos bases de données en local, ces paramètres sont les suivants : *''localhost''*, *''root''* et *''''*.

Le code suivant permet d'établir une connexion à MySQL :

```
<?

   mysql_connect("localhost","root”,"mdp");
?>
```

Pour ne pas avoir à modifier de dizaines de pages quand c'est nécessaire, il est conseillé de créer un fichier *.inc* dans un but de réutilisation. A l'intérieur de ce fichier que vous pouvez baptiser *sql.inc* placez les instructions suivantes :

```
<?

     $host    = “localhost”;

     $user    = “root”;

     $passwd  = “mdp”;

?>
```

Vous pouvez ainsi, à l'intérieur de toutes vos pages exploitant MySQL, placer la séquence suivante :

```
include("sql.inc");

mysql_connect($host, $user, $passwd);
```

2. Sélection d'une base de données et affichage de son contenu

Une fois la connexion effectuée, il faut ouvrir une de vos bases de données. Il faut utiliser pour cela la fonction *mysql_select_db* :

```
$bdd = mysql_select_bd("comptes");
```

Cette fonction renvoie une valeur *TRUE* ou *FALSE* selon qu'elle a aboutit ou non. Il est donc d'usage d'ajouter un test pour vérifier que la base a été ouverte.

```
$bdd = mysql_select_bd("comptes");

if (!$bdd)

   echo 'impossible d'ouvrir la base comptes';
```

Si tout se passe bien, vous pouvez lancer une requête SQL en utilisant la fonction *mysql_query ()* afin de lire le contenu d'une table telle que clients. La requête placée entre les parenthèses est une requête SQL *SELECT*.

Pour dire à MySQL que contient la table *clients* on utilise la fonction *mysql_query ()* comme ceci :

```
$result = mysql_query("SELECT * FROM clients");
```

Cette instruction lit les données de la table, mais elle ne les affiche pas sur la sortie. Pour afficher les lignes de la requête une à une, il faut utiliser l'instruction *mysql_fetch_object ()*. Une fois cette fois fonction est appelée, elle permet d'accéder au contenu des colonnes de la table par leur nom. Ci-dessous un exemple de l'utilisation de la fonction *mysql_fetch_object ()*.

```
while($line = mysql_fetch_assoc($result)){

   echo $line['date_derniere_consultation'].'<br>';

   echo $line['num_client'];

}
```

3. Déconnexion de la base de données

Pour fermer une connexion de base de données vous pouvez utilisez la fonction *mysql_close ()*. L'utilisation de cette commande n'est pas strictement nécessaire, car la connexion est de toute façon fermée lorsqu'un script termine son exécution.

3.10 Principales fonctions pour l'accès à MySQL

Voici la liste des fonctions intégrés à PHP et spécialisées dans l'accès aux bases MySQL. Chacune fait l'objet d'une description dans la documentation disponible sur le site PHP à http://www.php.net/docs.php. La traduction française à jour se trouve à

$connexion=mysql_connect ($serveur, $login, $mdp)	L'utilisateur $login ouvre une connexion au serveur $serveur, et reçoit en retour l'identifiant $connexion
mysql_close ($connexion)	Ferme la connexion MySQL identifiée par $connexion
mysql_select_db($bd,$connexion)	Sélectionne la base de données nommée $bd
$resultat = mysql_query ($requete [, $connexion])	La requête SQL $requete, écrite sous forme de chaîne, est envoyée à la base déjà sélectionnée, et reçoit un identifiant de résultat. Celui-ci est un pseudo booléen TRUE si la requete a été correctement exécutée (même si aucune ligne n'a été modifiée ou sélectionnée). En cas de requete SELECT, $resultat permet d'accéder aux données extraites.
mysql_db_query ($bd, $requete [, $connexion])	Idem, mais la base est d'abord sélectionnée
$nbchamps=mysql_num_fields($resultat)	Retourne dans $nbchamps le nombre de champs du résultat
$nblignes = mysql_num_rows ($resultat)	Retourne dans $nblignes le nombre de lignes du résultat
$nb=mysql_affected_rows($connexion)	Retourne le nombre de lignes affectées lors de la dernière requête de type INSERT, UPDATE, DELETE, CREATE ou DROP
$ligne = mysql_fetch_row($resultat)	Retourne la ligne suivante du résultat dans un tableau indicé, dont les éléments contiennent les valeurs de cette ligne
$ligne = mysql_fetch_array($resultat)	Retourne la ligne suivante du résultat dans un tableau associatif.
$ligne = mysql_fetch_object ($resultat)	Retourne la ligne suivante du résultat, dans un objet dont les attributs sont exactement les champs obtenus
$liste_tables = mysql_list_tables($bd)	Retourne la liste des tables de la base $bd
$nom = mysql_db_name($resultat,$i)	Retourne le nom de la $i ème base de donnée
$tab[$i] = mysql_tablename($liste_tables, $i)	Lit le nom du champ numéro $i dans la table
mysql_data_seek ($resultat,0)	Déplace le pointeur interne de résultat au début du résultat
$num=mysql_errno()	Retourne le numéro de message d'erreur de la dernière opération MySQL.
$texte=mysql_error()	Retourne le texte associé avec l'erreur générée lors de la dernière requête.

4 Travaux pratiques

PHP – TP N°0

L'objectif de ce TP est de se familiariser avec les bases du langage PHP.

Pré-requis : langage de description html

Exercice N°1 : **Affichage en php**

1. Après avoir installé et configurer le programme ***EasyPHP***, Ecrire un script en langage php (**hello.php**) qui permet d'afficher la phrase *«Hello World»*. Appeler ce script par un browser et vérifier qu'il s'exécute correctement.
2. Editer le code source à partir de votre navigateur. Commenter.
3. Utiliser les différents styles de balisage vus dans le cours pour afficher *«Hello World»*.

   Vous pouvez dupliquer le fichier **hello.php** en quatre autres fichiers *(*exemple *hello_js.php, hello_asp.php, hello_xml.php et hello_php.php).*

   Rendez vous dans le fichier php.ini :

   - Positionner les clefs *short_open_tag* et *asp_tags* à *On* puis exécuter les scripts.
   - Positionner les clefs *short_open_tag* et *asp_tags* à *Off* ensuite exécuter les scripts.

   Commenter.

4. Insérer une ligne de commentaire dans votre script **hello.php**.

Exercice N°2 : **Variables, constantes et portées.**

1. Ecrire un script (**cercle.php**) qui permet de calculer et d'afficher la surface d'un cercle de Rayon 4 cm (utiliser la fonction **define** pour donner une valeur approchée à Pi).
2. Réécrire le script **cercle.php** en utilisant une fonction *(surafceCercle)*.
3. Etudier la visibilité de la variable «*rayon*». Je vous conseil de faire des traces avant, pendant et après l'exécution de la fonction *surafceCercle*.
4. Reprendre la question 3 avec la variable prédéfinie *DOCUMENT_ROOT* ou *REMOTE_ADDR*.

Exercice N°3 : **Variables statiques**

Une variable statique a une portée locale, en revanche elle garde sa valeur lorsque le script appelle la fonction.

Les variables statiques sont essentielles lorsque vous faîtes des appels récursifs à une fonction.

1. Qu'affiche le code source suivant

```
<?php
     function cumul ($prix) {
          $cumul = 0 ;
```

```
            $i     = 1 ;
            echo "Total des achats $i = ";
            $cumul += $prix;
            $i++ ;
            return $cumul ;
        }
        echo cumul (175), "<br />" ;
        echo cumul (65) , "<br />" ;
        echo cumul (69) , "<br />" ;
?>
```

2. Modifier le script ci-dessus en précédant la définition des variables $cumul et $i par le mot ***static***. Ré-exécuter le script et commenter.

<u>Exercice N°4</u> : **Variables dynamiques**

Ecrire un script «**dynamique.php**» qui permet d'afficher le résultat suivant :

Indication :

Définir les variables dynamiques $ma_var1, $ma-var2, $ma_var3 en ajoutant un numéro $i à la valeur à une variable auxiliaire : "ma_var".

<u>Mini projet</u> : **Création d'un compteur de visite**

Ecrivez un script PHP **compteur.php** qui, à chaque appel, incrémente le nombre d'appel de la page. Ce nombre sera stocké dans un fichier texte **valeur_compteur.txt**.

PHP – TP N°1

Exercice 1

Exécuter le script suivant :

```
<?php
   echo ("<I>Bonjour</I> tout le monde !<BR>\n");
   echo ("Cet affichage est <U> produit </U> par<B>PHP</B>,<BR>\n");
   echo ("il contient des <FONT COLOR=\"red\">balises HTML</FONT>.\n");
?>
```

Exercice 2

Modifier le script ci-dessous à fin qu'il affiche les éléments des tableaux,

Exploiter au maximum les fonctions de manipulations de tableaux (trie, navigation, …).

```
<?php

   $article1 = array ("3.0","4.5","5.0","4.0","3.5") ;

   $article2 [0] = "3.0";

   $article2 [1] = "4.5";
   $article2 [2] = "5.0";
   $article2 [3] = "4.0";
   $article2 []  = "3.5";

   $article3 = array ("Carottes"=>"3.0", "Tomates"=>"4.5", "Oignons"=>"5.0",

                "Navets"=>"4.0", "Courgettes"=>"3.5");

?>
```

Exercice 3 : Stockage et récupération de données

Ecrire un script PHP **compteur.php** qui, à chaque appel, incrémente le nombre d'appel de la page. Ce nombre sera stocké dans un fichier texte **valeur_compteur.txt**.

Exercice 4 : Utilisation de PHP pour afficher le contenu d'un fichier de commandes

Créez un fichier texte comportant quelques lignes de commandes clients. A chaque ligne correspond une commande.

