Programação para a Web utilizando

Autores:
Alexandre Arroyo
Fabio Santos

Divisão de Serviços à Comunidade
Centro de Computação
Unicamp

Licenciamento de Uso

Índice

Última atualização em 21/06/2002

O que é PHP?

A abreviação PHP vem de “Hypertext PreProcessor”, que é uma linguagem de programação de código aberto muito utilizada para a criação de scripts, que são executados no servidor web para a manipulação de páginas HTML. Apesar de ser mais utilizado em aplicativos para a web, o PHP também suporta programação na linha de comando e aplicações gráficas cliente para serem executadas em interfaces gráficas com o PHP-GTK.

História

O PHP foi criado por volta de 1994 por Rasmus Lerdorf, que inicialmente o utilizava em sua home page pessoal (Personal Home Page). Em meados de 1995 ele passou a ser utilizado por outras pessoas e foi reescrito com novos recursos, sendo renomeado para Personal Home Page Tools/FI (Form Interpreter), e entre os novos recursos, passou a contar com suporte ao mSQL. Dois anos mais tarde o PHP deixou de ser um projeto pessoal de Rasmus Lerdorf e passou a ser desenvolvido por uma equipe de colaboradores, e neste período, foi lançada a versão 3 da linguagem. A partir da versão 4 o PHP passou a utilizar a engine de scripting da Zend, para melhorar a performance e suportar uma variedade maior de bibliotecas externas e extensões. Até março de 2002, o PHP estava sendo utilizado em 9.000.000 de domínios.

Vantagens

O PHP tem inúmeras vantagens, como veremos a seguir:

- É uma linguagem de fácil aprendizado;
- Tem performance e estabilidade excelentes;
- Seu código é aberto, não é preciso pagar por sua utilização, e é possível alterá-lo na medida da necessidade de cada usuário;
- Tem suporte nos principais servidores web do mercado, e suporte nativo no servidor web Apache (o mais utilizado no mundo);
- Suporta conexão com os bancos de dados mais utilizados do mercado, como por exemplo, MySQL, PostgreSQL, Oracle e DB2;
- É multiplataforma, tem suporte nos sistemas operacionais mais utilizados no mercado;
- Suporta uma variedade grande de padrões e protocolos, como o XML, DOM,
- IMAP, POP3, LDAP, HTTP, entre outros.

Comunicação cliente X servidor web

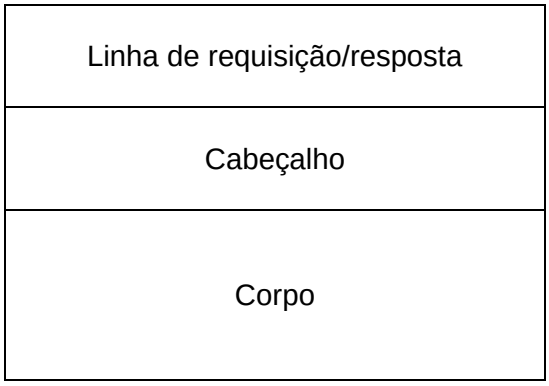
Quando é digitado um endereço no navegador para acessar uma página na internet, o que acontece é uma requisição (request) do cliente (navegador) ao servidor web. O servidor processa essa requisição e retorna uma resposta (response) ao cliente, que por sua vez interpreta o código retornado e formata a página para a sua visualização. Esse procedimento acontece em todas as requisições feitas pelo navegador.

TCP/IP e HTTP

O procedimento anterior só é possível através dos protocolos TCP/IP e HTTP. O TCP/IP é o protocolo básico para a comunicação entre as máquinas conectadas à internet, que gerencia toda a parte de transmissão e distribuição dos dados na rede. O HTTP (Hypertext Tranfer Protocol) é o protocolo que gerencia e formaliza as requisições e as respostas trafegadas entre o cliente e o servidor web. Caso o servidor web encontre a página, ela será enviada em partes ao navegador, caso contrário, o servidor enviará uma mensagem de erro.

Formato das requisições e respostas HTTP

O formato das requisições e das respostas HTTP são idênticas, como mostramos a seguir:



A diferença entre as duas é o conteúdo de cada parte descrita, as quais vamos falar separadamente:

Formato da requisição HTTP

- *Linha de requisição* : É sempre a primeira linha da requisição, a qual precisa conter um comando HTTP válido, o caminho da página requerida e a versão do protocolo HTTP:

  Exemplo:

```
GET /artigos/artigos1.html HTTP/1.1
```

Os comandos HTTP mais usados são:

GET – faz requisições específicas e sua funcionalidade é limitada, porém é o método mais usado.

POST – este método é mais abrangente que o GET, e é usado para passar informações para o servidor. Normalmente usado em formulários, que enviam dados ao servidor para serem manipulados.

- *Cabeçalho:* trecho composto por várias linhas, que carregam informações sobre o cliente, como por exemplo, o tipo e a versão do navegador, a data e as informações gerais dos clientes. Seu conteúdo pode ser variado, contendo outros tipos de linhas, e para saber quando o cabeçalho termina e o corpo começa, utilizamos uma linha em branco. No mínimo, uma requisição deve conter uma linha de requisição e um cabeçalho HOST.

  Exemplo:

```
Accept: */*
Accept-Language: pt-br
Connection: keep-alive
Host: www.phpteste.com.br
Referer: http://www.phpteste.com.br/index.php?id=1
User-Agent: Mozilla (X11; I ; Linux I686)
```

- *Corpo:* Caso o método GET seja usado na requisição, o corpo estará vazio, mas se o método utilizado for o POST e a página em questão contiver um formulário HTML com alguns campos, esses valores serão passados pelo corpo da requisição.

Formato da resposta HTTP

- *Linha de resposta* : Apenas uma linha indicando a versão do HTTP e o código de resposta do servidor:

  Exemplo:

  ```
  HTTP/1.1 200 OK
  ```

100-199	informativo, indica que a requisição está sendo processada
200-299	requisição bem-sucedida, o servidor enviará o código HTML sem nenhum problema
300-399	redirecionamento
400-499	o cliente passou uma requisição incorreta ao servidor, no qual não pôde ser executada
500-599	a requisição foi enviada corretamente, porém o servidor não pôde executá-la por estar com problemas internos

- *Cabeçalho*: Idem ao cabeçalho de requisição, porém este enviará as informações sobre os aplicativos utilizados no servidor:

Exemplo:

```
Date: Mon, 11st Feb. 2001, 08:02:43 GMT
Server: Apache/1.3.22 (Unix) PHP/4.1.1
Last-modified: Fri, 08TH Feb 2001, 06:10:00 GMT
```

- *Corpo:* Caso a requisição seja aceita e executada sem problemas pelo servidor web, o código HTML requerido será enviado ao navegador.

