

# PHP

Desenvolvida exclusivamente para o Apostilando.com  
por Marcos Paulo Furlan

## ÍNDICE

<b>ÍNDICE .....</b>	<b>2</b>
<b>INSTALANDO O WAMP SERVER.....</b>	<b>3</b>
<b>PHP .....</b>	<b>4</b>
Sintaxe Básica .....	6
Variáveis .....	7
Tipos de dados .....	8
Constantes.....	9
Operadores.....	10
Operadores lógicos .....	14
Estruturas de Controle.....	15
switch / case.....	16
For.....	17
While .....	17
do-while.....	18
Matrizes .....	18
<b>INCLUSÃO DE ARQUIVOS .....</b>	<b>19</b>
Parâmetros .....	21
<b>FORMULÁRIOS.....</b>	<b>22</b>
<b>ENVIO DE E-MAILS .....</b>	<b>23</b>
<b>BANCOS DE DADOS.....</b>	<b>25</b>
Introdução ao MySQL.....	26
Exibição.....	32
<b>ENTENDENDO UM POUCO SOBRE SQL .....</b>	<b>33</b>
<b>CONECTANDO AO BANCO DE DADOS PELO DREAMWEAVER .....</b>	<b>41</b>
<b>SISTEMA DE CADASTRO SIMPLES .....</b>	<b>49</b>
Criando o banco de dados no phpMyAdmin .....	54
Criando a conexão do Banco de Dados com o Dreamweaver .....	57
Criando o formulário de cadastro no Banco de Dados.....	59
Edição dos dados cadastrados.....	65
Exclusão de Registro .....	69

## INSTALANDO O WAMP SERVER.

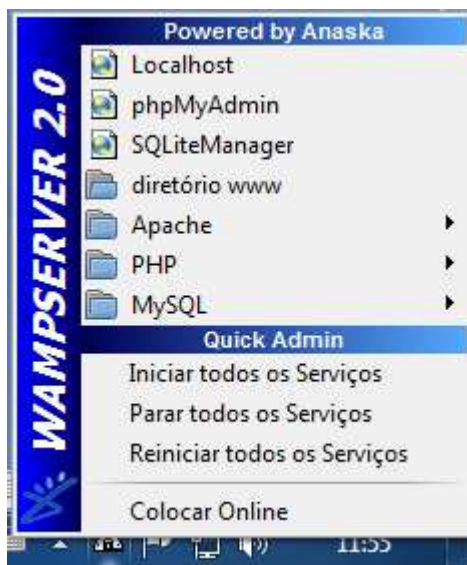
Para podermos testar nossos códigos em PHP é necessária a utilização de um servidor com suporte a PHP. Caso ainda não possua um serviço de hospedagem, ou caso seu serviço de hospedagem não de suporte à linguagem, você pode testar suas páginas em PHP em um servidor local.

Vamos para dar andamento ao conteúdo da apostila usar o WAM Server, que é um pacote que já instala para você o servidor internet local com suporte a PHP e MySQL.

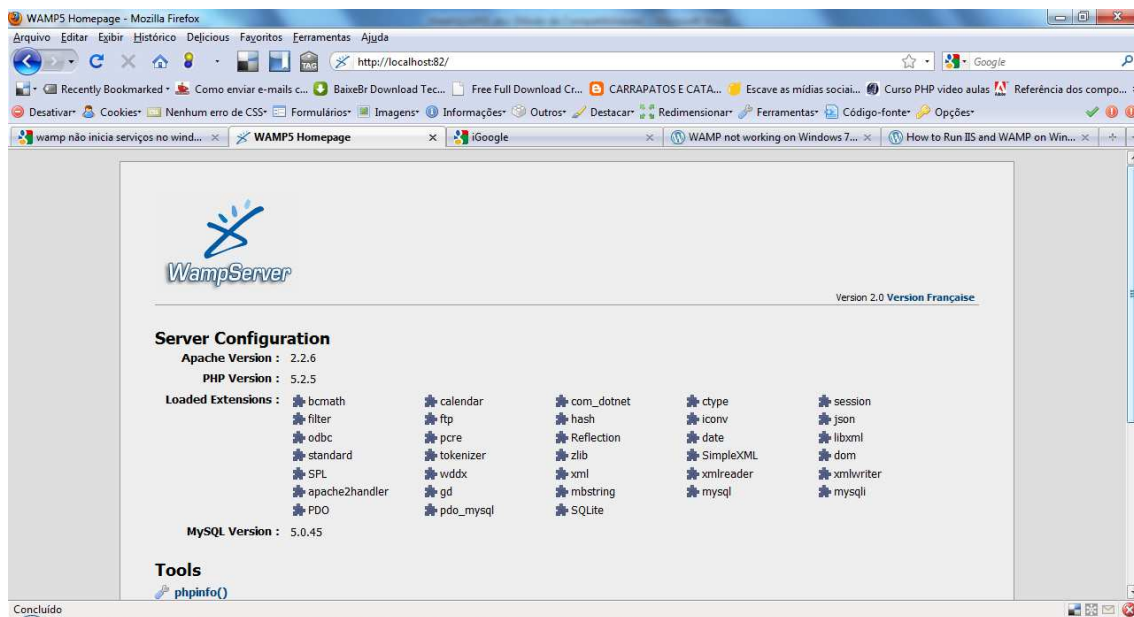
Basta ir clicando e avançando. Após a instalação será mostrado próximo ao relógio o ícone de seu servidor WEB.



Ao clicar sobre este ícone você terá acesso ao menu de opções de seu servidor local.



No seu navegador digite <http://localhost/>.



Vamos a um exemplo simples de teste.

Abra o Dreamweaver ou mesmo o bloco de notas e digite o código

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Untitled Document</title>
</head>
<?php echo "teste de texto"
?>
<body>
</body>
</html>
```

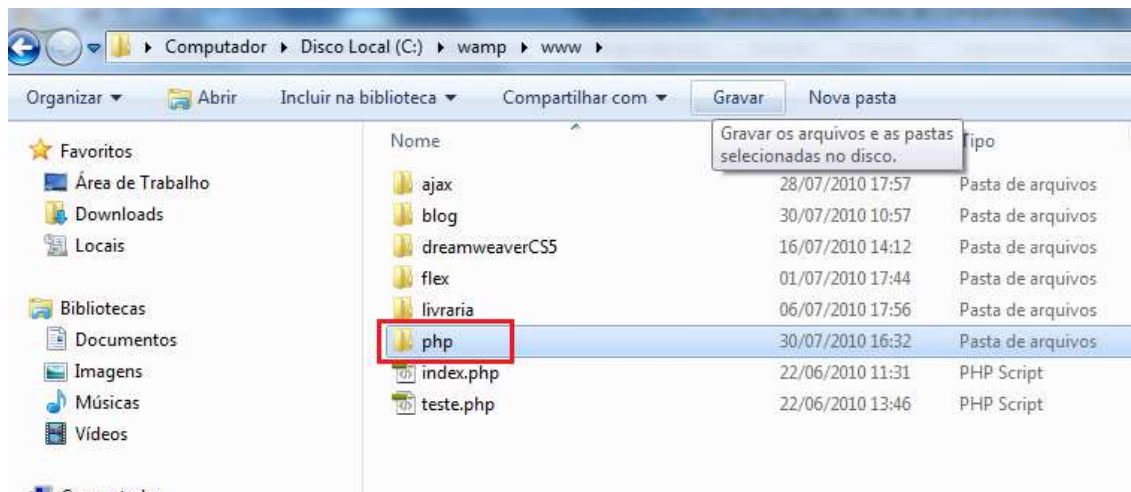
Salve como **teste.php** e salve na pasta: **C:\wamp\www**, depois digite em seu navegador: <http://localhost/teste.php>.

## PHP

PHP é uma sigla recursiva que significa PHP **HyperText Preprocessor**. O PHP é uma linguagem de código-fonte aberto, muito utilizada na Internet e

especialmente criada para o desenvolvimento de aplicativos Web.

Crie uma pasta chamada PHP dentro do diretório WWW do wamp, será nesta pasta que colocaremos nossos scripts., podendo assim testá-los.



O PHP é focado para ser uma linguagem de script do lado do servidor, portanto, você pode fazer qualquer coisa que outro programa CGI pode fazer, como: coletar dados de formulários, gerar páginas com conteúdo dinâmico ou enviar e receber cookies.

Esses são os maiores campos onde os scripts PHP podem se utilizados:

- Script no lado do servidor (server-side). Este é o mais tradicional e principal campo de atuação do PHP. Você precisa de três coisas para seu trabalho. O interpretador do PHP (como CGI ou módulo), um servidor web e um browser. Basta rodar o servidor web conectado a um PHP instalado. Você pode acessar os resultados de seu programa PHP com um browser, visualizando a página PHP através do servidor web.
- Script de linha de comando. Você pode fazer um script PHP funcionar sem um servidor web ou browser. A única coisa necessária é o interpretador. Esse tipo de uso é ideal para script executados usando o cron ou o Agendador de Tarefas (no Windows). Esses scripts podem ser usados também para rotinas de processamento de texto.
- Escrevendo aplicações GUI no lado do cliente (client-side). O

PHP não é (provavelmente) a melhor linguagem para produção de aplicações com interfaces em janelas, mas o PHP faz isso muito bem, e se você deseja usar alguns recursos avançados do PHP em aplicações no lado do cliente poderá utilizar o PHP-GTK para escrever esses programas. E programas escritos desta forma ainda serão independentes de plataforma. O PHP-GTK é uma extensão do PHP, não disponível na distribuição oficial. Se você está interessado no PHP-GTK, visite o site <http://gtk.php.net>.

### ***Sintaxe Básica***

Tags especiais indicam ao PHP onde estão os blocos de código. A tag de abertura é formada por um sinal de “menor que” (<), um sinal de interrogação (?) e a sigla php. A tag de fechamento é formada por um ponto interrogação (?) e sinal de “maior que” (>).

Ex:

```
<?php
```

```
...
```

```
?>
```

Monte o exemplo abaixo:

```
<body>
```

```
<?php
```

```
$a = 10;
```

```
$b = 15;
```

```
$c = $a + $b;
```

```
echo "$a mais $b é igual a $c";
```

```
?>
```

```
</body>
```

Explicando, criamos as variáveis “a” e “b” e atribuímos a ela um valor, depois criamos a variável “c” e definimos a ela a soma de a e b, depois escrevemos isso em tela com o comando **echo**.

Podemos adicionar isso ao próprio código como comentários de bloco **/\* e \*/**. de apenas uma linha são obtidos através de **//**.

```

10 <?php
11 /*criamos aqui a as variaveis
12 a e b e atribuimos um valor a elas*/
13 $a = 10;
14 $b = 15;
15 //aqui criamos a variável c
16 $c = $a + $b;
17 //escrevemos em tela o valor das avriáveis
18 echo "a mais b é igual a c";
19 ?>
20 </body>
21 </html>

```

## Variáveis

Variáveis armazenam valores. Pode-se referir a variáveis para obter seu valor ou para alterar seu conteúdo.

No PHP elas são representadas por um cifrão (\$) mais o nome da variável. Os nomes de variáveis válidos são iniciados por letras ou por um subscrito ( \_ ). Existe diferenciação entre nomes de variáveis maiúsculas e minúsculas.

Quando a variável é declarada dentro de uma função, ela só estará disponível para o código desta função. O código a seguir gera um erro devido a essa característica.

```

<?php
function soma($a){
    $b = $a + 5;
}
soma(10);
echo "o valor de 'b' é $b";
?>

```

Para evitar este tipo de problema, deve-se definir a variável como global. O código a seguir resolve o problema do código anterior. Compare os resultados dos dois scripts.

```

<?php
function soma($a)
{
    global $b;
    $b = $a + 5;
}

```

```
}  
soma(10);  
echo "o valor de 'b' é $b";  
?>
```

## ***Tipos de dados***

O PHP suporta vários tipos de dados:

- Inteiro – Números inteiros (isto é, números sem ponto decimal)
- Números de dupla precisão – Números reais (isto é, números que contêm um ponto decimal)
- String – Texto entre aspas simples ( ' ') ou duplas ( " ")
- Booleanos – armazenam valores verdadeiros ou falsos, usados em testes de condições
- Array – Grupo de elementos do mesmo tipo
- Objeto – Grupo de atributos e métodos
- Recurso – Uma origem de dados externa
- Nulo – Nenhum valor

```
<?php  
//dados booleanos  
$a = True;  
if ($a)  
{  
echo "Verdadeiro";  
}  
else  
{  
echo "Falso";  
}  
echo '<p>';  
// dados com hexadecimais
```



```

$a = 0x1A; //Corresponde ao decimal 26
$b = -16;
$c = $a + $b;
echo "a + b = $c";
echo '<p>';
//ponto flutuante são representados por (.)
$preco = 11.90;
$soma = $preco * 4;
echo "Quatro revistas W custam R$ $soma<br>";
echo '<p>';
//strings são representadas por aspas simples e duplas
$texto1 = 'Esse é o primeiro texto.<br>';
$texto2 = "Esse é o segundo texto.";
echo $texto1;
echo $texto2;
echo '<p>';
/*As variáveis do tipo matriz ou array permitem o armazenamento de
diversos elementos
referenciados por uma mesma referência*/
$frutas = array(
1 => "Laranja",
2 => "Maçã",
3 => "Uva");
echo "<li> $frutas[1]<br>";
echo "<li> $frutas[2]<br>";
echo "<li> $frutas[3]<br>";
?>

```

### **Constantes**

Constantes são identificadores para valores simples. O seu conteúdo não muda durante a execução do código. Elas são criadas com a função `define` e, por convenção, são escritas com letras maiúsculas e não usam o cifrão no início.

```
<?php
define("CONSTANTE", "Alô mundo.");
echo CONSTANTE;
?>
```

## **Operadores**

São usados para efetuarem operações sobre as variáveis e constantes. Os operadores do PHP são:

- + soma
- - subtração
- multiplicação
- / divisão
- ^ exponenciação
- % módulo, resto da divisão
- ++ acrescenta um a uma variável
- -- subtrai um de uma variável
- += soma um valor a uma variável e lhe atribui o resultado

```
<?php
$x = 2;
echo($x + 2);
echo "<br>";
$x = 2;
echo(5 - $x);
echo "<br>";
$x = 4;
echo($x * 5);
echo "<br>";
$x = 15;
echo($x / 5);
echo "<br>";
$x = 10;
echo($x % 8);
```

```
echo "<br>";  
$x = 5;  
$x++;  
echo($x);  
echo "<br>";  
$x = 5;  
$x--;  
echo($x);  
echo "<br>";  
$x = 8;  
echo($x);  
echo "<br>";  
$x = 8;  
$x = $x + 10;  
echo($x);  
echo "<br>";  
$x = 8;  
$x += 10;  
echo($x);  
?>
```

Há também os operadores de comparação. Uma comparação sempre gera um dos dois valores possíveis: vazio, que corresponde a falso, e 1, que corresponde a verdadeiro.

- == é igual a
- != não é igual a
- > é maior que
- < é menor que
- >= é maior ou igual a
- <= é menor ou igual a

```
<?php
```

```
$x = 5;  
$resultado = ($x == 8);
```

```
if($resultado == 1)
{
echo "verdadeiro";
}
else
{
echo "falso";
}
echo "<br>";
$x = 5;
$resultado = ($x != 8);
if($resultado == 1)
{
echo "verdadeiro";
}
else
{
echo "falso";
}
echo "<br>";
$x = 5;
$resultado = ($x > 8);
if($resultado == 1)
{
echo "verdadeiro";
}
else
{
echo "falso";
}
echo "<br>";
$x = 5;
```

```
$resultado = ($x > 8);  
if($resultado == 1)  
{  
    echo "verdadeiro";  
}  
else  
{  
    echo "falso";  
}  
echo "<br>";  
$x = 5;  
$resultado = ($x >= 8);  
if($resultado == 1)  
{  
    echo "verdadeiro";  
}  
else  
{  
    echo "falso";  
}  
echo "<br>";  
$x = 5;  
$resultado = ($x <= 8);  
if($resultado == 1)  
{  
    echo "verdadeiro";  
}  
else  
{  
    echo "falso";  
}  
?>
```

## Operadores lógicos

**and ou &&** - operador lógico “e”, apenas retornando verdadeiro quando as duas condições envolvidas no teste forem verdadeiras **or ou ||** operador lógico “ou”, retornando verdadeiro quando uma ou as duas condições envolvidas no teste forem verdadeiras **!** operador lógico “não”, invertendo o resultado de um teste **xor** – operador lógico “ou exclusivo” que determina se uma de duas condições é verdadeira mas não ambas. Se ambas forem verdadeiras, o teste final será falso.

```
<?php
$x = 6;
$y = 3;
$resultado = ($x < 10 && $y > 1);
if($resultado == 1)
{
    echo "verdadeiro";
}
else
{
    echo "falso";
}
echo "<br>";
$x = 6;
$y = 3;
$resultado = ($x == 5 || $y == 5);
if($resultado == 1)
{
    echo "verdadeiro";
}
else
{
    echo "falso";
}
```

```

echo "<br>";
$x = 6;
$y = 3;
$resultado = (!($x == $y));
if($resultado == 1)
{
echo "verdadeiro";
}
else
{
echo "falso";
}
?>

```

### ***Estruturas de Controle***

No PHP, as estruturas de controle são formadas por declarações condicionais e de looping:

**if** – executa uma ação se uma condição for atendida. O bloco de comandos a ser executado deve ser escrito entre chaves;

**else** – pode-se colocar um conjunto de comandos alternativos caso o teste do if seja negativo. A declaração else deve vir logo após o bloco de código relacionado ao. O comando if também pode ser usado após a declaração else

```

<body>
<?php
$x = 20;
if ($x > 10)
{
echo("O valor da variável é maior que 10.");
}
?>
</body>

```

Outro exemplo usando If e else

```

<body>

```

```

<?php
$cor = "branco";
if ($cor == "vermelho")
{
echo("A variável contém o valor 'vermelho'.");
}
else if ($cor == "azul")
{
echo("A variável contém o valor 'azul'.");
}
else if ($cor == "amarelo")
{
echo("A variável contém o valor 'amarelo'.");
}
else
{
echo("O valor da variável não foi identificado.");
}
?>
</body>

```

### switch / case

Forma de testar uma dentre várias possibilidades. A declaração **default** executa caso nenhuma das opções for verdadeira. A declaração **break** faz com que o restante do código não seja executado caso o teste seja verdadeiro.