A titre d'exemple vous pouvez insérer les lignes ci-dessous dans votre fichier « txt »

202005001 | 1236 | 20 octobre 2008 | Tomates | 4 | 4.0
212005001 | 1235 | 21 octobre 2008 | Tomates | 6 | 3.5
212005002 | 1234 | 21 octobre 2008 | Tomates | 8 | 3.0

Ecrire un script PHP qui permet de charger l'intégralité du fichier « txt » dans un tableau, ensuite afficher le contenu du fichier en utilisant les fonctionnalités des tableaux PHP.

Exercice 5

Créer une fonction qui prend en paramètre une chaîne $string et un délimiteur $car. La fonction aura comme tâche de couper la chaîne en utilisant le délimiteur et renvoie le résultat sous forme d'un tableau.

Exercice 6 : Utilisation de PHP pour séparer, mettre en forme et afficher les commandes clients

Une Centrale d'achats, souhaite centraliser les commandes clients provenant de ces points de ventes à fin des les traitées ensemble.

En utilisant l'exercice 4, écrire le script PHP qui permet l'affichage des commandes clients ci-dessous. Il vous est demandé de scinder chaque ligne et de mettre en forme le résultat sous la forme ci-dessous.

La fonction *explode* permet de scinder une chaîne de caractère en plusieurs parties

explode (« caractère délimiteur », chaîne de donnée)

Numéro de commande	Numéro client	Date de commande	Désignation article	Quantité	Prix U.
202005001	1234	20 octobre 2008	Carottes	4	4.0
212005001	1235	21 octobre 2008	Tomates	6	3.5
212005002	1234	21 octobre 2008	Tomates	8	3.5

Figure 1 : Visualisation de commandes

Exercice 7

Ecrire un script en langage PHP qui permet de reproduire les écrans ci-dessous :

Fonctionnement :

L'utilisateur saisi les données via l'interface ci-dessous (figure 1) et presse sur le bouton « Envoyer la commande » pour validation.

Quand l'utilisateur clique sur le bouton de validation, les données saisis sont envoyées et récupérées par un script PHP « gestionCommande.php », elles sont traitées et les résultats sont affichés sur la sortie standard.

Figure 2 : Interface de saisie de données (creeCommande.html)

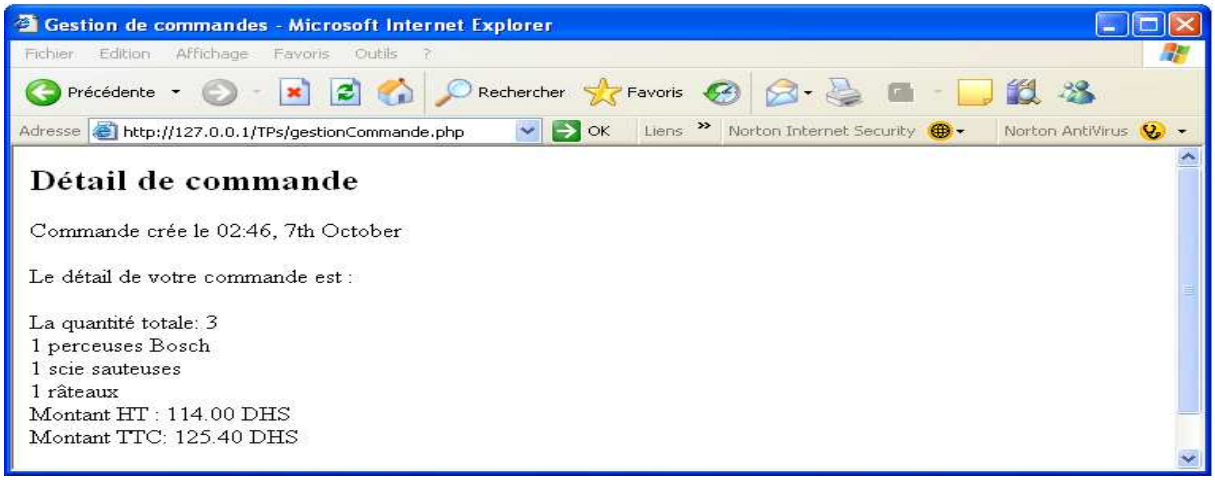


Figure 3 : Interface de validation et visualisation de données (gestionCommande.php)

Exercice 8 : **Interface de saisie d'une date**

Le formulaire de saisi (voir figure 1) est composé de :

- trois listes pour la saisie du jour, du mois et de l'année
- un bouton d'envoie des informations saisies au serveur pour validation

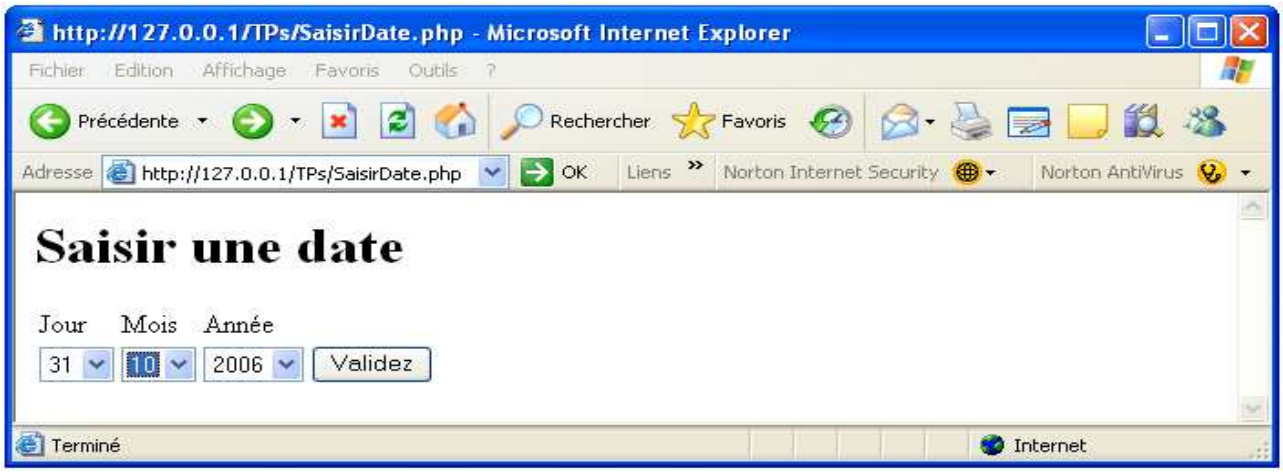


Figure 4 Interface de saisie d'une date

Exercice 9 : **Validation du formulaire**

L'utilisateur commence par choisir une date, il valide ensuite en cliquant sur le bouton « Validez ». Le script *valideDate*.php est alors appelé en affichant la date saisie (voir figure ci-dessous)

Figure 5 Validation d'une date

Lorsque l'utilisateur saisi une date incorrect, l'application doit afficher un message d'erreur signalant l'anomalie (voir figure ci-dessous)

Figure 5 Gestion des erreurs

PHP – TP N°2

Ce TP à pour bu la manipulation des formulaires en utilisant le langage PHP.

Pré-requis : langage de description html et le langage C.

Ecrire un script en langage PHP qui permet de reproduire les écrans ci-dessous :

Fonctionnement :

L'utilisateur saisi les données via l'interface ci-dessous (figure 1) et presse sur le bouton « Envoyer la commande » pour validation.

Quand l'utilisateur clique sur le bouton de validation, les données saisis sont envoyées et récupérées par un script PHP « gestionCommande.php », elles sont traitées et les résultats sont affichés sur la sortie standard.

Figure 6 : Interface de saisie de données (creeCommande.html)

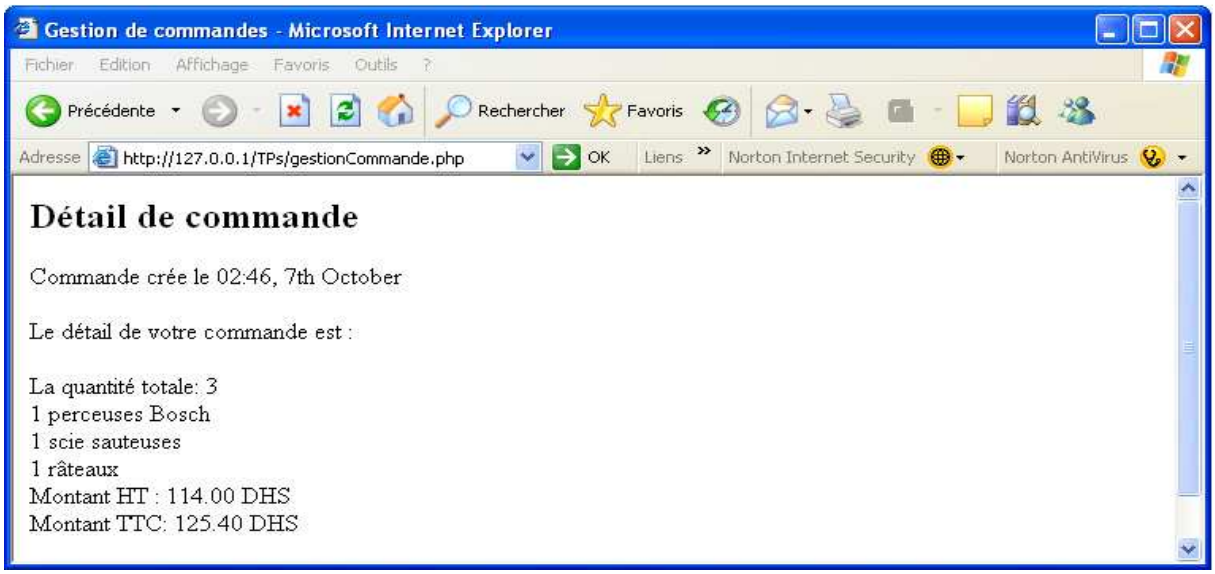


Figure 7 : Interface de validation et visualisation de données (gestionCommande.php)

PHP – TP N°3

Ce TP à pour but la réalisation de quelques applications en utilisant les tableaux PHP.

