



**GOVERNO DO  
ESTADO DO CEARÁ**

*Secretaria da Educação*

**ESCOLA ESTADUAL DE  
EDUCAÇÃO PROFISSIONAL - EEEP**  
**ENSINO MÉDIO INTEGRADO À EDUCAÇÃO PROFISSIONAL**

**CURSO TÉCNICO DE INFORMÁTICA**

**PHP/MySQL**





# GOVERNO DO ESTADO DO CEARÁ

*Secretaria da Educação*

**Governador**  
Cid Ferreira Gomes

**Vice Governador**  
Domingos Gomes de Aguiar Filho

**Secretária da Educação**  
Maria Izolda Cela de Arruda Coelho

**Secretário Adjunto**  
Maurício Holanda Maia

**Secretário Executivo**  
Antônio Idilvan de Lima Alencar

**Assessora Institucional do Gabinete da Seduc**  
Cristiane Carvalho Holanda

**Coordenadora da Educação Profissional – SEDUC**  
Andréa Araújo Rocha





# GOVERNO DO ESTADO DO CEARÁ

*Secretaria da Educação*

## Coordenação Técnica Pedagógica

Renanh Gonçalves de Araújo

### Equipe de Elaboração

Adriano Gomes da Silva  
Cíntia Reis de Oliveira  
Fernanda Vieira Ribeiro  
Francisco Aislan da Silva Freitas  
João Paulo de Oliveira Lima  
Liane Coe Girão Cartaxo  
Mirna Geyla Lopes Brandão  
Moribe Gomes de Alcântara  
Niltemberg Oliveira Carvalho  
Paulo Ricardo do Nascimento Lima  
Renanh Gonçalves de Araújo  
Renato William Rodrigues de Souza

### Colaboradores

Maria Analice de Araújo Albuquerque  
Maria Danielle Araújo Mota  
Sara Maria Rodrigues Ferreira Feitosa



# **PHP/MYSQL**

**MANUAL DO (A) ALUNO (A)**

**Agosto/ 2013  
FORTALEZA/CEARÁ**

## Sumário

Apresentação .....	4
1. Formulários e listagens .....	5
1.1. Inputs .....	5
1.1.1. Checkbox .....	8
1.1.2. Radio .....	9
Exercícios Práticos .....	11
Desafio .....	11
1.2. Select .....	11
1.3. Tabelas .....	12
1.4. Listas .....	13
Exercício de Aprendizagem .....	14
1.1. Formulários .....	14
1.1.1. Formulário de cadastro .....	15
1.1.2. Listar dados de cadastro .....	15
1.1.3. Remover elementos da lista .....	17
Exercícios Práticos .....	18
2. Funções em PHP .....	19
2.1. Escrevendo Funções .....	20
2.2. Require e Include .....	21
2.3. Função date( ) .....	21
3. Manipulação de arquivos e diretórios .....	23
3.1. Trabalhando com Arquivos .....	23
3.2. Abrir e Fechar um Arquivo .....	25
Exercícios Práticos .....	25
3.3. Ler a partir de um Arquivo .....	25
3.4. Escrevendo uma String em um Arquivo .....	27
3.5. Upload .....	28
3.6. Download .....	30
Exercícios Práticos .....	31
4. Rede de Comunicações .....	32

4.1.	DNS, Serviços e Servidores.....	32
4.2.	Mail.....	36
4.5.	Diretrizes de Configuração .....	37
4.6.	Enviando Email Usando um Script PHP .....	38
5.	Manipulação de dados .....	40
5.1.	Projeto Vídeo Locadora.....	40
	Exercício Prático .....	42
5.2.	Acesso nativo .....	43
5.3.	Interface do phpmyadmin .....	45
	Exercícios Práticos .....	46
5.4.	Tratamento de erros .....	47
5.5.	Sessão.....	49
	Exercício de Aprendizagem .....	51
5.6.	CRUD.....	56
5.6.1.	Inserção.....	56
	Exercício Prático .....	59
5.6.2.	Alteração .....	59
5.6.3.	Exclusão .....	64
	Exercício Prático .....	68
5.6.4.	Usando SQL no PHP.....	70
5.6.5.	Consulta .....	76
5.7.	Relatórios.....	79
6.	Frameworks para desenvolvimento em PHP .....	85
6.1.	Soluções de framework do PHP .....	85
6.2.	O CakePHP Framework .....	86
6.3.	O Solar Framework.....	87
6.4.	O symfony Framework .....	89
6.5.	O Zend Framework.....	90
	Referências Bibliográficas.....	92
	Índice de Tabelas .....	93

# Apresentação

O manual apresenta uma serie de práticas dividido em seis fases, inicialmente trabalhamos com a união de conhecimentos adquiridos em disciplinas anteriores, e vamos aprofundando o conhecimento da linguagem PHP e a integração com o banco de dados.

Abaixo a divisão das fases

1. Formulários e listagens.
  - a. Trabalhamos com a integração da linguagem PHP com os formulários, criando exemplos dinâmicos de formulários com o uso de arrays.
2. Funções em PHP.
  - a. Iremos conhecer algumas funções próprias do PHP que podem ser úteis em nossa vida profissional como desenvolvedor web.
3. Manipulação de arquivos e diretórios.
  - a. Estudaremos a manipulação de arquivos no servidor web, bem como construir scripts para realizar o upload de arquivos entre outros.
4. Rede de Comunicações.
  - a. Conheceremos recursos e serviços que podemos trabalhar com nosso servidor, iremos trabalhar com scripts de envio de email.
5. Manipulação de dados.
  - a. Este é um dos grandes pontos deste material, iremos realizar várias práticas de integração com o banco de dados MySQL, aprender a criar um CRUD.
6. Frameworks para desenvolvimento em PHP.
  - a. Estudaremos alguns frameworks utilizados em PHP para aperfeiçoar o trabalho do programador web.

Elaborado no intuito de qualificar o processo de formação, este Manual é um instrumento que se constitui como um mediador para facilitar o processo de ensino-aprendizagem em sala de aula.

É importante que o (a) aluno (a) compreenda o propósito do método do curso, e assim, se aproprie do conteúdo e da metodologia proposta por meio das atividades, *Esperamos contribuir com a consolidação do compromisso e envolvimento de todos (professores e alunos) na formação desses profissionais.*

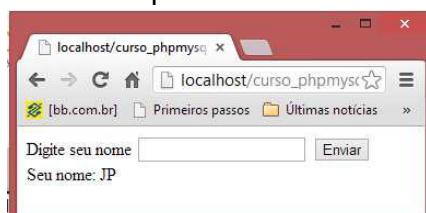
# Formulários e listagens

Nesta aula vamos codificar formulários HTML, realizando o processamento dos inputs do usuário através do PHP, estes processamentos serão feitos agora no lado servidor (Server-side) nas disciplinas passadas trabalhamos com formulários web, folhas de estilos CSS, JavaScript e um pouco da sintaxe PHP.

Nesta Fase vamos praticar a construção de páginas com formulários, sendo tratados no lado servidor trabalhando com requisições do usuário.

## 1.1. Inputs.

Nesta aula vamos trabalhar com o elemento input e com seus atributos, sendo processados com scripts PHP, Vamos começar com uma situação bem simples, precisamos criar um formulário que receba o nome de uma pessoa e mostre-o abaixo. Como na imagem abaixo.



Vamos executar esta tarefa de duas formas uma chamando a página **form\_inputs.php** e outra através de uma variável global.

Vamos criar um formulário como mostrado na imagem acima, o nome do arquivo PHP será **form\_inputs.php**.

Agora vamos à codificação.

```

12<form method="POST" action="form_inputs.php">
13    <label>Digite seu nome</label>
14    <input type="text" id="tx_nome" name="tx_nome" value="" />
15    <input type="submit" value="Enviar" name="enviar" />
16    <?php
17        $qtdde_lista = $_POST['tx_nome'];
18        echo '<br>Seu nome: ', $qtdde_lista, '</br>';
19    ?>
20    </form>
21</body>
```

Neste exemplo na linha 16 a 19 tenho o código em PHP, onde é definido que a variável **\$qtdde\_lista** vai receber uma variável do método post [**'tx\_nome'**], este valor é recebido do elemento input o qual tem o nome e id **tx\_nome**, quando formulário é submetido pelo post este valor é armazenado na variável **\$qtdde\_lista**, depois exibido na página pelo **echo** na linha 18. Observe que o action do formulário vai chamar ele mesmo, ou seja, tem o mesmo nome do seu arquivo PHP criado no inicio desta aula, desta forma fazemos a chamada dele mesmo, tudo que estiver dentro do bloco PHP será executado.

Agora vamos modificar este exemplo, usaremos agora uma variável global já definida no PHP, mais será que é possível?

Em situações que se estiver postando dados de volta ao mesmo script ou formulário que ele originou, podemos utilizar a variável superglobal **<?php echo \$\_SERVER['PHP\_SELF'];?>** o script PHP será automaticamente designado a esta variável.

Vamos ao exemplo;

```

11<body>
12  <form method="POST" action=<?php echo $_SERVER['PHP_SELF'];?> ">
13    <label>Digite seu nome</label>
14    <input type="text" id="tx_nome" name="tx_nome" value="" />
15    <input type="submit" value="Enviar" name="enviar" />
16  <?php
17    $qtda_lista = $_POST['tx_nome'];
18    echo '<br>Seu nome: ', $qtda_lista, '<br>';
19  ?>
20  </form>
21</body>

```

Nas situações acima é uma boa prática utilizar-se de instruções condicionais para especificar o que será executado, caso não tenha ou não necessite tudo que estiver no script será executado.

### Variáveis superglobais

Há várias variáveis pré-definidas no PHP são chamadas de "**superglobais**", que significa que elas estão disponíveis em todos os escopos para todo o script. Não há necessidade de fazer global **\$variavel**; para acessá-la dentro de funções ou métodos.

Abaixo temos uma tabela com as variáveis superglobais:

Variáveis	Descrição
<b>\$GLOBALS</b>	Um array associativo contendo referências para todas as variáveis que estão atualmente definidas no escopo global do script. O nome das variáveis são chaves do array.
<b>\$_SERVER</b>	<b>\$_SERVER</b> é um array contendo informação como cabeçalhos, paths(caminhos), e localizações do script. As entradas neste array são criadas pelo servidor web. Neste link temos a lista completa de variáveis reservadas usadas em PHP <a href="http://www.php.net/manual/pt_BR/reserved.variables.server.php">http://www.php.net/manual/pt_BR/reserved.variables.server.php</a>
<b>\$_GET</b>	Um array associativo de variáveis passadas para o script atual via o método HTTP GET.
<b>\$_POST</b>	Um array associativo de variáveis passado para o script atual via método HTTP POST.
<b>\$_FILES</b>	Um array associativo de itens enviado através do script atual via o método HTTP POST, utilizado em arquivos de variáveis para Upload.
<b>\$_COOKIE</b>	Um array associativo de variáveis passadas para o atual script via HTTP Cookies <sup>1</sup> .
<b>\$_SESSION</b>	Um array associativo contendo variáveis de sessão disponíveis para o atual script. Neste manual nas próximas aulas entraremos com mais detalhes sobre esta variável global.
<b>\$_REQUEST</b>	Um array associativo que por padrão contém informações de <b>\$_GET</b> , <b>\$_POST</b> and <b>\$_COOKIE</b> .
<b>\$_ENV</b>	Um array associativo de variáveis passadas para o script atual via o método do ambiente. Estas variáveis são importadas para o PHP do ambiente sob o qual o <b>parser</b> do PHP é executado.

Tabela 1 - Variáveis Globais – Fonte: [http://php.net/manual/pt\\_BR/language.variables.superglobals.php](http://php.net/manual/pt_BR/language.variables.superglobals.php)

<sup>1</sup> É um grupo de dados trocados entre o navegador e o servidor de páginas.

As variáveis globais acima citadas serão utilizadas nas próximas aulas, onde veremos exemplos de como podemos trabalhar com elas.

Neste exemplo vamos mostrar o valor de uma variável PHP sendo mostrado dentro do input, para revisar vamos utilizar uma situação problema que envolve a utilização de uma estrutura de decisão.

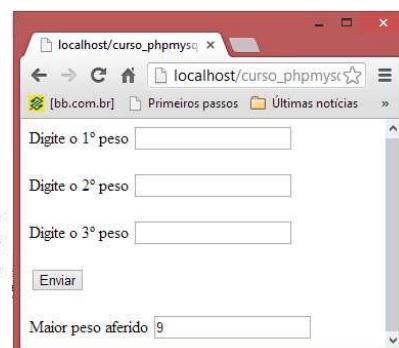
**2º Exemplo:** Criar um formulário que efetue a entrada dos valores de medida de três pesos aferidos de forma aleatória. No formulário deve ser mostrado em um input bloqueado com o maior peso aferido.

### Entendendo o problema:

- Para realizarmos esta tarefa vamos criar três variáveis referentes aos valores que serão recebidos pelos pesos aferidos.
- Será necessário utilizarmos uma estrutura de decisão neste exemplo o **IF**.

```

24 <?php
25 $maior_peso;
26 $peso1 = $_POST['field_pes01'];
27 $peso2 = $_POST['field_pes02'];
28 $peso3 = $_POST['field_pes03'];
29 $maior_peso = $peso1;
30 if ($maior_peso < $peso2) {
31     $maior_peso = $peso2;
32 }
33 if ($maior_peso < $peso3) {
34     $maior_peso = $peso3;
35 }
36 ?>
```



- No input onde receberá o valor iremos colocar o código PHP no atributo value (**value="<?php echo \$maior\_peso; ?>"**).

Agora vamos ver como ficou o formulário completo.

```

11 <body>
12     <form method="POST" action="form_inputs_2.php">
13         <label>Digite o 1º peso</label>
14         <input type="text" id="field_pes01" name="field_pes01" value="" />
15         <br></br>
16         <label>Digite o 2º peso</label>
17         <input type="text" id="field_pes02" name="field_pes02" value="" />
18         <br></br>
19         <label>Digite o 3º peso</label>
20         <input type="text" id="field_pes03" name="field_pes03" value="" />
21         <br></br>
22         <input type="submit" value="Enviar" name="enviar" />
23         <br></br>
24     <?php
25     $maior_peso;
26     $peso1 = $_POST['field_pes01'];
27     $peso2 = $_POST['field_pes02'];
28     $peso3 = $_POST['field_pes03'];
29     $maior_peso = $peso1;
30     if ($maior_peso < $peso2) {
31         $maior_peso = $peso2;
32     }
33     if ($maior_peso < $peso3) {
34         $maior_peso = $peso3;
35     }
36     ?>
37
38     <label>Maior peso aferido</label>
39     <input type="text" id="resultado_peso" name="resultado_peso" readonly="true" value="<?php echo $maior_peso; ?>"/>
```

### 1.1.1. Checkbox.

Neste exemplo vamos trabalhar com o input do tipo checkbox, iremos criar um formulário com opções de atividades esportivas, o usuário seleciona suas atividades, que serão mostradas em outra página.