Ambiente CGI

Muitas pessoas tem uma visão distorcida sobre o Common Gateway Interface, pensam que é uma linguagem de programação, o que não é correto. CGI é um interface de comunicação entre o servidor web e programas externos, que normalmente são utilizados para gerar contextos dinâmicos em páginas HTML. Estes tipos de programas podem ser desenvolvidos em qualquer linguagem que o sistema operacional do servidor web usado possa executar, como por exemplo, C, Perl, Python, PHP, Delphi entre outros. Apesar de seu uso ainda ser muito utilizado, este recurso já está se tornando obsoleto, dando lugar aos módulos embutidos nos servidores web, que na verdade, podem conter o interpretador inteiro da linguagem, ou somente parte dele. Linguagens como Perl, Python e PHP já suportam esse recurso, que tem como vantagem, a maior velocidade de processamento em relação aos programas CGI, por serem executados pelo próprio servidor web e não por processos externos, que demandam mais tempo.

Configuração

Usaremos como sistema operacional a distribuição do GNU/Linux Conectiva 7.0, e os pacotes necessários para o andamento do curso já se encontram instalados. Os pacotes necesários são:

- apache-1.3.22
- php4-4.1.1
- mod_php4-4.1.1
- php4-mysql-4.1.1
- MySQL-3.23.36

Dividiremos em duas partes a configuração do PHP, a primeira referente a ligação entre o servidor web e a linguagem, e a segunda entre a linguagem e o banco de dados.

- *Servidor web e a linguagem PHP (Apache+mod_php4+PHP4)* : Para que o servidor web Apache possa reconhecer as requisições para a execução de scripts PHP, é preciso que o módulo ***mod_php4-4.1.1*** esteja instalado e configurado no Apache.

Para efetuar esta configuração é necessário apenas descomentar algumas linhas no arquivo de configuração do servidor web Apache. Este arquivo se chama "***httpd.conf*** "e se encontra em "***/etc/httpd/conf***" . No editor de texto, faça uma busca pelas linhas listadas abaixo e retire o caracter "**#** " no início de cada linha encontrada:

LoadModule php4_module modules/libphp4.so

AddModule mod_php4.c

AddType application/x-httpd-php .php
AddType application/x-httpd-php-source .phps

DirectoryIndex index.php index.html index.wml

Depois de finalizada a operação anterior é necessário reiniciar o servidor web Apache com os seguintes comandos:

```
$ cds
$ ./httpd restart
```

- *Banco de dados e a linguagem PHP (PHP4+php4-mysql+MySQL)* : Como na configuração anterior, o PHP necessita de um módulo para conectar e executar instruções SQL no banco de dados MySQL. Para isso é necessário estar instalado o módulo "***php4-mysql-4.1.1***" e o banco de dados MySQL "***MySQL-3.23.36***":

Para efetuar a configuração é necessário tirar o comentário no arquivo de configuração "***php.ini***" , que se encontra em "***/etc/php4/apache***" , a seguinte linha:

extension=mysql.so

Teste de funcionamento

Depois de configurados os aplicativos necessários para a execução dos scripts PHP, precisaremos fazer um teste de funcionamento. No editor de textos, digite o código a seguir e salve como "***teste.php***" em "***/home/httpd/html***" :

```
<?php

phpinfo();

?>
```

Sintaxe Básica

O PHP tem uma sintaxe muito simples e enxuta, o que facilita muito a organização dos scripts a serem desenvolvidos. Outro ponto interessante que veremos é que os códigos em PHP são embutidos no HTML, ao invés de gerá-lo por completo, facilitando muito a análise de possíveis erros nos scripts desenvolvidos. A seguir, exemplos da sintaxe do PHP:

1	2	3	4
<?php ... . ?>	<? .... ?>	<% .... %>	<script language="PHP"> .... </script>

Variáveis

Manipular variáveis em PHP é uma atividade simples, como veremos a seguir:

- não é necessário declarar as variáveis, isto é feito quando atribuímos algum valor para elas;
- para declará-las, é necessário apenas colocar como primeiro caracter o '$' , juntamente com a string referente ao nome da variável, e esta string deve começar com uma letra ou o caracter '_';
- PHP é case sensitive, isto é, '$a' é diferente de '$A'. É aconselhável utilizar os nomes de variáveis com letras minúsculas, por causa das variáveis pré-definidas da linguagem, que são declaradas com maiúsculas;

- PHP suporta os seguintes tipos de variáveis:

  - inteiros (integer ou long);
  - ponto flutuante (double ou float);
  - strings
  - arrays
  - objetos *

**** Como se trata de um curso básico, não entraremos em detalhes sobre este tipo***

Tipos suportados

- *Inteiros*

  Sintaxe:

```
$curso = 1000;
$curso = -1000;
$curso = 0234; (inteiro base octal)
$curso = 0x34; (inteiro na base hexadecimal)
```

- *Ponto flutuante*

  Sintaxe:

```
$curso = 1.050;
$curso = 52e3; (equivale a 52000)
```

- *Strings*

Sintaxe:

```
$curso = 'PHP';

# desta maneira, o valor da variável será exatamente o texto
contido entre as aspas

$curso= “PHP”;

# desta maneira, qualquer variável ou caracter de escape será
expandido antes de ser atribuído
```

- *Caracteres de Escape*

\n	nova linha;
\r	retorno de carro (semelhante a \n)
\t	tabulação horizontal
\\	a própria barra (\)
\$	o símbolo $
\’	aspas simples
\”	aspas duplas

- *Arrays* : Array é um tipo de variável que possui seu conteúdo agrupado por índices, como um vetor ou um dicionário. Estes índices podem ser de qualquer tipo suportado pelo PHP, como é mostrado a seguir:

Sintaxe:

```
$estilo_musical[0] = 'pagode';
$estilo_musical[1] = “drum \'n\' bass”;
$estilo_musical[“MPB”] = 'Gilberto Gil';
$estilo_musical[“Rock”] = 'Blind Guardian';
```

- *Listas* : Utilizadas em PHP para realizar atribuições múltiplas, como por exemplo, atribuir valores de um array para variáveis, como mostra a seguir:

  Sintaxe:

```
list($a,$b,$c) = array(0=>”a”, 1=>”b”, 2=>”c”);
```

O trecho de código acima atribuirá simultânea e respectivamente os valores do array às variáveis passadas como parâmetros para o comando **list**. É muito importante lembrar que só serão passadas ao comando **list** os elementos do array que possuírem os índices com valores inteiros e não negativos.

- *Booleans* : Em PHP, não existe um tipo específico para as variáveis do tipo **boolean**, ele trata este tipo com valores inteiros: 0 para false e valores diferentes deste como true.