Pré-requis : langage de description html et le langage C.

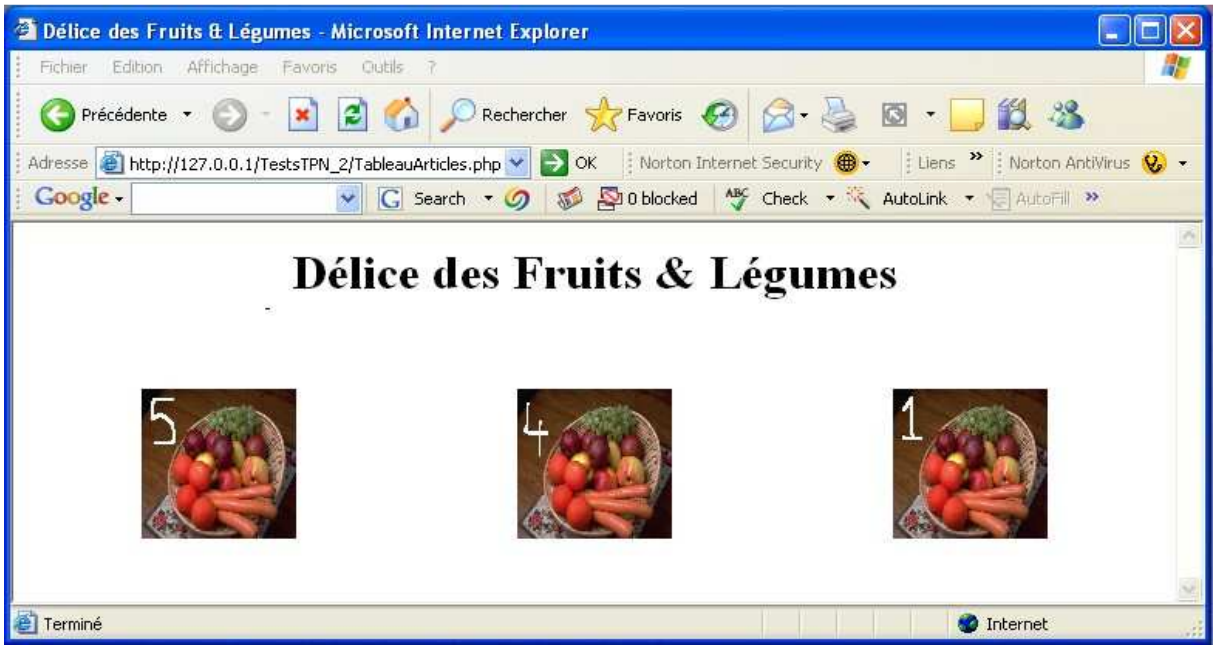
Exercice N°1 : **Utilisation de PHP pour générer une page d'accueil dynamique**

Une société spécialisée dans la vente des fruits et légumes aimerait que la page d'accueil de son site présente quelques-uns des produits proposés à la vente. Elle voudrait montrer trois des articles de son catalogue choisis au hasard et faire en sorte que ceux-ci soient différents à chaque nouvelle visite de ses clients.

Ecrire un script PHP qui permet l'affichage d'une page différente à chaque chargement.

Pour ce faire vous pouvez utiliser la fonction ***shuffle()*** qui permet de trier de manière aléatoire les éléments d'un tableau. Comme vous le devinez, cette fonction prend en paramètre un tableau.

La figure ci-dessous représente la capture d'écran correspondant à une exécution du script demandé.



Exercice N°2 : **Utilisation de PHP pour afficher le contenu d'un fichier de commandes**

Créez un fichier texte comportant quelques lignes de commandes clients. A chaque ligne correspond une commande.

A titre d'exemple vous pouvez insérer les lignes ci-dessous dans votre fichier « txt »

242005001 | 1236 | 24 octobre 2005 | Tomates | 4 | 4.0
252005001 | 1235 | 25 octobre 2005 | Tomates | 6 | 3.5
252005002 | 1234 | 25 octobre 2005 | Tomates | 8 | 3.0

Ecrire un script PHP qui permet de charger l'intégralité du fichier « txt » dans un tableau, ensuite afficher le contenu du fichier en utilisant les fonctionnalités des tableaux PHP.

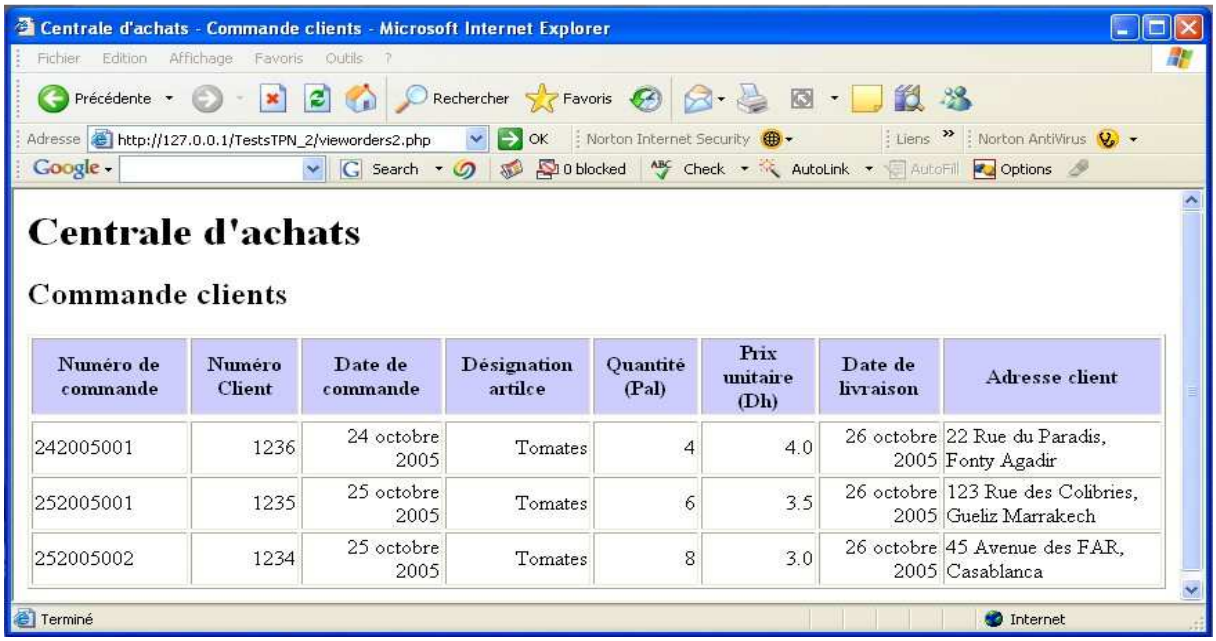
Exercice N°3 : **Utilisation de PHP pour séparer, mettre en forme et afficher les commandes clients envoyées à une centrale d'achats**

Une Centrale d'achats (Marjane par exemple), souhaite centralisée les commandes clients provenant de ces points de ventes à fin des les traitées ensemble.

En utilisant l'exercice N°2, écrire le script PHP qui permet l'affichage des commandes clients ci-dessous. Il vous est demandé de scinder chaque ligne et de mettre en forme le résultat sous la forme ci-dessous.

La fonction explode permet de scinder une chaîne de caractère en plusieurs parties

explode (« caractère délimiteur », chaîne de donnée)



Centrale d'achats

Commande clients

Numéro de commande	Numéro Client	Date de commande	Désignation artilce	Quantité (Pal)	Prix unitaire (Dh)	Date de livraison	Adresse client
242005001	1236	24 octobre 2005	Tomates	4	4.0	26 octobre 2005	22 Rue du Paradis, Fonty Agadir
252005001	1235	25 octobre 2005	Tomates	6	3.5	26 octobre 2005	123 Rue des Colibries, Gueliz Marrakech
252005002	1234	25 octobre 2005	Tomates	8	3.0	26 octobre 2005	45 Avenue des FAR, Casablanca

PHP – TP N°4

Exercice N°1 : **Manipuler les chaines de caractères**

Reprendre le TP N°2.

Le fichier de données contient les commandes envoyées par tous les clients (dans notre cas deux clients "CLI1001" et "CLI1004").

Ecrire un script PHP qui permet d'archiver les commandes du client CLI1001 dans un fichier de données pscde01_CLI1001.txt et celle du client CLI1004 dans le fichier psccl01_CLI1004.txt.

Utiliser au maximum les fonctions de manipulations des chaines de caractères vus en cours.