Em PHP iremos criar um array para armazenar as atividades esportivas, iremos criar um checkbox e usaremos a estrutura de repetição **foreach**, para criar a quantidade de checkbox correspondente ao array, logo se no array tivermos seis atividades serão criados seis checkboxes.

Este será o resultado final do nosso formulário, agora vamos à construção do mesmo.

Na estrutura do código abaixo temos:

- Uma página chamada **form\_chkbox.php** e um formulário chamado **form\_list\_ativ**.
- Teremos que criar uma página em PHP que será chamada de **sel\_atividades.php**, nela será mostrado o resultado das preferências selecionadas.

```

11 <body>
12   <form name="form_list_ativ" method="post" action="sel_atividades.php">
13     <?php
14       //criação do array com as atividades esportivas
15       $esportes[1] = "Futebol";
16       $esportes[2] = "Basquetebol";
17       $esportes[3] = "Esgirma";
18       $esportes[4] = "Ciclismo";
19       $esportes[5] = "Boxe";
20       $esportes[6] = "Natação";
21     ?>
22     <h1>Lista de Preferências</h1>
23     <h3> Marque as Atividades esportivas do seu interesse</h3>
24     <?php foreach ($esportes as $value) {
25
26       ?>
27       <input type="checkbox" name="ativ_esport[]" value="<?php echo $value;?>" />
28       <label><?php echo $value;?></label>
29       <?php
30     }
31   ?>
32   <p></p>
33   <input type="submit" value="Enviar" name="enviar" />
34   </form>
35 </body>
```

valores do array até que todos os itens tenham sido recuperados, ou outra condição interna seja encontrada.

Observe que a chave { do loop inicia na linha 24 e termina na linha 30, tudo que estiver dentro destas chaves será repetido até que sejam apresentados todos os valores do loop.

**Na linha 27:** Temos o input do tipo checkbox com o nome **ativ\_esport[ ]**, estes colchetes indicam que o mesmo receberá um array, no atributo **value** deste elemento inserimos o trecho PHP com a variável **\$value** que recebeu cada valor do array **\$esportes**, desta forma o valor de cada checkbox foi atribuído.

O elemento label é usado para exibir os valores do array na página.

No action do formulário chamamos a página **sel\_atividades.php**, com o código PHP mostrado abaixo.

```

12   <body>
13     <h1>Minhas Preferências</h1>
14     <?php
15       $lista_ativ = $_POST['ativ_esport'];
16       foreach ($lista_ativ as $value) {
17         echo "<p></p>", $value;
18       }
19     ?>
20   </body>
```

### Lista de Preferências

Marque as Atividades esportivas do seu interesse

Futebol  Basquetebol  Esgirma  Ciclismo  Boxe  Natação

Enviar

Na página inserimos um trecho PHP, no qual criamos um array chamado **\$esportes** nas linhas 13 a 21.

**Linha 24:** Usamos uma estrutura de repetição **foreach** nesta estrutura definimos dentro do parêntese o array **\$esportes** para **\$value**, desta forma tenho para cada chave um par de

Nesta página embutidos o código PHP, para mostrar as atividades selecionadas pelo usuário na página anterior.

**Linha 15:** Declaramos a variável **\$lista\_ativ** que recebe os valores da requisição post **\$\_POST['ativ\_esport']** ativ\_esport é o nome do checkbox criado na página anterior.

**Linha 16:** Através do **foreach** listamos os valores do array que o usuário selecionou.  
Ao lado temos o resultado da execução desta prática.

## Minhas Preferências

Basquetebol  
Esgrima  
Natação

### 1.1.2. Radio.

Nesta aula vamos trabalhar com o input do tipo radio, com ele criaremos um array de estado civil, onde enviaremos para outra página o código do estado civil, simulando o envio destas informações a um banco de dados.

Neste exercício vamos criar duas páginas uma será chamada de form\_radio.php e a outra result\_radio.php.

#### Na página form\_radio.php.

**1º passo:** Criaremos um array multidimensional, onde teremos em uma posição deste array outro array, no primeiro teremos a descrição do estado civil e no outro o código de identificação. Um array multidimensional pode ser entendido como uma matriz.

```

9   <title></title>
10  <?php
11    //criação do array multidimensional
12    $est_civil = array(
13      'Solteiro' => array('id' => '20'),
14      'Casado' => array('id' => '10'),
15      'Separado' => array('id' => '30'),
16      'Divorciado' => array('id' => '34'),
17      'Viúvo' => array('id' => '50')
18    );
19    ?>
20  </head>
```

**2º Passo:** Agora vamos montar a estrutura da página, exibindo os valores do array criado anteriormente, usaremos dois laços de repetição, um para mostrar os valores do array e outro para enviar o valor do id para outra página.

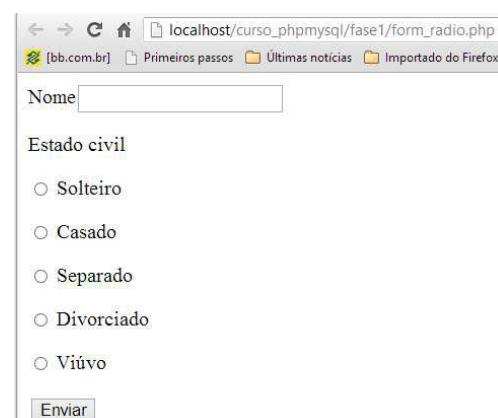
```

21 <body>
22     <form name="form_radio" method="post" action="result_radio.php">
23         <label>Nome</label><input type="text" name="nome" />
24         <p></p>
25         <label>Estado civil</label>
26         <p></p>
27         <?php
28             foreach ($est_civil as $value => $id_chave) {
29                 echo '<p></p>';
30             ?>
31             <input type = "radio" name = "radioEstC[]" value = "
32                 <?php foreach ($id_chave as $id => $v1) {
33                     echo $v1;
34                 ?>" />
35                 <label><?php echo $value; ?></label>
36                 <?php
37             }//Fim loop id_chave
38         } // Fim loop $est_civil
39     ?>
40     <p></p>
41     <input type="submit" value="Enviar" name="enviar" />
42
43 </body>

```

**Linha 28:** Por se tratar de um array dimensional é importante saber quantos níveis ele tem, neste exemplo temos dois, o primeiro nível temos os tipos estado civil e no segundo temos o ID correspondente, logo teremos que ter dois laços de repetição em cadeia, ou seja, um dentro do outro. Na declaração desta linha fazemos a interação do array **\$est\_civil** para cada interação **\$value** uma nova interação (**=> \$id\_chave**), neste exemplo o **\$value** armazena o tipo de estado civil (ex: casado, solteiro, divorciado...).

**Linha 31 e 32:** Nesta Linha temos o input do tipo radio com o nome **radioEstC[]**, no atributo **value** deste input, inserimos o outro loop que irá armazenar os ID's dos tipos de estado civil, agora temos a interação **\$id\_chave** para cada interação **\$id** uma nova interação (**=>\$v1**), esta por sua vez agora armazena os ID's dos tipos de estado civil. Na imagem ao lado vemos como ficará a página.



```

11 <body>
12     <h1>Resultado Estado civil</h1>
13     <a>Código estado civil: </a>
14     <?php
15         $idRadio = $_POST['radioEstC'];
16         $nomereresult = $_POST['nome'];
17
18         foreach ($idRadio as $valor) {
19             echo $valor;
20         }
21         echo '<p></p>Nome: ', $nomereresult;
22     ?>
23 </body>

```

### Na página result\_radio.php

**Linha 15:** Nesta linha recebemos o valor do input da página anterior através do método post.

**Linha 18:** Como o nosso array dimensional já foi devidamente tratado na página anterior, passando apenas o valor do campo ID para a variável **\$idRadio** fazendo as interações para **\$valor**, sendo assim exibido na página o ID do estado civil escolhido.



## Exercícios Práticos

- 1) A loja MOVEISVIP esta vendendo os seus produtos parcelados no cartão de credito, Faça um aplicativo web que receba o valor da compra e mostre o valor das parcelas, o usuário deverá escolher em quantas vezes que o parcelamento, que podem ser em 5x, 8x e 10x sem juros.
- 2) Crie um array multidimensional baseado na tabela abaixo.

Código da categoria	Nome da categoria
1	Produtos de Limpeza
2	Produtos Alimentícios
3	Produtos de consumo



### Desafio

1. Crie uma pagina web em PHP com um questionário de 10 perguntas, onde o usuário responderá as perguntas e no final deverá ser emitido o resultado dele, informar quantas perguntas ele acertou.

#### 1.2. Select.

Para exemplificar a utilização deste componente, criaremos uma página web onde o usuário deve escolher uma cidade para visitar, após a sua escolha será mostrada uma mensagem abaixo agradecendo a visita, como no exemplo abaixo.

The screenshot shows a browser window with the URL `localhost/curso_phpmysql/fase1/visitar_cidade.php`. A dropdown menu is open, showing options like 'Caucaia', 'Juazeiro do Norte', 'Maracanaú', etc. The option 'Caucaia' is selected. Below the form, a message reads 'Obrigador por visitar Pacajus'.

Vamos distribuir esta tarefa em passos.

**1º Passo:** Criar uma página chamada **form\_cidades.php**, nela vamos inserir o elemento **<select>** dentro de um form.

**2º Passo:** Criar um array usando a função **array()**, mostramos aqui uma outra forma de criar um array.

```

9 <title></title>
10 <?php
11 $cidades=array('Caucaia',
12     'Juazeiro do Norte',
13     'Maracanaú',
14     'Sobral',
15     'Crato',
16     'Itapipoca',
17     'Maranguape',
18     'Iguatu',
19     'Quixadá',
20     'Pacajus');
21 ?>
22 </head>
23 <body>
24     <form name="f_cidades" method="post" action="visitar_cidade.php">
25         Escolha uma cidade para visitar
26         <select name="lista_cidades[]>
27             <?php foreach ($cidades as $valor) { ?>
28                 <option value=<?php echo $valor; ?>> <?php echo $valor; ?></option>
29             </?php } ?>
30         </select>
31         <input type="submit" value="Visitar"/>
32     </form>
33 </body>

```

**3º Passo:** Vamos entender a estrutura desta página, o nosso **<select>** tem o nome de **lista\_cidades[]**, usamos o **foreach** para listar todos os valores existentes no array .

**Linha 24:** No **action** do form chamamos um script PHP que será criado nos próximos passos.

**Linha 28:** Nesta linha temos dois pontos importantes, o atributo **value** recebe o valor do array, isso é utilizado para passar o valor selecionado pelo usuário, o que esta entre o **<option>** **</option>** será mostrado ao usuário quando a página for carregada.

**4º Passo:** Agora vamos criar o script PHP chamado **visitar\_cidade.php**, ele será responsável por receber o valor selecionado pelo usuário na página anterior e exibir uma mensagem.

```

1 <?php
2 $cidade_vis = $_POST['lista_cidades'];
3 require 'form_cidades.php';
4 foreach ($cidade_vis as $vl_cidade) {
5     echo '<p></p>Obrigador por visitar ', $vl_cidade;
6 }
7 ?>

```

### 1.3. Tabelas.

Nesta aula usaremos um array multidimensional, criaremos uma página web com os cargos e salário em TI, que será mostrado em uma tabela, como na imagem abaixo.

Cargo	Salário
Líder de sistemas	8.155,00
Analista programador Java	3.900,00
Analista de testes	2.800,00
Analista de segurança da informação	8.155,00
Administrador de rede	4.480,00

Abaixo a estrutura da

página, onde temos uma tabela com os campos Cargo e Salário vindos do array.

```

20 <body>
21     <form name="form_tab" >
22         <h1>TABELA DE SALÁRIOS DE TI</h1>
23         <table border="1">
24             <thead>
25                 <tr>
26                     <th>Cargo</th>
27                     <th>Salário</th>
28                 </tr>
29             </thead>
30             <?php foreach ($cargos_salarios as $valor => $vl_sal) { ?>
31                 <tbody>
32                     <tr>
33                         <td><?php echo $valor; ?></td>
34                         <?php foreach ($vl_sal as $vlr => $vl_sals) { ?>
35                             <td><?php echo $vl_sals; ?></td>
36                         </tr>
37                     </tbody>
38                 <?php
39                     //Fim loop $vl_sal
40                     //Fim loop $cargos_salarios
41                 ?>
42             </table>
43         </form>
44     </body>

```

Este exemplo é bem parecido com o anterior, então podemos observar que a lógica de utilização dos arrays multidimensionais é a mesma, precisamos apenas saber onde queremos que as informações devam aparecer.

Criação do array multidimensional.

Criamos uma página PHP com o nome **form\_tabela.php**, e nela criaremos um array chamado **\$cargos\_salarios**, conforme abaixo;

```

<title></title>
<?php
$cargos_salarios = array(
    'Líder de sistemas' => array('sal' => '8.155,00'),
    'Analista programador Java' => array('sal' => '3.900,00'),
    'Analista de testes' => array('sal' => '2.800,00'),
    'Analista de segurança da informação' => array('sal' => '8.155,00'),
    'Administrador de rede' => array('sal' => '4.480,00')
);
?>
</head>

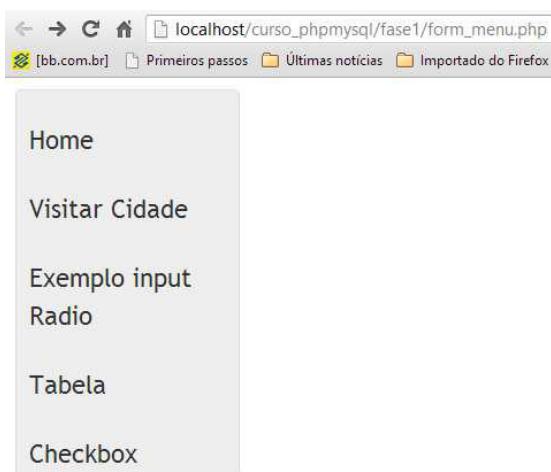
```

Neste exemplo fazemos a interação do array **\$cargos\_salarios** para cada interação **\$valor** uma nova interação ( $\Rightarrow \$vl\_sal$ ), onde **\$valor** recebe a descrição do cargo, no outro loop temos a interação de **\$vl\_sal** para cada interação **\$vlr** uma nova interação ( $\Rightarrow \$vl\_sals$ ) esta armazena o valor do salário de acordo com o cargo.

#### 1.4. Listas.

Neste exercício vamos criar um menu para acessar todos os exemplos anteriores, também usaremos um pouco de jquery ui.

Abaixo temos uma imagem de como ficará o nosso menu, este exercício vamos dividir em partes ou passos para melhor entendimento.



Para este exercício vamos criar em nosso projeto um diretório chamado jquery, nele vamos colocar os arquivos necessários jquery-1.8...js, jquery-ui.js e a folha de estilo jquery-ui.min.css.

Criaremos um array multidimensional para armazenar o texto do menu e o seu link.