Transformações de tipos

É possível fazer transformações de tipos de variáveis através das seguintes formas:

- *Coerções* : quando ocorrem determinadas operações matemáticas entre dois valores de tipos diferentes, como por exemplo a adição, o PHP converte um deles automaticamente. Um exemplo disso seria a conversão de uma string para um valor numérico (inteiro ou ponto flutuante), que segue as seguintes regras:

  - É analisado o ínicio da string, se contiver um número, ele será analisado, caso contrário, o valor será 0 (zero);
  - O número pode conter o sinal no início (+ ou -);

- Se a string contiver um ponto em sua parte numérica a ser analisada, ele será considerado, e o valor obtido será um ponto flutuante;

- Se a string contiver as letras "e" ou "E" em sua parte numérica a ser analisada, o valor seguinte será considerado como expoente da base 10, e o valor obtido será um ponto flutuante.

Exemplo de sintaxe:

```
$curso = 1 + “12.8”; ($curso == 13.8)
$curso = 1 + “15”; ($curso == 16)
$curso = 1 + “1.5e3”; ($curso == 1501)
$curso = 1 + “10curso”; ($curso == 11)
$curso = 1 + “  10curso”; ($curso == 11)
$curso = 1 + “+A10testes”; ($curso == 1)
```

- *Transformações explícitas de tipos* : desta forma precisaremos utilizar a sintaxe de typecast do PHP, como os exemplos a seguir:

```
$curso = 20;  (integer(20))
$curso = (double)$curso;  (double(20.0))
$curso = 3.9;  (double(3.9))
$curso = (int)$curso (o valor é truncado e fica como integer(3))
```

- *Tipos suportados nas transformações explícitas:*

```
(int), (integer) =  muda para inteiro;
(real), (double), (float) = muda para ponto flutuante;
(string) = muda para string
(array) = muda para array
(object) = muda para objeto
```

- *Função settype* : trabalha igualmente as tranformações explícitas, porém com sintaxe diferente, como o exemplo a seguir:

```
$curso = 20;  (integer)
settype($curso, double);

# o valor da variável $curso foi transformada em ponto flutuante
```

Operadores

- *Aritméticos:*

+	Adição
-	Subtração
*	Multiplicação
/	Divisão
%	Módulo

- *Strings:*

.	Concatenação

- *Atribuição:*

=	Atribuição simples
+=	Atribuição com adição
-=	Atribuição com subtração
*=	Atribuição com Multiplicação
/=	Atribuição com divisão
%=	Atribuição com módulo
.=	Atribuição com concatenação

Exemplo:

```
$curso = 7;
$curso += 2;  ($curso fica com o valor 9)
```

- *Lógicos:*

and	"e" lógico
or	"ou" lógico
xor	"ou" exclusivo
!	Não (inversão)
&&	"e" lógico
||	"ou" lógico

- *Comparação:*

==	igual a
!=	diferente de
<	menor que
>	maior que
<=	menor ou igual a
>=	maior ou igual a

- *Incremento e decremento:*

++	igual a
--	diferente de

Estes podem receber o valor antes ou depois da variável:

- *Antes:* retorna o valor da variável antes de incrementá-la ou decrementá-la:

  Exemplo:

```
$a = 1;
$b = ++a; ($b recebe 2, valor de $a já incrementado)
```

- *Depois:* retorna o valor da variável já incrementada ou decrementada:

  Exemplo:

```
$a = 1;
$b = a++; ($b recebe 1  e $a passa a ter 2)
```

Estruturas de controle

- ***If*** : O comando ***if*** testa a condição passada e executa o bloco de código caso o valor retornado da condição seja verdadeiro:

```
$a = 1;

if ($a == 1)
{
....
....
...
}
```

Caso a condição passada retorne um valor falso, e seja necessário executar um bloco de código diferente, utiliza-se a instrução ***else:***

```
$a = 1;
$b = 2;

if ($a > $b)
   {
     ....
     ...
   }
else
   {
    .....
    ....
   }
```

Ainda existe a instrução ***elseif,*** para situações onde precisa-se verificar mais que uma condição:

```
$a = 1;
$b = 2;
$c = 3;

if ($a > $b)
   {
      echo " a é maior que b " ;
   }
elseif ($a > $c)
    {
      echo " a é maior que c ";
    }
else
    {
      echo " a é menor que b e c ";
    }
```

- ***Switch*** : Comando utilizado para fazer múltiplos testes de condição. A idéia deste comando é igual ao do ***elseif*** , porém com algumas diferenças:

```
$a = 2;

switch ($a)
{
    case 0:
         echo " a é igual a 0 ";
         break;
    case 1:
         echo " a é igual a 1 ";
         break;
    case 2:
         echo " a é igual a 2 ";
         break;
}
```

A idéia do comando ***switch*** é achar a condição verdadeira e executar qualquer bloco de código que esteja abaixo dela, inclusive os que não forem do seu trecho, por esse motivo, utilizamos o comando ***break*** logo abaixo da última linha do bloco de código, como o

exemplo anterior. O comando ***switch*** também aceita testes de condição em qualquer tipo de variável suportado pelo PHP:

```
$a = “curso“;

switch ($a)
{
    case “PHP“:
         echo “ a é igual a PHP “;
         break;
    case “curso“:
         echo “ a é igual a Curso “;
         break;
    case “CCUEC“:
         echo “ a é igual a CCUEC “;
         break;
}
```

- ***While*** : Este comando é utilizado para realizar laços condicionais. Ele executa o bloco de código enquanto a condição passada for verdadeira, e caso a condição inicial que foi passada se torne falsa, o bloco não será executado:

```
$a = 1;

while ($a <= 10)
{
  echo “Número“.$a++.“<br>“;
}
```

- ***Do ... while*** : Este comando tem a mesma idéia que o comando ***while***, porém, seu teste de condição é feito no final do bloco de código:

```
$c = 0;
do
{
    echo “Número“.++$c.“<br>“;
} while ($c < 10);
```

- ***For*** : Como nos outros comando que realizam laços condicionais, o comando ***for*** também precisa de uma condição para ser testada a cada laço realizado, porém, este comando necessita de mais dois parâmetros, que seriam a declaração da variável contadora e a instrução de incremento:

```
for ($a=0; $a<=10; $a++)
    {
        echo “Número”.$a.”<br>”;
    }
```

Quebra de fluxo

- ***Break*** : O comando ***break*** pode ser utilizado em comandos de laços condicionais e no comando switch, e sua função é parar imediatamente o fluxo do bloco de código:

```
$a = 20;

while ($a < 0)
    {
        if ($a == 5)
             {
           echo “Número inválido!”;
     break;
             }
        echo “Número  “.$a.”<br>”;
        $a--;
    }
```

- ***Continue*** : O comando ***continue*** também funciona dentro dos laços condicionais, porém, não pára o fluxo do bloco de código, e sim, volta para o início dele:

```
for ($a=0;$a<10;$a++)
   {
        if ($a == 5)
           {
             continue;
           }
       else
           {
              echo “Número  “.$a.”<br>”;
           }
    }
```