Commande clients

Numéro de commande	Numéro Client	Date de commande	Désignation artilce	Quantité(Pal)	Prix unitaire(Dh)	Date de livraison	Adresse client
20061109-000001	CLI1001	11-09-2006	Sucre morceaux	1	6	16-09-2006	1, Rue des Colibris Agadir
20061109-000001	CLI1001	11-09-2006	Huile végétale	1	15	16-09-2006	1, Rue des Colibris Agadir
20061109-000001	CLI1001	11-09-2006	Maïs Géant 250 g	1	4	16-09-2006	1, Rue des Colibris Agadir
20061109-000001	CLI1001	11-09-2006	Eau Sidi Ali	5	4	16-09-2006	1, Rue des Colibris Agadir
20061109-000001	CLI1001	11-09-2006	Fanta 1.5 L	1	9	16-09-2006	1, Rue des Colibris Agadir
20061309-000002	CLI1004	13-09-2006	Coca Cola 1.5 L	2	9	18-10-2006	31, Rue de l'université Marrakech
20061309-000002	CLI1004	13-09-2006	Haricots sec	1	6	18-10-2006	31, Rue de l'université Marrakech
20061309-000002	CLI1004	13-09-2006	Lentilles Extra	1	6	18-10-2006	31, Rue de l'université Marrakech
20061309-000002	CLI1004	13-09-2006	Huile d'olives	1	45	18-10-2006	31, Rue de l'université Marrakech
20061309-000002	CLI1004	13-09-2006	Confiture Aïcha 250 g	1	5	18-10-2006	31, Rue de l'université Marrakech
20061010-000003	CLI1001	10-10-2006	Tomates	2	3	13-10-2006	1, Rue des Colibris Agadir
20061010-000003	CLI1001	10-10-2006	Oignons	2	4	13-10-2006	1, Rue des Colibris Agadir
20061010-000003	CLI1001	10-10-2006	Courgettes	2	3	13-10-2006	1, Rue des Colibris Agadir
20061010-000003	cli1001	10-10-2006	Pomme Golden	1	12	13-10-2006	1, Rue des Colibris Agadir
20061010-000003	CLI1001	10-10-2006	Bananes	1	12	13-10-2006	1, Rue des Colibris Agadir
20061010-000004	CLI1004	10-10-2006	Tomates	2	3	13-10-2006	31, Rue de l'université Marrakech
20061010-000004	CLI1004	10-10-2006	Oignons	2	4	13-10-2006	31, Rue de l'université Marrakech
20061010-000004	CLI1004	10-10-2006	Courgettes	2	3	13-10-2006	31, Rue de l'université Marrakech
20061010-000004	CLI1004	10-10-2006	Pomme Golden	1	12	13-10-2006	31, Rue de l'université Marrakech
20061010-000004	CLI1004	10-10-2006	Bananes	1	12	13-10-2006	31, Rue de l'université Marrakech

Exercice N°2 : **Réutilisation de code**

Reprendre l'exercice N°1 en utilisant des fonctions.

Mini projet

Objectif : Réaliser une application qui permet la gestion d'une librairie virtuelle.

Le travail à réaliser consiste à développer une application qui permet la gestion d'une librairie virtuelle. L'application doit être la plus complète possible (création, ajout, suppression). Elle doit être élaborée en utilisant le langage PHP et le SGBD MySQL.

Le schéma de la base de données est donné en annexe.

1 : Créer un script d'enregistrement d'un lecteur

Un lecteur est défini par les caractéristiques suivantes :

* nom
* prénom
* adresse (numéro, rue)
* ville
* code postal

La figure 1, représente le formulaire de saisie permettant la saisie des informations relatif à un lecteur. La validation du formulaire (figure 2) consiste à vérifier les données saisies et l'enregistrement du lecteur dans la table lecteurs (voir le script de création de la base de données en annexe).

Figure 1 Enregistrement d'un lecteur

Figure 8 Validation d'un lecteur

2 : Créer une page d'enregistrement d'un livre

Un livre est défini par les caractéristiques suivantes :

* nom de l'auteur
* prénom de l'auteur
* titre du livre
* catégorie (Roman, Science-fiction, Policier...)
* numéro ISBN

Réalisez le formulaire de saisie permettant la saisie de ces informations (voir figure 3). Après validation du formulaire, l'action réalisée consiste à appeler le script *valideLivre.php* qui enregistre l'ensemble des valeurs saisies par le formulaire dans la table livres (voir le script de création de la base de données en annexe) et les affiches à l'écran (figure 4).

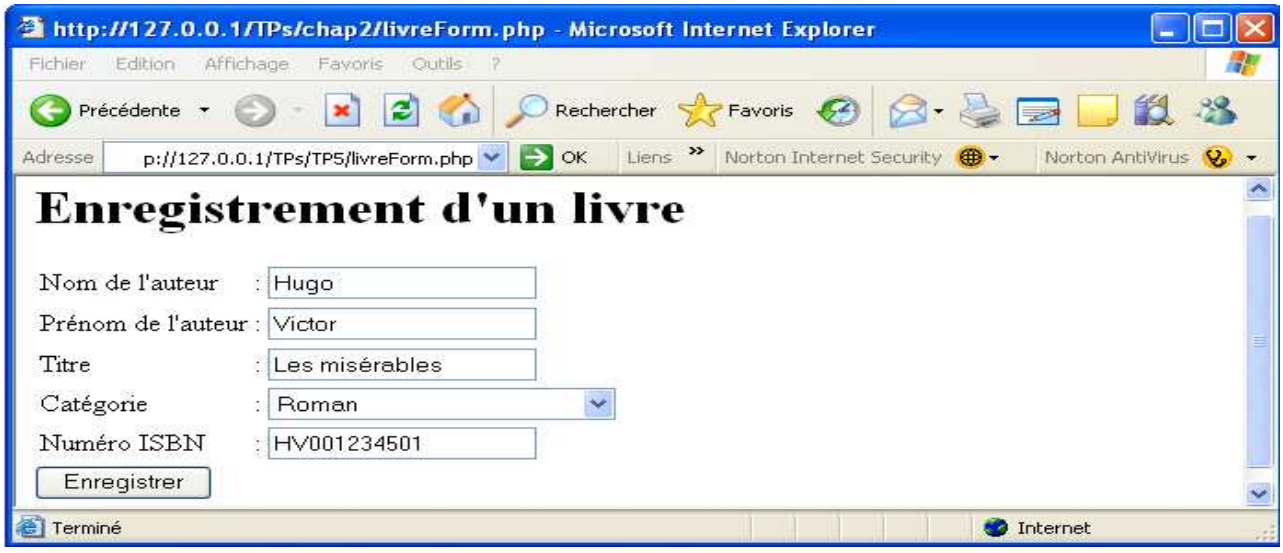


Figure 9 Enregistrement d'un livre

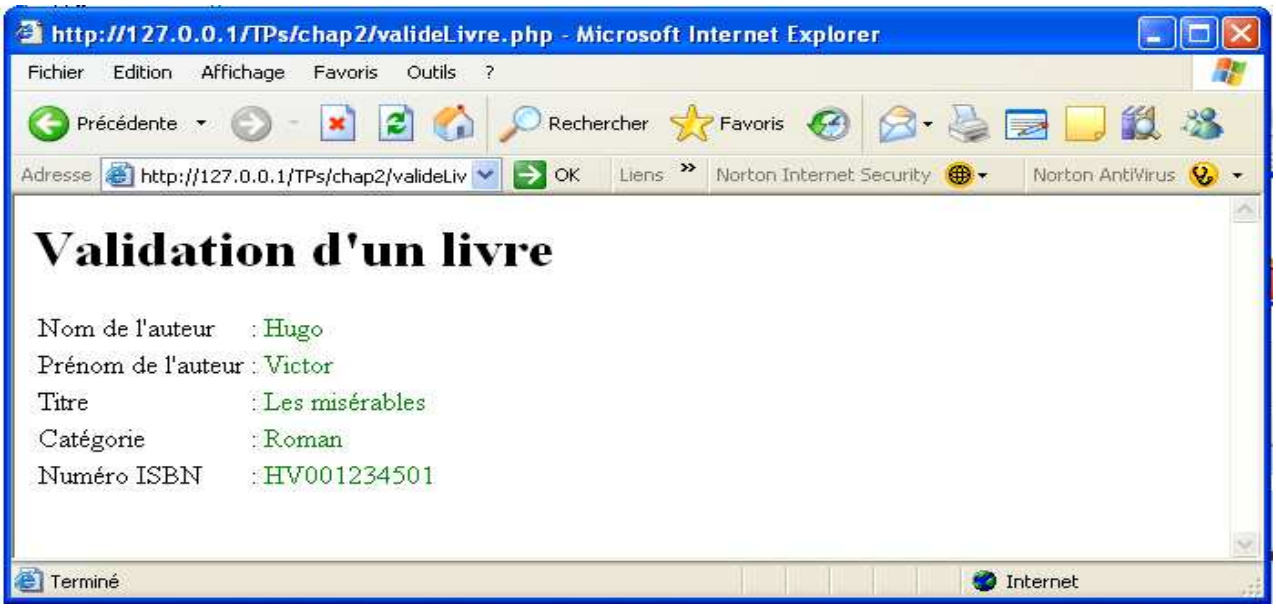


Figure 10 Validation d'un livre

3 : Réservation de livres en ligne

L'objectif et de développer la réservation des livres en ligne, sur le site de la librairie virtuelle.

Cahier de charges fonctionnel

Lorsque l'utilisateur se connecte sur le site de la librairie, un formulaire d'authentification s'affiche, lui demandant son nom et son mot de passe. Lorsque le lecteur est authentifié, la liste des livres à la réservation s'affiche par catégorie. Lorsque le lecteur valide le formulaire de réservation, une page affiche les dates de début et de fin de réservation.