Depois iremos incorporar o código PHP em uma lista, abaixo veremos os passos de forma detalhada.

**1º Passo:** Vamos inserir as tags script e chamar os scripts JavaScript como mostrados na imagem abaixo.

```

7 <head>
8     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
9     <title></title>
10    <script src="../jquery/jquery-1.8.3.min.js"></script>
11    <script src="../jquery/jquery-ui.js"></script>
12    <link href="../jquery/jquery-ui.min.css" rel="stylesheet">
13
14    <script>
15        $(function() {
16            $("#menu").menu();
17        });
18    </script>
19    <style>
20        .ui-menu { width: 150px; }
21    </style>

```

Nas linhas 14 a 18 Criamos uma função que chama outra função do jquery UI, onde a mesma defini que será aplicado no seletor css #menu.

Nas linhas 19 a 21 aplicamos o tamanho da área do menu.

**2º Passo:** Criação do array, abaixo criamos um array para armazenar o texto do menu e o link correspondente.

```

22 <?php
23 //array menu
24 $menu = array(
25     'Home' => array('link' => 'form_menu.php'),
26     'Visitar Cidade' => array('link' => 'form_cidades.php'),
27     'Exemplo input Radio' => array('link' => 'form_radio.php'),
28     'Tabela' => array('link' => 'form_tabela.php'),
29     'Checkbox' => array('link' => 'form_chkbox.php')
30 );
31 ?>

```

**3º Passo:** Na linha 39 iniciamos o laço de repetição para percorrer todos os elementos do array.

```

33 <body>
34
35     <form name="form" method="POST">
36         <ul id="menu"><!-- Início da lista onde deverá ser aplicado o efeito -->
37             <?php
38                 foreach ($menu as $valor => $link) {
39                     foreach ($link as $link_vl => $links) {
40                         ?
41                         <li><a href=<?php echo $links; ?>"><?php echo '<p></p>', $valor; ?></a></li>
42                     <?php
43                 //Fim do loop $link
44             //Fim do loop $menu
45         ?
46             </ul> <!-- Fim da lista -->
47         </form>
48     </body>

```

Como podemos observar na linha 41 que corresponde a tag **<li>**, adicionamos a tag **<a>** e nela incorporamos o código PHP, na variável **\$links** colocamos na propriedade **href**, o nosso link, a variável **\$valor** irá mostrar o texto do nosso link.



## Exercício de Aprendizagem

Estudo de caso da empresa de reserva de passagens aéreas Gato Ajato

### 1.1. Formulários.

Nesta aula vamos desenvolver formulários dinâmicos, em nossa problemática temos que desenvolver um web site, A empresa de reserva de passagens aéreas Gato Ajato foi quem solicitou o protótipo.

Foi solicitado aos projetistas que a web site efetue reserva de passagens aéreas da companhia.

Deverá cadastrar os números (ID) dos aviões e o número de lugares disponíveis em cada um. O home deverá mostrar o seguinte menu de opções:

1. Cadastrar aviões.
  - a. Com número de lugares disponíveis em cada avião.
2. Cadastrar Passageiros, com os campos nome, CPF, sexo, endereço, cidade.
3. Reservar passagem.

- a. Checar se no avião escolhido, ainda existe lugar disponível. Caso exista, o programa deverá diminuir o total de vagas e mostrar a mensagem *Reserva Confirmada*. Caso contrario, deve mostrara a mensagem *Voo lotado*.
4. Consultar avião.
5. Consultar passageiro.

### 1.1.1. Formulário de cadastro.

Nesta aula vamos praticar, desenvolvendo os formulários pedidos na problemática acima, o objetivo desta aula é dinamizar formulários utilizando os arrays e os elementos HTML.

Mão a obra, vamos montar o cadastro de passageiros, vamos agora iniciar a dinamizar um formulário, trabalhando com os recursos aprendidos das aulas anteriores.

Vamos observar as tarefas a serem cumpridas nesta aula.

A gato Ajato solicitou um web site, para este intuito devemos criar um layout para este projeto, e as páginas a seguir;

- Página index.php; com o menu usando jquery no menu será possível navegar nas outras páginas, conforme problemática apresentada anteriormente.
- Cadastro de Passageiros, com os campos nome, CPF, sexo, endereço, cidade.
- Cadastro de aviões, com os campos modelo, pais de origem, número de lugares.

### 1.1.2. Listar dados de cadastro.

Nesta aula vamos aprender a fazer uma busca em um array, com este conhecimento adquirido podemos cumprir as tarefas de consulta que necessitam o site em questão.

Abaixo temos a página de exemplo que será desenvolvida nesta aula.

Nome Passageiro
Ana Paula Mourão
Galberto Maia
Paulo Negreiros
Cassio Menezes
Paula Silva
Renata Ramos

Vamos criar uma página chamada de **consulta.php**, nela temos que criar um array com o nome de **\$passageiros** para realizar a busca pelo nome do passageiro.

```

8   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
9   <title>Passagens aéreas Gato Ajato</title>
10  <?php
11  $passageiros = array("Ana Paula Mourão", "Galberto Maia", "Paulo Negreiros",
12    "Cassio Menezes", "Paula Silva", "Renata Ramos");
13  ?>
14  </head>

```

No formulário no atributo **action** chamaremos a própria página de consulta.

```

15     <body>
16         <h1><a href="index.php">Passagens aéreas Gato Ajato - Consultar</a></h1>
17         <form name="form" method="POST" action="consulta.php">
18             <input type="text" name="busca" value="" />
19             <input type="submit" value="Pesquisar" name="enviar" />
20             <p></p>
21
22             <?php
23                 $buscar = $_POST['busca'];
24
25                 if ($buscar) {
26                     $cons_passageiros = preg_grep("/^$buscar/", $passageiros);
27                 }
28
29                 if ($buscar == NULL) {
30                     $cons_passageiros = $passageiros;
31                 }
32             ?>

```

**Linha 23:** Recebemos o valor do input busca através do método post, o valor que o usuário digitar será armazenado na variável **\$busca**.

**Linha 25:** Temos aqui uma instrução de decisão se a variável **\$busca** for verdadeira, ou seja conter um valor, a linha 26 é executada. Na linha 26 criamos a variável **\$cons\_passageiros**, esta recebe o valor da função **preg\_grep** que retorna as entradas do array **\$passageiros** que combinaram com um padrão repassado, “**"/^\$buscar/"**” neste exemplo usamos o que é conhecido como expressões regulares, onde se define um padrão a ser usado, então com o operador **^** indica que a expressão deve iniciar com a string dada, logo em nossa expressão será verificado os primeiros caracteres digitados no input de busca.

**Linha 29:** Nesta condição o valor do input for igual à **null** retorna todos os dados do array **\$passageiros**, onde os valores do array são armazenados em um outro array chamado **\$cons\_passageiros**.

Preenchendo os dados do array na tabela, vamos utilizar agora um velho conhecido das aulas anteriores o **foreach**, definiu-se na tabela o nome da coluna, e a linha onde devem aparecer os valores do array.

```

34         <table border="1">
35             <thead>
36                 <tr>
37                     <th>Nome Passageiro</th>
38                 </tr>
39             </thead>
40             <tbody>
41                 <?php
42                     foreach ($cons_passageiros as $valor) {
43                         ?>
44                         <tr>
45                             <td><?php echo $valor; ?></td>
46                         </tr>
47                         <?php } ?>
48                     </tbody>
49                 </table>
50             </form>
51         </body>

```

### 1.1.3. Remover elementos da lista.

Nesta aula vamos aprender a remover um elemento de um array, em nosso exemplo temos uma página chamada **remover.php**.

Vamos criar uma lista de modelos de aviões, o usuário quando de um clique no modelo o **id** que é a chave do array, será enviado através do método **get**, ao ser recebido o valor da chave será removido usando a função **unset()**.

```

11 <body>
12     <h2>Modelos de aviões</h2>
13     <?php
14         // Criando um array normal
15         $array = array("14Bis", "A330-300", "L-1049 - TAP", "ERK 190", "F16",
16             "Aeroflot Airbus A321",
17             "Airbus A320",
18             "Fed Ex ATR-72");
19         //Recebe a chave do array selecionado
20         $remov = $_GET['id'];
21
22         // Agora apagando o item do array,
23         foreach ($array as $i => $value) {
24             unset($array[$remov]);
25         }
26     ?>
27     <form name="form" method="get">
28         <?php
29             foreach ($array as $chave => $valor) {
30                 ?>
31                 <a href="remover.php?id=<?php echo $chave; ?>"> <?php echo '<p></p>', $valor; ?></a>
32             <?php }//Fim do loop do array que exibe os elementos ?>
33         </form>
34     </body>

```

**Nas linha 15 a18:** Temos a criação do nosso array, com os modelos de aviões.

**Linha 20:** Declaramos uma variável chamada **\$remov** que recebe um valor pelo método **get**.

**Nas linhas 21 a24:** Temos um **loop** onde para recuperar todos os valores do array, a função **unset()** recebe o array e a chave ou índice do array que será removido, este valor é recebido através da variável **\$remov**.

Para exibir na página criamos um novo **loop**, e neste definimos a variável que receberá a chave do array(**\$chave**), e a outra que será usada para exibir os modelos de aviões na página.

**Na linha 31:** Observamos que na tag **<a>** usamos o atributo **href** para passar o link da página que será chamada, no caso ela mesma e o parâmetro de URL chamado **id**, logo para passarmos parâmetros pelo URL utilizaremos esta sintaxe, nome da página (**remover.php**) mais o nome do parâmetro e o valor que será recebido (**?id=**).

#### Modelos de aviões





## Exercícios Práticos

- 1) Vamos criar um formulário web que para cadastrar uma lista de contatos, com os campos; nome, endereço, telefone, cidade.
- 2) Crie um formulário para cadastro de 15 palavras chaves, estas palavras devem ser armazenadas em um array e exibidas em outra página web, usando checkbox's.
- 3) Uma empresa prestadora de serviços tem uma lista de 10 serviços, onde desta lista um precisa ser removido, utilizando arrays e a função **unset()**, crie uma página para remover o serviço conforme a escolha do usuário.
- 4) Vamos criar um array com o modelo de aviões, e criar uma consulta de modelos de aviões.
- 5) Faça um web site que realize um cadastro de contas bancárias com as seguintes informações: número da conta, nome do cliente e saldo. O banco permitirá o cadastramento de contas e não poderá haver mais que uma conta com o mesmo número. Crie o menu de opções a seguir.

Menu de opções:

- Cadastro de contas de determinado cliente.
- Consultar contas de determinado cliente.
- Excluir um cliente.
- Sobre, com os dados dos desenvolvedores do projeto.

## 2. Funções em PHP

Nesta aula vamos conhecer algumas funções, que podem ser úteis em nossa caminhada como desenvolvedor PHP, abaixo uma relação de funções para arrays, que podemos utilizar como referência. No site da linguagem temos diversas referências sobre várias funções da linguagem, veja no link [http://www.php.net/manual/pt\\_BR/funcref.php](http://www.php.net/manual/pt_BR/funcref.php)

### Funções para Array

Função	Descrição
<b>array_change_key_case</b>	Modifica todas as chaves em um array
<b>array_chunk</b>	Divide um array em pedaços
<b>array_column</b>	Retorna os valores de uma única coluna no array de entrada
<b>array_combine</b>	Cria um array usando um array para chaves e outro para valores
<b>array_count_values</b>	Conta as frequências de cada valor de um array
<b>array_flip</b>	Inverte as relações entre chaves e valores
<b>array_key_exists</b>	Checa se uma chave ou índice existe em um array
<b>array_keys</b>	Retorna todas as chaves de um array
<b>array_merge</b>	Junta um ou mais arrays
<b>array_multisort</b>	Ordena múltiplos arrays ou arrays multidimensionais
<b>array_pop</b>	Retira um elemento do final do array
<b>array_product</b>	Calcula o produto dos valores de um array
<b>array_push</b>	Adiciona um ou mais elementos no final de um array
<b>array_rand</b>	Retorna um ou mais elementos aleatórios de um array
<b>array_replace</b>	Replaces elements from passed arrays into the first array
<b>array_reverse</b>	Retorna um array com os elementos na ordem inversa
<b>array_search</b>	Procura por um valor em um array e retorna sua chave correspondente caso seja encontrado
<b>array_shift</b>	Retira o primeiro elemento de um array
<b>array_slice</b>	Extrai uma parcela de um array
<b>array_splice</b>	Remove uma parcela do array e substitui com outros elementos
<b>array_sum</b>	Calcula a soma dos elementos de um array
<b>array_unique</b>	Remove valores duplicados de um array
<b>array_unshift</b>	Adiciona um ou mais elementos no início de um array
<b>array_values</b>	Retorna todos os valores de um array
<b>array</b>	Cria um array
<b>asort</b>	Ordena um array mantendo a associação entre índices e valores
<b>count</b>	Conta o número de elementos de uma variável, ou propriedades de um objeto.
<b>current</b>	Retorna o elemento corrente em um array
<b>each</b>	Retorna o par chave/valor corrente de um array e avança o seu cursor
<b>in_array</b>	Checa se um valor existe em um array

<b>key</b>	Retorna uma chave de um array
<b>ksort</b>	Ordena um array pelas chaves
<b>list</b>	Cria variáveis como se fossem arrays
<b>next</b>	Avança o ponteiro interno de um array
<b>prev</b>	Retrocede o ponteiro interno de um array
<b>range</b>	Cria um array contendo uma faixa de elementos
<b>reset</b>	Faz o ponteiro interno de um array apontar para o seu primeiro elemento
<b>rsort</b>	Ordena um array em ordem decrescente
<b>sort</b>	Ordena um array

Tabela 2 - Funções para Array

## 2.1. Escrevendo Funções.

Vamos escrever algumas funções, primeiramente criaremos uma função para converter palavras em maiúscula ou minúscula, usaremos duas funções do PHP a **strtoupper()** para maiúscula e **strtolower()** para minúscula.

Vamos criar uma página e criar o formulário abaixo;

```

11 <body>
12   <form name="form" method="POST">
13     Texto <input type="text" name="tx" value="" />
14     <select name="lista_convert">
15       <option value="mai">Maiúscula</option>
16       <option value="min">Minúscula</option>
17     </select>
18     <p></p>
19     <input type="submit" value="Converter" />
20   </form>

```

Temos no **select** os valores que serão passados para a nossa função.

Abaixo temos o trecho de código PHP que pode ser utilizado na mesma página, ou em outra página.

```

22 <?php
23   $vlr = $_POST['lista_convert'];
24   $texto = $_POST['tx'];
25   //chamar função
26   converter($vlr, $texto);
27
28   function converter($valor, $txt) {
29     if ($valor == "mai") {
30       echo $tx = strtoupper($txt);
31     }
32     if ($valor == "min") {
33       echo $tx = strtolower($txt);
34     }
35   }
36 ?>

```

Inicialmente recebemos através do **post**, o valor de duas variáveis, a **\$vlr** recebe o valor escolhido pelo usuário se a opção de conversão dele é maiúscula ou minúscula. A outra variável **\$texto** recebe o texto digitado pelo usuário.