Funções

Funções são pequenas seções independentes de código que podem ser chamadas a qualquer momento e em qualquer ordem, que servem para desempenhar tarefas específicas dentro dos scripts. O exemplo a seguir mostra a sua sintaxe básica:

```
function soma ($a, $b)
 {
       $c = $a + $b;
       return $c;
 }
```

A instrução ***return*** é opcional, já que não é obrigatório retornar algum valor em funções no PHP, outra regra é a de não permitir que sejam retornados múltiplos valores através desta instrução. Para resolver essa necessidade, pode-se retornar listas e arrays, como mostra o exemplo a seguir:

```
function soma ($a, $b)
{
     $c = $a + $b;
     $d = $c - 5;
     return array($c,$b,$d)
}

list ($f,$g,$h) = soma(10,10);

echo $f."<br>";
echo $g."<br>";
echo $h."<br>";
```

- *Passagem de parâmetros por referência* : Normalmente, a passagem de parâmetros em PHP é feita através dos valores das variáveis, não permitindo assim, a alteração do valor na variável original, como mostra o exemplo a seguir:

```
$cont = 10;

function contador ($a)
{
   $a++;
}

contador($cont);
echo $cont;
```

No exemplo acima, a variável original permanecerá com o mesmo valor porque não foi definido a passagem de parâmetros por referência, o que alteraria também o valor da variável original. Uma das maneiras de se utilizar esse recurso é colocar o caracter "&" antes do nome da variável na declaração da função, como mostra o exemplo a seguir:

```
$cont = 10;

function contador(&$a)
{
      $a++;
}

contador($cont);
echo $cont;
```

Poderíamos também utilizar a passagem de parâmetros por referência apenas quando fossemos chamar a função, e não em sua declaração:

```
contador(&$cont);
echo $cont;
```

Escopo das variáveis

Discutimos anteriormente sobre variáveis e os tipos suportados pelo PHP. Agora, discutiremos sobre os escopos destas variáveis, que podem ser dos seguintes tipos:

- globais;
- locais;
- estáticas;
- constantes.

- *Globais*: As variáveis globais são por definição, as variáveis que podem ser acessadas dentro de todo o script. Porém, quando cria-se escopos locais como nas funções, precisaremos utilizar um tipo de chamada especial, como no exemplo a seguir:

```
$curso = 'PHP';

function mostra()
{
     global $curso;
     echo $curso;
}

mostra();
```

O mesmo recurso pode ser acessado através da array GLOBALS, que nos permite acessar todas as variáveis globais do script. O exemplo acima pode ser reescrito da seguinte maneira:

```
$curso = 'PHP';

function mostra()
{
     echo $GLOBALS[“curso“];
     echo $curso;
}

mostra();
```

- *Locais*: As variáveis locais são o tipo mais restrito dentro do PHP. Elas funcionam apenas dentro deste escopo, como mostra o exemplo a seguir:

```
$curso = 'PHP';

function mostra()
{
     $var_local = 'variável local';
     echo $var_local;
}

echo “<b>   $var_local </b>”;
```

- *Estáticas*: As variáveis estáticas são variáveis que possuem o mesmo tempo de vida das variáveis globais, com a diferença de funcionarem apenas em escopos locais e serem inicializadas uma só vez. A seguir, um exemplo deste recurso:

```
function contador()
{
    static $i = 0;
   echo $i++.“<br>“;
}


for ($a=0; $a<=5; $a++)
{
    contador();
}
```

Projeto

Desenvolveremos um site dinâmico utilizando a linguagem PHP e o servidor de banco de dados MySQL. O objetivo desse site será a localização de funcionários na Unicamp. Fazendo-se uma busca pelo nome do funcionário, serão disponibilizadas informações (unidade, telefone, fax, e-mail e cargo) que permitam localizá-lo na universidade.
O site permitirá inclusão, consulta, alteração e exclusão de dados. Todas as páginas e programas ficarão armazenados no diretório **home/httpd/html/cursophp**.

1 - Criação da base de dados e tabelas

Utilizando o servidor de banco de dados MySQL, o primeiro passo será definir a base de dados e as tabelas em que guardaremos as informações.
Podemos criar nossa estrutura de dados diretamente no MySQL, da seguinte forma:

1. Abra uma janela de terminal
2. Digite o comando **mysql –u root –p** (será solicitada a senha do administrador)
3. Crie uma base de dados no MySQL, que conterá as tabelas a serem utilizadas no projeto.
   O comando é **create database unicamp;**
   Em que 'unicamp' é o nome da base de dados.
4. Depois de criada, acesse a base de dados com o seguinte comando:
   use unicamp;

Agora já podemos pensar nas tabelas que serão necessárias para esse projeto. Para facilitar, vamos utilizar uma única tabela, chamada **funcionarios**.

Dicas:

Tipos de campos:

varchar(N) : um campo caractere variável de no máximo N caracteres;

integer : um inteiro padrão;

char(N) : um campo caractere com exatamente N caracteres;

text : um campo com um comprimento máximo de 65535 caracteres;

date : uma data no formato 'AAAA-MM-DD'.

Not null: significa que o campo não pode ser nulo.

Primary key: significa que é campo chave.

Para criar essa tabela, utilizaremos a seguinte sintaxe:

```
create table funcionarios (
nome varchar(50) not null primary key,
unidade varchar(40) not null,
telefone varchar(10) not null,
email varchar(40),
cargo varchar(40) not null
);
```

Dicas:

Para adicionar ou excluir campos da tabela, depois que ela foi criada:

Supondo que quiséssemos excluir o campo *cargo* da tabela funcionarios:
alter table funcionarios drop column cargo;

Supondo que quiséssemos adicionar novamente o campo *cargo* na tabela funcionarios:
alter table funcionarios add column cargo;

Para visualizar as bases de dados existentes:
show databases;

Para visualizar as tabelas pertencentes a uma base de dados:
use base de dados;
show tables;

Para visualizar os campos de uma tabela:
desc tabela;

Para visualizar todos os registros de uma tabela:
select * from tabela;

2 - Criação da home page do site

A página principal (homepage) do site será bastante simples e trará um menu com as opções de inclusão, consulta, alteração e exclusão. As opções terão links para seus respectivos módulos. Essa página inicial terá a extensão html e será criada utilizando-se um editor de páginas (**Netscape Composer**).