Script de création de la base de données

```
USE mysql;
DROP DATABASE IF EXISTS `librairie`;
CREATE DATABASE `librairie`;
USE librairie;

-- Structure de la table emprunts
DROP TABLE IF EXISTS `emprunts`;
CREATE TABLE `emprunts` (
        empnum          char(255),
        empdate         date ,
        empdateret      date,
        empcodelivre    char(255) ,
        empnumlect      char(255),
        PRIMARY KEY  (`empnum`)
) ;
```

```
-- Structure de la table `lecteurs`
DROP TABLE IF EXISTS `lecteurs`;
CREATE TABLE `lecteurs` (
        lecnum          char(16),
        lecnom          char(16),
        lecprenom               char(16),
        lecadresse      char(80),
        lecville        char(16),
        leccodepostal   char(10),
        lecmotdepasse char(80),
        PRIMARY KEY  (`lecnum`)
);
-- Données enregistrées pour la table `lecteurs`
insert into `lecteurs` values ('216', 'Lamy', 'Elena', '7 rue du Paradis', 'Paris', '75012', 'Elena');
insert into `lecteurs` values ('221', 'Theos', 'Pablo', '3 passage Secret', 'Paris', '75004', 'Pablo');
insert into `lecteurs` values ('342', 'Camden',      'Nicolas', '24 av du Papillon', 'Paris', '75013', 'Nicolas');
insert into `lecteurs` values ('528', 'Line', 'Margo', '22 rue de la Liberté', 'Paris', '75005', 'Margo');

-- Structure de la table `livres`
DROP TABLE IF EXISTS `livres`;
CREATE TABLE `livres` (
        livcode         char(255) ,
        livnomaut               char(255) ,
        livprenomaut  char(255) ,
        livtitre        char(255) ,
        livcategorie    char(255) ,
        livISBN                 char(255) ,
        livdejareserve  tinyint(1) NOT NULL default '0',
        PRIMARY KEY  (`livcode`)
);

-- Données enregistrées pour la table `livres`
insert into `livres` values ('KaElRo58', 'Kazan', 'Elia', 'L'arrangement', 'Roman', '2234023858', 1);
insert into `livres` values ('AsIsSc08', 'Asimov', 'Isaac', 'Fondation', 'Science-fiction', '2070415708', 1);
insert into `livres` values ('DiPhSc43', 'Dick', 'Philip K.', 'Blade Runner', 'Science-fiction', '2290314943', 1);
insert into `livres` values ('WaAlRo37', 'Walker', 'Alice', 'La couleur pourpre', 'Roman', '2290021237', 1);
insert into `livres` values ('KuMiRo38', 'Kundera', 'Milan', 'La plaisanterie', 'Roman', '2070703738', 0);
insert into `livres` values ('BaJaJu63', 'Barrie', 'James M.', 'Peter Pan', 'Junior', '2290333263', 0);
insert into `livres` values ('VeJuRo22', 'Verne', 'Jules', 'L île mysterieuse', 'Roman', '0812966422', 0);
```

5 Installation et Configuration d'Apache, de PHP et de MySQL sous Windows XP

5.1 Introduction

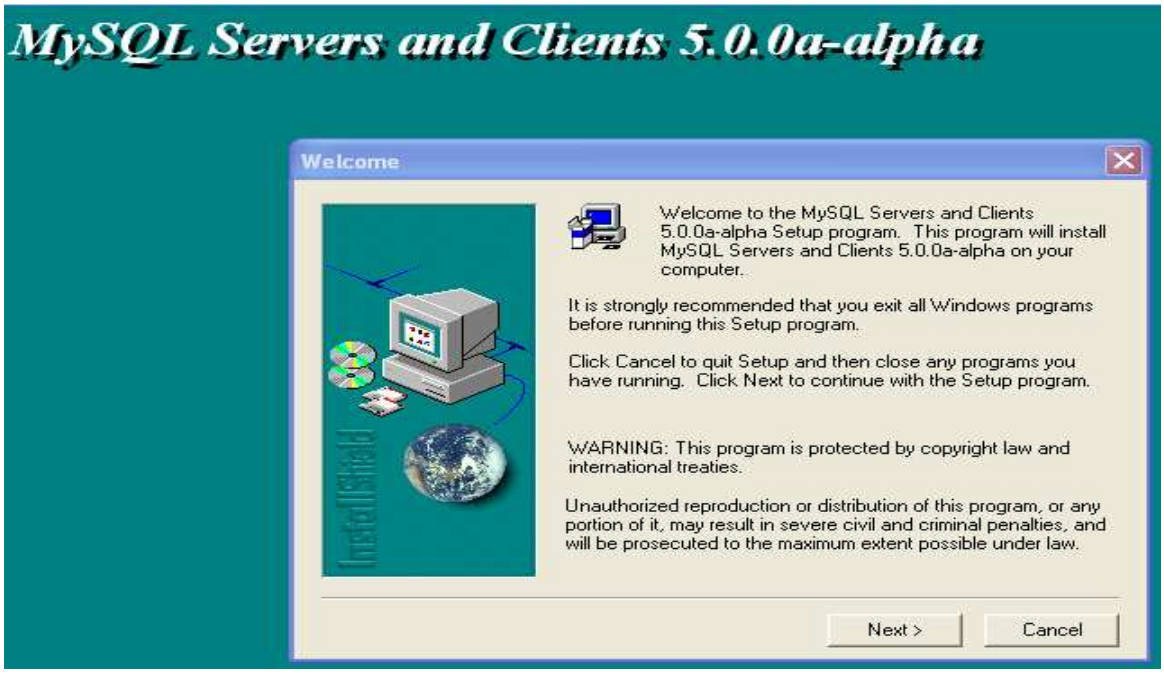
Apache, PHP et MySQL sont disponibles pour un grand nombre de combinaisons de systèmes d'exploitation et de serveurs Web. Ce manuel explique comment installer et configurer Apache, PHP et MySQL sur le système Windows.

5.2 Installation de MySQL sous Windows

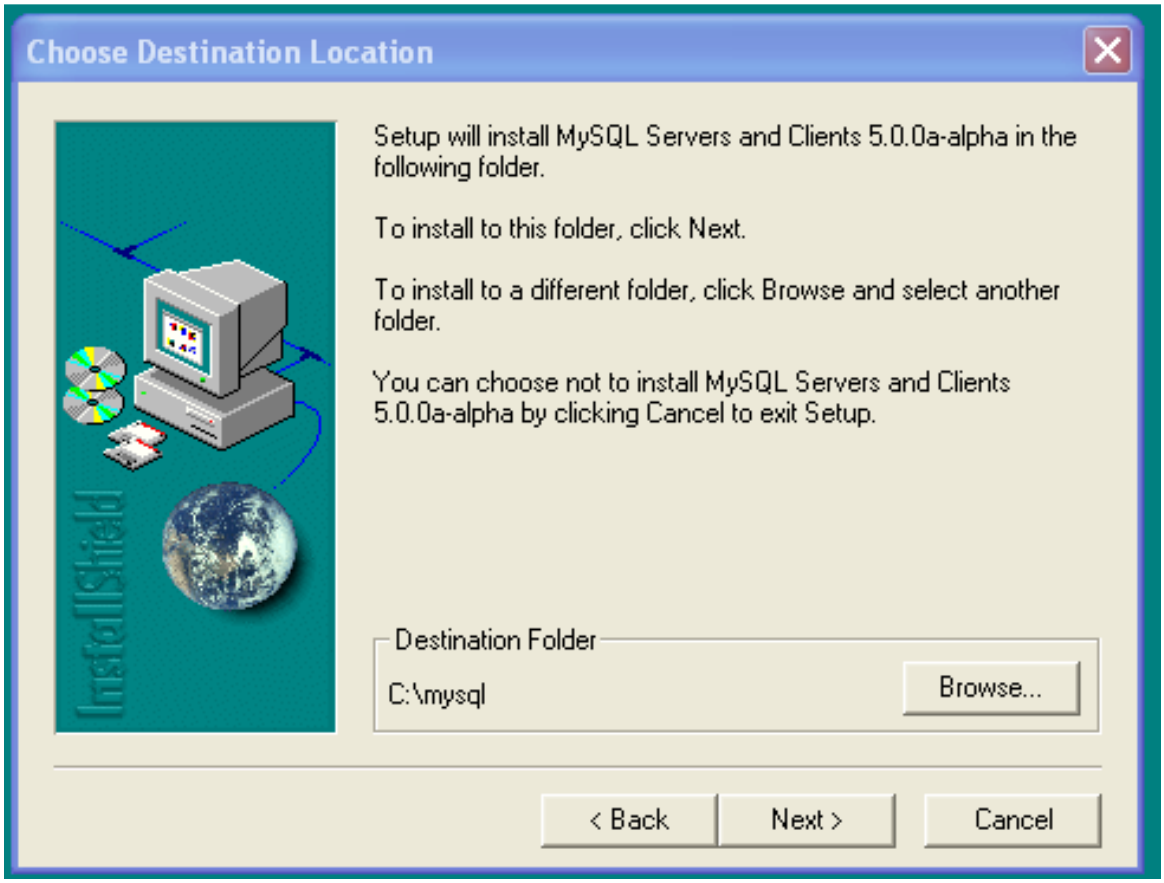
Les instructions suivantes ont été écrites pour une installation sous Windows. Commencer par créer un répertoire temporaire (***C:\Temp\Download\MySQL*** par exemple). Vous pouvez télécharger le fichier ZIP requis de http://www.MySQL.com. Décompresser le fichier ZIP de MySQL vers le répertoire temporaire et exécuter le programme setup.exe. Le programme d'installation est un InstallShield standard; il devrait présenter le même aspect que les nombreux autres installateurs que vous avez déjà rencontrés.