Na linha 26 chamamos a função e passamos dois parâmetros, um com o valor para ser comparado no **if**, e outra com o texto que será convertido, conforme a opção do usuário.

## Outras Funções

### Addslashes

**addslashes (string \$variavel)** - Retorna uma string com barras invertidas antes de caracteres que precisam ser escapados para serem escapados em query do banco de dados e etc. Estes caracteres são aspas simples ('), aspas duplas ("), barra invertida (\) e **NUL** (o byte NULL).

#### Exemplo:

```

11 <body>
12     <?php
13     $string = "Minha string com caracteres indevidos ' veja ''";
14 echo addslashes($string);
15 // retorno: Minha string com caracteres indevidos \' veja \\
16     ?>
17 </body>
```

### Stripslashes

**stripslashes (string \$variavel )** – Caso queira retirar caracteres indevidos utilizamos esta função, ele retira as \";

#### Exemplo:

```

11 <body>
12     <?php
13     $string = "Minha string com caracteres indevidos \'veja\'";
14 echo stripslashes($string);
15 // retorno: Minha string com caracteres indevidos ' veja '
16     ?>
17 </body>
```

## 2.2. Require e Include

**include()**: Esta função tenta incluir uma página. Em caso de algum erro, o script retorna um warning (aviso) e prossegue com a execução do script. Aceita a passagem de variáveis (**GET**) na string.

**Require()**: Tenta incluir uma página. Em caso de erro, o script retorna um **fatal error** (erro fatal) e aborta a execução do script. E não prossegue com a execução do script. Não aceita a passagem de variáveis (**GET**) na string.

**include\_once() e require\_once()**: Idênticas as suas funções simples, porém se o arquivo referenciado já foi incluso na página anteriormente, a função retorna ‘**false**’ e o arquivo não é incluído.

### 2.3. Função date( )

Nesta aula vamos trabalhar com uma função de manipulação de datas, com esta função podemos obter a data e hora do servidor onde o PHP esta rodando.

**Exemplo 01:** Neste exemplo temos a aplicação da função **date()**, o resultado será a data no formato **dd/mm/aaaa**, caso o meu objetivo fosse obter o ano por exemplo, basta usar **date("Y")**.

```

13      <?php
14      //Exibindo a data no formato 16/02/2004.
15      //FUNÇÃO DATE()
16      echo $data = date("d/m/Y");
17

```

**Exemplo 02:** O resultado do exemplo abaixo é este: **Hojé é July 23, 2013**, apresentando o mês por extenso.

```

20      <?php
21      echo 'Hojé é ' . date("F d, Y");
22
23

```

**Exemplo 03:** Verifica se o servidor esta em horário de verão ou normal.

```

25      <?php
26      //Verificar horário de verão
27      //FUNÇÃO DATE(i em maiúsculo)
28      $data = date("I");
29
30      //Exibindo O RESULTADO
31      echo $data ? "Horário de Verão" : "Horário Normal";
32

```

**Exemplo 04:** Exibir a data por extenso.

```

38      <?php
39      //Exibindo a data por extenso
40      //FUNÇÃO DATE()
41      $data = date("D, d de M de Y");
42

```

**Exemplo 05:** Exibindo a data e a hora no formato 27/05/2013 22:16:02

```

46      <?php
47      //Exibindo a data e a hora no formato 27/05/2013 22:16:02
48      //FUNÇÃO DATE()
49      $data = date("d/m/Y H:i:s ");
50

```

### 3. Manipulação de arquivos e diretórios

Nesta fase vamos aprender a manipular arquivos no servidor, desde a criação, leitura de arquivos, aprender um pouco sobre download e upload.

#### 3.1. Trabalhando com Arquivos

Vamos conhecer uma série de funções para manipulação de arquivos, abaixo temos uma tabela com algumas funções.

Função	Descrição
<code>fopen()</code>	Abre um arquivo para que possa ser manipulado.
<code>fgets()</code>	Pega uma linha do arquivo até o máximo de 1024 bytes.
<code>feof()</code>	Durante a leitura de um arquivo, avisa se chegou ao final.
<code>file_get_contents()</code>	Pega todo conteúdo do arquivo aberto como uma string.
<code>ftruncate()</code>	Reduz o tamanho do arquivo. Usado para apagar seu conteúdo.
<code>fwrite()</code>	Escreve no arquivo.
<code>unlink()</code>	Apaga o arquivo indicado.

Tabela 3 – Funções de manipulação de arquivos

Antes de qualquer coisa precisamos ter um arquivo para manipula, ou entender como cria-lo, para que depois do mesmo existir, podermos acrescentar conteúdo, abri-lo e excluí-lo.

No momento em que o recurso é aberto ou criado, precisamos definir o nível de acesso disponível para o recurso em questão, abaixo temos os modos de arquivos retirados da documentação do PHP.

Modo	Descrição
'w'	Abre o arquivo somente para a escrita, coloca o ponteiro do arquivo no começo do arquivo e diminui (trunca) o tamanho do arquivo para zero. Caso o arquivo não exista, tenta criá-lo.
'w+'	Abre o arquivo para leitura e escrita, coloca o ponteiro do arquivo no início e diminui (trunca) o tamanho do arquivo para zero. Se o arquivo não existe, tenta criá-lo.
'r'	Abre o arquivo somente para leitura, coloca o ponteiro de escrita no começo do arquivo. Caso o arquivo não exista retorna um erro e não tenta cria-lo.
'r+'	Abre para leitura e escrita, coloca o ponteiro de escrita no começo do arquivo. Retorna um erro caso o arquivo não exista e o mesmo não é criado.
'a'	Abre somente para escrita, coloca o ponteiro do arquivo no final. Se o arquivo não existir, tenta criá-lo.
'a+'	Abre o arquivo para leitura e escrita; coloca o ponteiro do arquivo no final. Se o arquivo não existir, tenta criá-lo.
'x'	Cria e abre o arquivo somente para escrita, coloca o ponteiro no início do arquivo. Se o arquivo já existe, a chamada <code>fopen()</code> irá falhar, retornando FALSE, gerando um erro nível E_WARNING. Se o arquivo não existe, tenta criá-lo.
'x+'	Cria e abre um arquivo para escrita e leitura; coloca o ponteiro do arquivo no início. Se o arquivo já existe, a chamada <code>fopen()</code> irá falhar, retornando FALSE, gerando um erro nível E_WARNING. Se o arquivo não existe, tenta criá-lo.

Tabela 4 – Modos de Arquivos

Agora que já entendemos como funciona o modo de arquivos, vamos criar um arquivo para ser manipulado em outras aulas. Utilizaremos a função **fopen()**.

Neste exemplo vamos criar um formulário, nele teremos um input para digitar o nome do arquivo, e um botão que o chamaremos de **criar arquivo**.

Podemos também criar um script em PHP, para fazer a criação do arquivo com as especificações, como nome e extensão do arquivo dentro do código.

Neste exemplo temos um script para criação do arquivo.

```

1 <?php
2 $arquivo = fopen('meutexto.txt', 'w');
3 if ($arquivo == false)
4     die('Não foi possível criar o arquivo.');
5 ?>

```

**Linha 2:** Definimos uma variável que vai receber o valor da função **fopen()**, que retornará **true** ou **false**, na função passamos dois parâmetros, o primeiro é o nome do arquivo e sua extensão, o segundo passamos o modo de operação do arquivo, que neste exemplo está abrindo o arquivo somente para escrita, caso não exista o cria.

**Nas linhas 3 e 4:** Se a variável **\$arquivo** retornar **false** a função **die()** é chamada para apresentar o erro.

Agora vamos para outro exemplo, agora usando mais recursos.

No exemplo abaixo, damos ao usuário a possibilidade de definir qual o nome do arquivo, para isso criamos uma função chamada **criar\_arquivo()**.

```

11 <body>
12     <form name="form" method="POST">
13         Digite o nome do arquivo texto do seu artigo
14         <input type="text" name="nome_arq" value="" />
15         <input type="submit" value="Criar arquivo" name="btn" onclick="<?php criar_arquivo() ?>" />
16     </form>
17     <?php
18
19     function criar_arquivo() {
20         $nome_arq = $_POST['nome_arq'];
21
22         if (isset($_POST['nome_arq'])) {
23             $arquivo = fopen($nome_arq.'.txt', 'w');
24         }
25         if ($arquivo == false)
26             die('Não foi possível criar o arquivo.');
27     }
28     ?>
29 </body>

```

**Linha 20:** Armazenamos o nome do arquivo na variável **\$nome\_arq**, que obteve o valor do input **nome\_arq** através do método **post**.

**Linha 22:** Criamos um **if** para que a partir daquele trecho só execute a linha 23, se a variável do **post** estiver preenchida, para isso usamos a função **isset()**.

**Linha 23:** Nesta linha pegamos o nome do arquivo armazenado na variável **\$nome\_arq**, usamos o ponto final para concatenar as strings (**juntar**), adicionado a extensão do arquivo, e depois passamos o segundo parâmetro do método que é o modo de operação do arquivo.  
Resultado final deste exemplo.

### 3.2. Abrir e Fechar um Arquivo

Agora nesta aula vamos aprender a abrir e fechar um arquivo, na aula anterior criamos um arquivo, neste momento vamos verificar se o arquivo pode ser aberto, em algumas circunstâncias um arquivo pode ser corrompido, se caso o arquivo esteja corrompido não será possível abri-lo.

No exemplo abaixo é semelhante ao anterior, porém neste usamos outro modo de operação “r”.

```

11 |     <body>
12 |         <?php
13 |             $arquivo = fopen('meutexto.txt', 'r');
14 |             if ($arquivo == false){
15 |                 die('Não foi possível abrir o arquivo.');
16 |             }
17 |
18 |             if($arquivo==true){
19 |                 echo 'Arquivo lido com sucesso!';
20 |                 //Fecha o arquivo
21 |                 fclose($arquivo);
22 |             }
23 |
24 |         ?>
25 |     </body>

```

Neste exemplo usamos uma ação para quando a condição for falsa, caso não seja possível abrir ou encontrar o arquivo. E **true** caso seja possível abrir o arquivo, exibindo uma mensagem ao usuário.

**Na linha 21:** Usamos a função **fclose()**, para fechar o arquivo se ele tiver sido aberto.



### Exercícios Práticos

- 1) Crie uma página para criar arquivos para artigos com a extensão **\*.html**, onde o usuário que deve especificar o nome do arquivo.
- 2) Crie uma função para verificar se o arquivo pode ser aberto.

### 3.3. Ler a partir de um Arquivo

Para realizar a leitura de um arquivo, podemos utilizar duas funções que veremos abaixo, outro detalhe importante é quando lemos um arquivo texto pelo PHP, ele é apresentado na estrutura HTML e ocorre um problema com os caracteres acentuados, apresentando caracteres indesejados.

Para resolver este problema, na página que iremos mostrar a leitura do arquivo, temos que o setar no cabeçalho da página PHP o **charset**, isso é necessário também quando tivermos trabalhando com banco de dados.

```

8  <?php header("Content-Type: text/html; charset=ISO-8859-1", true); ?>
9  <html>
10 <head>
11     <!-- Resolver Problemas na acentuação-->
12     <meta http-equiv="content-Type" content="text/html; charset=iso-8859-1" />

```

No inicio da página embutimos o código PHP acima na linha 8, onde estamos passando o **charset ISO-8859-1**.

**Na linha 12:** No HTML fazemos a alteração na tag meta, onde é inserido o mesmo **charset ISO-8859-1**.

A função **fgets()** pega uma linha do arquivo, no exemplo abaixo fazemos a leitura do arquivo **texto.txt**, o mesmo encontra-se no raiz do site por este motivo passamos seu nome direto, caso esteja dentro de algum outro diretório deverá ser informado.

```

28      <?php
29          //ler a primeira linha do arquivo
30          $arquivo = fopen('texto.txt', 'r');
31          if ($arquivo == false)
32              die('Não foi possível abrir o arquivo.');
33          $linha = fgets($arquivo);
34          echo $linha;
35          fclose($arquivo);
36      ?>

```

**Linha 30:** Nesta linha fazemos a leitura do arquivo.

**Linha 33:** Criamos a variável **\$linha**, onde ele recebe o valor da função **fgets()** que realizou a leitura do arquivo, pegando a primeira linha. Na linha 34 mostramos o que foi lido pela função.

A função **file\_get\_contents()** pega todo conteúdo do arquivo aberto como uma string, vamos fazer o exemplo abaixo.

Neste exemplo vamos fazer a leitura de um arquivo com a extensão HTML, o mesmo deve ter um conteúdo um texto, por exemplo, o conteúdo do arquivo será mostrado dentro de um **<textarea>**.

```

16 <body>
17     <h3>Conteúdo do Arquivo</h3>
18     <textarea name="textos" rows="10" cols="40">
19         <?php
20             //ler todas as linhas do arquivo
21             $arquivo = fopen('artigo.html', 'r');
22             if ($arquivo == false)
23                 die('Não foi possível abrir o arquivo.');
24             $string = file_get_contents('artigo.html');
25             echo $string;
26             fclose($arquivo);
27         ?>
28     </textarea>
29 </body>

```

Como podemos observar na imagem acima, é praticamente a mesma estrutura, o que muda é a função, que a mesma permite que seja lido todo o arquivo.

#### Conteúdo do Arquivo

```
Uma das habilidades mais
importantes de um Líder é a capacidade de
perceber e desenvolver o potencial das
pessoas. Todos nós temos limitações em
determinadas áreas, em contra-partida
somos muito bons em várias outras.
```

Ao lado temos o resultado desta aula, o arquivo HTML sendo apresentado em um <textarea>.

### 3.4. Escrevendo uma String em um Arquivo

Nesta aula vamos criar uma página para o usuário escrever um artigo, usaremos a função **fwrite()** para inserir um conteúdo no arquivo, e o modo de operação do arquivo “**W+**”, para se o arquivo não existir o mesmo ser criado.

```
11  <body>
12    <form name="form" method="post">
13      <h3>Escreva o texto do seu artigo</h3>
14      <p></p>
15      <textarea name="texto" rows="20" cols="60"></textarea>
16      <p></p>
17      <input type="submit" value="Salvar" onclick="<?php salvarArquivo() ?>" />
18    <?php
19    function salvarArquivo() {
20        $txt = $_POST['texto'];
21        $arquivo = fopen('meuartigo.html', 'w+');
22
23        if (isset($_POST['texto'])) {
24
25            if (!fwrite($arquivo, $txt))
26                die('Não foi possível atualizar o arquivo.');
27
28            fclose($arquivo);
29        } //Fim do IF
30    } //Fim da função
31    ?>
32    </form>
33  </body>
```

Vamos criar uma função que chamaremos de **salvarArquivo()**, ela será chamada no botão pelo evento **onclick**.