Abra o Netscape e na barra de menus escolha as opções **arquivo – nova – página em branco** (uma nova página será aberta)	
Defina a cor de fundo da página: - clique sobre a página com o botão direito do mouse - selecione **cores e propriedades da página** - clique no botão **usar cores personalizadas** - escolha a cor branca - clique em **Aplicar** e **Ok**	
Na barra de ferramentas do Composer, utilize a opção para centralizar o cursor	
Na barra de ferramentas do Composer, utilize a opção para inserir imagem e selecione a imagem **topo.gif** (que está em home/httpd/html/cursophp)	
Pule 1 linha e mantenha o cursor no centro da página. Digite os ítens do menu (em negrito): “Inclusão de funcionários”, “Consulta de funcionários”, “Alteração dos dados de funcionários” e “Exclusão de funcionários”.	
Transforme as opções do menu em links e direcione para seus respectivos endereços: inclusao.html consulta.html alteracao.html exclusao.html	
Salve a página como **index.html** e teste-a digitando o endereço http://localhost/cursophp/index.html	

3 - Módulo de Inclusão

Vamos criar a página para o formulário de inclusão. Os recursos do Netscape Composer são muito limitados para a confecção de formulários, sendo assim digitaremos o código utilizando um editor de texto. Esse arquivo terá extensão html.

3.1) Formulário inclusao.html

```
<html>
<head>
<title>Inclusao.html</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>
<body bgcolor="#FFFFFF">
<table width="640" border="0" cellspacing="0" align="center">
  <tr>
    <td>
      <p><img src="topo.gif" width="640" height="44"></p>
      <p><b>Formulário de inclusão: <br></b></p>
```

```
      <form method="post" action="inclusao.php">
        <p>Nome completo:
          <input type="text" name="nome" size="25" maxlength="50">
        </p>
        <p>Unidade: <input type="text" name="unidade" size="40"
           maxlength="40"> </p>
        <p>Telefone:
          <input type="text" name="telefone" maxlength="10" size="10">
        </p>
        <p>E-mail:
          <input type="text" name="email" size="25" maxlength="40">
        </p>
        <p>Cargo: <input type="text" name="cargo" size="40"
       maxlength="40"> </p>
        <p>
          <input type="submit" name="Submit" value="Enviar">
          <center> <b> <a href="index.html">Home</a> </b> </center>
        </p>
      </form>
    </td>
  </tr>
  <tr>
    <td> </td>
  </tr>
</table>
</body>
</html>
```

Após digitar o código, salve-o e teste-o. Quando o formulário for submetido, dará um erro, alertando que o programa "inclusao.php" (para o qual estamos encaminhando os dados) não existe. Precisamos, então, criá-lo. Notem que a próxima página não terá mais a extensão html e sim php, pois o código vai conter programação PHP.

Dicas:

Quando o servidor recebe a requisição de uma página HTML, ele apenas envia a página requisitada. Por outro lado, quando a requisição é de uma página com extensão PHP, o servidor processa o código antes de enviá-la.
Pode-se combinar os códigos HTML e PHP.

3.2) Programa inclusão.php

O programa "inclusao.php" vai tratar os dados recebidos através do formulário, incluindo-os no banco de dados.

Relembrando:

As **tags** que identificam o início e o fim da programação PHP são:

<?php

código php

?>

A sintaxe do **IF/ELSE**:

```
if (condição) {
   echo ("mensagem1");
   echo ($variavel1); }
else {
   echo ("mensagem2");
   echo ($variavel2); }
```

Variáveis: as variáveis devem ser precedidas pelo caractere "**$**".

Dicas:

No código referente ao programa "inclusão.php", utilizaremos algumas funções do PHP:

Trim: tira espaços em branco de uma variável.

A expressão "**or die**" pode ser usada como uma alternativa para o "if/else".

```
<html>
<head>
<title>Inclusao.php</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>
<body bgcolor="#FFFFFF">
<?php

// Tirar espaço em branco das variáveis recebidas pelo formulário
$nome = trim($nome);
$unidade = trim($unidade);
$telefone = trim($telefone);
$email = trim($email);
$cargo = trim($cargo);

echo ("<p><center><img src=\"topo.gif\" width=\"640\"
height=\"44\"></center></p>");
```

```
// Consiste Nome
if (empty($nome) || empty($unidade) || empty($telefone) || empty($cargo))
{
 echo ("<font color=\"#FF0000\">
 <b>Campo(s) obrigatório(s) não preenchido(s)</b></font>");
 echo ("
 <table width=\"640\" border=\"0\" cellspacing=\"0\" align=\"center\">
  <tr>
    <td>
      <p><b>Formulário de inclusão: <br>
        </b></p>
      <form method=\"post\" action=\"inclusao.php\">
        <p>Nome completo:
          <input type=\"text\" name=\"nome\" value=\"$nome\" size=\"25\"
maxlength=\"50\">
        </p>
        <p>Unidade: <input type=\"text\" name=\"unidade\"
value=\"$unidade\" size=\"40\"
           maxlength=\"40\"> </p>
        <p>Telefone:
          <input type=\"text\" name=\"telefone\" value=\"$telefone\"
maxlength=\"10\" size=\"10\">
        </p>
        <p>E-mail:
          <input type=\"text\" name=\"email\" value=\"$email\"
size=\"25\" maxlength=\"40\">
        </p>
        <p>Cargo: <input type=\"text\" name=\"cargo\" value=\"$cargo\"
size=\"40\"
       maxlength=\"40\"> </p>
        <p>
          <input type=\"submit\" name=\"Submit\" value=\"Enviar\">
          <center> <b> <a href=\"index.html\">Home</a> </b> </center>
        </p>
      </form>
    </td>
  </tr>
  <tr>
    <td> </td>
  </tr>
 </table>
");
}
else {

 // Inclui os dados recebidos do formulário na tabela funcionarios

 // Cria uma conexão com o servidor MySQL passando host, username e senha
 $conec = mysql_connect ("localhost","root","Unicamp") or die
 ("Falha na conexão com o banco de dados");

 // Declaração SQL
 $declar = "INSERT into funcionarios values ('$nome', '$unidade',
'$telefone', '$email', '$cargo')";

 // Roda a query e trata o resultado
```

```
 if (mysql_db_query ("unicamp", $declar, $conec)) {
   echo ("<BR><BR>");
   echo ("<center> <b> <font size = 4> Inclusão Efetuada </font> </b>
   </center>");
   echo ("<BR>");
   echo ("<center> <b> <a href=\"inclusao.html\">Voltar</a> </b>
</center>");
 }
 else {
   echo ("<BR><BR>");
   echo ("<center> <b> <font size = 4> Erro - Inclusão não Efetuada
</font>
   </b> </center>");
   echo ("<BR>");
   echo ("<center> <b> <a href=\"inclusao.html\">Voltar</a> </b>
</center>");
 }

// Fecha a conexão com o servidor MySQL (Opcional)
mysql_close ($conec);
}
?>
</body>
</html>
```

3.3) Testando o módulo de inclusão

Abra o navegador (Netscape) e digite o endereço do site: http://localhost/cursophp/index.html
No menu da página principal, clique na opção **inclusão**.
Deixe os campos do formulário em branco. Clique em enviar. Deverá mostrar uma mensagem de erro. O único campo que não é obrigatório é o e-mail.
Preencha o formulário com os dados do funcionário: *nome completo*, *unidade*, *telefone*, *e-mail* e *cargo*. Clique em *enviar*. Deverá mostrar a mensagem “Inclusão Efetuada”.
Volte para a página do formulário e entre com outros dados, só que desta vez entre com um nome que já existe no banco de dados. Clique em *enviar*. Deverá mostrar a mensagem “Inclusão não efetuada”, pois o campo *nome* é chave e não aceita valores duplicados.
Insira pelo menos 5 funcionários.