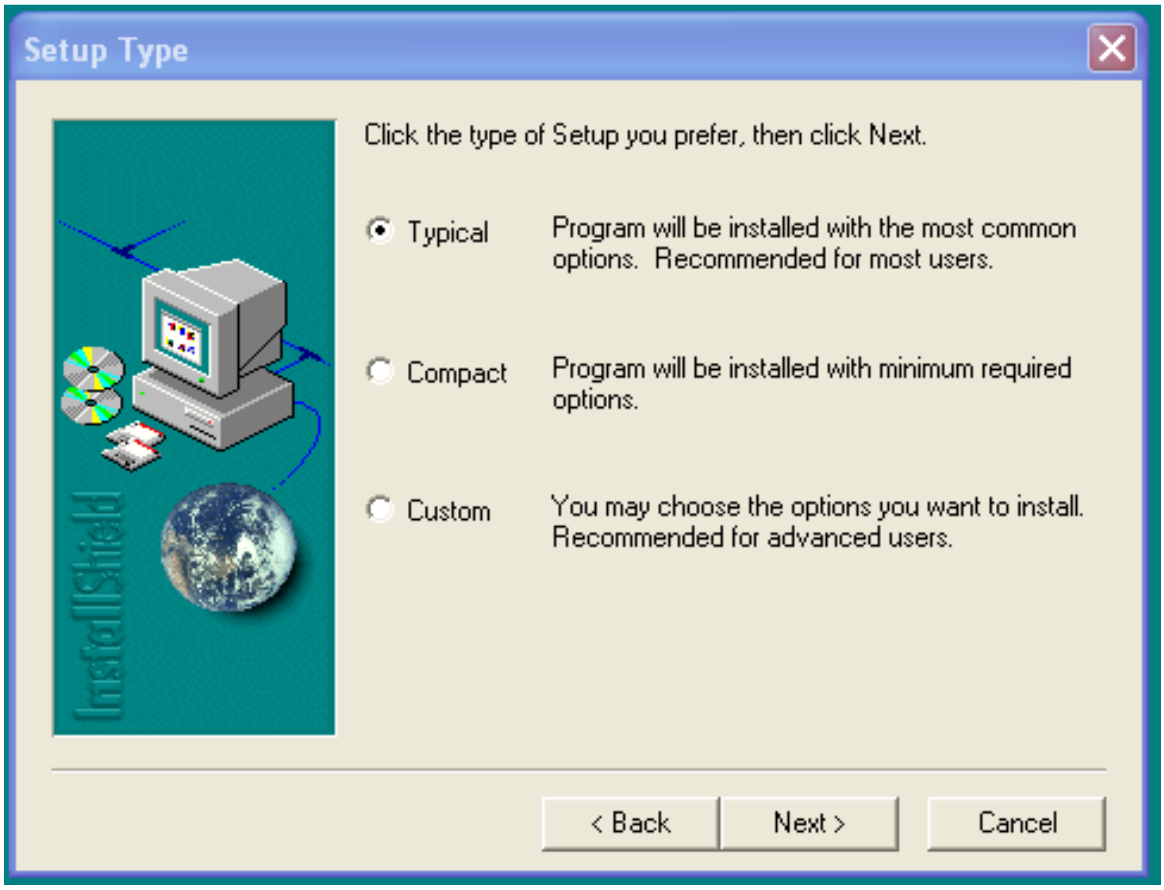
Si vous choisissez l'installation type, l'assistant ne devrait pas vous poser de question autres que celle relative à l'emplacement où installer MySQL. Par défaut, MySQL s'installe dans le répertoire ***C:\MySQL***.

Ci-dessous l'enchaînement relative à l'installation de MySQL




Information
This is a release of MySQL 5.0.0a-alpha for Win32.
NOTE: If you install MySQL in a folder other than C:\MYSQL or you intend to start MySQL on NT/Win2000 as a service, you must create a file named C:\MY.CNF or \Windows\my.ini or \winnt\my.ini with the following information::
[mysqld]
basedir=E:/installation-path/
datadir=E:/data-path/
After your have installed MySQL, the installation directory will contain 4 files named 'my-small.cnf, my-medium.cnf, my-large.cnf, my-huge.cnf'.
You can use this as a starting point for your own
InstallShield
< Back
Next >
Cancel


Choose Destination Location
Setup will install MySQL Servers and Clients 5.0.0a-alpha in the following folder.
To install to this folder, click Next.
To install to a different folder, click Browse and select another folder.
You can choose not to install MySQL Servers and Clients 5.0.0a-alpha by clicking Cancel to exit Setup.
Destination Folder
C:\mysql
Browse...
InstallShield
< Back
Next >
Cancel



Dans l'installation sous Windows NT/2000/XP, le nom du serveur MySQL est MySQL-nt et celui-ci sera normalement installé en tant que service. Un service est un programme qui s'exécute de manière permanente à l'arrière-plan et qui est destiné à fournir des services à d'autres programmes.

Vous pouvez installer le serveur MySQL en tant que service en ouvrant une fenêtre d'invite de commande et en entrant les lignes suivante :

cd C:\MySQL\bin

MySQLd-nt -install

Après avoir exécuter cette commande, vous devriez obtenir la réponse suivante :

```
C:\mysql\bin> mysqld-nt -install
Service successfully installed.

C:\mysql\bin>
```

Utilisez les commandes en ligne

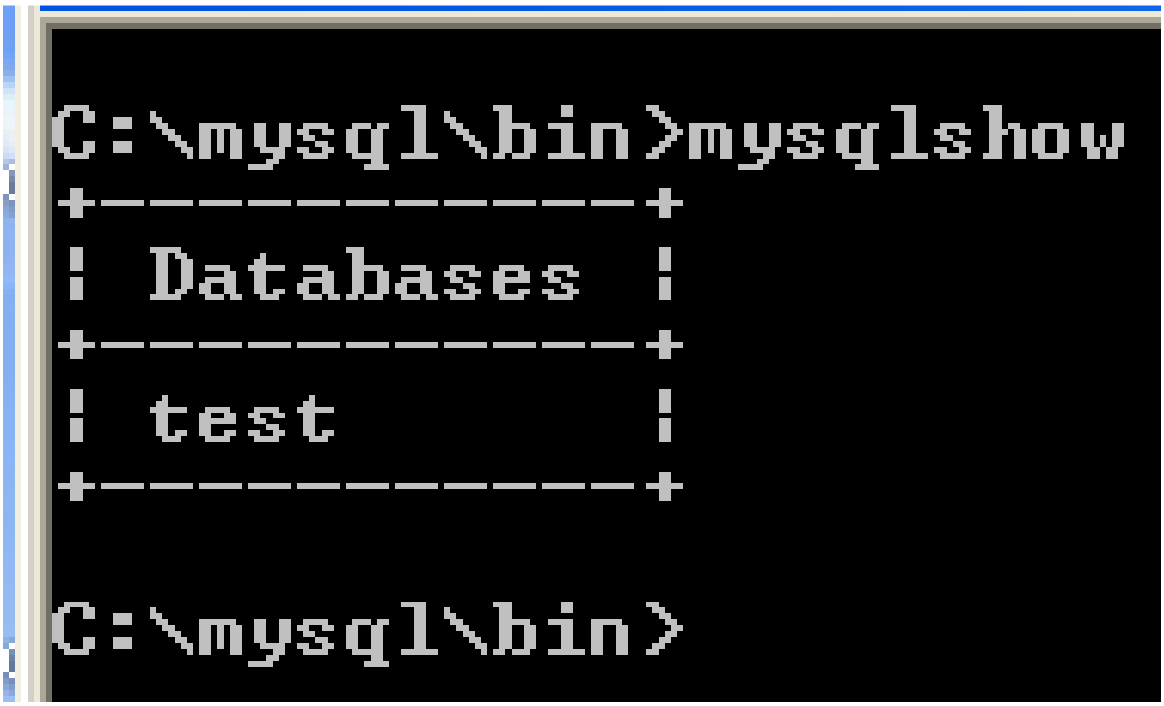
NET START MySQL pour démarrer le service MySQL

NET STOP MySQL pour arrêter le service MySQL

Vous devriez obtenir la réponse suivante :

Pour afficher le contenu de la base de donnée (par défaut), tapez la commande suivante :

MySQLshow



- **Suppression de l'utilisateur anonyme**

  La configuration par défaut de MySQL permet à n'importe quel utilisateur d'accéder au système sans avoir à fournir un nom d'utilisateur ou un mot de passe.

  Exécutez les ordres SQL suivantes pour afficher les utilisateurs de la table user, la figure ci-dessous montre que la table user contient le super utilisateur root et un utilisateur anonyme.

```
C:\mysql\bin>mysql -u root
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 2 to server version: 5.0.0-alpha-nt

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use mysql
Database changed
mysql> select User from user;
+------+
| User |
+------+
|      |
| root |
|      |
| root |
+------+
4 rows in set (0.05 sec)

mysql>
```

Pour supprimer l'utilisateur anonyme, ouvrez une invite de commande et tapez les lignes suivantes :

```
C:\mysql\bin>mysql -u root
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3 to server version: 5.0.0-alpha-nt

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use mysql
Database changed
mysql> delete from user where User='';
Query OK, 2 rows affected (0.00 sec)

mysql>
```

Vérifions si l'utilisateur anonyme a été supprimé en utilisant la requête ci-dessous

```
C:\mysql\bin>mysql -u root
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 6 to server version: 5.0.0-alpha-nt

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> use mysql
Database changed
mysql> select User from user;
+------+
| User |
+------+
| root |
| root |
+------+
2 rows in set (0.00 sec)

mysql>
```

- **Définition du mot de passe root**

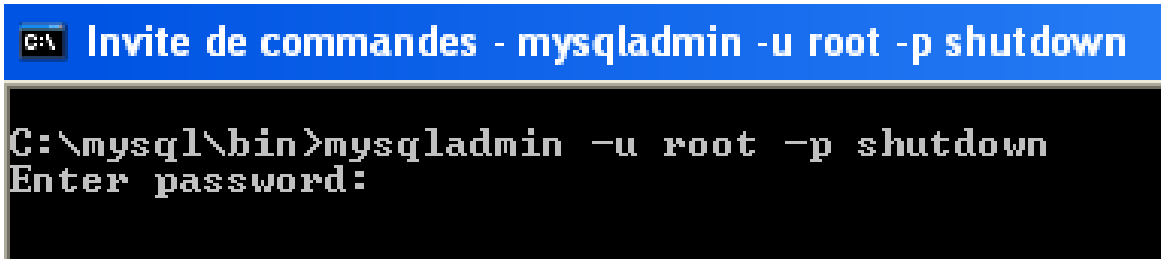
A ce stade, le compte du super-utilisateur est encore dépourvu de mot de passe. Le mot de passe pour l'administrateur (root) de la base MySQL peut être défini par la commande suivant :