**Linha 21:** Aqui passamos o nome do arquivo e o modo de operação, com este modo de operação se o arquivo não existir o mesmo será criado.

**Linhas 23 e 25:** Nestas linhas temos duas estruturas condicionais, a primeira verifica se a variável do método **post** esta preenchida, caso ela esteja passa para a outra condição, o método **fwrite()** é usado para preencher um arquivo com o conteúdo de uma string, nesta condição temos, se a função não conseguir escrever no arquivo, e chamada a função **die()**, exibindo uma mensagem.

Na imagem abaixo temos como ficara a página, o usuário poderá escrever o seu conteúdo, que será armazenado no arquivo meuartigo.html no raiz do site.

The screenshot shows a web browser window with the URL `localhost/curso_phpmysql/fase3/escrever_arquivo.php`. The page contains a large text input area with the placeholder text "Escreva texto do seu artigo". Below the input area is a "Salvar" button.

### 3.5. Upload

Seja em sites de armazenamento online ou mesmo em perfis de redes sociais, o termo “**upload**” tem estado presente em boa parte do nosso cotidiano online. Nesta aula vamos entender de que se trata um upload, implementaremos um script de upload completo, com diversos recursos de tratamento.

Upload ou carregamento é a transferência de dados de um computador local para um servidor, o que pode ser feito pelo protocolo FTP ou pelo HTTP, em nosso exemplo vamos criar uma página para fazer o upload.

Vamos criar uma página que chamaremos de **form\_upload.php**, abaixo temos a imagem de como esta página será apresentada ao usuário.

The screenshot shows a web browser window with the URL `localhost/curso_phpmysql/fase3/`. The page title is "Upload de Arquivos". It features a file input field with the placeholder "Escolher arquivo" and the message "Nenhum arquivo selecionado". Below the input field is an "Enviar" button.

Na estrutura HTML vamos adicionar dois inputs, um com o **type file**, este é usado quando desejo carregar arquivos do navegador para o meu servidor, outro é o **submit** para enviar as informações do input file.

```

11 <body>
12   <form method="post" action="recebe_upload.php" enctype="multipart/form-data">
13     <h1>Upload de Arquivos</h1>
14     <input type="file" name="arquivo" />
15
16     <input type="submit" value="Enviar" />
17
18   </form>
19 </body>

```

Abaixo temos o script PHP para upload de arquivos, chamaremos este script de **recebe\_upload.php**, inicialmente no script abaixo temos um array com o nome de **`$_arq`**, onde passamos uma chave para este array e um conteúdo.

```

1 <?php
2 // Pasta onde o arquivo vai ser salvo
3 $_arq['pasta'] = 'upload/';
4 // Tamanho máximo do arquivo (em Bytes)
5 $_arq['tamanho'] = 1024 * 1024 * 2; // 2Mb
6
7 // Array com as extensões permitidas
8 $_arq['extensoes'] = array('jpg', 'png', 'gif');
9
10 // Renomeia o arquivo? (Se true, o arquivo será salvo como .jpg e um nome único)
11 $_arq['renomeia'] = false;
12 // Array com os tipos de erros de upload do PHP
13 $_arq['erros'][0] = 'Não houve erro';
14 $_arq['erros'][1] = 'O arquivo no upload é maior do que o limite do PHP';
15 $_arq['erros'][2] = 'O arquivo ultrapassa o limite de tamanho especificado no HTML';
16 $_arq['erros'][3] = 'O upload do arquivo foi feito parcialmente';
17 $_arq['erros'][4] = 'Não foi feito o upload do arquivo';
18
19 // Verifica se houve algum erro com o upload. Se sim, exibe a mensagem do erro
20 if ($_FILES['arquivo']['error'] != 0) {
21 die("Não foi possível fazer o upload, erro:<br />" . $_arq['erros'][$_FILES['arquivo']['error']]);
22 exit; // Para a execução do script
23 }

```

A chave ‘**pasta**’ definimos que pasta no servidor os arquivos serão armazenados, depois na chave ‘**tamanho**’ é definido o tamanho máximo do arquivo.

**Linha 8:** Nesta linha passamos para a chave ‘**extensoes**’ um array com as extensões permitidas.

**Linhas 13 a 17:** Criamos um array com os tipos de erros que podem ocorrer no upload.

**Linha 20 a 22:** Aqui temos uma estrutura condicional, que será verificado se ocorreu algum erro com o upload, se sim exibir mensagem de erro. Neste trecho usamos a variável global **`$_FILES`**, que trata especificamente de manipulação de arquivos entre o servidor e o navegador.

Continuando o script, na linha 25 temos a variável **`$extensao`**, onde ela vai receber a extensão do arquivo, com a função **`strtolower()`** o nome do arquivo que é recebido é transformado em minúsculo, depois entra em ação a função **`explode()`**, esta vai quebrar o nome do arquivo para extrair a extensão do mesmo. Caso o arquivo não tenha as extensões definidas no inicio do script, é exibida uma mensagem de erro.

```

25 // Faz a verificação da extensão do arquivo
26 $extensao = strtolower(end(explode('.', $_FILES['arquivo']['name'])));
27 if (array_search($extensao, $_arq['extensoes']) === false) {
28 echo "Por favor, envie arquivos com as seguintes extensões: jpg, png ou gif";
29 }
30 // Faz a verificação do tamanho do arquivo
31 else if ($_arq['tamanho'] < $_FILES['arquivo']['size']) {
32 echo "O arquivo enviado é muito grande, envie arquivos de até 2Mb.";
33 }
34 // O arquivo passou em todas as verificações, hora de tentar movê-lo para a pasta
35 else {
36 // Primeiro verifica se deve trocar o nome do arquivo
37 if ($_arq['renomeia'] == true) {
38 // Cria um nome baseado no UNIX TIMESTAMP atual e com extensão .jpg
39 $nome_final = time() . '.jpg';
40 } else {
41 // Mantém o nome original do arquivo
42 $nome_final = $_FILES['arquivo']['name'];
43 }

```

**Linha 31:** Tenho uma condição que verificar se o tamanho do arquivo corresponde pelo especificado na variável `$_arq['tamanho']`, a chave `size` do super global `$_FILE` retorna o tamanho do arquivo.

**Linhas 37 a 42:** Nestas linhas verificamos se é necessário mudar o nome do arquivo, caso não seja necessário, o nome do arquivo é mantido.

Continuando o script, agora já estamos no final do script, depois de passadas todas as verificações, é hora de mover o arquivo para o diretório especificado, para realizar esta tarefa é utilizado a função `move_uploaded_file()`, onde passamos o arquivo, um nome temporário caso seja necessário e o diretório onde o arquivo será armazenado.

```

44 // Depois verifica se é possível mover o arquivo para a pasta escolhida
45 if (move_uploaded_file($_FILES['arquivo']['tmp_name'], $_arq['pasta'] . $nome_final)) {
46 // Upload efetuado com sucesso, exibe uma mensagem e um link para o arquivo
47 echo "Upload efetuado com sucesso!";
48 echo "<br /><a href='".$arq['pasta'] . $nome_final . '">Clique aqui para acessar o arquivo</a>";
49 } else {
50 // Não foi possível fazer o upload, provavelmente a pasta está incorreta
51 echo "Não foi possível enviar o arquivo, tente novamente";
52 }
53 }
54 ?>

```

**Nas linhas 47 e 51:** São apresentadas algumas mensagens, também é criado um link para o usuário verificar o arquivo que foi enviado.

Upload efetuado com sucesso!  
[Clique aqui para acessar o arquivo](#)

### 3.6. Download

Agora vamos trabalhar com um velho conhecido de muitos, Download ou descarregar que significa baixar, em português, é a transferência de dados de um computador remoto (servidor) para um computador local.

Nesta aula vamos criar uma página de download, listando os arquivos de um determinado diretório.

Vamos criar uma página web chamada de `listar_arquivos.php`, abaixo temos como esta página ficará.

Lista de Arquivos do diretório 'arquivos':

```

Aide-32.png
arquivo.txt
arquivos.rar
Clip-Splitter-32.png
Mes-Videos-Bleu-32.png
Mov-32.png
movie Grev-32.png
Recycler-Bin-32.png
Scroll left-32.png
texto.txt
Trash White Empty-32.png

```

11	<body>
12	<?php
13	\$path = "arquivos/";
14	\$diretorio = dir(\$path);
15	
16	echo "Lista de Arquivos do diretório '<strong>\$path.</strong>' ";
17	while(\$arquivo = \$diretorio -> read()) {
18	echo "<a href='".\$path.\$arquivo."'>\$arquivo.</a> ";
19	}
20	\$diretorio -> close();
21	?>
22	</body>

Na body de nossa página vamos inserir o trecho de código PHP acima, na linha 13 definimos o caminho, ou seja, o diretório que será listado os arquivos, temos que criar um diretório chamado arquivos e colocarmos alguns arquivos nele.

**Linha 14:** Criamos uma variável para armazenar o conteúdo do diretório da variável `$path`.

**Linha 17:** Criamos um laço de repetição com o `while`, listando assim todos os arquivos encontrados no diretório especificado. Para cada arquivo encontrado é criando um link do mesmo, com a tag `<a>`.

**Na linha 20:** Após concluir a leitura do conteúdo do diretório especificado, é executado a função **close()**, encerrando a comunicação com aquele diretório.



## Exercícios Práticos

- 1) Crie uma pagina onde o usuário poderá escrever um texto e armazena-lo em um arquivo com a extensão HTML, o próprio usuário é quem irá definir o nome do seu arquivo, você deve criar no mínimo três arquivos com textos usando sua página deste exercício.
- 2) Crie três páginas onde será mostrado o texto dos artigos criados, então deveremos ter três arquivos com conteúdo gerado no exercício anterior e sendo exibidas em outras três páginas, uma página para cada arquivo.
- 3) Crie uma área de download para baixar os arquivos criados no exercício anterior.
- 4) Crie um novo script de upload a partir do que vimos na aula anterior, modifique o tamanho de arquivo para upload, os tipos de arquivos aceitos.

## 4. Rede de Comunicações

Nesta aula vamos conhecer algumas funções capazes de recuperar informações de servidores, aprenderemos a trabalhar com a função **mail()** do PHP e outros recursos de envio de e-mail. O conhecimento adquirido nesta aula poderá ser usado em outra necessidade para sistemas web, o objetivo desta aula é demonstrar algumas práticas com o uso do script PHP interagindo com recursos de redes de computadores.

Abaixo temos algumas funções utilizadas em PHP para interação de recursos e serviços de rede, das funções citadas abaixo utilizaremos algumas.

Função	Descrição
<b>checkdnsrr</b>	Verificar os registros de DNS correspondente a um determinado nome de host Internet ou endereço IP.
<b>dns_get_record</b>	Buscar DNS Resource Records associados a um hostname
<b>fsockopen</b>	Abre um socket de conexão Internet ou de domínio Unix
<b>gethostbyaddr</b>	Obtém nome do host de Internet correspondendo ao endereço de IP fornecido.
<b>gethostbyname</b>	Obter o endereço IPv4 correspondente a um determinado nome de host Internet
<b>gethostname</b>	Obtém o nome do host
<b>getprotobynumber</b>	Obter o número de protocolo associado com o nome de protocolo
<b>getprotobyname</b>	Obter nome do protocolo associado com o número de protocolo
<b>getservbyname</b>	Obter o número da porta associada a um serviço de Internet e protocolo
<b>getservbyport</b>	Obter serviço de Internet que corresponde a porta e o protocolo
<b>setcookie</b>	Envia um cookie
<b>setrawcookie</b>	Enviar um cookie sem url encoding o valor do cookie

### 4.1. DNS, Serviços e Servidores

**DNS** é um sistema de nome de domínio, é através dele que podemos usar os nomes de domínios, por exemplo, tenho uma locadora e desejo ter um nome na internet meu domínio então seria **sbfilmes.com.br**, o nome de domínio é mais amigável ao usuário, este domínio por exemplo corresponde a um endereço IP por exemplo **200.3.23.166** este sendo o IP do servidor que esta sendo acessado, que não é tão amigável como o nome do domínio.

Na disciplina de rede estudamos este assunto mais afundo então nesta disciplina é apenas para relembrar.

Vamos neste exercício usar a função **checkdnsrr()** para verificar a existência de registro DNS de um domínio.

Esta função vai retorna verdadeira quando o registro DNS for encontrado e falso caso não encontre o registro DNS solicitado.

Agora vamos à prática, vamos criar dois arquivos um como a imagem abaixo que chamaremos de **fase4\_dns.php**, onde o usuário deve digitar o nome do domínio que quer consultar.

Digite o um domínio para verificar

No formulário deste exemplo iremos chamar no **action="verific\_dns.php"**, este é o script PHP usado para verificar a existência do domínio através da função **checkdnsrr()**.

```

6 <html>
7   <head>
8     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
9     <title></title>
10    </head>
11    <body>
12      <form name="form" method="POST" action="verific_dns.php">
13        <label>Digite o um domínio para verificar</label>
14        <input type="text" id="dominio" name="dominio" value="" />
15        <input type="submit" value="Enviar" name="enviar" />
16      </form>
17    </body>
18  </html>

```

Agora na imagem abaixo vamos criar o script PHP para verificar o domínio.

```

1 <?php
2 $dns_dom = $_POST['dominio'];
3 $dominioexist = checkdnsrr($dns_dom, "ANY");
4
5 if($dominioexist){
6
7   require 'fase4_dns.php';
8   echo "<br>Este nome de domínio foi reservado</br>";
9
10 } else {
11   require 'fase4_dns.php';
12   echo "<br>O nome de domínio está disponível</br>";
13 }
14 ?>

```

**Na linha 2:** Pegamos o valor do input **dominio** através do método POST e armazenamos na variável **\$dns\_dom**.

**Na linha 3:** A variável **\$dominioexist** irá receber o valor verdadeiro ou falso da função **checkdnsrr()**, esta função recebe dois parâmetros o primeiro é o hostname que será verificado, o segundo é o tipo de registro usado, neste exemplo usaremos o **ANY** que procura por qualquer tipo de registro.

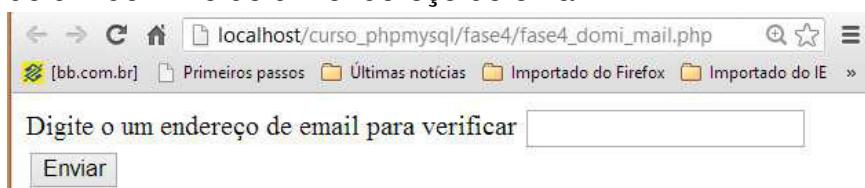
**Nas linhas 5 a 13:** Temos uma estrutura de decisão **if** onde passamos a variável **\$dominioexist**, que recebeu um valor boolean, sendo verdadeiro (**true**) redireciona para a página **fase4\_dns.php** e acrescenta a mensagem da linha 8, caso falso (**false**) o **else** (senão) é executado e redireciona para a página **fase4\_dns.php** e acrescenta a mensagem da linha 12.