```

<body>
<?php
$d = getdate();
switch ($d['wday'])
{
case 5:
echo("Sexta-feira");
break;

```



```
case 6:
echo("Sábado");
break;
case 0:
echo("Domingo");
break;
default:
echo("Outro dia da semana");
}
?>
</body>
```

## For

Estrutura de looping que executa um bloco de código quantas vezes for indicado em uma variável. Deve-se definir a variável que será testada no looping, uma condição de teste e o incremento (ou decremento) da variável de controle.

```
<body>
<?php
for ($i = 1; $i < 10; $i++)
{
echo("Linha $i <br>");
}
?>
</body>
```

## While

Estrutura de looping que não necessita de um número determinado de iterações. Ele é executado enquanto uma condição for verdadeira.

```
<body>
<?php
$i = 1;
```

```

while ($i < 10000)
{
echo($i);
$i *= 2;
echo(" vezes 2 é igual a $i <br>");
}
?>
</body>

```

### **do-while**

Outra forma de looping que executa um bloco de código, testa uma condição e repete novamente o bloco de código (ou não).

```

<?php
$i = 1;
do
{
echo ("Linha $i <br>");
$i++;
}
while ($i < 10)
?>

```

### **Matrizes**

Matrizes são variáveis que armazenam mais de um valor simultaneamente. Uma matriz no PHP é atualmente um mapa ordenado. Um mapa é um tipo que relaciona valores para chaves. Este tipo é otimizado de várias maneiras, então você pode usá-lo como um array real, ou uma lista (vetor), hashtable (que é uma implementação de mapa), dicionário, coleção, pilha, fila, etc.

As referências aos elementos da matriz podem ser declaradas como valores numéricos ou strings

```

<body>
<?php
$colaboradores = array(0 => "Marcos",

```

```

1 => "Eduardo",
2 => "Maria",
3 => "Sérgio",
4 => "Rosangela");
echo "<b>Colaboradores</b>";
echo "<ul>";
echo "<li>" . $colaboradores[0];
echo "<li>" . $colaboradores[1];
echo "<li>" . $colaboradores[3];
echo "<li>" . $colaboradores[2];
echo "<li>" . $colaboradores[4];
echo "</ul>";
?>
</body>

```

## INCLUSÃO DE ARQUIVOS

O comando **include** permite a inclusão de outros arquivos php dentro do script que está sendo executado. Pode-se criar uma função que imprime a data atual e pode-se reusá-lo sem precisar reescrever o código cada vez que for necessário. No exemplo a seguir, pode-se chamar o primeiro script de **topo.php** e o próximo script o inclui através do comando include.

Código do arquivo **topo.php**.

```

<body>
<?php
$meses = array(1 => "Janeiro",
2 => "Fevereiro",
3 => "Março",
4 => "Abril",
5 => "Maio",
6 => "Junho",
7 => "Julho",
8 => "Agosto",
9 => "Setembro",

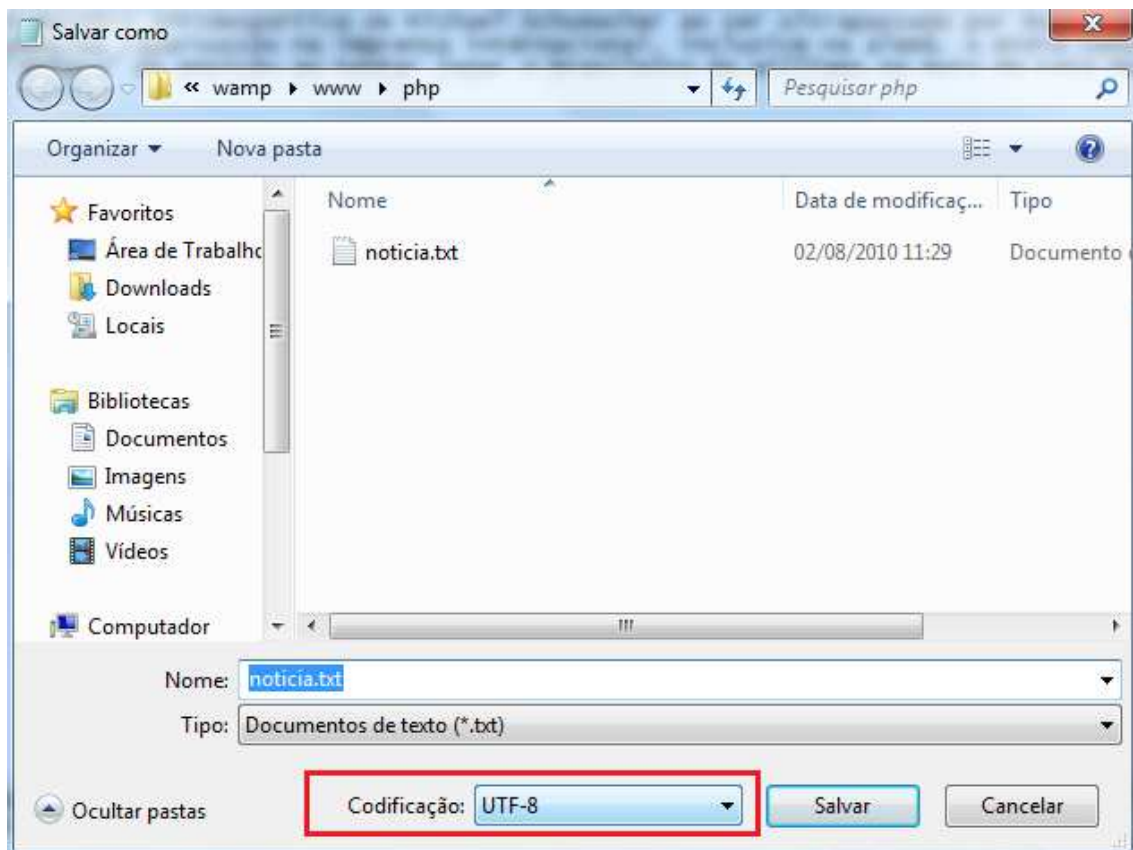
```

```
10 => "Outubro",  
11 => "Novembro",  
12 => "Dezembro");  
$hoje = getdate();  
$dia = $hoje["mday"];  
$mes = $hoje["mon"];  
$nomeMes = $meses[$mes];  
$ano = $hoje["year"];  
echo "Hoje é dia $dia de $nomeMes de $ano."  
?>  
</body>
```

Para chamar o arquivo **topo.php**, usamos o seguinte código em um novo arquivo php.

```
<body>  
<?php  
include("topo.php");  
?>  
</body>
```

Este recurso pode ser bem interessante, para facilitar a atualização de setores do seu site, pois podemos importar arquivos no formato texto como, por exemplo, arquivos em **TXT** , apenas é necessário salvá-lo no formato UTF-8, para que o navegador consiga interpretar corretamente a acentuação.



Na chamada do Include coloque:

```
<?php  
include("noticia.txt");  
?>
```

### **Parâmetros**

O uso de parâmetros facilita a programação porque permite a passagem de dados entre o browser e o script ou entre scripts. A passagem de parâmetros entre o browser e o script é feita dentro da URL, por exemplo, e é manipulada pela função **\$\_GET**.

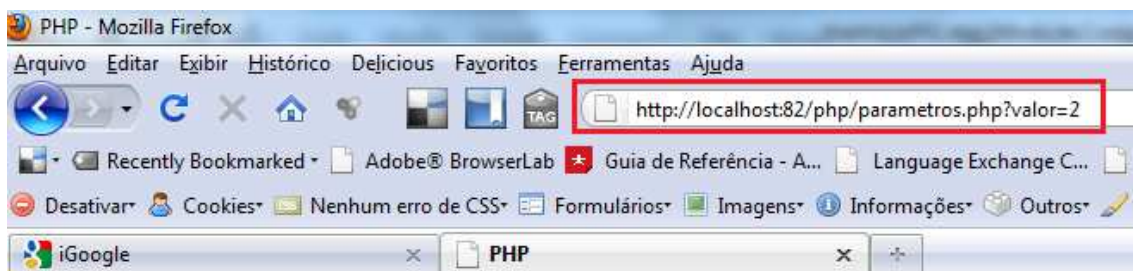
Nesse exemplo a seguir, cada um dos links envia um valor diferente para a página que é aberta **parametros.php**. Para enviar um parâmetro, a sintaxe inclui um sinal de interrogação, o nome da variável, um sinal de igual e o valor da variável.

```
<body>  
<?php  
if (isset($_GET["valor"]))
```

```

{
$valor = $_GET["valor"];
echo "Você clicou no valor $valor <p>";
}
else
{
echo "Clique em um dos valores:<p>";
}
?>
<a href="parametros.php?valor=1">Valor01</a><br>
<a href="parametros.php?valor=2">Valor02</a><br>
<a href="parametros.php?valor=3">Valor03</a><br>
<a href="parametros.php?valor=4">Valor04</a><br>
<a href="parametros.php?valor=5">Valor05</a><br>
</body>

```



Você clicou no valor 2

[Valor01](#)  
[Valor02](#)  
[Valor03](#)  
[Valor04](#)  
[Valor05](#)

## FORMULÁRIOS

Os valores enviados através de um formulário podem ser recuperados pela variável predefinida **\$\_POST**. Através dela é possível obter os dados que foram enviados através do método POST do HTML, bastando indicar o nome do campo do formulário. No comando action do formulário, deve-se indicar a página PHP que irá receber os valores. O mesmo documento pode conter o

código e o formulário

```
<body>
<?php
if (isset($_POST["pnome"]) && isset($_POST["snome"]))
{
    $pnome = $_POST["pnome"];
    $snome = $_POST["snome"];
    echo "Olá $pnome $snome.<p>";
}
else
{
    echo "Digite o seu nome.<p>";
}
?>
<form method="post" action="formulario.php">
Nome: <input type="text" name="pnome">
<br>
Sobrenome: <input type="text" name="snome">
<br><br>
<input type="submit" value="Enviar">
</form>
</body>
```

## ENVIO DE E-MAILS

O PHP permite que se enviem e-mails de forma automatizada. Para isso, deve-se ajustar o arquivo de configuração (php.ini) para se indicar o servidor SMTP:

*[mail function]*

*SMTP = localhost ;for win32 only*

*sendmail\_from = me@localhost.com ;for win32 only*

*;sendmail\_path = ;for unix only, may supply*

*arguments as well (default is 'sendmail -t -i')*

A opção SMTP indica o endereço ou número IP do servidor SMTP. A opção **sendmail\_from** indica o endereço do remetente da mensagem. É necessário

reiniciar o servidor a cada modificação em algum arquivo de configuração.



É importante se atentar como o seu servidor de hospedagem trata o envio de mensagens por e-mail, atualmente é comum os servidores atribuírem uma regra para estes envios, focando assim na segurança de seus servidores e procurando também evitar a pratica de SPAM.

Abaixo um exemplo básico de envio de e-mail via formulário:

**O formulário de contato:**

```
<tr>
<td><p>Nome</p></td>
<td><input name="nome" type="text" size="30" /></td>
</tr>
<tr>
<td><p>E-mail</p></td>
<td><input name="email" type="text" /></td>
</tr>
<tr>
<td><p>Assunto</p></td>
<td><input name="assunto" type="text" /></td>
</tr>
<tr>
<td><p>Coment&acutes;rios</p></td>
<td><textarea name="mensagem" cols="29" rows="5"></textarea></td>
</tr>
<tr>
<td><input name="enviar" type="submit" value="Enviar" />
<input name="limpar" type="reset" value="Limpar" /></td>
</tr>
```

O arquivo PHP que processa a informação e faz o envio:

```
<? php
```

```
//Recebendo os dados do formulário.
```



```

$nome = $_POST["nome"];
$email = $_POST["email"];
$assunto = $_POST["assunto"];
$mensagem=$_POST["mensagem"];
//Setando o restante das variáveis para o disparo do email
$destinatario = "seuemail@provedor.com";
$formato = "\nContent-type: text/html\n";
//Incluindo os campos nome e email no corpo da mensagem.
$msg = "- Nome: ".$nome."<br>- Assunto: ".$assunto."<br>- Mensagem:
".$mensagem;
//Enviando o email
mail("$destinatario","$assunto","$msg","from: ".$email.$formato);
//Criando a mensagem de confirmação de envio de email.
echo "<div align=center>Mensagem enviada com sucesso! Aguarde um
retorno. Clique <a href='index.php'> <b>aqui</b> </a> para
retornar.</div>";
?>

```

## BANCOS DE DADOS

O ponto de partida para a criação de um banco de dados é o registro. Um registro é uma coleção de dados relacionados tratada como uma entidade única. Por exemplo: a ficha de funcionário pode ser chamada de um registro: ela contém o nome, fotografia, função, salário, etc. Empregando a terminologia de banco de dados, cada um desses itens relacionados seria chamado de campo: cada "registro" de ficha do funcionário contém um campo de nome, um campo de fotografia, um campo de função etc.

Uma coleção de registros que compartilham os mesmos campos é chamada de tabela, porque esses tipos de informações podem ser facilmente apresentados no formato de tabela: cada coluna representa um campo e cada linha representa um registro. Na verdade, a palavra coluna é sinônima de campo e linha é sinônima de registro.

Um banco de dados pode conter mais de uma tabela, cada uma delas com um nome exclusivo. Essas tabelas podem ser relacionadas ou independentes.

Um conjunto de registros é um subconjunto de registros extraídos de uma ou mais tabelas de um banco de dados. Um conjunto de registros é também uma tabela, por ser uma coleção de registros que compartilham as mesmas colunas.

Para criar um conjunto de registros, é necessário fazer uma consulta ao banco de dados. Um consulta consiste em um critério de pesquisa.

### ***Introdução ao MySQL***

O **MySQL** é o gerenciador de banco de dados mais usado com o PHP. Existem muitas funções pré-definidas para manipulação de conexões com bancos de dados.

A função **mysql\_connect** tenta uma conexão com um servidor MySQL. Deve-se passar como parâmetros: o nome do servidor (ou número IP) onde o MySQL está sendo executado, o nome de usuário e a senha deste usuário. O comando alternativo `die` trata um possível fracasso na conexão.

A função **mysql\_selectdb** seleciona qual base será selecionada dentro do banco de dados que foi conectado. O comando alternativo `die` trata um possível fracasso na seleção da base, podendo ser incluída uma mensagem customizada.

A função **mysql\_query** faz consultas à base previamente selecionada. Deve-se passar, como parâmetros, os comandos SQL apropriados. Novamente, o comando alternativo `die` pode tratar um não sucesso na consulta.

**<body>**

**<?php**