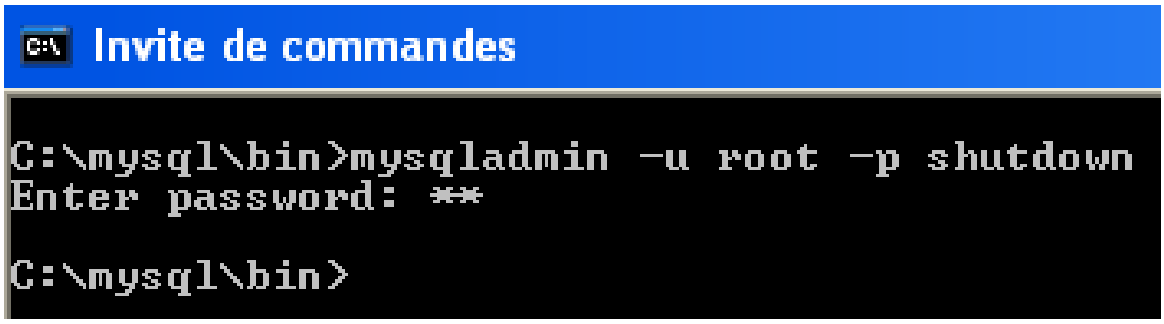
```
C:\mysql\bin>mysqladmin -u root password lk

C:\mysql\bin>
```

Désormais, vous serez invité à spécifier le nom de l'utilisateur ainsi que le mot de passe. Vous pouvez tester la commande suivante :

```
C:\mysql\bin>mysqladmin shutdown
mysqladmin: shutdown failed; error: 'Access denied. You need the SHUTDOWN privil
ege for this operation'

C:\mysql\bin>
```

Pour arrêter votre serveur MySQL, vous devez saisir votre mot de passe.





5.3 Installation d'Apache

Rendez-vous dans le répertoire ***C:\TEMP\DOWNLOAD\Apache***. Vous y trouverez le fichier ***apache_1.3.31-win32x86-no_src***. Cette version contient la version courante (dans la hiérarchie 1.3) pour Windows. Ce fichier contient le serveur Apache prêt pour l'installation.

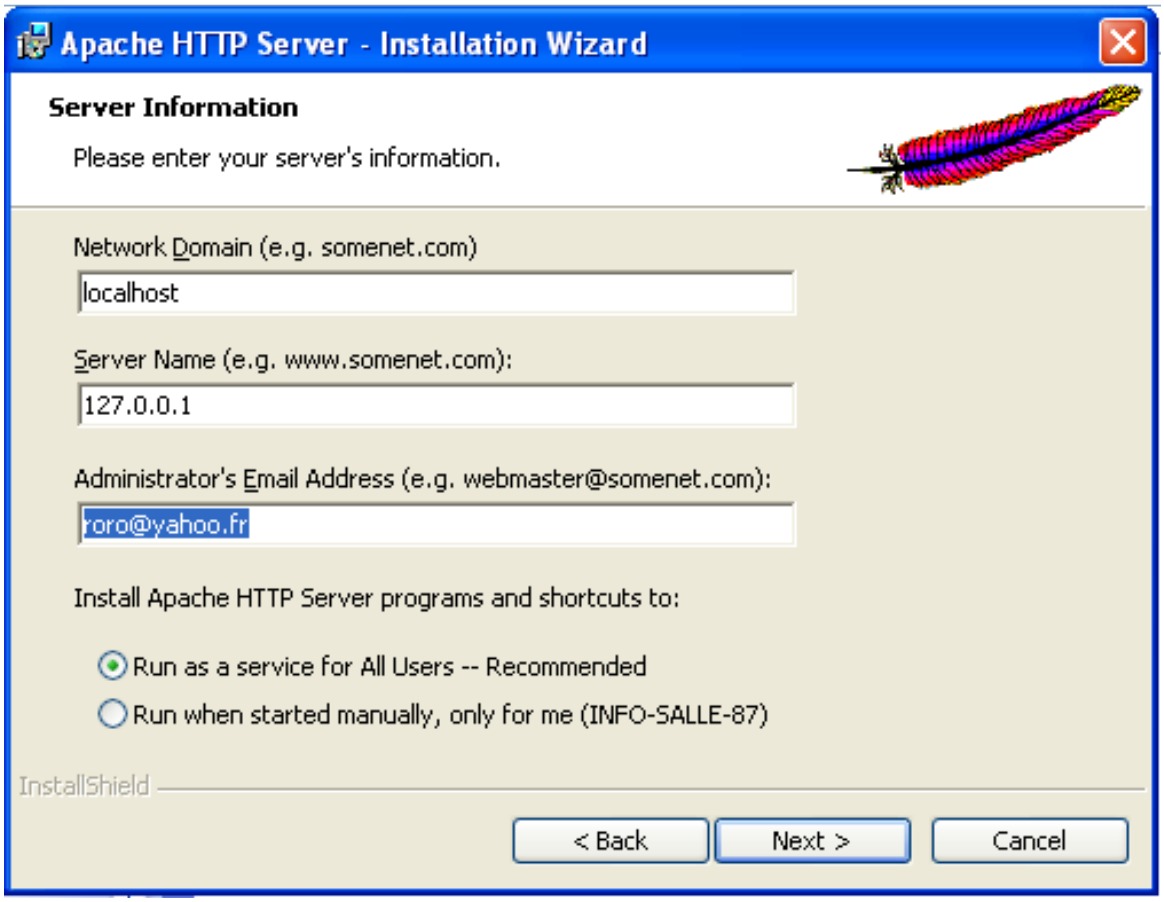
Double cliquer sur le fichier pour démarrer la procédure d'installation. Celle-ci devrait vous être familière puisqu'elle est semblable à de nombreuses autres utilisant l'installateur Windows (Figure ci-dessous)

Le programme d'installation vous interrogera sur les points suivants :

- **Network Domain** → nom du domaine de votre ordinateur, mettez

  localhost

- **Server Name** → nom donné au serveur, mettez 127.0.0.1
- **Administrator's Email Adress** → adresse email de l'administrateur



Si vous voulez exécuter Apache en tant que service, comme pour MySQL, il est généralement préférable de le configurer de la manière suivante :

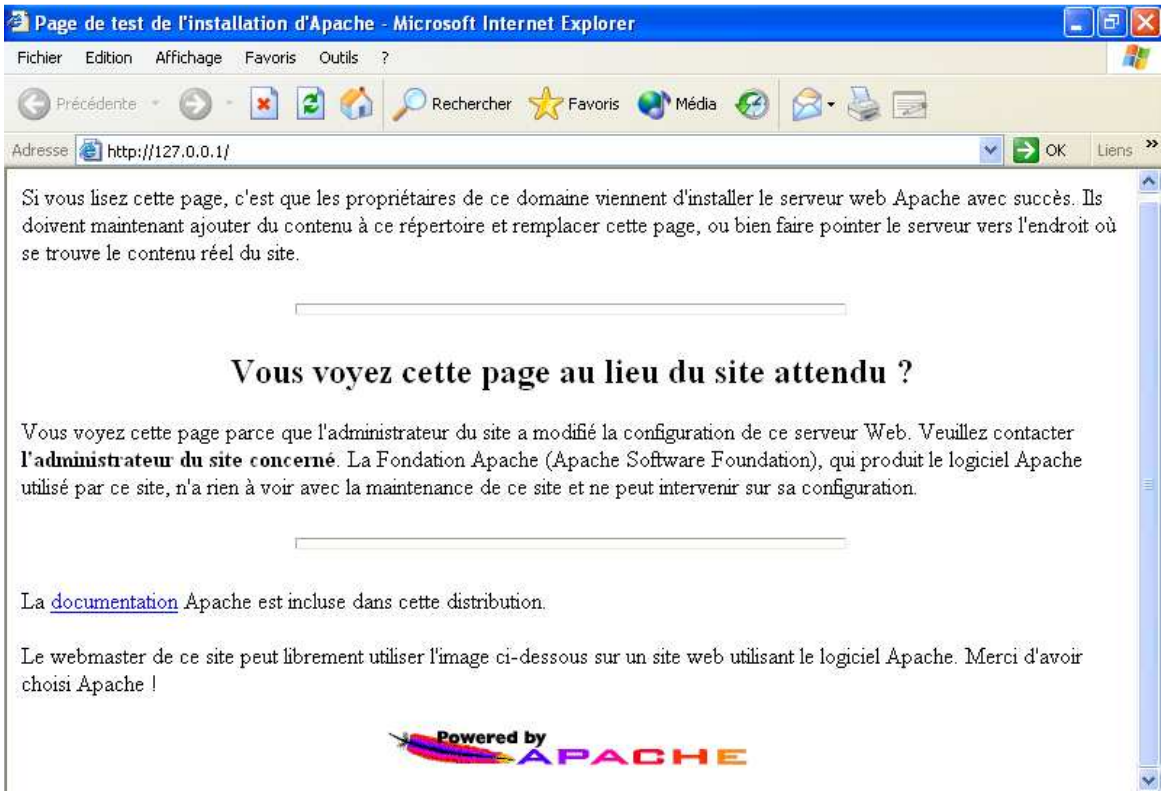
Le type d'installation. Il est recommandé de choisir *l'installation comp*lète

Le répertoire dans lequel installer Apache. (Le répertoire par défaut est **C:\Program Files \Apache Group \Apache**.)