Na imagem abaixo vemos o resultado final deste exercício.

Digite o um domínio para verificar

Este nome de domínio foi reservado

Agora vamos fazer mais um exercício, nesta situação problema precisa-se verificar a existência de um domínio de um endereço de email.



Na imagem acima vemos como ficará a página **fase4\_domi\_mail.php**, nesta caixa de texto iremos digitar o endereço de e-mail.

Temos dois pontos importantes nesta página, no action chamaremos o script PHP **verific\_domin\_mail.php**, e o input **dominio\_mail**, que receberá o endereço de email, para ser verificado.

```

6 <html>
7   <head>
8     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
9     <title></title>
10    </head>
11    <body>
12      <form name="form" method="POST" action="verific_domin_mail.php">
13        <label>Digite o um endereço de email para verificar</label>
14        <input type="text" id="dominio_mail" name="dominio_mail" value="" />
15        <input type="submit" value="Enviar" name="enviar" />
16
17      </form>
18    </body>
19  </html>

```

Para este exercício usaremos a função **explode()** que retorna um array contendo as partes da string com valores separados.

```

1 <?php
2 $email = $_POST['dominio_mail'];
3
4 $dominio_mail = explode("@", $email);
5
6 $validar = checkdnsrr($dominio_mail[1], "ANY");
7
8 if($validar){
9
10   require 'fase4_domi_mail.php';
11   echo "<br>Este email tem um dominio existente<br>";
12
13 }else {
14   require 'fase4_domi_mail.php';
15   echo "<br>Este e-mail tem um dominio inexistente<br>";
16 }
17
18 ?>

```

**Na linha 4:** A variável **\$dominio\_mail** recebe o nome do domínio obtido depois que a função **explode** dividiu as partes antes do @ e depois onde encontra-se o domínio.

**Na linha 6:** Aqui usamos a função para verificar a existência do domínio, esta verificação é somente do domínio, e não da existência do email.

**Nas linhas 8 e 16:** Nesta condição a variável **\$validar** retorna um valor **boolean**, sendo verdadeiro, o domínio existe a pagina que requisitou é chamada e uma mensagem é exibida. Caso seja falso a pagina que requisitou é chamada e uma mensagem é exibida.

## Serviços

A internet é composta de diversos serviços, que é através destes que essa plataforma de comunicação é definida, esses serviços que são utilizados diariamente em nossa vida online, são estes **HTTP, FTP, POP3, IMAP E SSH**, estes serviços trabalham com um protocolo, que é o conjunto de regras sobre o modo como se dará a comunicação entre as partes envolvidas.

Vamos criar um exemplo que verifica a conectividade com o servidor, pois todos esses serviços são inicializados no servidor.

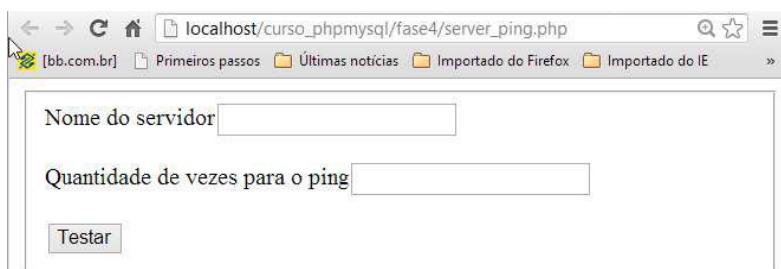
Ao lado o exemplo que será criado.

Vamos criar uma página PHP com o nome de `server_ping.php`, abaixo temos a estrutura da página.

```

6  <html>
7  	<head>
8    	<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
9    	<title></title>
10    </head>
11  	<body>
12    	<form name="form" method="POST" action="teste_ping.php">
13
14      <fieldset>
15        Nome do servidor<input type="text" name="host_ping" value="" />
16        <p></p>
17        Quantidade de vezes para o ping<input type="text" name="qtde_ping" value="" />
18        <p></p>
19        <input type="submit" value="Testar" />
20
21      </fieldset>
22    </form>
23  </body>
24 </html>

```



Agora vamos criar o script PHP, que irá executar o teste com o servidor.

```

1 <?php
2
3 $server = $_POST['host_ping'];
4 $qtde = $_POST['qtde_ping'];
5 require 'server_ping.php';
6
7 echo "<pre>";
8 //No linux
9 // system("/bin/ping -c $qtde $server");
10 //No windows
11 system("ping -n $qtde $server");
12 echo "</pre>";
13 // finalizar a tarefa
14 system("killall -q ping");
15

```

**\$server** o hostname do servidor ou o IP.

No caso do Linux devemos usar desta forma `system("/bin/ping -c $qtde $server");`

**Linha 3 e 4:** Recebemos através do método POST, o endereço do servidor que iremos realizar o teste, e a quantidade de vezes que irá pingar no servidor.

**Linha 11:** Usamos a função `system()`, que permite executar vários comandos no sistema operacional do servidor, de acordo com o sistema operacional do servidor o comando pode mudar um pouco. Neste exemplo usamos um servidor Windows, então passamos o comando ping -n, a variável `$qtde` defini a quantidade de vezes e a

## 4.2. Mail

Nos dias atuais uma característica muito forte das aplicações web é o poder de comunicação, uma das mais usadas é o envio de email, em todos os sites de empresas temos o formulário de contato, e em alguns deles são enviados email's para determinados departamentos.

Em PHP temos a função **mail()**, a sua sintaxe é bem básica, nesta função são passados três parâmetros, destinatário, assunto e o corpo da mensagem.

Abaixo temos um exemplo de envio de email bem simples.

```
19  <?php  
20  mail("jplimamaster@gmail.com", "assunto", "Este é o corpo da mensagem") ;  
21 ?>
```

Neste exemplo chamamos a função **mail()** e passamos os três parâmetros, isso é o suficiente para o envio de email, porém para testarmos este exemplo precisamos ter um servidor de email.

#### 4.5. Diretrizes de Configuração

A função **mail()** tem algumas diretrizes de configuração, o comportamento dessas funções podem ser modificado pelas configurações do **php.ini**, este é o arquivo de configuração do PHP.

Opções de configuração de e-mail			
Nome	Padrão	Modifica	Versão PHP
<b>mail.add_x_header</b>	"0"	PHP_INI_PERDIR	Disponível desde PHP 5.3.0.
<b>mail.log</b>	NULL	PHP_INI_PERDIR	Disponível desde PHP 5.3.0.
<b>SMTP</b>	"localhost"	PHP_INI_ALL	
<b>smtp_port</b>	"25"	PHP_INI_ALL	Disponível desde PHP 4.3.0.
<b>sendmail_from</b>	NULL	PHP_INI_ALL	
<b>sendmail_path</b>	"/usr/sbin/sendmail -t -i"	PHP_INI_SYSTEM	

**Tabela 5** - Opções de configuração de e-mail

Abaixo algumas configurações que podem ser feitas no **php.ini** com relação a envio de email's.

##### **mail.add\_x\_header**

Adicionar **X-PHP-Originário-Script** que irá incluir **UID** do script, seguido pelo nome do arquivo.

##### **mail.log**

O caminho para um arquivo de log, que irá registrar todas as chamadas da função **mail ()**. As entradas de log incluem o caminho completo do script, o número da linha, o endereço e cabeçalhos do email.

##### **SMTP**

Usado apenas no Windows:

Passando o nome do host ou endereço IP do servidor, usado para emails enviados com a função **mail()**.

##### **smtp\_port**

Usado apenas sob Windows:

Passando o número da porta para se conectar ao servidor especificado, com a configuração SMTP ao enviar e-mail com a função **mail ()**;

Porta padrão 25. Apenas disponível desde o PHP 4.3.0.

##### **sendmail\_from**

Configura o campo do cabeçalho **"From:"** o endereço de email que deve ser usado em emails enviados do PHP no Windows. Esta diretiva também define o **"Return-Path:"** cabeçalho (**header**).

##### **sendmail\_path**

configura o caminho para o programa sendmail, normalmente / usr / sbin / sendmail ou / usr / lib / sendmail (no Linux).

Esta diretiva também funciona no Windows. Se definido, **smtp\_port** e **sendmail\_from** são ignorados e o comando especificado é executado.

#### 4.6. Enviando Email Usando um Script PHP

Nesta aula vamos criar um script PHP para o envio de email, abaixo temos o formulário de que será criado.

E-mail

Assunto

Mensagem

Estrutura HTML, no action vamos chamar o script PHP **enviar\_email.php**.

```

11 <body>
12   <form name="form" method="POST" action="enviar_email.php">
13     <label>E-mail</label>
14     <input type="text" name="mail" id="mail" value="" />
15     <br></br>
16     <label>Assunto</label>
17     <input type="text" name="assunto" id="assunto" value="" />
18     <br></br>
19     <label>Mensagem</label>
20     <br></br>
21     <textarea name="mensagem" id="mensagem" rows="4" cols="20">
22       </textarea>
23     <br></br>
24     <input type="submit" value="Enviar" name="btn_enviar" />
25   </form>
26 </body>
```

Abaixo temos o script **enviar\_email**, temos três variáveis uma para o destinatário, outra para o assunto e uma para a mensagem, ambos dados vindos do formulário.

```

1 <?php
2 $destinatario = $_POST['mail'];
3 $assunto = $_POST['assunto'];
4 $mensagem = $_POST['mensagem'];
5
6 mail($destinatario, $assunto, $mensagem);
7
8 echo "E-mail enviado!";
9 echo $assunto;
10 echo "<a href=\"fase4_mail.php\">Voltar</a>";
11 ?>
```

**Linha 6:** Nesta linha chamamos a função **mail()**, passando os três parâmetros necessários.

Para realizarmos os testes de envio de email, podemos baixar alguns programas servidores de email, abaixo alguns disponíveis.

<http://www.baixaki.com.br/download/hmailserver.htm>

<http://ultradownloads.com.br/listagem/servidores-de-e-mail/13,687,1,,2,2,1.html>

Outra solução é configurar o localhost para envio de emails com WampServer.

<http://www.youtube.com/watch?v=P0KdnquezO0>

## 5. Manipulação de dados

Nesta aula vamos aprender a manipular dados com a linguagem de programação PHP, criaremos dois projetos, um será o produto da disciplina, neste poderemos aplicar todo conhecimento adquirido nas disciplinas de web, agora é hora de por em prática todos os fragmentos aprendidos desta disciplina e de outras relacionadas.

O outro projeto é para realizarmos as práticas em um contexto mais simples, para podermos aprender e depois aplicar no projeto que é o produto da disciplina.

### 5.1. Projeto Vídeo Locadora.

Aqui iniciamos o projeto web de uma vídeo locadora, em nosso estudo de caso vamos desenvolver um site para a **SBFilmes** uma empresa de locação de filmes, Neste sistema web será possível fazer a locações de filmes, reservas, terá um ambiente do cliente para ele acompanhar suas reservas e suas locações, este sistema é composto de uma área administrativa que será usada pelos funcionários da empresa.

Devido ao tempo não poderemos desenvolver este sistema por completo, utilizamos esta proposta para aprendizado, dessa forma iremos desenvolver algumas partes do sistema com o intuito didático. Com a experiência que adquirirmos neste projeto poderá ser aplicado em outros projetos de sistemas web.

Uma fase muito importante de um projeto seja qual for a plataforma web ou desktop, é o planejamento, para então experimentarmos esta experiência nesta disciplina, a proposta é desenvolver um projeto chamado de produto da disciplina, nesta aula propomos um projeto de vídeo locadora, nele mostraremos algumas práticas através de exercícios.

Mostraremos um exemplo de uma funcionalidade do site, e as outras semelhantes serão feitas através de exercícios.

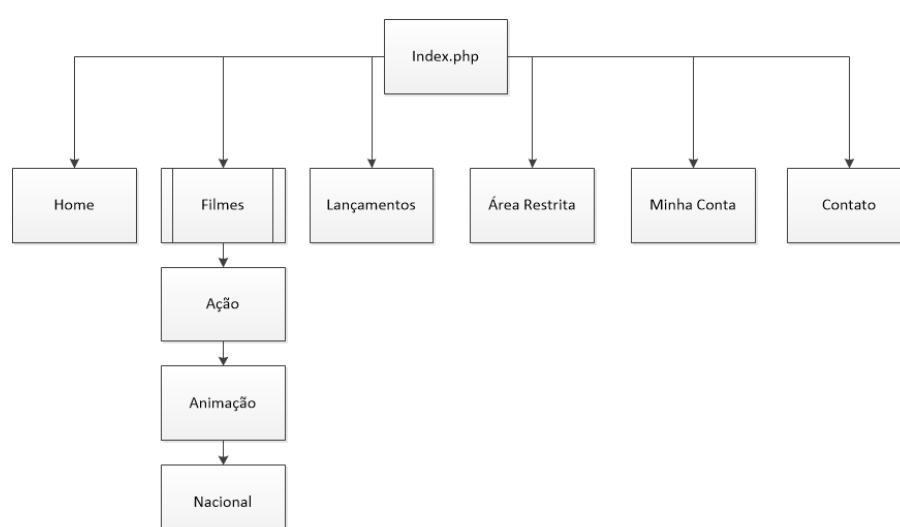
O template (modelo) deste projeto pode ser baixado neste link

<https://mega.co.nz/#!3U4QzKyQ!Q4SIWRR2rIYimthpZwCELHZLZ9xCyc1W79ABFw07rU>

Neste Link

<https://mega.co.nz/#!LVRzIR7D!ZxFPQhdGa9IYLWPwYY56V3whR3xe0Opqd2MeuQn9bqc>

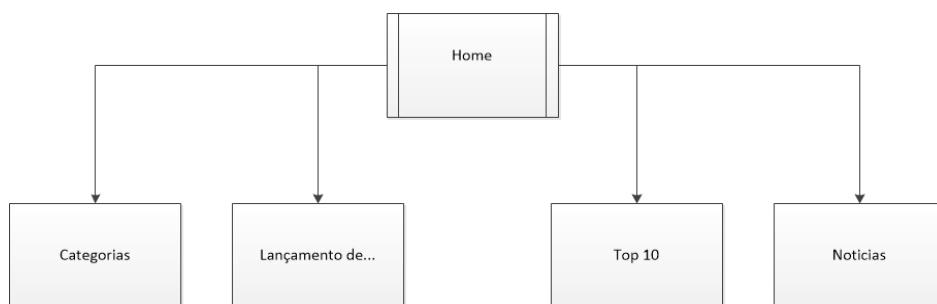
temos o script do banco de dados que será usado.