```
$link = mysql_connect("localhost", "root", "");  
or die("Não foi possível conectar");  
mysql_select_db("curso")  
or die("Não foi possível selecionar o banco de dados");  
$consulta = "SELECT * FROM Modulos";  
$resultado = mysql_query($consulta)  
or die("Falha na execução da consulta");  
echo "Consulta executada com sucesso";  
?>
```

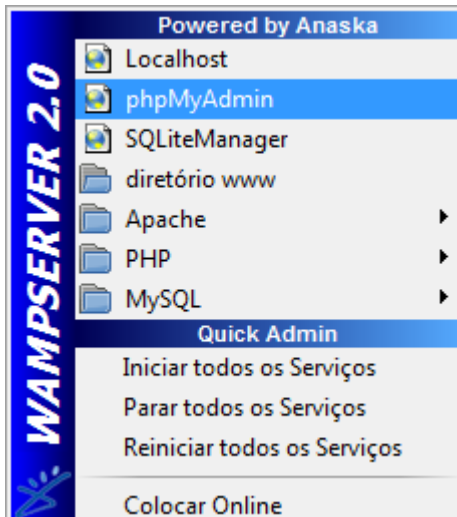
</body>



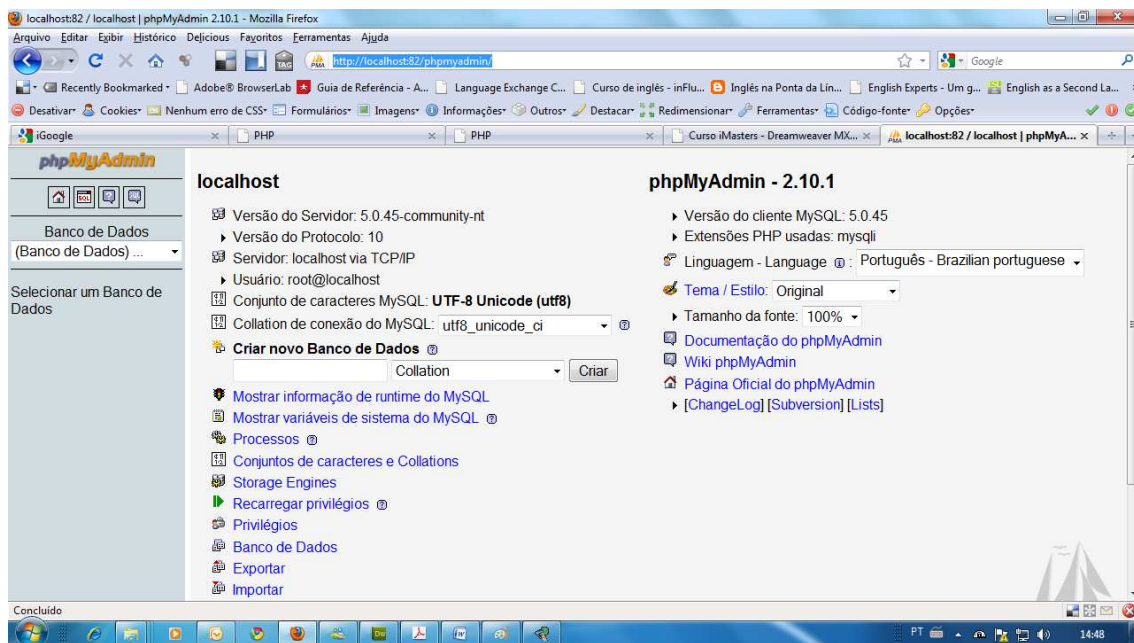
No exemplo como estamos usando o WAMP, o usuário é root e a senha vazia, ao enviar para um servidor, é necessário preencher com os dados corretos.

Embora seja possível criar os Bancos de Dados, tabelas, etc. diretamente pelo código, este processo se torna mais fácil de forma visual, utilizando, por exemplo, o PHPMyAdmin.

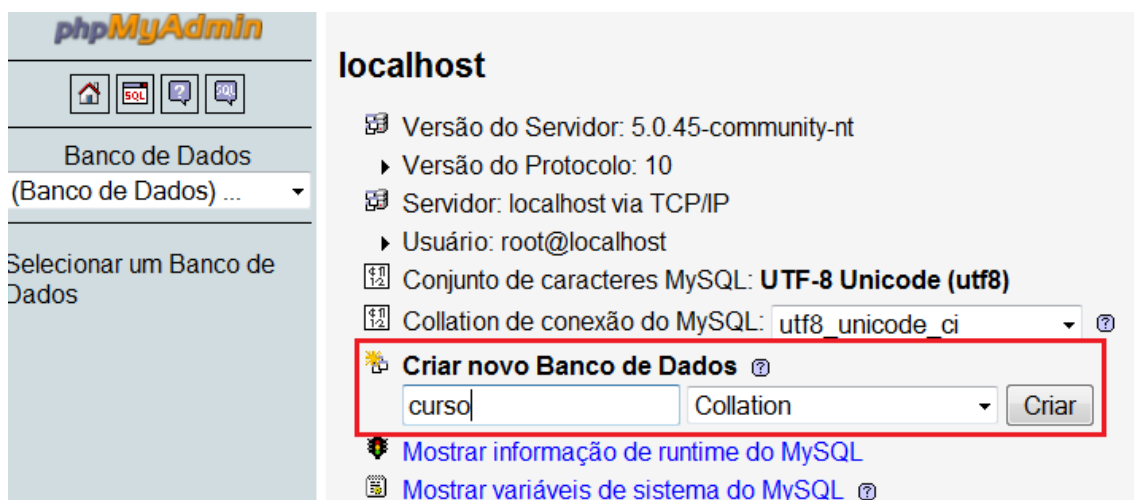
Pelo gerenciador do WAMP, selecione phpMyAdmin.



Caso ele não abra a página correta mude para: <http://localhost/phpmyadmin/>, caso você tenha precisado mudar a porta de acesso.



Vamos criar o banco de dados e a tabela que tentamos conectar no exemplo acima.



Clique em Criar

Será criado o Banco porém sem nenhuma tabela.

Na janela que você havia feito para se logar ao banco de dados, pressione F5 e veja que agora ele mostra a mensagem de falha na consulta, pois temos o Banco de dados.



Falha na execução da consulta

A barra de navegação do phpMyAdmin oferece diversos recursos para você administrar seu banco de dados e/ou tabela.

Explicando-a:

- Estrutura: Exibe a relação das tabelas contidas no banco de dados, uma breve estatísticas de cada uma e diversas opções para administrá-las.
- SQL: Exibe um painel para você executar instruções SQL no seu banco de dados e/ou tabela.
- Procurar: Um painel para você realizar uma busca e encontrar determinado registro em sua tabela.
- Exportar: Exportar os dados do seu banco de dados e/ou tabela em diversos formato como: SQL, Latex, Microsoft Excel 2000, Microsoft Word 2000, CSV for MS Excel, CSV e XML. Você pode exportar a estrutura somente e/ou os dados.
- Importar: Inserir dados na tabela.
- Operações: Permite-lhe administrar sua tabela com diversas opções como: renomear, mover ou copiar para outro banco de dados, inserir comentários, modificar o tipo da tabela, reparar, otimizar e diversos outros recursos.
- Eliminar: Deleta o banco de dados e/ou tabela.

Como é apenas um exemplo, vamos determinar que nosso banco de dados curso tenha a tabela Modulos e dentro desta tabela os campos ID, NOME, DESCRICAO E CATEGORIA.

No phpMyAdmin, defina o nome da tabela e quantidade de campos.

Banco de Dados

curso (0)

curso (0)

Nenhuma tabela encontrada no Banco de Dados

Banco de Dados curso foi criado.

consulta SQL:

CREATE DATABASE `curso` ;

Nenhuma tabela encontrada no Banco de Dados

Criar nova tabela no Banco de Dados curso

Nome: modulos

Número de arquivos: 4

Preencha os campos de cabeçalho de sua tabela.

Servidor: localhost > Banco de Dados: curso > Tabela: modulos

Campo	Tipo	Tamanho/Definir <sup>1</sup>	Collation	Atributos	Nulo
ID	INT				not null
NOME	TEXT				not null
DESCRICAO	TEXT				not null
CATEGORIA	TEXT				not null

Comentários da tabela: Storage Engine: InnoDB Collation:

Para o campo ID, vamos deixá-lo como Auto Incremento e com atributo de chave primária.

n	Atributos	Nulo	Padrão <sup>2</sup>	Extra		
1		not null		auto_increment		
2		not null				
3		not null				
4		not null				





Dê um refresh em sua página de conexão e veja que agora ele apresenta sucesso.



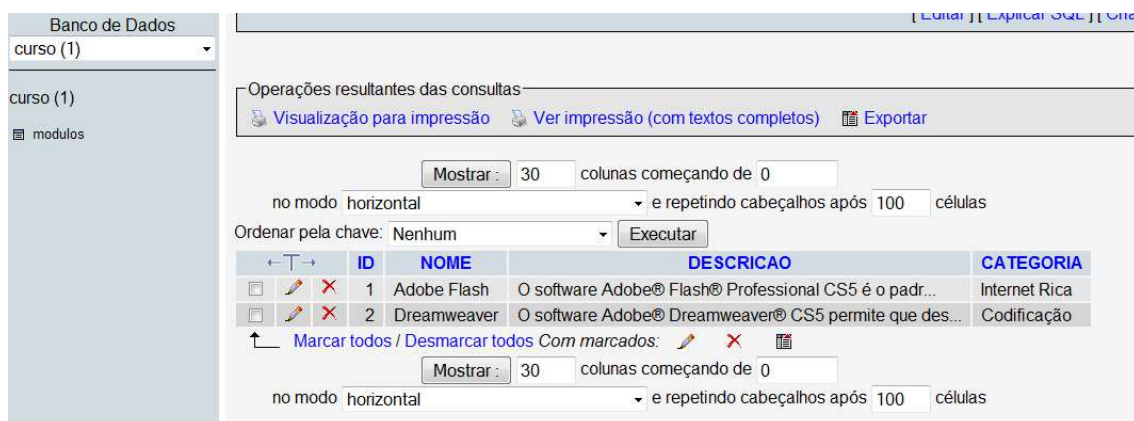
Consulta executada com sucesso

Na página do PHP MyAdmin, adicione algumas informações, para isso clique no botão Inserir.

Campo	Tipo	Funções	Nulo	Valor
ID	int(11)	<input type="text"/>	<input type="checkbox"/>	<input type="text"/>
NOME	text	<input type="text"/>	<input type="checkbox"/>	Adobe Flash
DESCRICAO	text	<input type="text"/>	<input type="checkbox"/>	O software Adobe® Flash® Professional CS5 é o padrão de mercado para autoria interativa e fornecimento de experiências imersivas que se apresentam de modo consistente entre computadores pessoais, dispositivos móveis e telas de praticamente qualquer tamanho e resolução.
CATEGORIA	text	<input type="text"/>	<input type="checkbox"/>	Internet Rica

No final da página clique em Executar.

Clique na barra no botão visualizar para ver os registros dentro do Banco.



## Exibição

Para que os registros da consulta sejam exibidos, deve-se usar a função **mysql\_fetch\_assoc**, que retorna uma matriz com a linha atual e move para a próxima. Para se imprimir todos os resultados de uma query, é necessária a construção de uma estrutura de repetição (while) até que a função **mysql\_fetch\_assoc** não retorne nenhum valor (vazio).

Para melhorar a apresentação dos resultados, é possível usar tags HTML que incluam os dados dentro de tabelas, por exemplo.

**<body>**

**<?php**

```
$link = mysql_connect("localhost", "root", "");
```

```
or die("Não foi possível conectar");
```

```
mysql_select_db("curso")
```

```
or die("Não foi possível selecionar o banco de dados");
```

```
$consulta = "SELECT * FROM modulos";
```

```
$resultado = mysql_query($consulta)
```

```
or die("Falha na execução da consulta");
```

```
$linha = mysql_fetch_assoc($resultado);
```

```
$Nome = $linha["NOME"];
```

```
$Categoria = $linha["CATEGORIA"];
```

```
echo "<b>Nome do Módulo:</b> $Nome<br>";
```



```
echo "<b>Categoria:</b> $Categoria";  
?>  
</body>
```



**Nome do Módulo:** Adobe Flash  
**Categoria:** Internet Rica

## ENTENDENDO UM POUCO SOBRE SQL

Para um melhor aproveitamento do curso, vamos entender um pouco sobre o SQL

A instrução SQL mais comumente utilizada para criar um conjunto de registros é **SELECT**, que extrai determinadas colunas de uma ou mais tabelas de bancos de dados para criar um conjunto de registros. A sintaxe básica da instrução SELECT é apresentada abaixo:

*SELECT NomeDaColuna FROM NomeDaTabela*

É possível adicionar quebras de linha, guias e outros tipos de espaço em branco às instruções para tornar clara a lógica: O SQL ignora todos os tipos de espaço em branco. Por exemplo: a seguinte instrução é válida:

```
SELECT nome  
FROM modulos
```

Abaixo tabela com as palavras-chave mais utilizados pelos comandos SQL

SELECT	Recupera os registros especificados no banco de dados.
INSERT	Inclui um novo registro em uma tabela de banco de dados.
UPDATE	Altera os valores nos registros de banco de dados especificados.
DELETE	Remove os registros de banco de dados especificados.

As seguintes palavras-chave são utilizadas para refinar as instruções SQL:

FROM	Denomina a fonte de dados de uma operação.
------	--------------------------------------------

WHERE Define uma ou mais condições para a operação.

ORDER BY Classifica as linhas de conjuntos de registros em uma determinada ordem.

GROUP BY Agrupa o conjunto de registros pelos itens especificados, selecionados em uma lista.

Os seguintes operadores especificam condições e executam funções numéricas e lógicas:

Operador Significado

= Igual a

LIKE Como (caracteres curingas são aceitos)

<> Diferente de

NOT LIKE Não igual a (curingas são aceitos)

< Menor que

> Maior que

<= Menor ou igual a

>= Maior ou igual a

AND Ambas as condições devem ser atendidas, como Minas Gerais AND Bahia

OR Pelo menos uma das condições deve ser atendida, como Matos OR Mattos

NOT Exclui a condição seguinte, como Paris NOT França

Se o item que estiver sendo comparado for texto, coloque-o entre aspas simples, como no exemplo a seguir:

WHERE pais = 'Brasil'

O SQL pode ser utilizado para definir os conjuntos de registros de uma página.

Um conjunto de registros é um subconjunto extraído de um banco de dados.

Esta é a sintaxe SQL básica para definir as colunas de um conjunto de registros:

*SELECT NomeDaColuna, NomeDaColuna2, NomeDaColuna3 FROM modulos*

Para incluir todas as colunas de uma tabela no conjunto de registros, utilize o caractere curinga (\*) da seguinte forma:

```
SELECT * FROM NomeDaTabela
```

Por exemplo: suponha que você esteja trabalhando com uma tabela denominada funcionários. Para extrair todas as colunas, digite a seguinte instrução SELECT:

```
SELECT * FROM funcionarios
```

Suponha que você necessite apenas dos dados contidos em duas colunas da tabela funcionarios: nas colunas nome e cargo. Para criar um conjunto de registros que contenha apenas os dados destas duas colunas, digite a seguinte instrução SELECT:

```
SELECT nome, cargo  
FROM funcionários
```

Utilize uma cláusula **WHERE** para limitar o número de registros no conjunto de registros. Por exemplo: apenas os funcionarios que ganham mais de R\$ 2mil mensais serão incluídos. Suponha que uma coluna da tabela denominada salário informa os rendimentos de cada funcionário. A instrução SELECT teria a seguinte aparência:

```
SELECT nome, salario FROM funcionarios WHERE salario > 2000
```

É possível filtrar registros em um banco de dados com base na igualdade do valor de um parâmetro em relação ao valor de uma coluna de registros.

Suponha que você decidiu permitir que os usuários pesquisassem o banco de dados por quantidade de cursos de atualização de cada funcionário. Siga a lógica a seguir para criar o conjunto de registros resultante da busca:

1. Verifique um registro na tabela do banco de dados.
2. Se o valor na coluna "quantidade\_cursos" do registro for igual a

quantidade enviada pelo usuário, inclua esse registro no conjunto de registros resultante da busca.

3. Verifique o próximo registro da tabela.

A lógica pode ser expressa com a seguinte cláusula WHERE:

*WHERE NomeDaColuna = ValorDoParametro*

**ValorDoParametro** é uma variável SQL que contém um parâmetro de busca. Em um aplicativo para a Web, o usuário normalmente fornece esse parâmetro utilizando um formulário HTML.

Esta consulta ao banco de dados poderia ser expressa inteiramente na linguagem SQL, da seguinte forma:

*SELECT nome FROM funcionarios WHERE quantidade\_cursos = 'NomeDaVariavel'*

A instrução SQL localiza todos os registros na tabela de funcionarios que contenham a quantidade de cursos igual àquela encontrada na variável especificada. Por exemplo: se o usuário especificar a quantidade de certificados como 4, a instrução SQL deverá gerar o seguinte conjunto de registros:

Marcos	4
Maria	4
Ronaldo	4
Beth	4

É possível filtrar registros em um banco de dados com base na semelhança do valor de um parâmetro em relação ao valor de uma coluna de registros.

A utilização da semelhança em vez da igualdade permite aos usuários uma maior flexibilidade ao especificar o valor desses parâmetros. Por exemplo: as palavras da busca não necessitam distinguir maiúsculas de minúsculas. Se o usuário digitar "curitiba" e a coluna da tabela contiver o valor "Curitiba", a

correspondência será estabelecida.

Além disso, a semelhança possibilita o uso de caracteres curinga para que os usuários possam fazer buscas de partes de palavras e também em ordem alfabética. Por exemplo: se o usuário digitar "m" e a coluna da tabela contiver os valores "Cuiabá", "Curitiba" e "Curitibanos", será possível utilizar um caractere curinga na instrução SQL para que as três correspondências sejam estabelecidas.

O caractere curinga padrão é o sinal de percentagem (%):

*WHERE sobrenome LIKE 'Mc%'*

Suponha que você decidiu permitir que os usuários pesquisem o banco de dados por sobrenome. Siga a lógica a seguir para criar o conjunto de registros resultante da busca:

1. Verifique um registro na tabela do banco de dados.
2. Se o valor na coluna "sobrenome" do registro contiver um valor semelhante ao que o usuário enviou, inclua esse registro no conjunto de registros resultante da busca.
3. Verifique o próximo registro da tabela.

A lógica pode ser expressa com a seguinte cláusula WHERE:

*WHERE NomeDaColuna LIKE ValorDoParametro*

**ValorDoParametro** é uma variável SQL que contém um parâmetro de busca. Em um aplicativo para a Web, o usuário normalmente fornece esse parâmetro utilizando um formulário HTML.

Esta consulta ao banco de dados poderia ser expressa inteiramente na linguagem SQL, da seguinte forma:

*SELECT nome, sobrenome, idade, cursos*

*FROM funcionarios WHERE sobrenome LIKE 'NomeDaVariavel'*

Para que os usuários possam executar buscas utilizando partes de palavras, combine a variável com um caractere curinga. O caractere curinga da SQL utilizado nesse caso é o sinal de percentagem (%). Por exemplo:

*WHERE sobrenome LIKE 'NomeDaVariavel%'*

Por exemplo: se o usuário digitar o parâmetro de busca "s", todos os registros contendo sobrenomes iniciados com a letra "f" serão incluídos no conjunto de registros, conforme mostrado abaixo:

Marcos	Furlan	04
Maria	Furlan	04
Ronaldo	Franca	02

Os registros de um banco de dados podem ser filtrados estabelecendo-se que um valor da coluna de registros pertença a uma faixa limitada por dois valores de parâmetros.

Suponha que você decidiu permitir que os usuários pesquisassem o banco de dados por uma faixa de datas. Siga a lógica a seguir para criar o conjunto de registros resultante da busca:

1. Verifique um registro na tabela do banco de dados.
2. Se o valor na coluna "data" do registro se encontrar entre dois valores de data enviados pelo usuário, inclua esse registro no conjunto de registros resultante da busca.
3. Verifique o próximo registro da tabela.

A lógica pode ser expressa com a seguinte cláusula WHERE:

*WHERE NomeDaColuna BETWEEN ValorDoParametro AND ValorDoParametro2*

**ValorDoParametro e ValorDoParametro2** são variáveis SQL que contêm um parâmetro de busca. Em um aplicativo para a Web, o usuário normalmente

fornece esse parâmetro utilizando um formulário HTML.

Esta é a maneira de expressar em SQL esse tipo de consulta ao banco de dados:

```
SELECT nome, sobrenome, data_nascimento  
FROM funcionarios  
WHERE data_nascimento BETWEEN 'NomeDaVariavel' AND  
'NomeDaVariavel2'
```

Por exemplo: se o usuário digitar os parâmetros "01/05/66" e "30/06/1995", todos os funcionários que nasceram a partir "01/05/66" até 30/06/1995" serão incluídos no conjunto de registros, como mostrado abaixo:

Marcos	Furlan	30/04/73
Beth	Oliveira	12/05/1992

Esta seção descreve como incluir registros no conjunto de registros resultante da busca com base em uma combinação de condições de busca. Combine as condições na linguagem SQL utilizando os operadores lógicos **AND**, **OR** e **NOT**.

Se desejar que todas as condições sejam verdadeiras (true) para um registro incluído no conjunto de registros, utilize o operador **AND** da seguinte forma:

```
WHERE gols LIKE 'NomeDaVariavel' AND time LIKE 'NomeDaVariavel2'
```

Se desejar que qualquer uma das condições seja verdadeira (true) para um registro no conjunto de registros, utilize o operador **OR** da seguinte forma:

```
WHERE gols LIKE 'NomeDaVariavel' OR time LIKE 'NomeDaVariavel2'
```

Se desejar que uma condição seja verdadeira (true), mas não outra, utilize o operador **NOT** da seguinte forma:

```
WHERE gols LIKE 'NomeDaVariavel' AND NOT time LIKE 'NomeDaVariavel2'
```

Utilize os parênteses para agrupar as condições de busca:

```
WHERE (nome LIKE 'NomeDaVariavel' AND time = 'NomeDaVariavel2')  
OR data_nascimento BETWEEN 'NomeDaVariavel3' AND 'NomeDaVariavel4'
```

Utilize a cláusula **ORDER BY** para ordenar os registros do conjunto de registros. Por exemplo: suponha que deseja ordenar os registros do conjunto de registros de acordo com a renda do funcionário, da mais baixa para a mais alta. No SQL, ordene os registros da seguinte forma:

```
SELECT nome, sobrenome, salario FROM funcionarios  
ORDER BY salario
```

Como padrão, a cláusula **ORDER BY** ordena os registros na ordem ascendente (1, 2, 3... ou A, B, C...). Se desejar classificá-los na ordem descendente, da renda mais alta para a mais baixa, utilize a palavra-chave DESC da seguinte forma:

```
ORDER BY salario DESC
```

### **Unindo Tabelas**

É possível utilizar uma única instrução SELECT para recuperar dados de uma ou mais tabelas de um banco de dados. Esta instrução une as tabelas e retorna um único conjunto de registros que contém dados selecionados em cada tabela.

Por exemplo: um determinado banco de dados contém uma tabela com os dados pessoais dos funcionarios e outra tabela contém os convênios que a empresa possui. Para criar um catálogo de jogadores com os respectivos nomes, sobrenomes e quando estes foram artilheiro, é necessário recuperar informações das duas tabelas simultaneamente.

Para isso, crie uma união especificando todas as tabelas a serem incluídas e como se relacionam. Por exemplo:

```
SELECT nome, sobrenome, convenios, data_nascimento_convenio  
FROM funcionarios, convenios
```



*WHERE funcionarios.funcionarios\_id = convenios.convenios\_id*

A primeira linha especifica as colunas a serem recuperadas. As duas primeiras colunas, nome, sobrenome, constam na tabela funcionarios, enquanto que a quarta, data\_nascimento\_convenio está contida apenas na tabela convenio.

A segunda linha especifica as duas tabelas das quais os dados serão recuperados, funcionarios e convenios.

A última linha especifica os registros a serem unidos e recuperados nas duas tabelas. Cada tabela contém uma coluna com dados únicos e exclusivos de cada jogador como: funcionario\_id (na tabela funcionario, esta coluna é a chave primária). A cláusula WHERE compara o valor de funcionario\_id de uma tabela com esse mesmo valor em outra tabela. Quando uma correspondência for encontrada, todos os campos do registro na tabela funcionarios serão unidos a todos os campos do registro na tabela convenios.

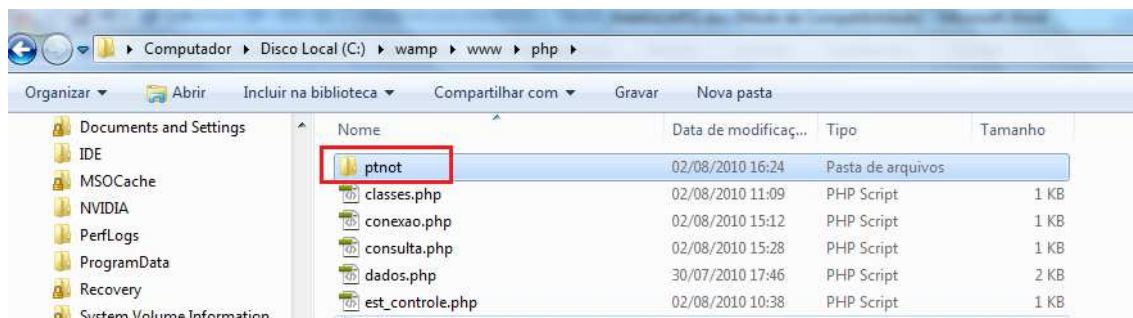
A utilização de uma sintaxe de união um pouco diferente poderá ser mais conveniente em alguns sistemas de banco de dados. Por exemplo: a seguinte instrução SQL utiliza as palavras-chave **SQL INNER JOIN...ON** para obter os mesmos resultados que o exemplo anterior:

```
SELECT nome, sobrenome, convenios, data_nascimento_convenio  
FROM funcionarios INNER JOIN convenios  
ON funcionarios.funcionarios_id = convenios.convenios_id
```

## **CONECTANDO AO BANCO DE DADOS PELO DREAMWEAVER**

Para que isto seja possível é preciso inicialmente que o Dreamweaver, trabalhe com tecnologia servidor.

Crie uma pasta em seu servidor para que possamos ligar o Dreamweaver a ela.



Clique em Dreamweaver site



Preencha os dados com o nome de seu site e pasta de origem.

Site Definition for sistnot

Basic Advanced

## Site Definition

Editing Files Testing Files Sharing Files

A site, in Dreamweaver, is a collection of files and folders that corresponds to a website on a server.

What would you like to name your site?

sistnot

Example: mySite

What is the HTTP Address (URL) of your site?

http://localhost:82/php/ptnot/

Example: http://www.myHost.com/mySite

If you want to work directly on the server using FTP or RDS, you should [create an FTP or RDS server connection](#). Working directly on the server does not allow you to perform sitewide operations like link checking or site reports.

< Back Next > Cancel Help

Site Definition for sistnot

Basic Advanced

## Site Definition

Editing Files, Part 2 Testing Files Sharing Files

Do you want to work with a server technology such as ColdFusion, ASP.NET, ASP, JSP, or PHP?

☐ No, I do not want to use a server technology.

☒ Yes, I want to use a server technology.

Which server technology?

PHP MySQL

< Back Next > Cancel Help

Site Definition for sistnot

Basic Advanced


## Site Definition

Editing Files, Part 3 Testing Files Sharing Files

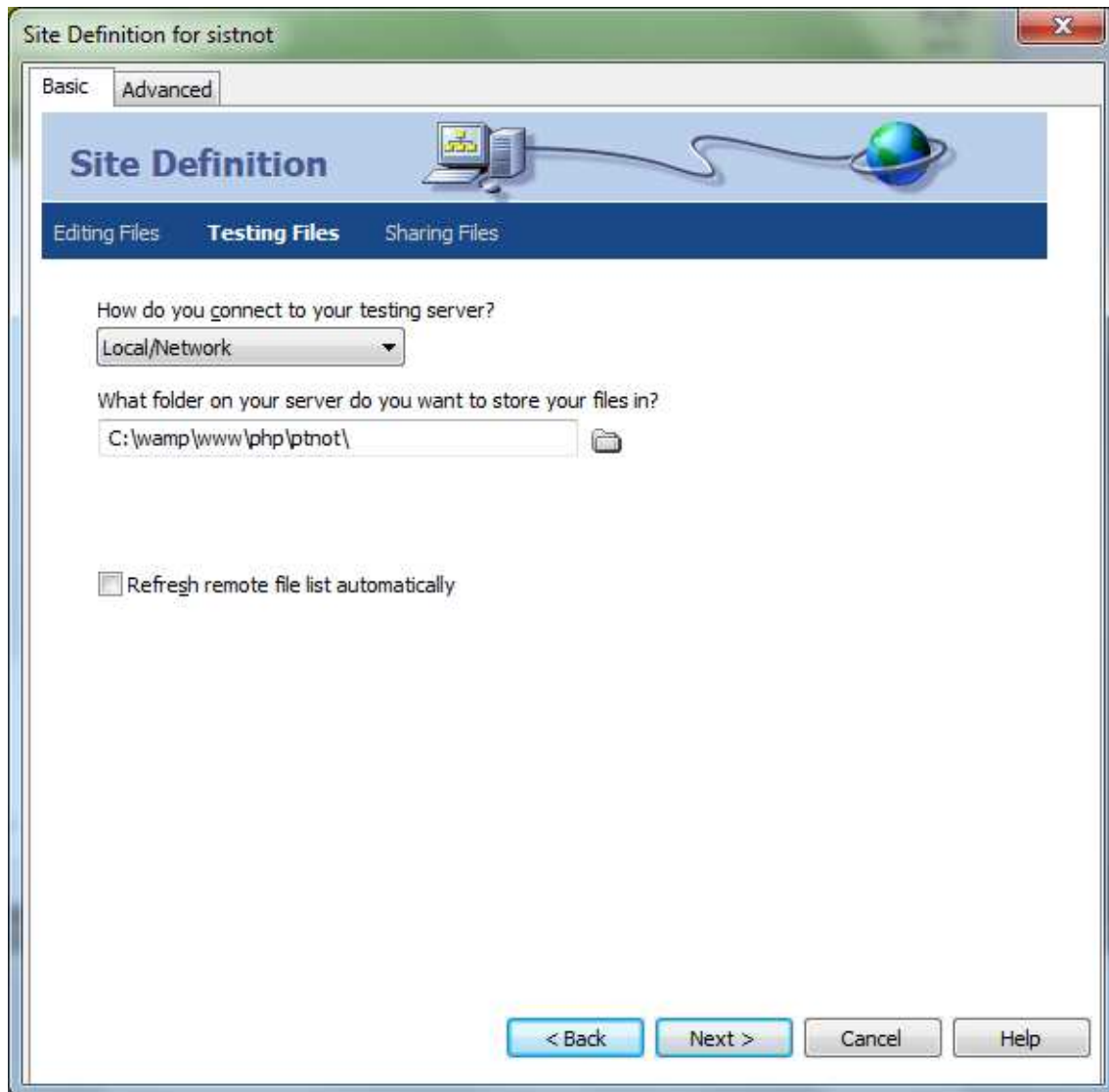
How do you want to work with your files during development?

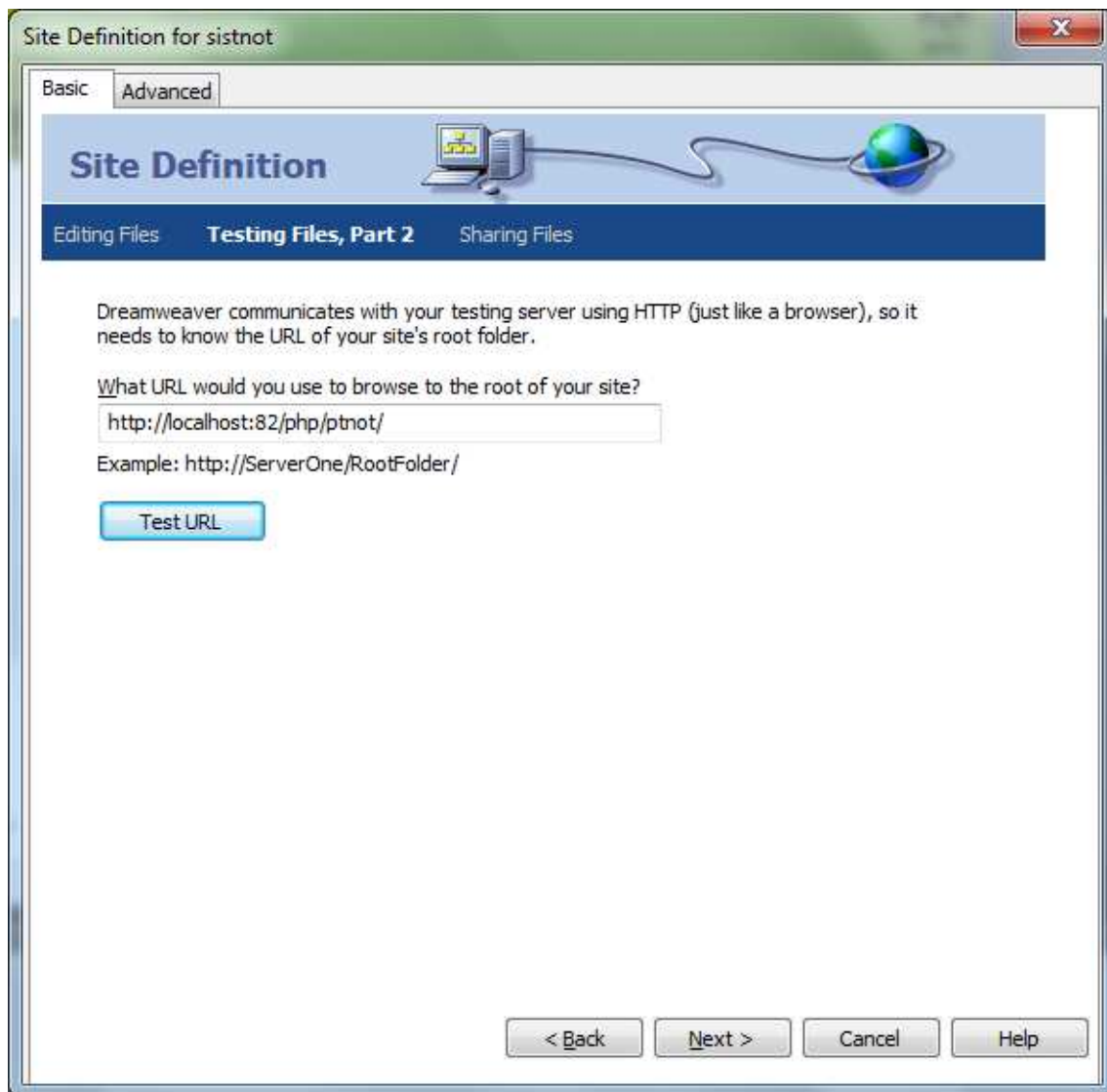
- ☐ Edit and test locally (my testing server is on this computer)
- ☒ Edit locally, then upload to remote testing server
- ☐ Edit directly on remote testing server using local network

Where on your computer do you want to store your files?

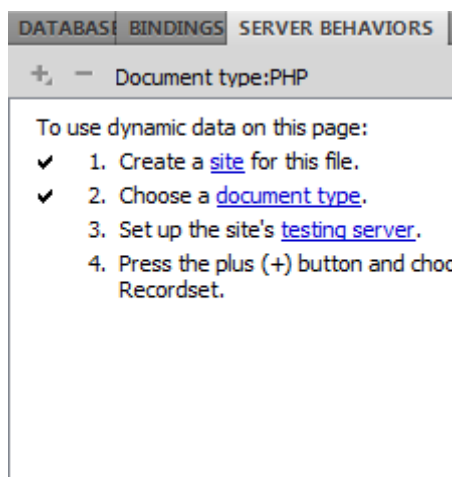
C: \wamp\www\php\ptnot\ 

< Back Next > Cancel Help



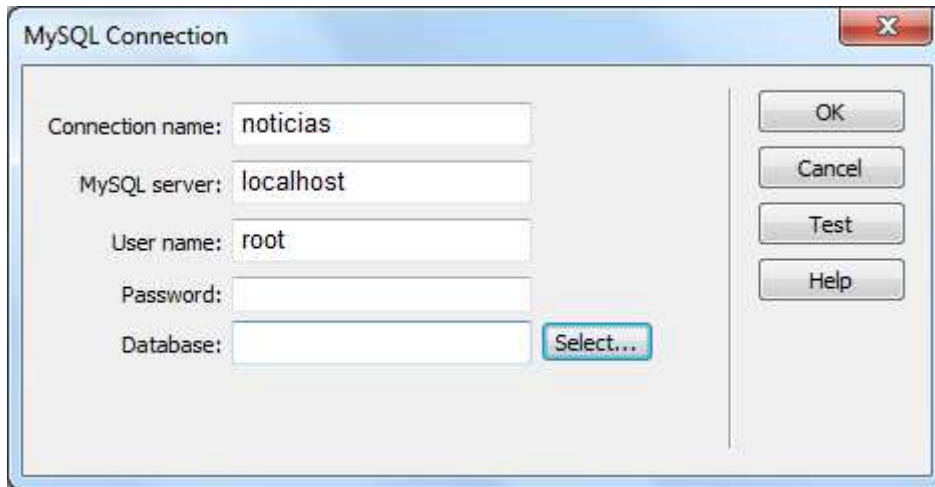


Clique no menu Window e chame o painel **Server Behaviors**.

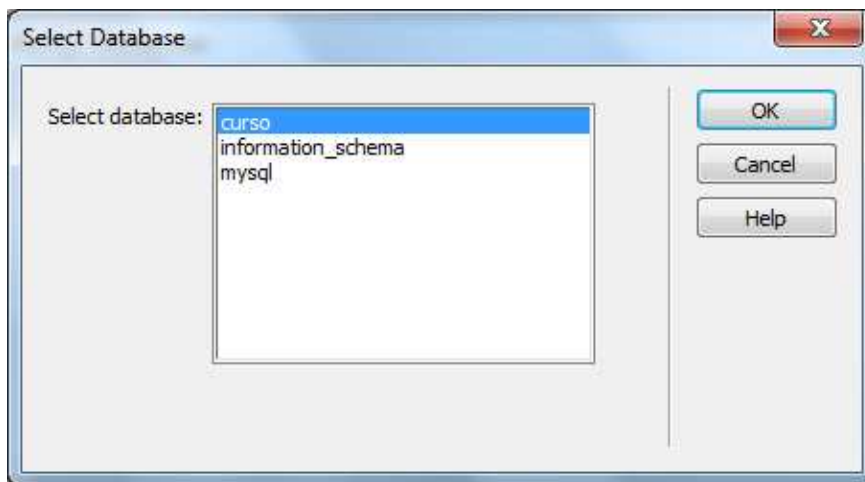


Clique na Guia DataBases, depois clique na opção **MySQL Connection**

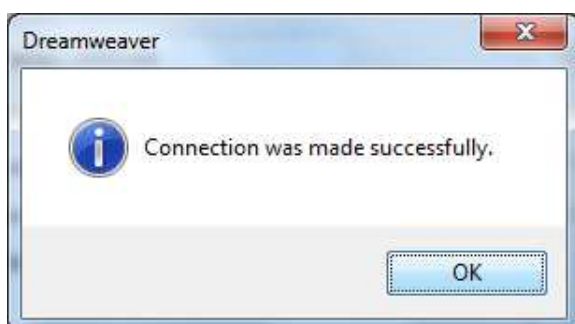
Configure a conexão da seguinte forma:



Clique no botão select para selecionar o banco de dados

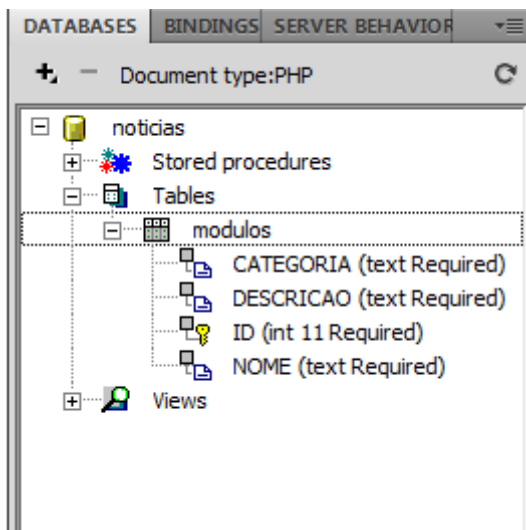


Clique em Test



Observe agora o painel DataBases



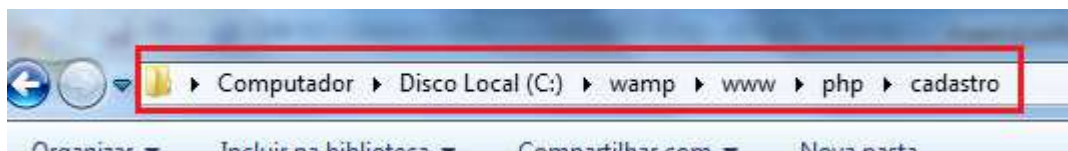


Você pode observar também que o Dreamweaver cria uma pasta chamada Connections e dentro desta pasta ele vai criar um arquivo de configuração com a nossa conexão.

## SISTEMA DE CADASTRO SIMPLES

Vamos criar um sistema de cadastro simples com auxílio do Dreamweaver.

Crie uma pasta chamada **cadastro** dentro da pasta **PHP**.



Vamos criar um novo site direcionando para esta pasta.



Na primeira tela coloque

Site Definition for cadastro

Basic Advanced

### Site Definition

Editing Files Testing Files Sharing Files

A site, in Dreamweaver, is a collection of files and folders that corresponds to a website on a server.

What would you like to name your site?

cadastro

Example: mySite

What is the HTTP Address (URL) of your site?

http://localhost:82/php/cadastro/

Example: http://www.myHost.com/mySite

If you want to work directly on the server using FTP or RDS, you should [create an FTP or RDS server connection](#). Working directly on the server does not allow you to perform sitewide operations like link checking or site reports.

< Back Next > Cancel Help

Clique em Next

Site Definition for cadastro

Basic Advanced

## Site Definition

Editing Files, Part 2 Testing Files Sharing Files

Do you want to work with a server technology such as ColdFusion, ASP.NET, ASP, JSP, or PHP?

☐ No, I do not want to use a server technology.

☒ Yes, I want to use a server technology.

Which server technology?

PHP MySQL

< Back Next > Cancel Help

Site Definition for cadastro

Basic Advanced


## Site Definition

Editing Files, Part 3 Testing Files Sharing Files

How do you want to work with your files during development?

- ☐ Edit and test locally (my testing server is on this computer)
- ☒ Edit locally, then upload to remote testing server
- ☐ Edit directly on remote testing server using local network

Where on your computer do you want to store your files?

C: \wamp\www\php\cadastro\ 

< Back Next > Cancel Help

Site Definition for cadastro

Basic Advanced

### Site Definition

Editing Files **Testing Files** Sharing Files

How do you connect to your testing server?

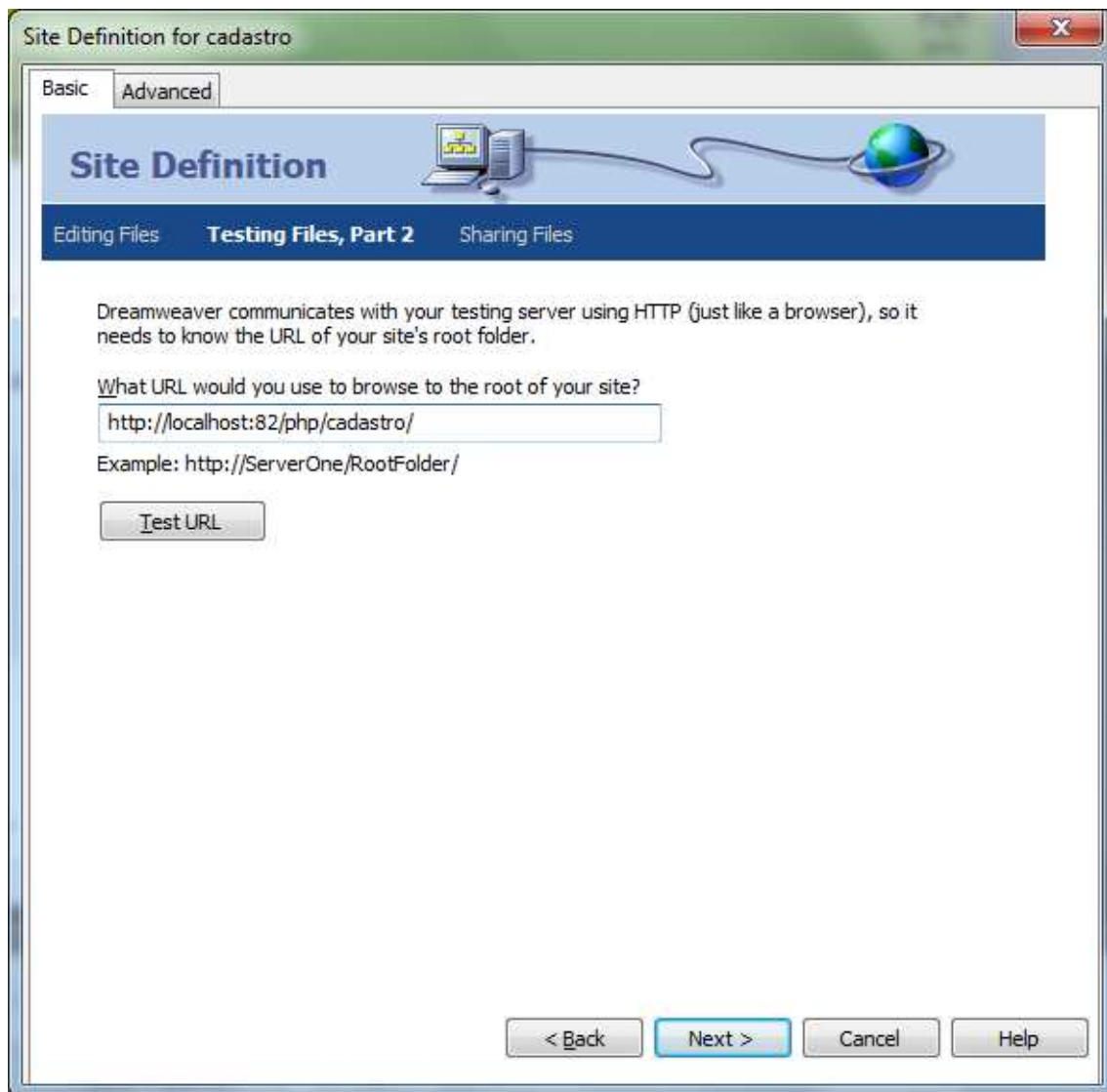
Local/Network

What folder on your server do you want to store your files in?

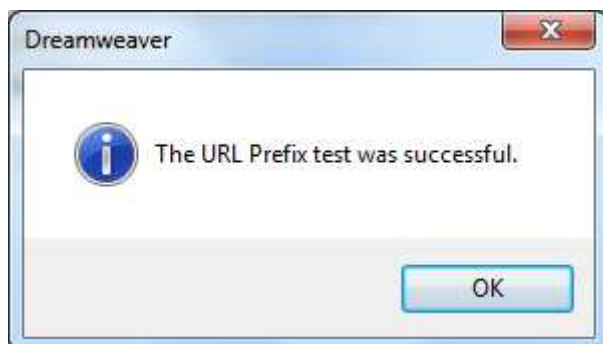
C:\wamp\www\php\cadastro\

☐ Refresh remote file list automatically

< Back Next > Cancel Help



Clique no botão **Test URL**



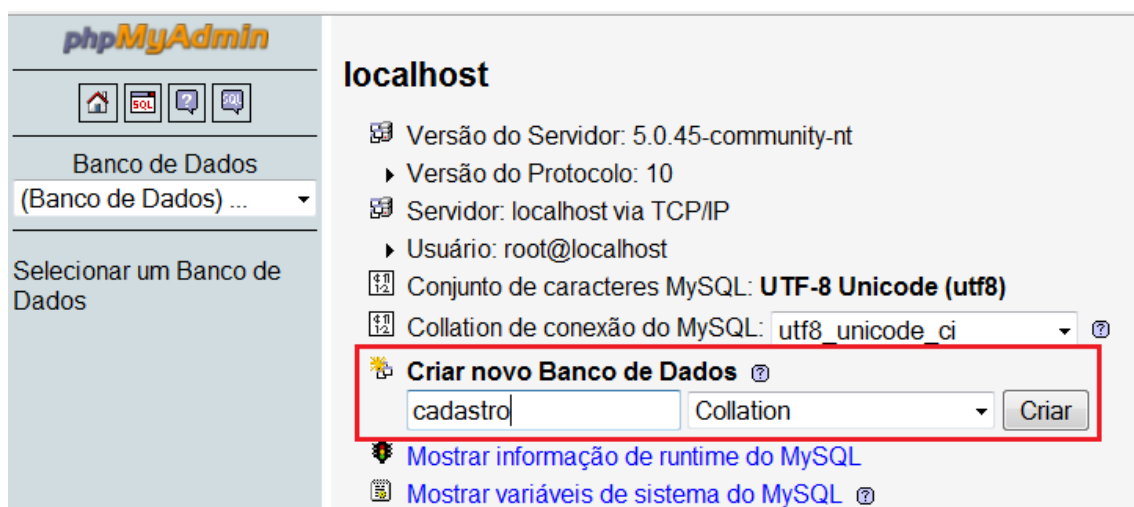
### ***Criando o banco de dados no phpMyAdmin***

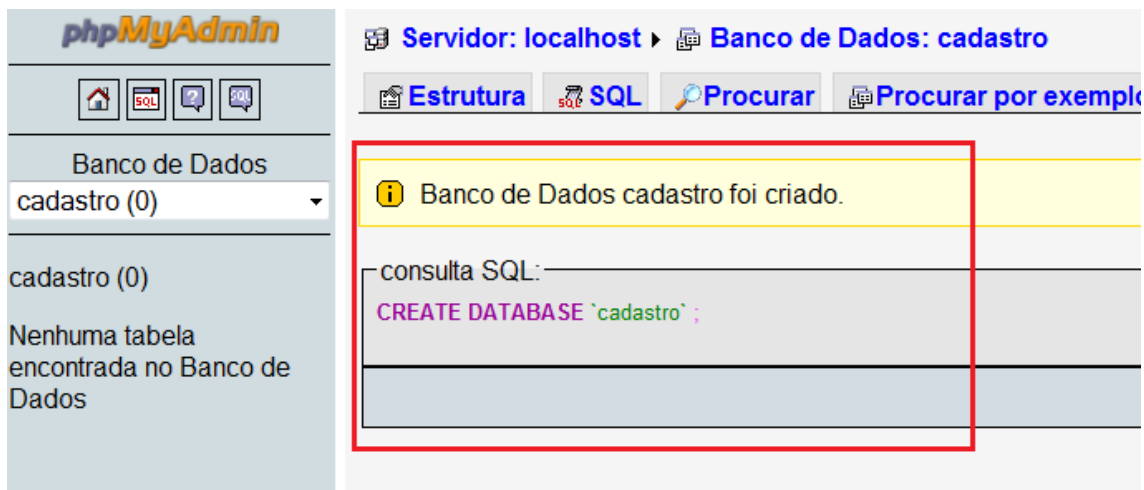
Abra o phpMyAdmin para configurarmos o nosso Banco de Dados de cadastro.



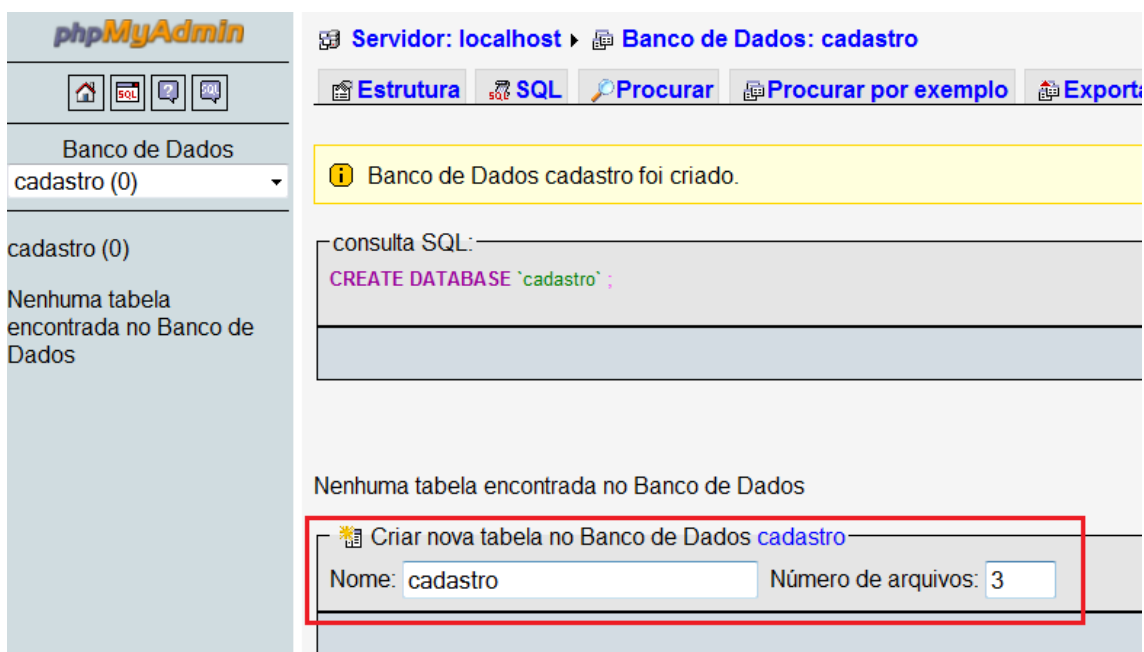


Vamos criar um novo banco de dados chamado cadastro.





Precisamos agora criar a tabela de cadastro em nosso Banco de Dados, vou chamá-la também de cadastro com 3 campos.



Clique em Executar.

Vamos inserir os seguintes campos:

1. cad\_id – responsável pela identificação individual de cada cadastro
2. cad\_nome – responsável pelo armazenamento do nome do cadastrado
3. cad\_email – responsável pelo armazenamento do e-mail do cadastrado

Colocamos o tipo (**Type**) de dados para o campo cad\_id, **smallint** com 5 caracteres. Isto significa que nosso sistema pode comportar mais de 65.000 cadastros. Quando selecionamos o atributo (**Atributtes**) **UNSIGNED**, estamos garantindo que a numeração gerada será a partir do zero e não incluirá



números negativos. Em Extra, ao marcarmos a opção **auto\_increment** (**auto\_incremento**), estamos definindo que o banco de dados gerará automaticamente, uma sequência de números para cada cadastro. E por fim, selecionamos a opção **Primary Key (Chave primária)**. Esta ação garante a identificação única e exclusiva em cada linha da tabela.

E quanto ao **VARCHAR** selecionado para os outros dois campos da tabela, significa que podemos inserir até 255 caracteres. Porém, delimitamos para 100 ao nome e 75, ao e-mail.

Servidor: localhost Banco de Dados: cadastro Tabela: cadastro

Campo	Tipo	Tamanho/Definir <sup>1</sup>	Collation	Atributos	Nulo
cad_id	SMALLINT	5		UNSIGNED	not null
cad_nome	VARCHAR	100			not null
cad_email	VARCHAR	75			not null

Nulo	Padrão <sup>2</sup>	Extra						Comentários
not null		auto_increment	<input checked="" type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="checkbox"/>	
not null			<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="checkbox"/>	
not null			<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input checked="" type="radio"/>	<input type="checkbox"/>	

Clique em salvar.

phpMyAdmin

Servidor: localhost Banco de Dados: cadastro Tabela: cadastro

Visualizar Estrutura SQL Procurar Inserir Exportar Importar Operações Limpa Eliminar

Tabela 'cadastro' 'cadastro' foi criado.

consulta SQL:

```
CREATE TABLE `cadastro` (
  `cad_id` SMALLINT(5) UNSIGNED NOT NULL AUTO_INCREMENT PRIMARY KEY,
  `cad_nome` VARCHAR(100) NOT NULL,
  `cad_email` VARCHAR(75) NOT NULL
) ENGINE = INNODB;
```

[ Editar ] [ Criar código F

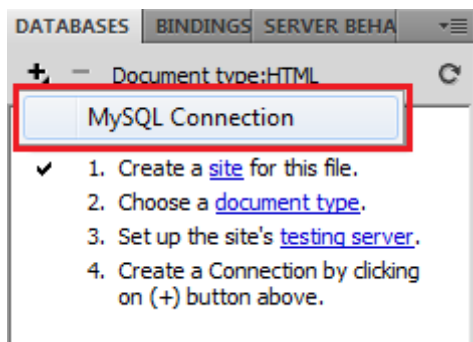
	Campo	Tipo	Collation	Atributos	Nulo	Padrão	Extra	Ações					
<input type="checkbox"/>	cad_id	smallint(5)		UNSIGNED	Não		auto_increment	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	cad_nome	varchar(100)	latin1_swedish_ci		Não			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
<input type="checkbox"/>	cad_email	varchar(75)	latin1_swedish_ci		Não			<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

↑ Marcar todos / Desmarcar todos Com marcados: ☐ ☐ ☐ ☐ ☐ ☐

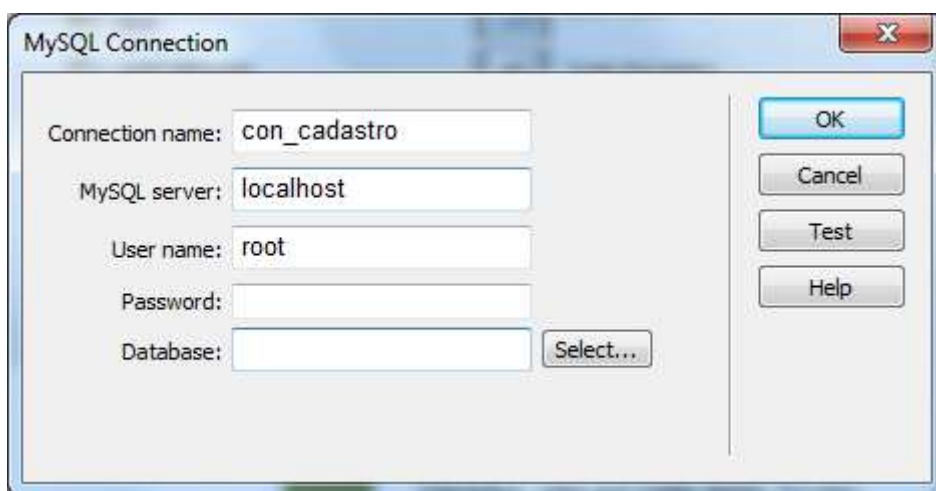
## Criando a conexão do Banco de Dados com o Dreamweaver

Para trabalharmos com dados no Dreamweaver usaremos praticamente três painéis: DataBases, Bindings e Server Behaviors.

Clique em DataBases e clique no sinal de mais (+) e escolha MySQL Connection.

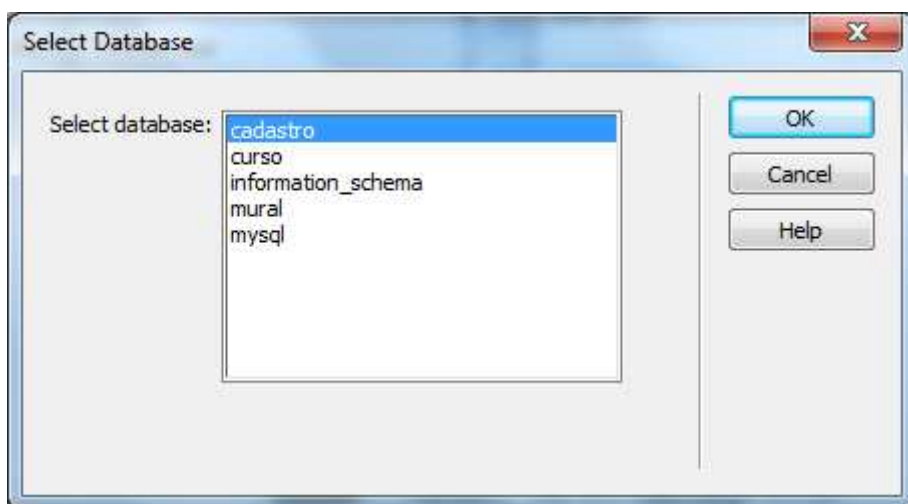


É necessário preencher os dados na próxima tela

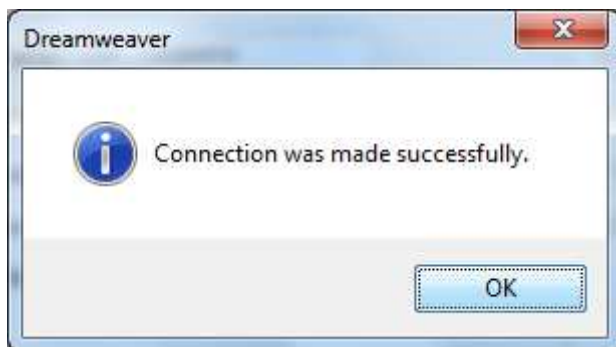


Como estamos testando localmente o User name é **root** e a senha em branco, ao testar em seu servidor é preciso usar as configurações que o mesmo te forneceu.

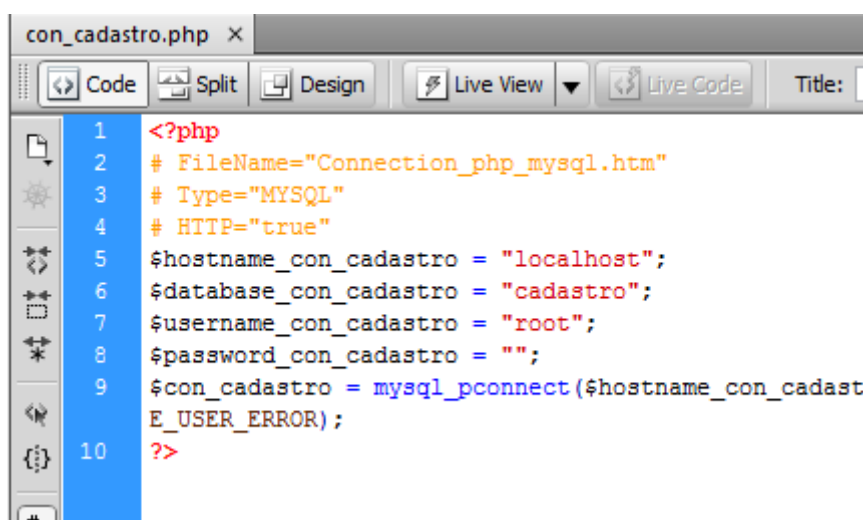
Clique em Select para selecionarmos a base de dados a qual vamos nos conectar.



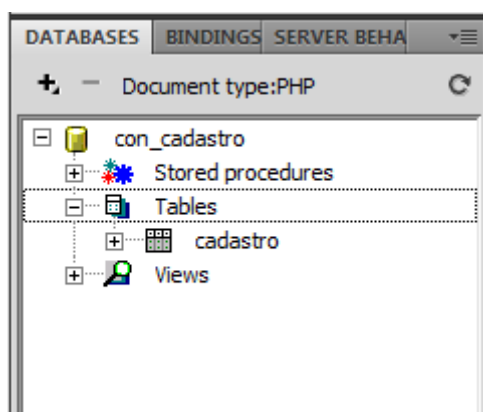
Clique em Test para verificar se a conexão foi efetuada com sucesso.



Clique em Ok e depois novamente em OK, observe que foi criada uma pasta chamada **Connections** em seu projeto, abra esta pasta e abra este arquivo, observe que ele criou toda a estrutura php para conectar-se ao seu Banco de Dados.



Observe também no painel Databases a estrutura com a tabela criada.

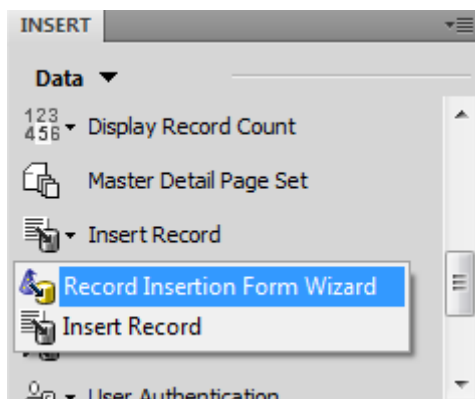


## Criando o formulário de cadastro no Banco de Dados

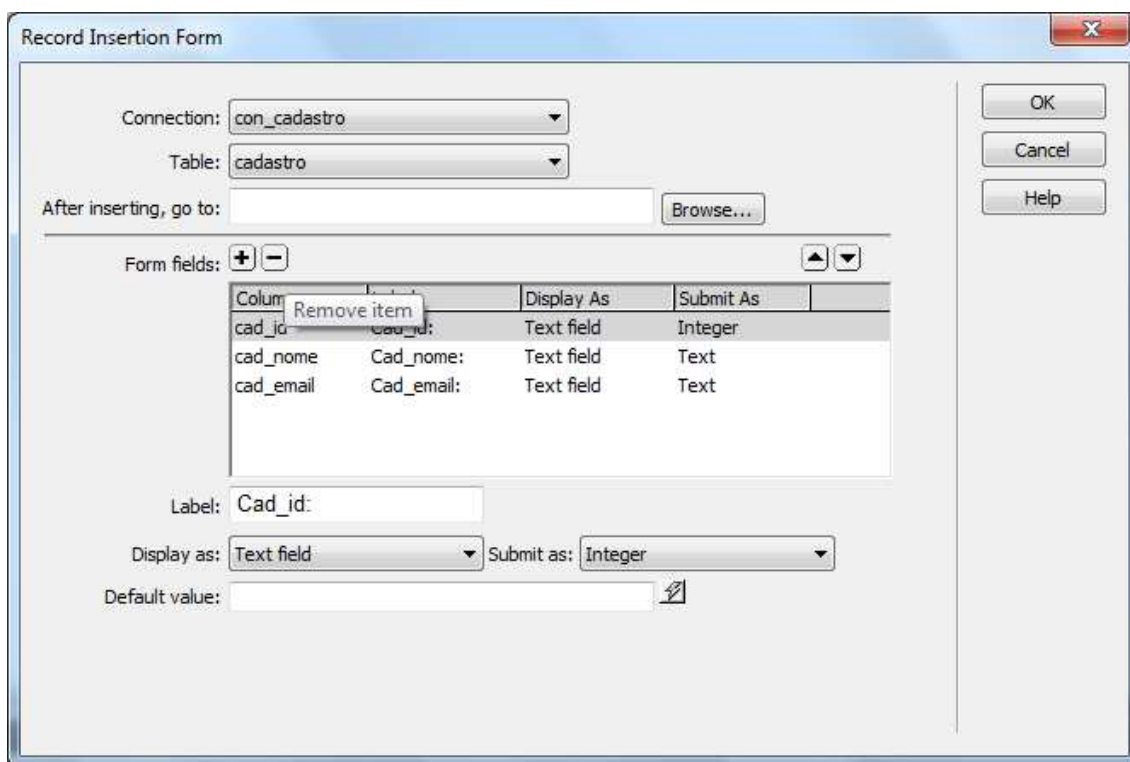
Crie uma página nova PHP e salve-a como index.php.

Vamos agora criar o formulário de cadastro.

Existe um recurso no Dreamweaver chamado **Record Insertion Form Wizard** e, através dele, de uma só vez, criaremos o formulário e os códigos PHP. Acessemos então no painel **Insert**, aba **Data**, a opção Insert record > Record Insertion Form Wizard.



Na tela que vai aparecer, vamos remover do formulário o campo ID, pois ele é criado automaticamente.



No campo **After inserting, go to** coloque **cad\_realizado.php**.  
Mude também os Labels dos campos para Nome e E-mail.

Record Insertion Form

Connection: con\_cadastro

Table: cadastro

After inserting, go to: cad\_realizado.php

Form fields:

Column	Label	Display As	Submit As
cad_nome	Nome	Text field	Text
cad_email	E-mail	Text field	Text

Label: E-mail

Display as: Text field Submit as: Text

Default value:

Clique em OK.

Seu formulário será criado.

Nome

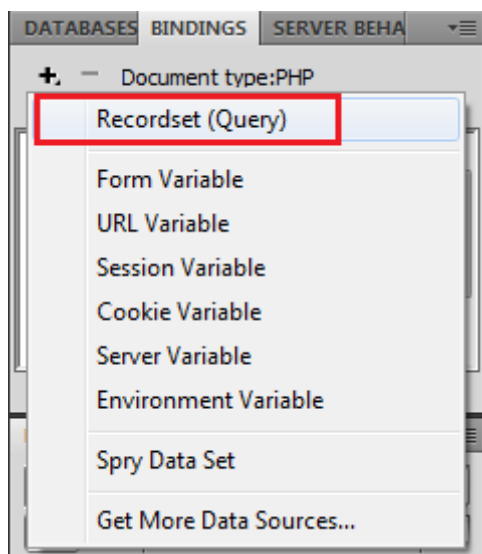
E-mail

Se quiser pode configurar seu formulário, como, por exemplo, mudar cores, mudar texto do botão, etc.

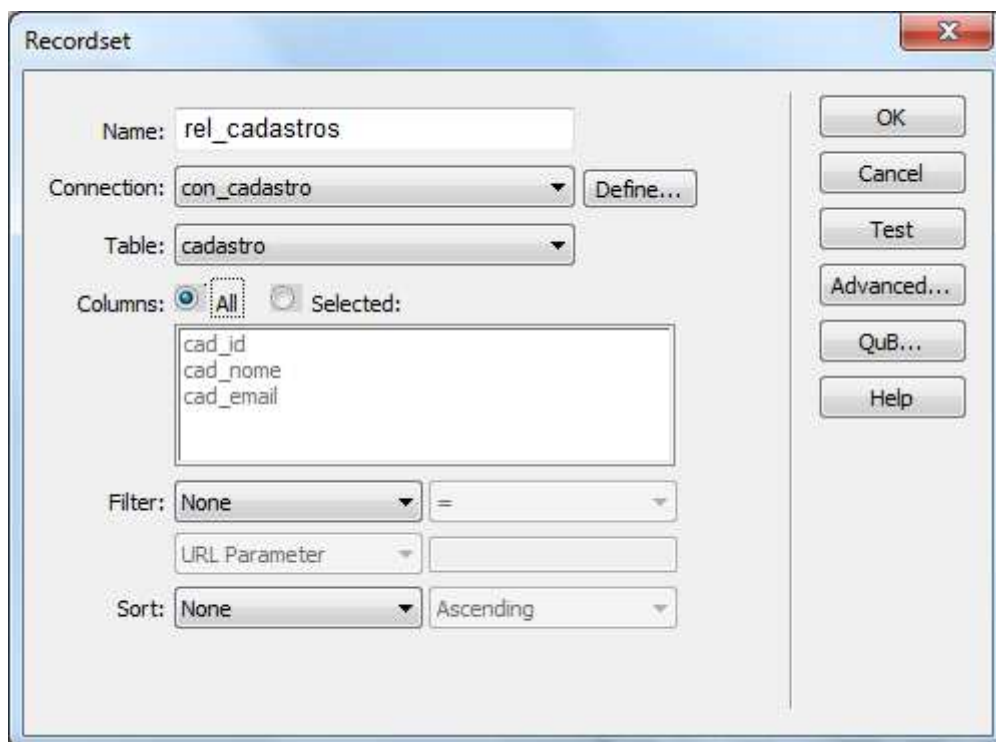
Precisamos agora criar a página de confirmação a página **cad\_realizado.php**, Crie um novo arquivo PHP e salve-o com este nome.

Adicione um link chamado cadastros e direcione para uma página chamada **cadastros.php**, crie também esta página.

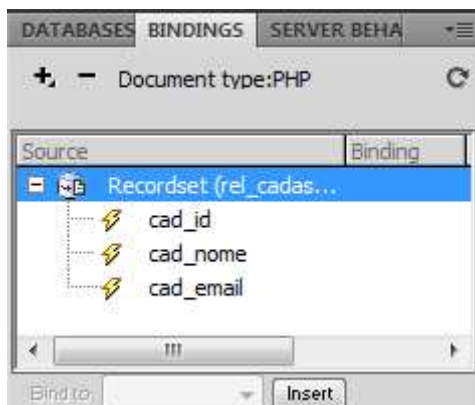
Clique no painel **Bindings** e escolha **RecordSet (Query)**.



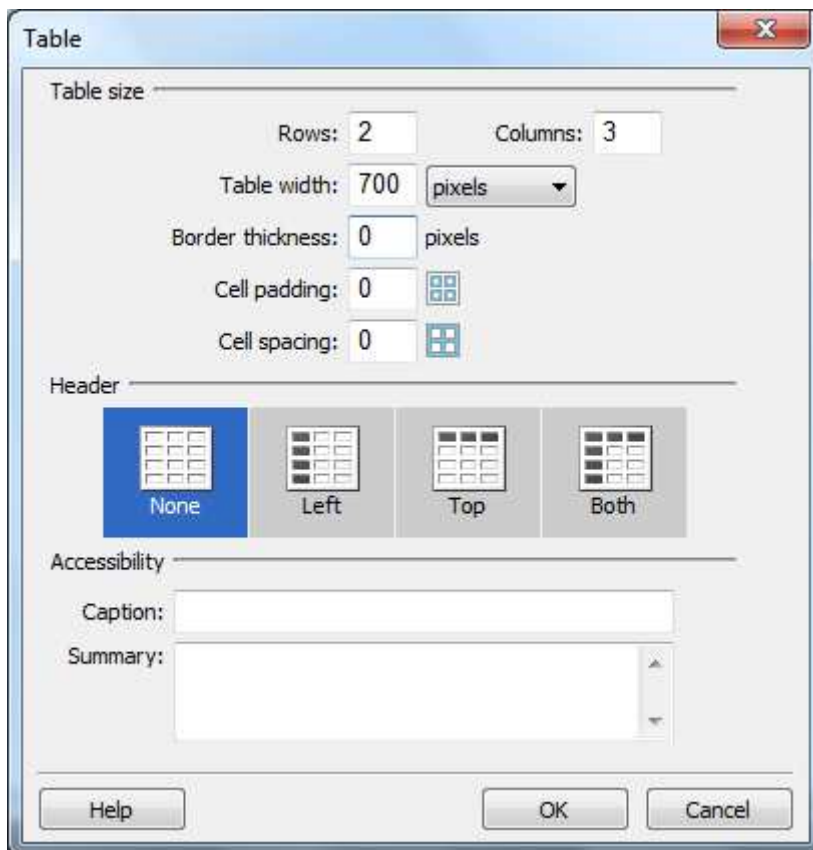
Dê um nome ao seu Recordset.



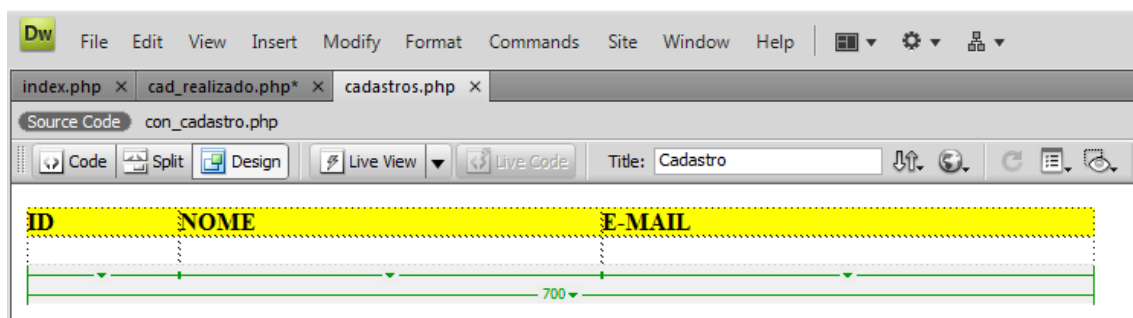
Clique em OK.



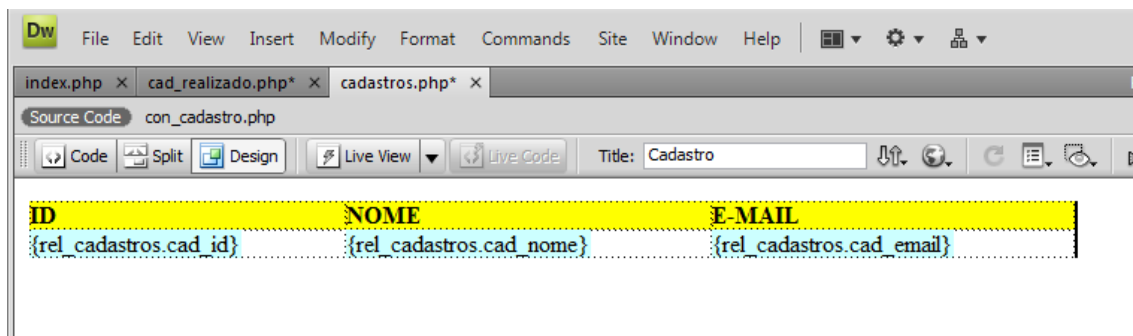
Crie neste arquivo uma tabela com 2 linhas e 3 colunas, com largura de 700 px:



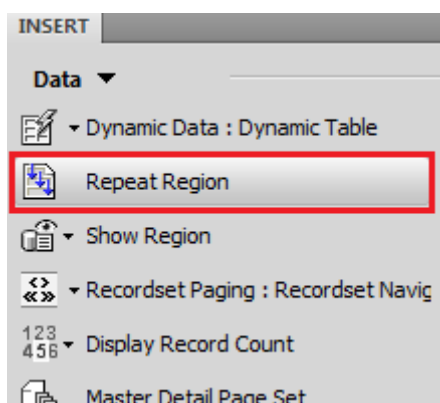
Formate-a como for necessário.



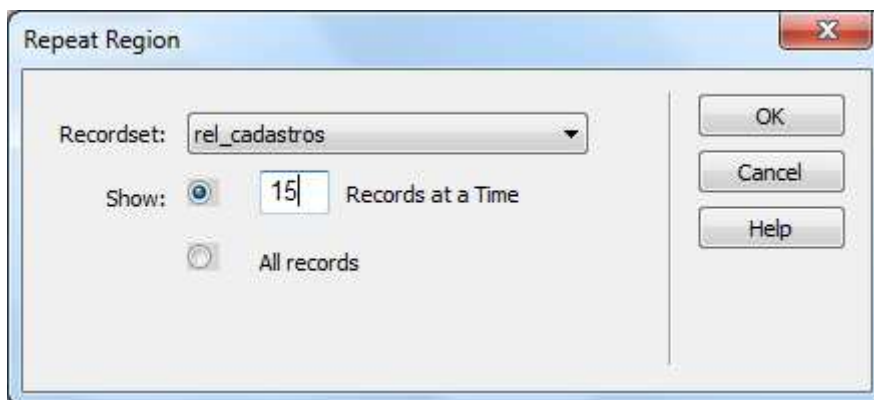
É necessário agora adicionarmos os **Recordsets** para cada campo na tabela.



Para podermos visualizar todos os campos de nosso banco de dados, selecione a linha inteira onde estão os recordsets, depois na ABA **Data** clique em **Repeat Region**.



Defina a quantidade de dados a serem mostrados por tela.



Aqui se deve ter um cuidado, mesmo selecionando a linha, o Dreamweaver pode se perder e ao testar sua repetição, ele, adicionar os dados ao lado e não abaixo.



ID	NOME	E-MAIL
1	Marcos	marcospfurlan@gmail.com
2	Maria	maria@teste.com.br

[Retornar ao cadastramento](#)

Caso isso ocorra abra o código de seu formulário e mude as linhas do **looping** para pegar toda a linha.