4 - Módulo de Consulta

Vamos criar a página com o formulário de consulta.

4.1) Formulário consulta.html

```
<html>
<head>
<title>Consulta.html</title>
```

```
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>
<body bgcolor="#FFFFFF">
<table width="640" border="0" cellspacing="0" align="center">
  <tr valign="top">
    <td>
      <p><img src="topo.gif" width="640" height="44"></p>
      <!mensagem>
      <p><b>Formulário de consulta<br>
        </b></p>
      <form method="post" action="consulta.php">
        <p>Nome completo:
          <input type="text" name="nome" size="25" maxlength="50">
        </p>
        <p>
          <input type="submit" name="Submit" value="Enviar"> </p>
        </form>
        <center> <b> <a href="index.html">Home</a> </b> </center>
    </td>
  </tr>
  <tr>
    <td> </td>
  </tr>
</table>
</body>
</html>
```

Após digitar o código, salve e teste. Quando o formulário for submetido, dará um erro, alertando que o programa "consulta.php" (para o qual estamos encaminhando os dados) não existe. Precisamos, então, criá-lo. Notem que essa página não será mais html e sim php, pois o código vai conter programação PHP.

4.2) Programa consulta.php

O programa “consulta.php” vai receber o dado do formulário, pesquisar no banco de dados e mostrar as informações referentes ao dado.

Dicas:

No código referente ao programa “consulta.php” utilizaremos mais algumas funções do PHP:

File: lê um arquivo, retornando o seu conteúdo como um array; cada linha do arquivo será representada por um elemento do array.
Implode: armazena todo o conteúdo de um array como uma string, concatena os conteúdos de cada elemento do array em uma string, utilizando ou não um delimitador entre eles.
Str_replace: vai ler uma string e substituir um determinado valor por outro. No nosso caso, essa função substituirá a expressão “<!mensagem>”, contida no código do “consulta.html”, por uma mensagem de erro. Portanto, não se

esqueça de colocar no html a expressão a ser substituída (como comentário).
mysql_num_rows: obtem o número de registros que retornou do select.
mysql_fetch_row: obtem os campos do registro que retornou do select.

```
<html>
<head>
<title>Consulta.php Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>
<body bgcolor="#FFFFFF">
<?php
// Tirar espaço em branco das variáveis recebidas pelo formulário
$nome = trim($nome);
// Consiste Nome
if (empty($nome)) {
$html = file("consulta.html");
$html = implode(" ",$html);
$erro = "<center><font color=\"#FF0000\"> Preencha o campo
<b>Nome</b></font></center>";
$html = str_replace("<!mensagem>",$erro,$html);
echo ($html);
}
else {
echo ("<p><center><img src=\"topo.gif\" width=\"640\"
height=\"44\"></center></p>");

// Cria uma conexão com o servidor MySQL
// Parâmetros: host, username, senha
$conec = mysql_connect ("localhost","root","Unicamp");

// Declaração do SQL
$declar = "SELECT unidade, telefone, email, cargo from funcionarios
where nome = '$nome'";

// Roda a query e verifica se encontrou registro
$query = mysql_db_query ('unicamp', $declar, $conec) or die ("Erro no
acesso ao banco");
$achou = mysql_num_rows($query);
// echo ($achou);

// Se encontrou, guarda as variáveis
if ($achou > 0) {
$row = mysql_fetch_row ($query);
    $unidade = $row[0];
    $telefone = $row[1];
    $email = $row[2];
    $cargo = $row[3];
    echo ("<BR>");
    echo ("<table width=\"640\" border=\"0\" cellspacing=\"0\"
align=\"center\"> <tr> <td>");
    echo ("<b> Resultado da Consulta </b>");
```

```
    echo ("<BR><BR>");
    echo ("<b> Nome: </b> $nome <BR>");
    echo ("<b> Unidade: </b> $unidade <BR>");
    echo ("<b> Telefone: </b> $telefone <BR>");
    echo ("<b> E-mail: </b> $email <BR>");
    echo ("<b> Cargo: </b> $cargo <BR>");
    echo ("</td> </tr> </table>");
    echo ("<center> <b> <a href=\"consulta.html\">Voltar</a> </b>
</center>");
}
else {
   echo ("<BR>");
   echo ("<center> <b> Funcionário não cadastrado </b> </center>");
   echo ("<BR>");
   echo ("<center> <b> <a href=\"consulta.html\">Voltar</a> </b>
</center>");
}
}
?>
</body>
</html>
```

4.3) Testando o módulo de consulta

Abra o navegador (Netscape) e digite o endereço do site: http://localhost/cursophp/index.html
No menu da página principal, clique na opção **consulta**.
Deixe o campo *nome do funcionário* em branco e clique em *enviar*. Deverá mostrar uma mensagem de erro.
Preencha o formulário com um nome de funcionário inexistente e clique em *enviar*. Deverá mostrar a mensagem “Funcionário não cadastrado”.
Preencha o formulário com um nome de funcionário válido e clique em *enviar*. Deverá mostrar os dados do funcionário.

5 - Módulo de Exclusão

Vamos criar a página com o formulário de exclusão.

5.1) Formulário exclusao.html

```
<html>
<head>
<title>Exclusao.html</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>

<body bgcolor="#FFFFFF">
<table width="640" border="0" cellspacing="0" align="center">
  <tr valign="top">
```

```
    <td>
      <p><img src="topo.gif" width="640" height="44"></p>
      <!mensagem>
      <p><b>Formulário de exclusão: <br>
        </b></p>
      <form method="post" action="exclusao.php">
        <p>Nome Completo:
          <input type="text" name="nome" size="25" maxlength="50">
        </p>
        <p>
          <input type="submit" name="Submit" value="Enviar">
        </p>
        </form>
        <center> <b> <a href="index.html">Home</a> </b> </center>
    </td>
  </tr>
  <tr>
    <td> </td>
  </tr>
</table>
</body>
</html>
```

Após digitar o código, salve-o e teste-o. Quando o formulário for submetido, dará um erro, alertando que o programa "exclusao.php" (para o qual estamos encaminhando os dados) não existe. Precisamos, então, criá-lo. Notem que esta página não será mais html e sim php, pois o código vai conter programação PHP.