## O planejamento

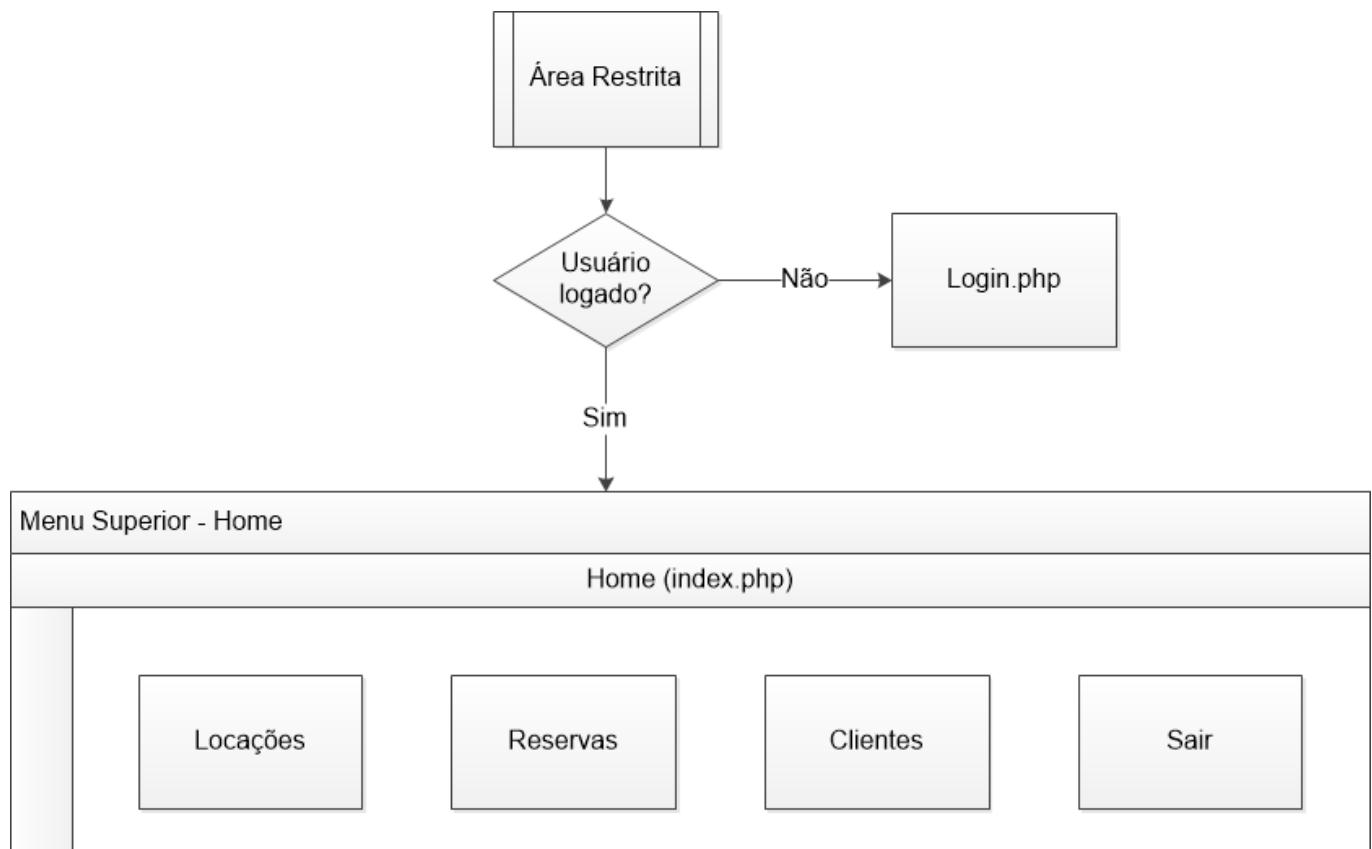
É sempre bom planejar o projeto de seu site, que é um rascunho do mesmo, Então o primeiro passo é identificar quais serão as páginas, link e recursos que deverão ter no site, em nosso exemplo acima definimos que páginas ou links deviam estar disponíveis na pagina inicial do site o nosso **index.php**, depois escolher a página e especificar que links ou que páginas estão disponíveis a partir dela, como podemos verificar no exemplo abaixo.

**Estrutura do site no Home**



Nesta estrutura temos o conteúdo **Lançamento de ...**, nesta área da página Home criaremos uma consulta para listar os lançamentos do mês vigente, mostrando o mês e os seus lançamentos.

Abaixo temos como irá funcionar o menu área restrita, este menu é dedicado ao administrador do sistema ou funcionário da empresa de locação de filmes, o usuário precisa se logar nesta parte do sistema, caso ele já esteja logado será direcionado para o home (**index.php**) caso contrário direcionado para a pagina de login de usuário.



### Exercício Prático

- 1) Para exercitarmos um pouco vamos planejar um site, vamos seguir os passos abaixo.
  - ✓ Escolher um tema para o seu site.
  - ✓ Criar o mapa do site, com as páginas previstas ou link e recursos para o site.
  - ✓ Desenhe o layout do seu site, da página principal **index** e outras páginas que tenham o layout diferente.
- 2) Crie o projeto SBfilmes na sua IDE de preferência, faça o download do template e script do banco de dados, execute o script do banco de dados.

## 5.2. Acesso nativo

O PHP pode se conectar a uma variedade de sistemas gerenciadores de banco de dados (SGBD) disponíveis. Para cada SGBD existe uma variedade de funções para realizar operações como conexão, consultas retorno, desconexão e outras. Nesta aula vamos conhecer algumas funções do SGBD Mysql, assim como uma lista de funções do mesmo.

Abaixo temos uma relação com alguns SGBD's e um link do site PHP com a lista completa de funções de acesso.

SGBD	Link do site
<b>MSQL (SQL Server)</b>	<a href="http://www.php.net/manual/pt_BR/book.msql.php">http://www.php.net/manual/pt_BR/book.msql.php</a>
<b>Firebird/InterBase</b>	<a href="http://www.php.net/manual/pt_BR/book.ibase.php">http://www.php.net/manual/pt_BR/book.ibase.php</a>
<b>MySQL</b>	<a href="http://www.php.net/manual/pt_BR/book.mysql.php">http://www.php.net/manual/pt_BR/book.mysql.php</a>
<b>Mysqli (Extensão MySQL Melhorada)</b>	<a href="http://www.php.net/manual/pt_BR/book.mysqli.php">http://www.php.net/manual/pt_BR/book.mysqli.php</a>
<b>ODBC</b>	<a href="http://www.php.net/manual/pt_BR/book.uodbc.php">http://www.php.net/manual/pt_BR/book.uodbc.php</a>
<b>Oracle (OCI7 e OCI8)</b>	<a href="http://www.php.net/manual/pt_BR/book.oci8.php">http://www.php.net/manual/pt_BR/book.oci8.php</a>
<b>PostgreSQL</b>	<a href="http://www.php.net/manual/pt_BR/book.pgsql.php">http://www.php.net/manual/pt_BR/book.pgsql.php</a>
<b>SQLite</b>	<a href="http://www.php.net/manual/pt_BR/book.sqlite.php">http://www.php.net/manual/pt_BR/book.sqlite.php</a>

Tabela 6 - Principais Bancos de Dados suportados em PHP - Fonte <http://www.php.net>

Agora vamos conhecer como podemos realizar o acesso a um SGBD, iremos criar uma base de dados para exercícios que serão realizados nas próximas aulas.

Abaixo temos as algumas funções de acesso ao SGBD Mysql.

Funções utilizadas para acesso e manipulação de dados.

Função	Descrição
<b>mysql_affected_rows</b>	Obtém o número de linhas atingidas na operação anterior do MySQL
<b>mysql_close</b>	Fechá a conexão MySQL
<b>mysql_connect</b>	Abre uma conexão com um servidor MySQL
<b>mysql_db_query</b>	Envia uma consulta MySQL
<b>mysql_drop_db</b>	Exclui um banco de dados MySQL
<b>mysql_error</b>	Retorna o texto da mensagem de erro da operação MySQL anterior
<b>mysql_escape_string</b>	Escapa uma string para usar em uma consulta MySQL
<b>mysql_fetch_array</b>	Obtém uma linha como uma matriz associativa, uma matriz numérica, ou ambas.
<b>mysql_fetch_assoc</b>	Obtém uma linha do resultado como uma matriz associativa
<b>mysql_fetch_field</b>	Obtém informações sobre uma coluna de um resultado e retorna como um objeto
<b>mysql_fetch_lengths</b>	Obtém o tamanho de cada saída no resultado
<b>mysql_fetch_object</b>	Obtém o resultado de uma linha como um objeto
<b>mysql_fetch_row</b>	Obtém uma linha como uma array numérica
<b>mysql_field_len</b>	Retorna o tamanho do campo especificado
<b>mysql_field_name</b>	Obtém o nome do campo especificado em um resultado

<b>mysql_field_table</b>	Obtém o nome da tabela na qual o campo esta especificado.
<b>mysql_field_type</b>	Obtém o tipo do campo especificado em um resultado
<b>mysql_free_result</b>	Libera um resultado da memória
<b>mysql_get_client_info</b>	Obtém informações do cliente MySQL.
<b>mysql_get_host_info</b>	Obtém informações do servidor MySQL.
<b>mysql_info</b>	Obtém informação sobre a consulta mais recente
<b>mysql_insert_id</b>	Obtém o ID gerado pela operação INSERT anterior
<b>mysql_list_dbs</b>	Lista os bancos de dados disponíveis em um servidor MySQL
<b>mysql_list_fields</b>	Lista os campos de uma tabela MySQL
<b>mysql_list_processes</b>	Lista os processos MySQL
<b>mysql_list_tables</b>	Lista as tabelas em um banco de dados MySQL
<b>mysql_num_fields</b>	Obtém o numero de campos em um resultado
<b>mysql_num_rows</b>	Obtém o número de linhas em um resultado
<b>mysql_pconnect</b>	Abre uma conexão persistente com um servidor MySQL
<b>mysql_ping</b>	Pinga uma conexão com o servidor ou reconecta se não houver conexão.
<b>mysql_query</b>	Envia uma consulta MySQL
<b>mysql_result</b>	Retorna dados do resultado.
<b>mysql_select_db</b>	Seleciona um banco de dados MySQL
<b>mysql_set_charset</b>	Define o cliente character set
<b>mysql_tablename</b>	Retorna o nome da tabela do campo

Tabela 7 - Funções PHP para SGBD MySql

### Criação do Script de criação do banco.

Neste exercício vamos criar um script em PHP para realizar a criação do banco de dados, que será utilizado na próxima aula e exercícios, chamaremos este script de **criarbd.php**.

```

1 <?php
2 $dbconn = mysql_connect("localhost", "root", "270184");
3 //Instrução SQL para criar um banco de dados MySQL
4 $sql="create database db_php_mysql";
5 mysql_query($sql,$dbconn);
6 echo '<p></p>Banco de dados criado com sucesso';
7
8 mysql_select_db("db_php_mysql",$dbconn);
9
10 echo '<p></p>Banco de dados selecionado: db_php_mysql';
11 //Instrução SQL para criar tabelas
12 $sql="CREATE TABLE IF NOT EXISTS t_cidade (
13     idt_cidade int(11) NOT NULL AUTO_INCREMENT,
14     nome_cidade varchar(100) DEFAULT NULL,
15     PRIMARY KEY (idt_cidade)
16 )";
17 mysql_query($sql,$dbconn);
18 $sql="CREATE TABLE IF NOT EXISTS t_cliente (
19     idt_cliente int(11) NOT NULL AUTO_INCREMENT,
20     nomeCli varchar(45) DEFAULT NULL,
21     PRIMARY KEY (idt_cliente)
22 )";
23 mysql_query($sql,$dbconn);
24 echo '<p></p>Tabelas criadas com sucesso';
25
26 //fecha a conexão aberta
27 mysql_close();
28 ?>

```

**Linha 2:** Nesta linha utilizamos a função **mysql\_connect()** e passamos os dados da conexão com o banco de dados, nesta ordem servidor (**localhost**), usuário do banco de dados (**root**), e a senha do banco.

**Linha 5:** Na linha anterior 4 criamos uma variável para receber a instrução SQL, e nesta linha usamos esta função para executar a instrução SQL, passamos dois parâmetros o primeiro a instrução SQL e o segundo os dados da conexão.

**Linha 8:** Esta função é utilizada para selecionar um determinado banco de dados, nela informo qual o nome do banco de dados e a variável **\$dbconn**

que contem os dados da conexão com o SGBD, depois na linha 12 passamos o novo valor para a variável **\$sql**.

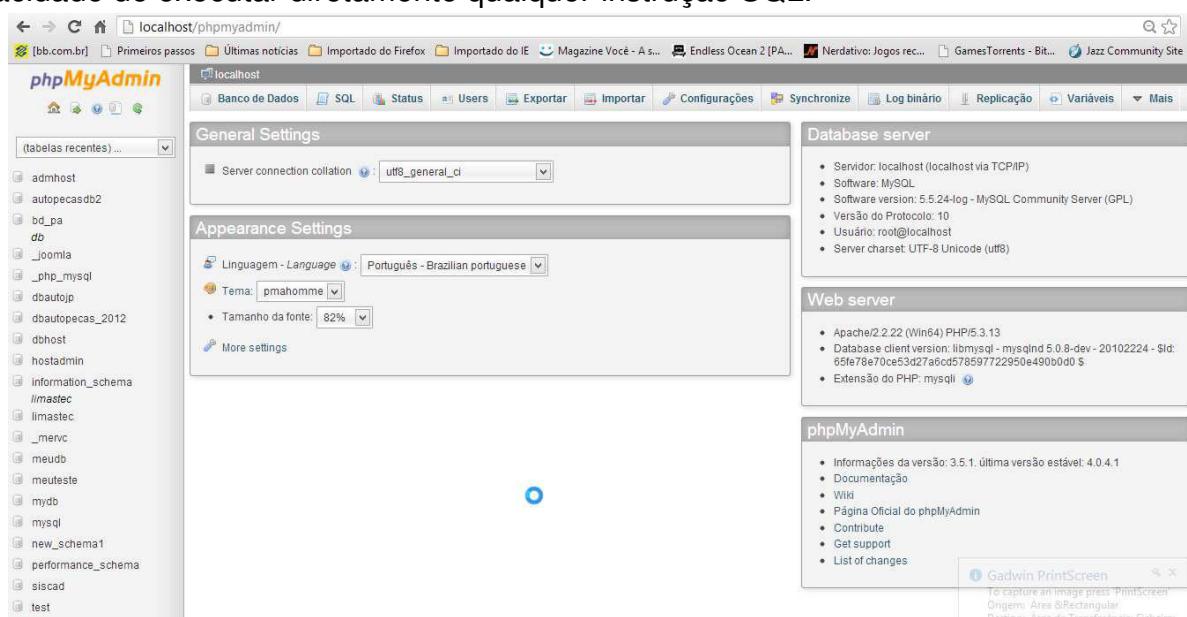
**Linha 27:** Esta função é utilizada para fechar a conexão aberta.

Este é um pequeno exemplo de um script de instalação de aplicativos web, com ele aprendemos que podemos realizar toda a criação de um banco de dados através de um script PHP.

### 5.3. Interface do phpmyadmin

O **phpMyAdmin** é uma ferramenta de software livre escrito em PHP, destinado a lidar com a administração do MySQL pela Internet.