```

68 </tr>
69 <?php do { ?>
70 <tr>
71 <td><?php echo $row_rel_cadastrados['cad_id']; ?></td>
72 <td><?php echo $row_rel_cadastrados['cad_nome']; ?></td>
73 <td><?php echo $row_rel_cadastrados['cad_email']; ?></td>
74 </tr>
75 <?php } while ($row_rel_cadastrados = mysql_fetch_assoc($rel_cadastrados)); ?>
76 </table>
77 <p><a href="index.php">Retornar ao cadastramento</a></p>

```

ID	NOME	E-MAIL
1	Marcos	marcospfurlan@gmail.com
2	Maria	maria@teste.com.br
3	João	joao@terra.com.br

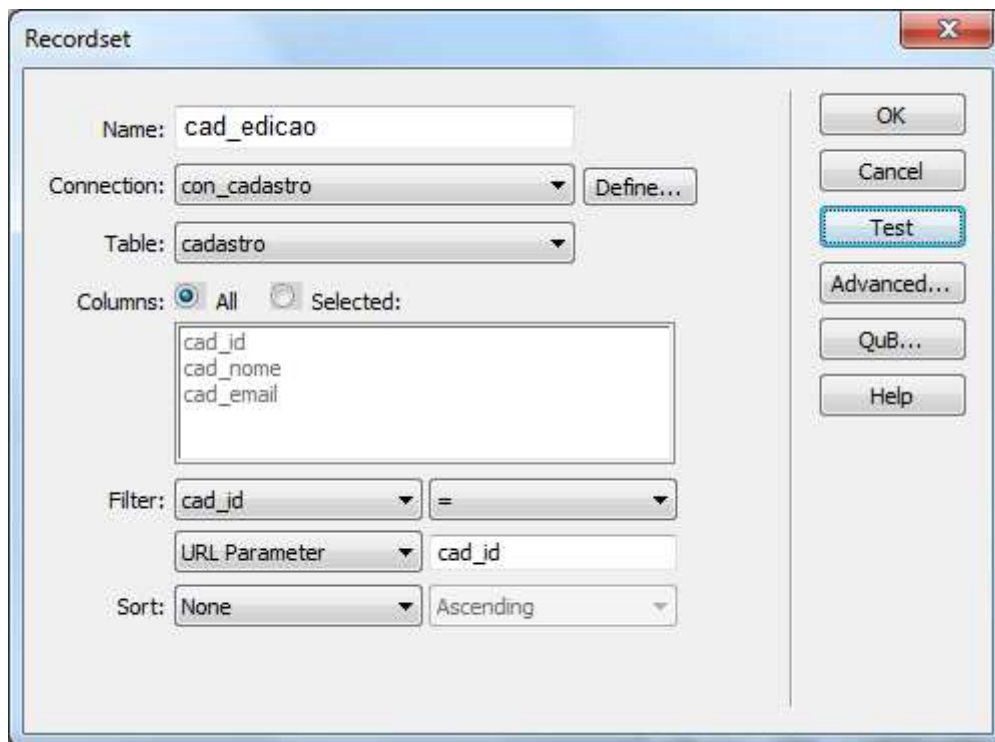
[Retornar ao cadastramento](#)

### ***Edição dos dados cadastrados***

Vamos criar um novo arquivo PHP com o nome de **cad\_edicao.php**.

Através do painel Bindings vamos criar um novo Recordset.

Defina o campo **Filter** como cad\_id.

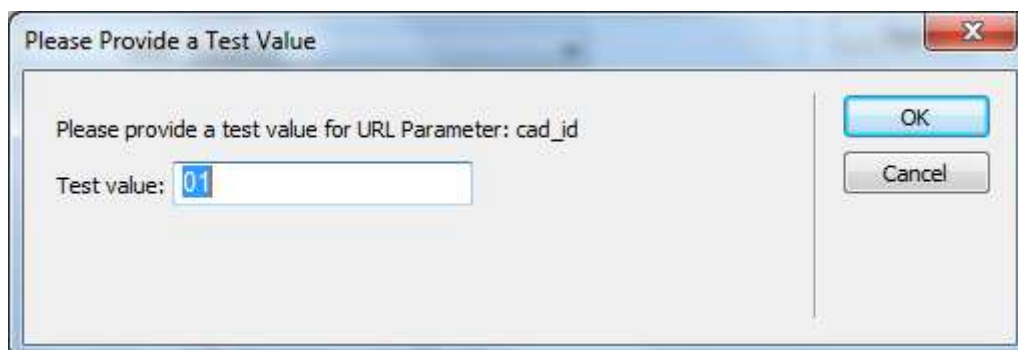


The 'Recordset' dialog box is shown with the following configuration:

- Name: cad\_edicao
- Connection: con\_cadastro
- Table: cadastro
- Columns: ☒ All, ☐ Selected: cad\_id, cad\_nome, cad\_email
- Filter: cad\_id = URL Parameter cad\_id
- Sort: None Ascending

Buttons on the right: OK, Cancel, Test (highlighted), Advanced..., QuB..., Help.

Clique em Test.

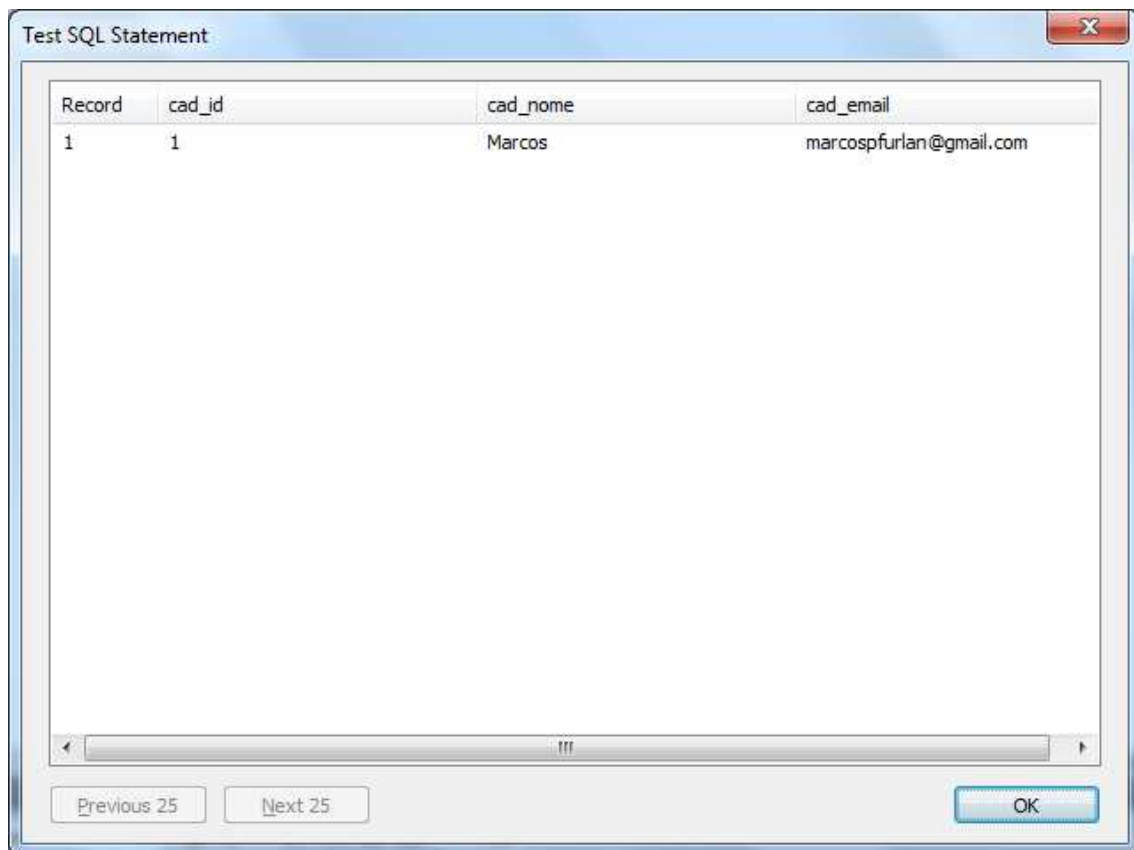


The 'Please Provide a Test Value' dialog box is shown with the following configuration:

- Please provide a test value for URL Parameter: cad\_id
- Test value: 01

Buttons on the right: OK, Cancel.

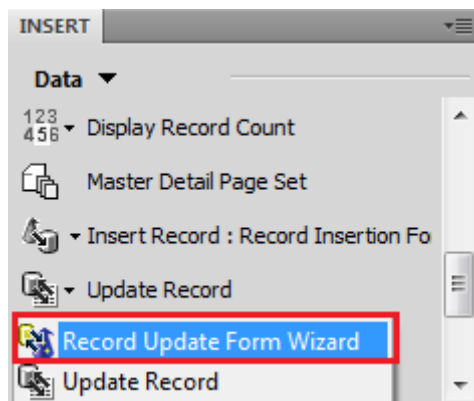
Clique em OK, para verificar o resultado do teste.



Clique em OK

E vamos criar o formulário de edição.

Clique em **Data, Update Record, Record Update Form Wizard.**



Configure o seu Record Update.

Record Update Form

Connection:

Table to update:

Select record from:

Unique key column:  ☒ Numeric

After updating, go to:

Form fields:

Column	Label	Display As	Submit As
cad_nome	Nome	Text field	Text
cad_email	E-mail	Text field	Text

Label:

Display as:  Submit as:

Default value:

Nome {cad\_edicao.cad\_nome}

E-mail {cad\_edicao.cad\_email}

Crie o arquivo **res\_edicao.php** e adicione nele.


index.php x cad\_realizado.php x cadastros.php x cad\_edicao.php x res\_edicao.php x

Code Split Design Live View Live Code Title: Cadastro - Edição

Edição de Cadastro |

[Cadastros](#)

Na pagina cadastros.php, insira uma nova coluna, nesta nova coluna adicione uma imagem.

Repeat	NOME	E-MAIL	EDITAR
{rel_cadastrados.cad_id}	{rel_cadastrados.cad_nome}	{rel_cadastrados.cad_email}	

[Retornar ao cadastramento](#)

Como link nesta imagem adicione:

**cad\_edicao.php?cad\_id=<?php echo \$row\_rel\_cadastrados['cad\_id']; ?>">**

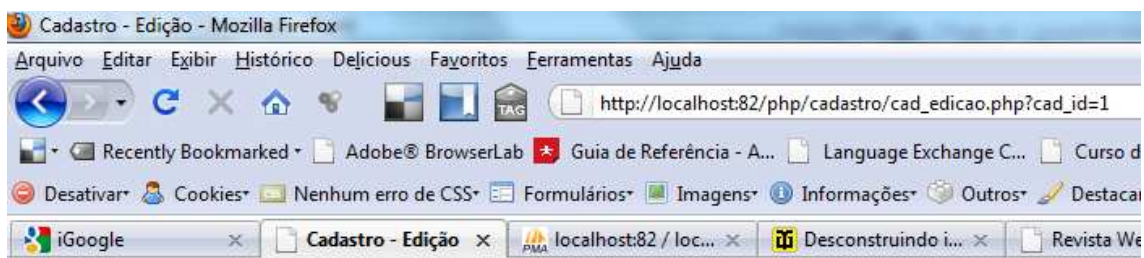
```