5.2) Programa exclusao.php

O programa “exclusao.php” vai receber o dado do formulário, confirmar através de consulta ao banco de dados se o funcionário está cadastrado, e excluir o registro.

```
<html>
<head>
<title>Exclusao.php</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>

<body bgcolor="#FFFFFF">

<?php

// Tirar espaço em branco das variáveis recebidas pelo formulário
$nome = trim($nome);

// Consiste Nome
if (empty($nome)) {
$html = file("exclusao.html");
$html = implode(" ",$html);
$erro = "<center><font color=\"#FF0000\"> Preencha o campo
<b>Nome</b></font></center>";
$html = str_replace("<!mensagem>",$erro,$html);
echo ($html);
}
```

```
else {
echo ("<p><center><img src=\"topo.gif\" width=\"640\"
height=\"44\"></center></p>");

// Cria uma conexão com o servidor MySQL
$conec = mysql_connect ("localhost","root","Unicamp");

// Declaração do SQL
$declar = "SELECT nome from funcionarios where nome = '$nome'";

// Roda a query, verifica se funcionário é cadastrado
$query = mysql_db_query ('unicamp', $declar, $conec) or die ("Erro no
acesso ao banco");
$achou = mysql_num_rows($query);
//echo ($achou);
// Se encontrou exclui, senão mostra mensagem
if ($achou > 0) {
   echo ("<BR><BR>");
   echo ("<center> Funcionário: $nome </center>");
   echo ("<BR>");

   // Exclui registro na tabela funcionarios

   $declar2 = "DELETE from funcionarios where nome = '$nome'";
   if (mysql_db_query ('unicamp', $declar2, $conec)) {
      echo ("<BR><BR>");
      echo ("<center> <b> <font size = 4> Exclusão Efetuada </font> </b>
           </center>");
      echo ("<BR><BR>");
echo ("<center> <b> <a href=\"exclusao.html\">Voltar</a> </b>
</center>");
}
   else {
      echo ("<BR><BR>");
      echo ("<center> <b> <font size = 4> Erro - Exclusão não Efetuada
</font>
      </b> </center>");
      echo ("<BR><BR>");
    echo ("<center> <b> <a href=\"exclusao.html\">Voltar</a> </b>
</center>");
}
}
else {
   echo ("<BR><BR>");
   echo ("<center> <b> Funcionário não cadastrado </b>
</center>");
   echo ("<BR><BR>");
echo ("<center> <b> <a href=\"exclusao.html\">Voltar</a> </b>
</center>");
 }

mysql_close ($conec);
}
?>
</body>
</html>
```

5.3) Testando o módulo de exclusão

Abra o navegador (Netscape) e digite o endereço do site: http://localhost/cursophp/index.html
No menu da página principal, clique em **exclusão**.
Deixe o campo *nome do funcionário* em branco e clique em *enviar*. Deverá mostrar uma mensagem de erro.
Preencha o formulário com o nome completo do funcionário e clique em *enviar*. Deverá mostrar a mensagem “Exclusão efetuada”.
Preencha o formulário com o nome do funcionário que você acabou de excluir e clique em *enviar*. Deverá mostrar a mensagem “Funcionário não cadastrado”.

6 - Módulo de Alteração

Vamos criar a página com o formulário de alteração.

6.1) Formulário alteracao.html

```
<html>
<head>
<title>Alteracao.html</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>
<body bgcolor="#FFFFFF">
<table width="640" border="0" cellspacing="0" align="center">
  <tr valign="top">
    <td>
      <p><img src="topo.gif" width="640" height="44"></p>
      <!mensagem>
      <p><b>Formulário de alteração </b></p>
      <form method="post" action="alteracao.php">
        <p>Nome completo:
          <input type="text" name="nome" size="25" maxlength="50">
        </p>
        <p>
          <input type="submit" name="Submit" value="Enviar">
        </p>
        </form>
        <center> <b> <a href="index.html">Home</a> </b> </center>
    </td>
  </tr>
  <tr>
    <td> </td>
  </tr>
</table>
</body>
</html>
```

Após digitar o código, salve-o e teste-o. Quando o formulário for submetido, dará um erro, alertando que o programa "alteracao.php" (para o qual estamos encaminhando os dados) não existe. Precisamos, então, criá-lo. Notem que esta página não será mais html e sim php, pois o código vai conter programação PHP.

6.2) Programa alteracao.php

O programa “alteracao.php” vai receber o dado do formulário, recuperar as informações do banco de dados e mostrá-las num formulário para que elas sejam alteradas. Para montar o formulário, criaremos uma função php.

```
<html>
<head>
<title>Alteracao.php</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>

<body bgcolor="#FFFFFF">

<?php

include ("funcoes.php");

// Tirar espaço em branco das variáveis recebidas pelo formulário
$nome = trim($nome);

// Consiste Nome
if (empty($nome)) {
$html = file("alteracao.html");
$html = implode(" ",$html);
$erro = "<center><font color=\"#FF0000\"> Preencha o campo
<b>Nome</b></font></center>";
$html = str_replace("<!mensagem>",$erro,$html);
echo ($html);
}
else {
echo ("<p><center><img src=\"topo.gif\" width=\"640\"
height=\"44\"></center></p>");

// Cria uma conexão com o servidor MySQL
// Parâmetros: host, username, senha
$conec = mysql_connect ("localhost","root","Unicamp");

// Declaração do SQL
$declar = "SELECT unidade, telefone, email, cargo from funcionarios
where nome = '$nome'";

// Roda a query e verifica se encontrou registro
$query = mysql_db_query ('unicamp', $declar, $conec) or die ("Erro no
acesso ao banco");
$achou = mysql_num_rows($query);
// echo ($achou);

// Se encontrou, guarda as variáveis
if ($achou > 0) {
```

```
$row = mysql_fetch_row ($query);
    $unidade = $row[0];
    $telefone = $row[1];
    $email = $row[2];
    $cargo = $row[3];
    monta_pagina($nome,$unidade,$telefone,$email,$cargo);
}
else {
   echo ("<BR><BR>");
   echo ("<center> <b> Funcionário não cadastrado </b> </center>");
   echo ("<BR>");
   echo ("<center> <b> <a href=\"alteracao.html\">Voltar</a> </b>
</center>");
}
}
?>

</body>
</html>
```

Notem que, no código que acabamos de digitar, estamos chamando a função **monta_pagina**, passando como parâmetros as variáveis **nome**, **unidade**, **telefone**, **email** e **cargo**.
As funções são úteis porque podem ser reutilizadas em vários programas, além disso, o tamanho do código do programa chamador diminui consideravelmente.
Podemos criar um único programa (exemplo: funcoes.php) que conterá todas as funções.
Um detalhe importante que não podemos esquecer é que precisamos incluir esse programa de funções em nosso programa chamador. No código visto anteriormente temos o comando **include ("funcoes.php")** logo no início do código php.