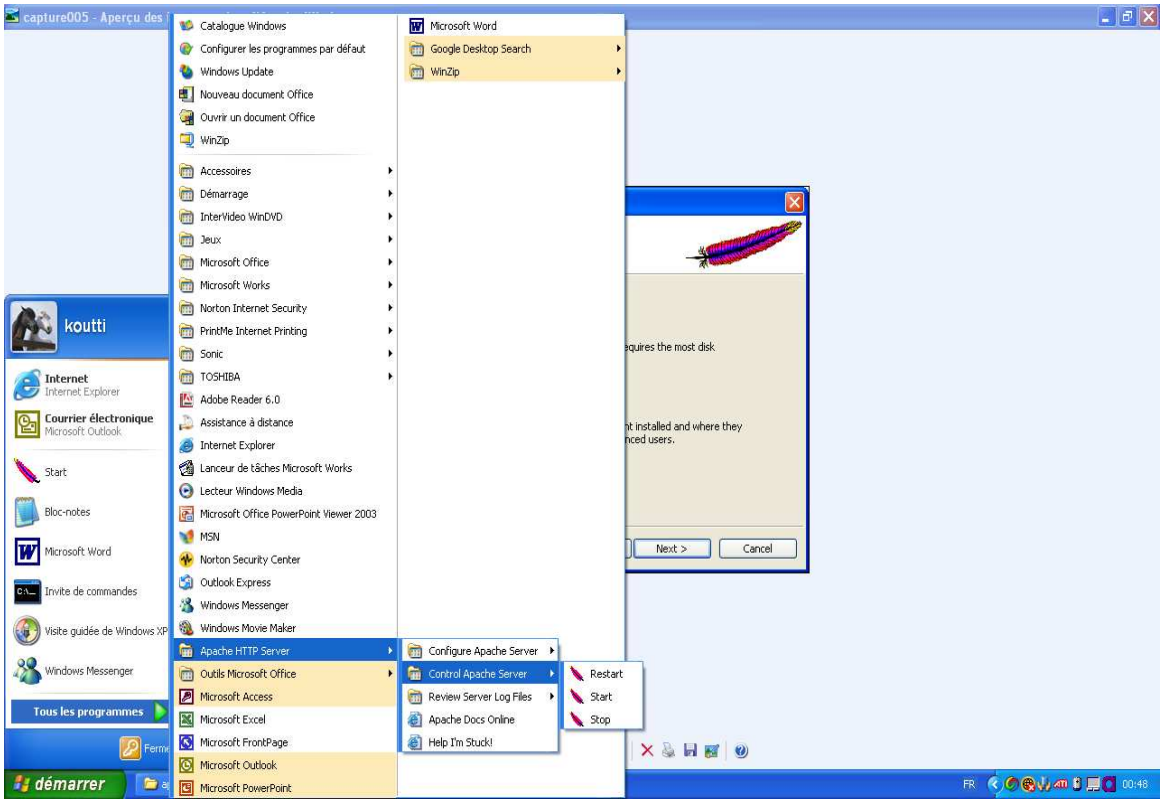
Une fois choisies toutes ces options, le serveur Apache sera installé et démarré. Apache écoutera sur le port **80** (excepté si vous avez modifié les directives Port, **Listen** ou **BindAddress** dans les fichiers de configuration) après son démarrage. Pour vous connecter au serveur et accéder à la page par défaut, lancez un navigateur Web et entrez l'URL suivante :

http://localhost/

Vous devrez alors obtenir l'affichage d'une page de bienvenue ci-dessous :



Vous pouvez démarrer et arrêter le service Apache à partir du menu Démarrer : Apache s'y ajoute lui-même sous l'intitulé "Apache http Server". Dans le panneau intitulé "Contrôle du serveur Apache", vous trouverez les options permettant de démarrer, d'arrêter ou de redémarrer le serveur.



Après avoir installé Apache, il peut être nécessaire d'éditer les fichiers de configuration conservés dans le répertoire conf.

5.4 Installation de PHP

Rendez vous dans le répertoire ***C:\Temp\Download\PHP***

Commencez par décompresser le fichier ZIP vers le répertoire de votre choix. L'emplacement habituel est ***C:\PHP***. C'est celui que nous utiliserons dans l'exemple suivant.

Vous pouvez installer les bibliothèques **PECL** en décompressant le fichier **PECL** vers votre répertoire d'extensions. Si C:\PHP est le répertoire de base, il s'agira de ***C:\PHP\ext***.

A présent, suivez ces étapes

- Copiez toutes les DLL vers le répertoire système de Windows.
- Installez le fichier de configuration php.ini. Faites une copie du fichier php.ini.recommended que vous renommerez php.ini. Placez votre fichier php.ini dans le répertoire Windows.
- Editez votre fichier php.ini. Celui-ci contient plusieurs paramètres. Les paramètres à modifier pour le moment sont les suivants :
- Modifiez la directive extension_dir de manière qu'elle pointe vers l'emplacement où résident les DLL de vos extensions. Dans le cadre d'une installation standard, il s'agit de ***C:\PHP\ext***. Votre php.ini contiendra en conséquence la ligne suivante :

  Extension_dir = C:/php/ext

- Définissez la directive ***doc_root*** de façon qu'elle pointe vers le répertoire racine à partir duquel opère votre serveur Web. Le plus souvent avec Apache, la ligne se présente comme suit :

  doc_root="C:\Program Files/Apache Groupe/ Apache /htdocs"

  Assurez-vous que les extensions suivantes sont activées :

  php_pdf.dll, php_gd2.dll, php_imap.dll et php_MySQLi.dll. Si ce n'est pas le cas vous devez décommenter ces lignes.

- Fermez et sauvegarde le fichier php.ini

5.5 Ajout de PHP à votre configuration Apache

Vous aurez certainement besoin d'éditer les fichiers de configuration d'Apache. Ouvrez le fichier ***httpd.conf*** dans votre éditeur de prédilection. Ce fichier est généralement situé dans le répertoire ***"C:/Program Files/Apache Groupe/Apache /conf"***. Recherchez les lignes suivantes :

LoadModule php5_module C:/php/php5apache.dll

AddModule mod_php5.c

AddType application/x-httpd-php .php

Action application/x-httpd-php "/php/php.exe"

Si ces lignes n'existent pas, ajoutez-les à la fin du fichier, enregistrez ce dernier et redémarrer votre serveur Apache.

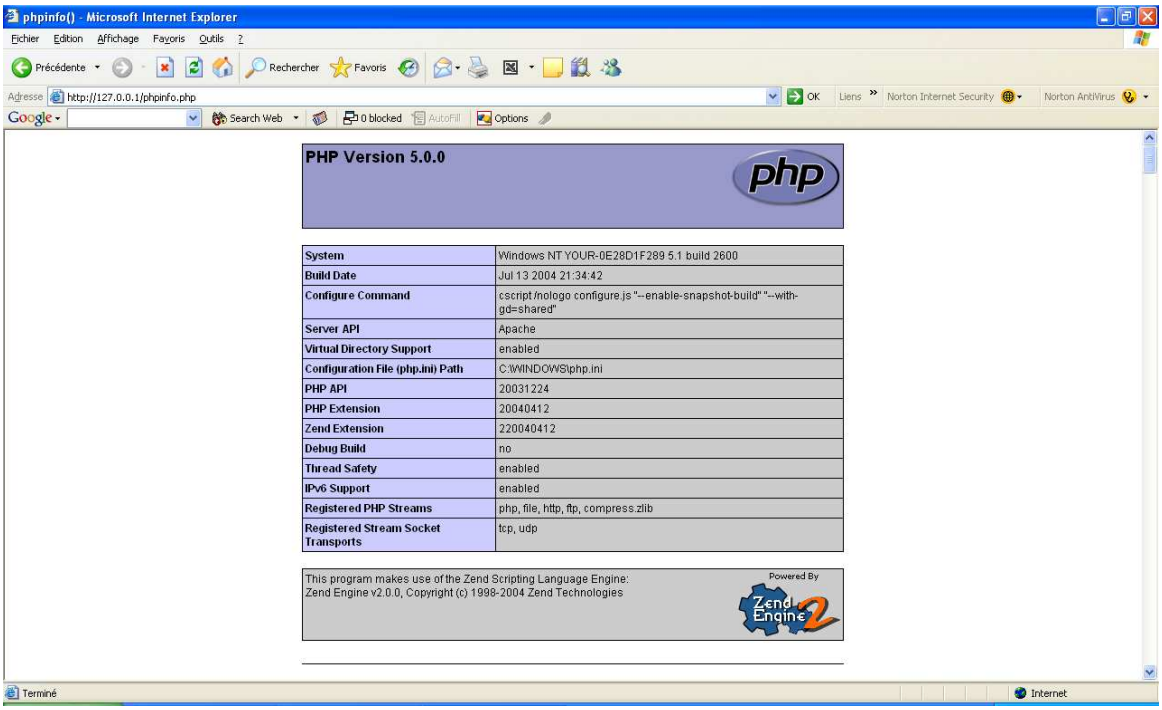
5.6 Test de l'installation de PHP

Démarrer le serveur Web et testez le bon fonctionnement de PHP. Créez un fichier ***test.php*** et insérez-y la ligne suivante :

< ? phpinfo() ?>

Placez ce fichier dans le répertoire racine des documents du serveur (généralement C:\ Program Files\ Apache Group\Apache\htdocs, puis chargez-le dans votre navigateur, comme suit :

http://localhost/test.php



5.7 Ressources bibliographiques et webo graphiques

Ci-dessous, une liste de quelques-unes de nombreuses ressources disponibles sur le Web, dont vous pouvez tirer profit pour trouver des cours, des articles, des informations récentes et des exemples de code PHP.

PHP 5 & MySQL 5 Luke Welling & Laura Thomson CampusPress
http://www.phpfrance.com
http://www.siteduzero.com/php/qcm.php
http://www.ac-creteil.fr/util/programmation/scripts/php-langage.php
http://www.mysql.com
http://dev.nexen.net/docs/mysql/chargement.hml
http://www.apache.org
http://www.apachefrance.com