70 <?php do { ?>
71 <tr>
72 <td><?php echo $row_rel_cadastrados['cad_id']; ?></td>
73 <td><?php echo $row_rel_cadastrados['cad_nome']; ?></td>
74 <td><?php echo $row_rel_cadastrados['cad_email']; ?></td>
75 <td align="center"><a href="cad_edicao.php?cad_id=<?php echo $row_rel_cadastrados['cad_id']; ?>"></a></td>
76 </tr>
77 <?php } while ($row_rel_cadastrados = mysql_fetch_assoc($rel_cadastrados)); ?>

```




ID	NOME	E-MAIL	EDITAR
1	Marcos	marcospfurlan@gmail.com	
2	Maria	maria@teste.com.br	
3	João	joao@terra.com.br	

[Retornar ao cadastramento](#)



Nome Marcos

Email marcos@teste.com

ID	NOME	E-MAIL	EDITAR
1	Marcos	marcos@teste.com	
2	Maria	maria@teste.com.br	
3	João	joao@terra.com.br	

[Retornar ao cadastramento](#)

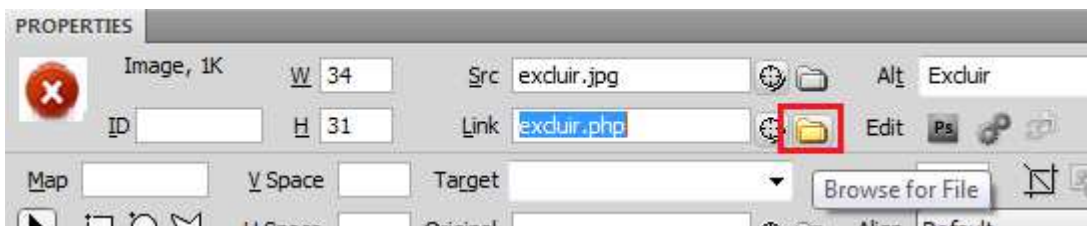
## Exclusão de Registro

Crie mais uma coluna ao lado de editar na tabela que existe em cadastro.php e adicione uma imagem para exclusão.

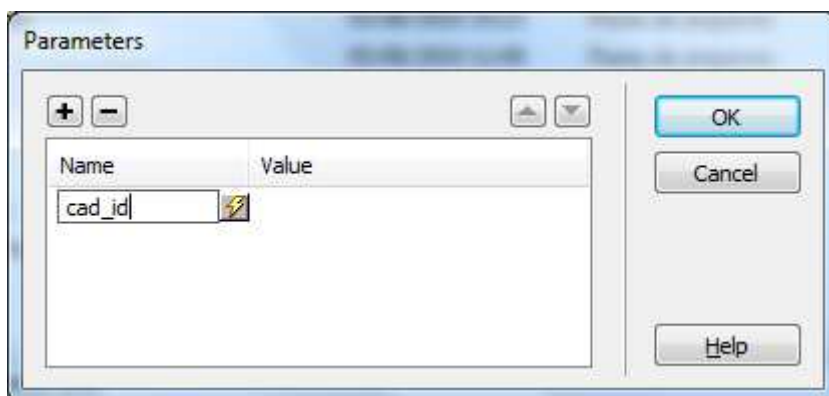
Repeat	NOME	E-MAIL	EDITAR	EXCLUIR
{rel_cadastros.cad_id}	{rel_cadastros.cad_nome}	{rel_cadastros.cad_email}		

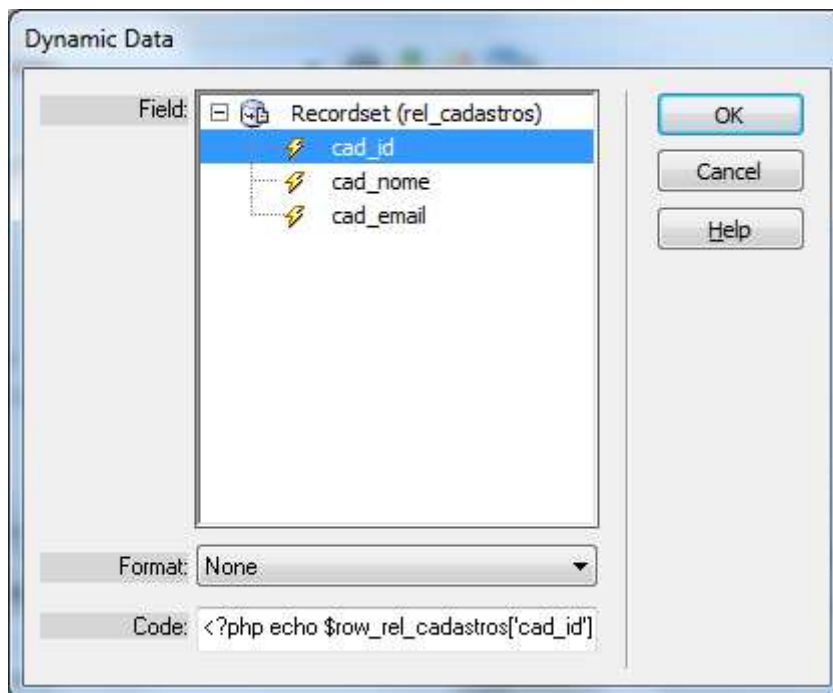
[Retornar ao cadastramento](#)

Em seguida, com a imagem selecionada, coloquemos o link da página responsável pela exclusão: **excluir.php** e em seguida, cliquemos na “pastinha” ao lado do campo link para fazermos a inclusão das instruções do registro.



Na primeira coluna (name), coloquemos a palavra **cad\_id** e na segunda (coluna value), cliquemos no “raio” para abrir uma nova janela, Dynamic Data e selecionemos o campo **cad\_id**.





Clique em Ok, e OK novamente.

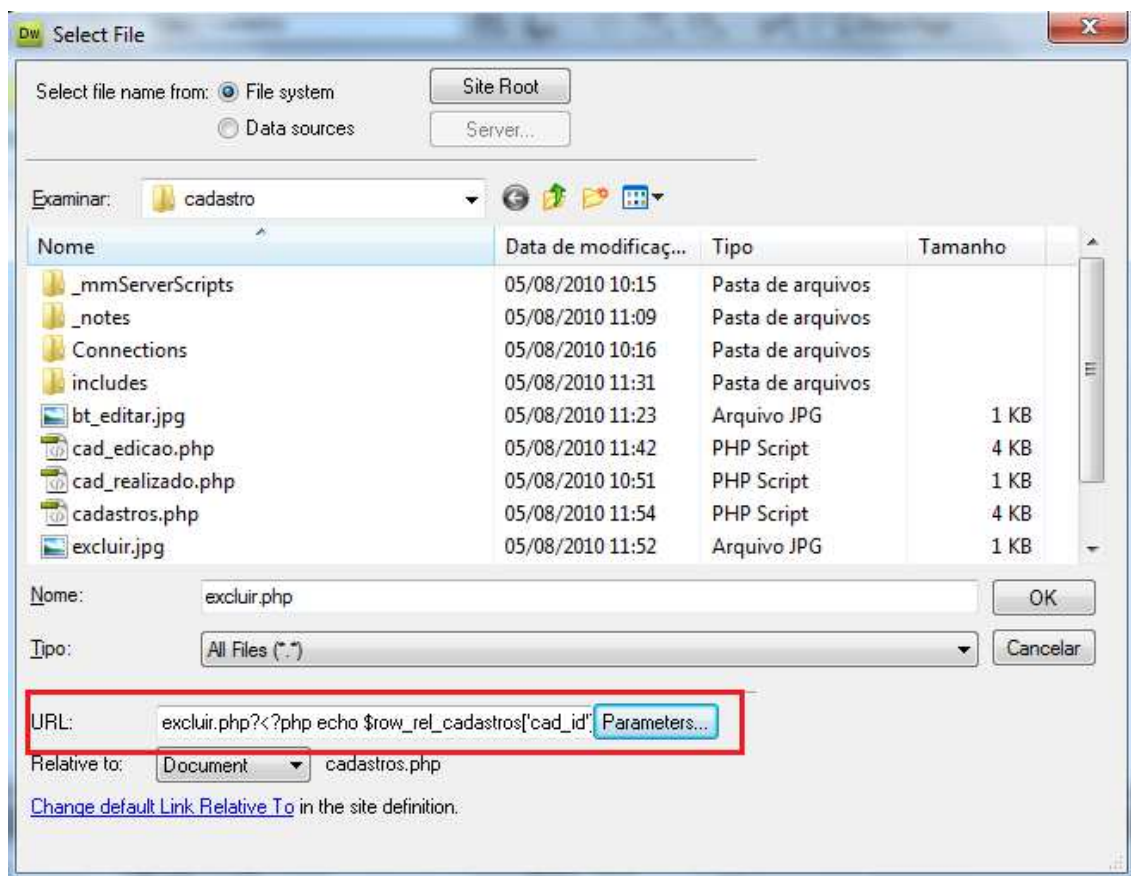


Aqui também ocorre do Dreamweaver se perder na hora de criar o código, verifique se a linha esta da seguinte forma:

```
<td align="center"><a href="excluir.php?cad_id=<?php echo $row_rel_cadastrros['cad_id']; ?>"></a></td>
```

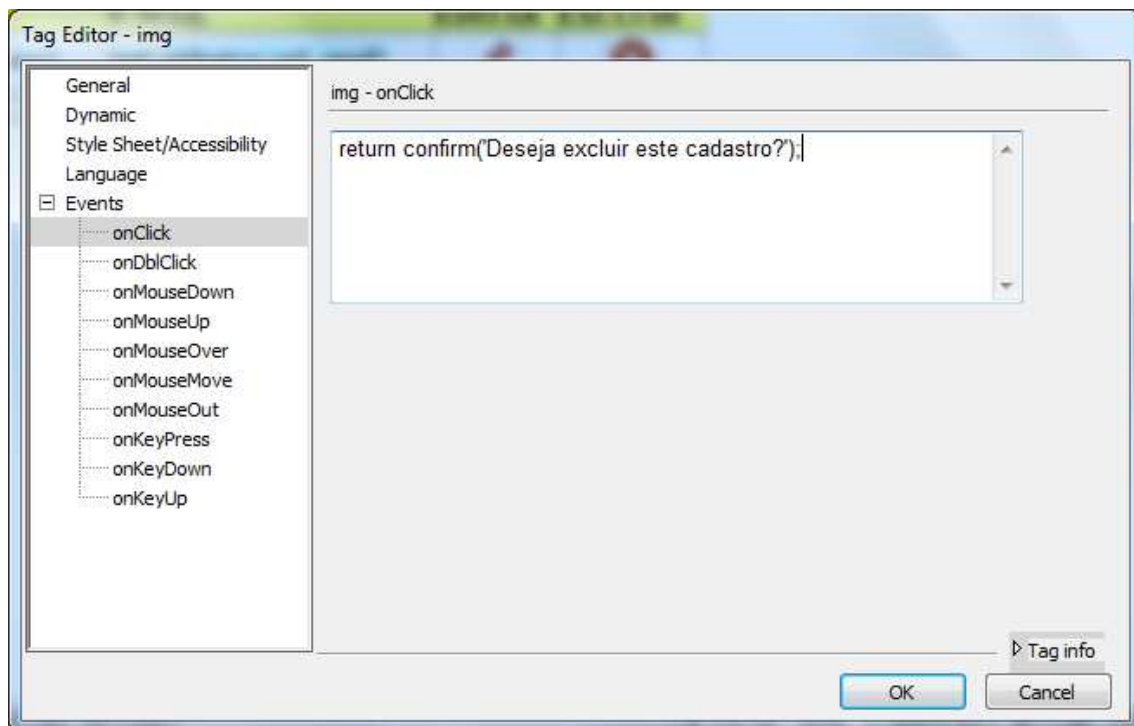
```
74 <td><?php echo $row_rel_cadastrros['cad_nome']; ?></td>
75 <td><?php echo $row_rel_cadastrros['cad_email']; ?></td>
76 <td align="center"><a href="cad_edicao.php?cad_id=<?php echo $row_rel_cadastrros['cad_id']; ?>"></a></td>
77 <td align="center"><a href="excluir.php?cad_id=<?php echo $row_rel_cadastrros['cad_id']; ?>"></a></td>
78 </tr>
```



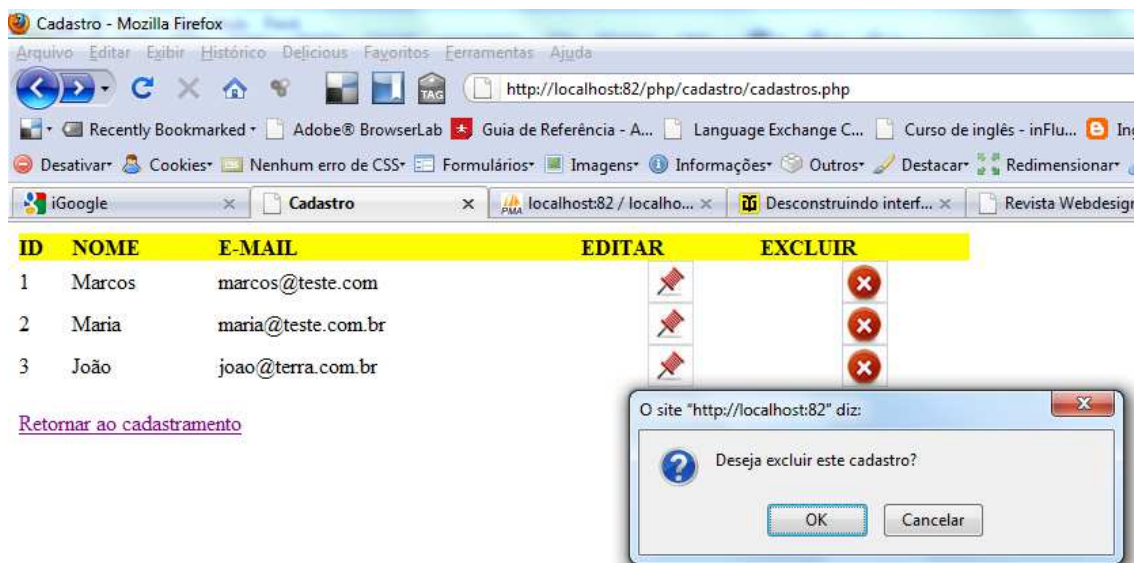


Como se trata de uma ação destrutiva, ou seja, que não há como voltar, é conveniente pedir ao usuário que confirme sua atitude de excluir ou não. E esta confirmação faremos utilizando Javascript. Com a imagem selecionada, acesse o menu **Modify > Edit TAG**, categoria **Events > onClick** e digitemos a mensagem conforme imagem:





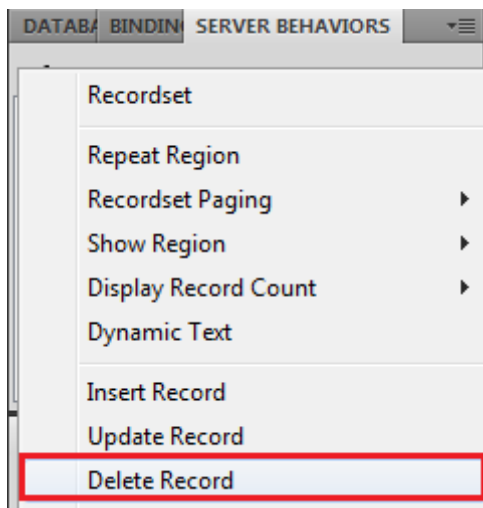
Mensagem: **return confirm('Deseja excluir este cadastro?');**



Esta pendente ainda a página **excluir.php**, para ela vamos proceder da seguinte forma.

Crie uma nova página PHP.

Clique no painel **Server Behaviors**. Clique no sinal de mais (+) e escolha **Delete Record**.



Configure da seguinte forma



Salve a página como **excluir.php**.

Visualize o código da página.

Localize a linha: **\$deleteGoTo = "cadastros.php";**

```

40 |
41 | $deleteGoTo = "cadastros.php";
42 | if (isset($_SERVER['QUERY_STRING'])) {
43 |     $deleteGoTo .= (strpos($deleteGoTo, '?')) ? "&" : "?";
44 |     $deleteGoTo .= $_SERVER['QUERY_STRING'];
45 | }
46 | header(sprintf("Location: %s", $deleteGoTo));
47 | }
48 | ?>

```

Adicione após o nome do arquivo:

**\$deleteGoTo = "cadastros.php?excluido=sucesso";**

```

40
41 $deleteGoTo = "cadastros.php?excluido=sucesso";
42 if (isset($_SERVER['QUERY_STRING'])) {
43     $deleteGoTo .= (strpos($deleteGoTo, '?') ? "&" : "?");
44     $deleteGoTo .= $_SERVER['QUERY_STRING'];
45 }
46 header(sprintf("Location: %s", $deleteGoTo));
47 }
48 ?>
49 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http
50 <html xmlns="http://www.w3.org/1999/xhtml">
51 </head>

```

Agora na página **cadastros.php** e adicione um parágrafo abaixo de sua tabela e mude para a visualização de código, adicione o seguinte código:

```
<?php
```

```
if ((isset($_GET['excluido'])) && ($_GET['excluido']=="sucesso")) {
```

```
echo "<p><i>Cadastro excluído com sucesso!</i>";
```

```
}
```

```
?>
```

```

81 <p>
82 <?php
83 if ((isset($_GET['excluido'])) && ($_GET['excluido']=="sucesso")) {
84     echo "<p><i>Cadastro excluído com sucesso!</i>";
85 }
86 ?>
87 </p>

```

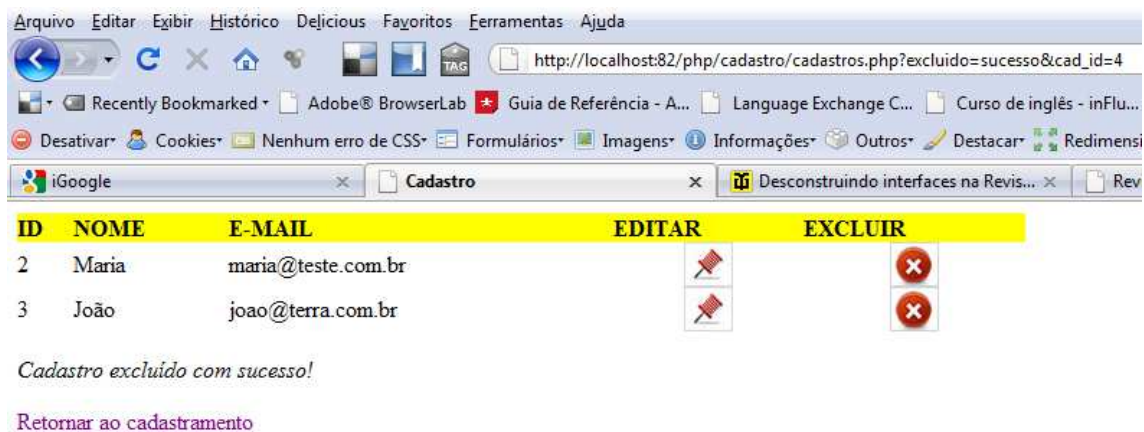
Salve todos seus arquivos e teste o sistema de cadastro.

The screenshot shows a web browser window with the address bar displaying `http://localhost:82/php/cadastro/cadastros.php`. The browser's address bar and tabs are visible, showing several open pages. Below the browser window, a table is displayed with the following structure:

ID	NOME	E-MAIL	EDITAR	EXCLUIR
2	Maria	maria@teste.com.br		
3	João	joao@terra.com.br		
4	Marcos	marcos@teste.com.br		

Below the table, there is a link: [Retornar ao cadastramento](#).

Overlaid on the bottom right of the browser window is a small dialog box with the title "O site 'http://localhost:82' diz:". The dialog box contains a question mark icon and the text "Deseja excluir este cadastro?". There are two buttons at the bottom: "OK" and "Cancelar".



Com isso finalizamos um primeiro exemplo, apenas como observação a ideia aqui foi mostrar a parte lógica de como funciona, por isso abrimos mão da parte de design.

Com este exemplo finalizamos nosso estudo em PHP, existem inúmeros tipos de aplicação que podemos fazer com o PHP.