6.3) Programa funcoes.php

O programa “funcoes.php” pode armazenar todas as funções que serão utilizadas no site. Neste curso usaremos apenas a função **monta_pagina**. Essa função serve para montar o formulário já preenchido, com as informações que foram passadas como parâmetros no programa anterior.
Observação: com algumas pequenas alterações, essa função também poderia ser usada para recriar o formulário do módulo de inclusão.

```
<?php
function
monta_pagina($nome,$unidade,$telefone,$email,$cargo)
{
echo "<table width=\"640\" border=\"0\" cellspacing=\"0\"
align=\"center\">";
echo "<tr>";
echo "<td>";
echo "<p><b>Formul&aacute;rio de altera&ccedil;&atilde;o: <br></b></p>";
echo "<form method=\"post\" action=\"alteracao2.php\">";
echo "<p>Nome: $nome </p>";
echo "<p>Unidade: <input type=\"text\" name=\"unidade\"
value=\"$unidade\"
     size=\"40\" maxlength=\"40\"> </p>";
echo "<p>Telefone: <input type=\"text\" name=\"telefone\"
```

```
value=\"$telefone\"
     maxlength=\"10\" size=\"10\"> </p>";
echo "<p>E-mail: <input type=\"text\" name=\"email\" value=\"$email\"
     size=\"25\" maxlength=\"25\"> </p>";
echo "<p>Cargo: <input type=\"text\" name=\"cargo\" value=\"$cargo\"
     size=\"40\" maxlength=\"40\"> </p>";
echo "<p> <input type=\"submit\" name=\"Submit\" value=\"Enviar\">
      </p>";
echo "<p> <input type=\"hidden\" name=\"nome\" value=\"$nome\"></p>";
echo "</form>";
echo "</td>";
echo "</tr>";
echo "<tr>";
echo "</tr>";
echo "</table>";
echo ("<center> <b> <a href=\"alteracao.html\">Voltar</a> </b>
</center>");
return; }
?>
```

Notem que o formulário criado pela função **monta_pagina** chama o programa **alteracao2.php**. Isso porque, para completarmos o módulo de alteração precisamos de mais um programa que pegue as informações que foram alteradas e as inclua no banco de dados.
Observação: como o campo **nome** não é passado para o programa **alteracao2.php**, por não se tratar de uma variável do formulário, temos que passá-lo como um campo escondido `input type="hidden"`.

6.4) Programa alteracao2.php

O programa “alteracao2.php” vai pegar as informações alteradas e fazer um update no banco de dados.

```
<html>
<head>
<title>Alteracao2.php Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>
<body bgcolor="#FFFFFF">
<p><center><img src="topo.gif" width="640" height="44"></center></p>
<?php
include ("funcoes.php");

// Tirar espaço em branco das variáveis recebidas pelo formulário
$nome = trim($nome);
$unidade = trim($unidade);
$telefone = trim($telefone);
$email = trim($email);
$cargo = trim($cargo);

if (empty($nome) || empty($unidade) || empty($telefone) || empty($cargo))
{
echo ("<font color=\"#FF0000\">
<b>Campo(s) obrigatório(s) não preenchido(s)</b></font>");
monta_pagina($nome, $unidade, $telefone, $email, $cargo);
```

```
}
else
{
// Cria uma conexão com o servidor MySQL
$conec = mysql_connect ("localhost","root","Unicamp") or die
("Falha na conexão com o banco de dados");
/*
comentário
*/
$declar = "UPDATE funcionarios SET unidade='$unidade',
telefone='$telefone', email='$email', cargo='$cargo'
WHERE nome='$nome'";

// Roda a query e trata o resultado
if (mysql_db_query ("unicamp", $declar, $conec)) {
   echo ("<BR><BR>");
   echo "<center> <b> <font size = 4> Alteração Efetuada! </font> </b>
   </center>";
   echo ("<BR>");
   echo ("<center> <b> <a href=\"alteracao.html\">Voltar</a> </b>
</center>");
}
else {
   echo ("<BR><BR>");
   echo ("<center> <b> <font size = 4> Erro - Alteração não Efetuada
</font>
   </b> </center>");
   echo ("<BR>");
   echo ("<center> <b> <a href=\"alteracao.html\">Voltar</a> </b>
</center>");
}

// Fecha a conexão com o servidor MySQL (Opcional)
mysql_close ($conec);
}
?>
</body>
</html>
```

6.5) Testando o módulo de alteração

Abra o navegador (Netscape) e digite o endereço do site: http://localhost/cursophp/index.html
No menu da página principal, clique em **alteração**.
Deixe o campo *nome do funcionário* em branco e clique em *enviar*. Deverá mostrar uma mensagem de erro.
Preencha o formulário com um nome de funcionário que não existe e clique em *enviar*. Deverá mostrar a mensagem “Funcionário não cadastrado”.
Preencha o formulário com um nome de funcionário válido e clique em *enviar*. Será mostrado um formulário com os dados desse funcionário. Altere alguns campos e clique em *enviar*. Deverá mostrar a mensagem “Alteração efetuada”.
Entre no módulo de consulta e confira se os dados foram realmente alterados.

Dicas:

Como obter data e hora do sistema.

No exemplo a seguir obtemos data e hora usando a função date, jogamos o conteúdo em variáveis e mostramos essas variáveis na tela.
Parâmetros utilizados na função date:
j: dia
m: mês
Y: ano
H: hora
i: minutos
s: segundos

```
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type"
content="text/html; charset=iso-8859-1">
</head>
<body bgcolor="#FFFFFF">
<?php
$data = date("j/m/Y");
$hora = date("H:i:s");

echo ("Data: $data");
echo ("<br><br>");
echo ("Hora: $hora");
?>
</body>
</html>
```

Observação: se fôssemos gravar a data num banco de dados (aaaa/mm/dd), ao invés de mostrá-la, a sintaxe seria a seguinte:

```
$data = date("Y/m/j");
```

Referência Bibliográfica

- ***Beginning PHP4 - Programando***
  Autores: Wankyu Choi, Allan Kent, Chris Lea, ganesh Prasad, Chris Ullman, Jon Blank e Sea Cazzell
  Editora: Makron Books

- **Curso de Aplicacoes Web em PHP**
  Autor: Mauricio Vivas (mauricio@cipsga.org.br)

- Colaboradores: Carlos Froldi e Marcelo G. Malheiros

Onde obter ajuda

Para ajudá-lo a solucionar dúvidas de informática, utilize o sistema Rau-Tu de perguntas e respostas, que foi desenvolvido pelo Centro de Computação da Unicamp em conjunto com o Instituto Vale do Futuro. Tem por objetivo possibilitar que um time de colaboradores possa responder a perguntas colocadas por qualquer pessoa no site, cobrindo diversas áreas de conhecimento.
Acesse: **www.rau-tu.unicamp.br**