Ele suporta varias operações do MySQL. Operações como (gerenciar bancos de dados, tabelas, colunas, relações, índices, usuários, permissões, etc.), e você ainda tem a capacidade de executar diretamente qualquer instrução SQL.



O phpMyAdmin esta incluso na instalação do **Wamp Server**, e pode ser feito o download no site <http://www.phpmyadmin.net>.



Agora com o projeto já criado na sua IDE e o banco de dados rodando, vamos criar um script para o projeto SBFilmes com os dados iniciais usuário e senha, estes dados são do usuário padrão do sistema.

Abaixo temos o script com o **insert** do usuário, nas aulas anteriores vimos como fazer a conexão e a criação do banco de dados e tabelas, neste exercício utilizará a instrução **insert** do SQL.

```

1 <?php
2 $hostname_db = "localhost";
3 $database_db = "sbfilmes";
4 $username_db = "root";
5 $password_db = "270184";
6
7 $dbconn = mysql_connect($hostname_db,$username_db,$password_db);
8
9 mysql_select_db($database_db, $dbconn);
10 //instrução sql para inserir dados na tabela
11 $sql="INSERT INTO funcionario VALUES(null,'Administrador','admin','123456')";
12 mysql_query($sql,$dbconn);
13 echo '<p></p>Configurações iniciais criadas';
14 ?>

```

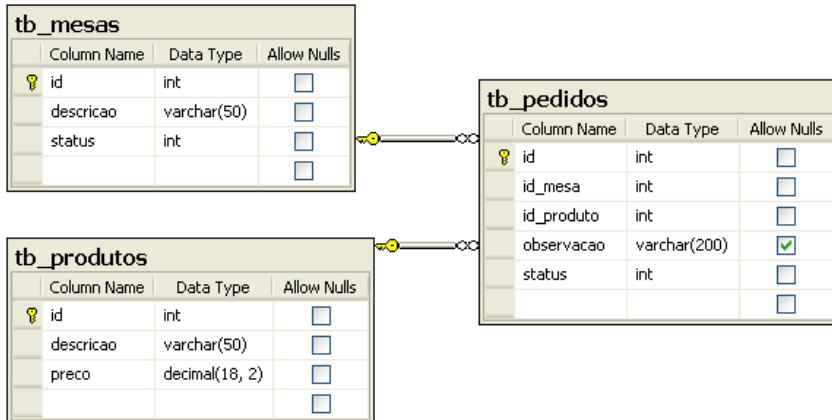
Crie um novo script PHP com o nome **insertuser.php**, insira o código acima e depois execute, pronto acabamos de inserir os dados de usuário e senha na base de dados sbfilmes.

Como podemos observar declaramos quatro variáveis com os dados da conexão, e inserimos a instrução SQL, com relação ao SQL vimos na disciplina de banco de dados.



## Exercícios Práticos

- 1) Crie um script de instalação para um determinado site, o nome da base de dados será **db\_restaurante**, abaixo o modelo de dados para a criação do banco, você deverá criar uma página nela deve existir um botão com o nome de instalar sistema, após o clique do usuário neste botão inicia a execução do script.



- 2) Crie um script que realize a inserção de cinco registros na tabela **tb\_produtos**.
- 3) Agora modifique o script da questão um para ele realizar o processo de instalação completo, o seu instalador agora terá que criar uma base de dados, criar as tabelas e inserir os registros na tabela.

## 5.4. Tratamento de erros

Tratamento de erros ou exceções é o nome que se dá a tarefa de elaborar rotinas que serão executadas sempre que um erro acontecer, durante a execução de um aplicativo,

É uma tarefa do desenvolvedor tentar imaginar em quais situações o aplicativo poderia passar por um erro, e prepara-lo para todas situações possíveis. Abaixo temos alguns métodos para serem usados no tratamento de exceções.

Método	Descrição
<code>getMessage()</code>	Retorna a mensagem de erro.
<code>getCode()</code>	Retorna o código de erro.
<code>getFile()</code>	Retorna o caminho do arquivo no qual o correu o erro.
<code>getLine()</code>	Retorna o array com a linha do erro ou as linhas com os erros.
<code>getTrace()</code>	Retorna as ações em forma de um array, consistido de informações pertinentes ao contexto em que o erro ocorreu.
<code>getTraceAsString()</code>	Retorna toda informação igual ao <code>getTrace()</code> , exceto que a informação é mostrada como uma string ao invés de um array.

Tabela 8 – Funções para tratamento de exceções.

Vamos criar um script e realizar o tratamento de exceção, vamos entender o script abaixo;

```

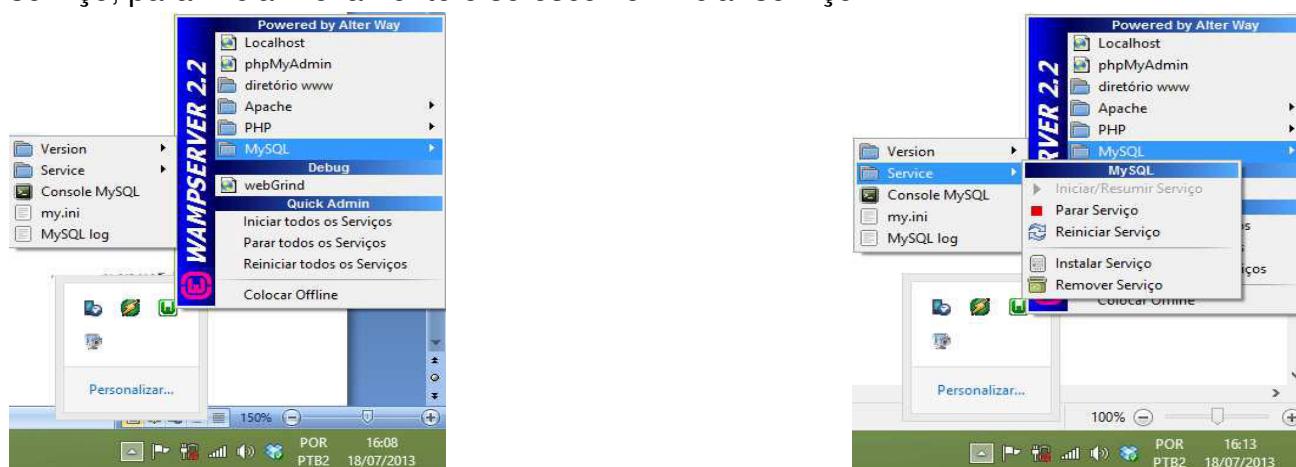
1 <?php
2 //cria a conexão com o SGBD
3 $dbconn = mysql_connect("localhost", "root", "270184") or die(mysql_error());
4
5 //Instrução SQL para criar um banco de dados MySQL
6 $sql="create database db_teste";
7
8 mysql_query($sql,$dbconn);
9 echo '<p></p>Banco de dados criado com sucesso';
10
11 mysql_close();
12 ?>

```

Neste script de criação de uma base de dados, realizamos a conexão na linha 3 usamos a função `die()`, caso a conexão falhe será apresentado ao usuário uma mensagem de erro.

Para testarmos o nosso script vamos parar o serviço do mysql e tentar rodar o script.

Para parar o serviço do mysql no windows usando o wamp Server, clique no ícone do wamp na barra de sistema, clique no meu Mysql e vá ate o menu service, depois clique em para serviço, para iniciar novamente é só escolher iniciar serviço.



No Linux devemos acessar o terminal e digitar o comando abaixo;

### Parar o MySQL

sudo /etc/init.d/mysql stop

### Iniciar o MySQL

sudo /etc/init.d/mysql start

## Construindo uma exceção

```

1  <?php
2   $filename="contato.txt";
3   try{
4       $arquivo= fopen($filename, "r");
5       if(!$arquivo){
6           throw new Exception("Não foi possível abrir o arquivo");
7       }else{
8
9           echo 'O arquivo foi lido com sucesso!';
10      }
11  } catch (Exception $ex){
12      echo "Erro (Arquivo: ".$ex->getFile().", Linha ".
13          $ex->getLine()." : ".$ex->getMessage());
14  }
15 }
16 ?>

```

Este script será chamado de **const\_excecao.php**, nele vamos testar alguns métodos de tratamento de exceção, quase todas as linguagens de programação utilizam uma instrução chamada de **try/catch**, onde o que está dentro do bloco **try** tentará ser executado caso não consiga, será direcionado para o **catch** onde será capturado o erro ocorrido.

**Linha 2:** Nesta linha definimos a localização do arquivo com o seu devido nome.

**Linhas 3 a 12:** Neste trecho a instrução **try** tentará executar a instrução de ler o arquivo, se o arquivo não existir a instrução **throw new** entra com a classe **Exception** e lança a mensagem de erro, se não é exibida uma mensagem de sucesso. A linha 12 entra em ação caso o arquivo não exista ou não seja encontrado, usando a instrução **catch**, é criado um objeto **\$ex** da classe **Exception**, com este objeto é possível chamar as funções desta classe.

## 5.5. Sessão

Sessão é um recurso do PHP que permite que você salve valores (variáveis), para serem usados ao longo da visita do usuário. Valores salvos na sessão podem ser usados em qualquer parte do script, mesmo em outras páginas do site.

Estas variáveis permanecem setadas até o visitante fechar o browser ou a sessão ser destruída.

Você precisa iniciar a sessão antes de poder setar ou pegar valores dela. Não há limite de valores salvos na sessão. A sessão é pessoal de cada visitante.

Quando um visitante acessa o site, é gerado um cookie no computador dele informando um id único de sessão e o PHP usa esse identificador pra ‘organizar’ as sessões entre os visitantes do seu site.

Esse cookie tem validade apenas enquanto o browser estiver aberto.

Você precisa iniciar a sessão antes de iniciar o output, ou seja, antes de retornar qualquer coisa para o HTML. Antes de dar qualquer **echo** ou antes de inserir qualquer HTML fora de blocos php. O início da sessão é uma das primeiras coisas no começo de todo site.

A sessão precisa ser iniciada em cada página que você for usar ou definir um valor dela, salvo arquivos que vieram por include, mas é preciso ter iniciado a sessão uma vez antes do include.

Agora vamos aprender como iniciar uma sessão, abaixo temos um exemplo com a sintaxe;

### Inicia a sessão



```
session_start();
```

Depois de iniciada a sessão, podemos definir valores dentro dela, exemplo abaixo:



```
$_SESSION['usuario'] = 'jplima';
```

Quando você precisar exibir o valor salvo na sessão (provavelmente em outras páginas), veja o exemplo abaixo:



```
echo $_SESSION['usuario'];
```

Você pode salvar quantos valores quiser, pode re-definir os valores e usa-los em echos, argumentos de funções e da forma que preferir.

Agora vamos aprender como deletar uma variável específica da sessão, vamos usar a função **unset()**, Exemplo abaixo;



```
unset($_SESSION['usuario']);
```

Deleta uma variável da sessão.

Também podemos destruir toda a sessão de uma só vez, eliminando todas as variáveis salvas, exemplo abaixo:



```
session_destroy();
```

Com isso você tem total controle das sessões no seu site e pode salvar, por exemplo, o nome de usuário depois que ele fez o login e salvar outra variável informando que o usuário está logado. Esta é uma prática muito comum em sistemas de autenticação de usuário.

Na tabela abaixo temos uma referência para função para sessões.

Função	Descrição
<b>session_cache_limiter</b>	Obtém e/ou define o limitador do cache atual
<b>session_cache_expire</b>	Retorna o prazo do cache atual
<b>session_commit</b>	Sinônimo de session_write_close
<b>session_decode</b>	Decifra dado de sessão de uma string
<b>session_destroy</b>	Destrói todos os dados registrados em uma sessão
<b>session_encode</b>	Codifica os dados da sessão atual como uma string
<b>session_get_cookie_params</b>	Obtém os parâmetros do cookie da sessão
<b>session_id</b>	Obtém e/ou define o id de sessão atual
<b>session_is_registered</b>	Descobre se uma variável global está registrada numa sessão
<b>session_module_name</b>	Obtém e/ou define o módulo da sessão atual
<b>session_name</b>	Obtém e/ou define o nome da sessão atual
<b>session_regenerate_id</b>	Atualiza o id da sessão atual com um novo gerado
<b>session_register_shutdown</b>	Função de desligamento da sessão registrada
<b>session_register</b>	Registrar uma ou mais variáveis globais na sessão atual
<b>session_save_path</b>	Obtém e/ou define o save path da sessão atual
<b>session_set_cookie_params</b>	Define os parâmetros do cookie de sessão
<b>session_set_save_handler</b>	Define a sequência de funções de armazenamento
<b>session_start</b>	Inicia dados de sessão
<b>session_status</b>	Retorna o status da sessão corrente.
<b>session_unregister</b>	Desregistra uma variável global da sessão atual
<b>session_unset</b>	Libera todas as variáveis de sessão
<b>session_write_close</b>	Escreve dados de sessão e termina a sessão.

Tabela 9 – Funções para sessão.



## Exercício de Aprendizagem

Vamos criar o login de usuário, no projeto SBFilmes criaremos o acesso do administrador ao sistema, o usuário entra no menu área restrita e se ele estiver com uma sessão aberta ele direciona para o painel administrativo caso não esteja logado (com uma sessão aberta) é apresentado a tela de login. No template baixado existe um diretório chamado **admin**, nele será salvo tudo relacionado à área administrativa.

**1º passo:** Será criar um script de conexão com nossa base de dados **sbfilmes**, desta forma toda vez que precisarmos abrir uma conexão há chamaremos.

Este script se chamará de **conect\_db.php**, e será salvo no raiz do site.

```
<?php  
1  
2  
3 $hostname_db = "localhost";  
4 $database_db = "sbfilmes";  
5 $username_db = "root";  
6 $password_db = "270184";  
7  
8 $dbconn = mysql_connect($hostname_db,$username_db,$password_db) or die(mysql_error());  
9  
10 mysql_select_db($database_db, $dbconn);  
11 ?>
```

Conforme podemos observar na imagem acima, nas linhas 3 a 6, passamos os dados da conexão.

**Na linha 8:** Passamos os dados para a função **mysql\_connect()**, e exibimos uma mensagem de erro caso o mesmo venha a acontecer.

**Na linha 10:** Passamos a variável correspondente a nossa base de dados, e os dados da conexão para a função **mysql\_select\_db()**.

**2º passo:** Vamos criar um script chamado **validarusuario.php** que será responsável pela validação de usuário, Vamos criar este script no diretório admin, que é o diretório onde será criado este exemplo.

**Na linha 3:** Fazemos a inclusão do script **conect\_db.php**.

```

1 <?php
2 //CONECTA COM O BANCO DE DADOS
3 require_once '../conect_db.php';
4 //RECEBE OS DADOS DO FORMULÁRIO
5 $usuario = $_POST['usuario'];
6 $senha = $_POST['senha'];
7
8 //PASSAR A INSTRUÇÃO SQL PARA DA CONSULTA
9 $querysql = mysql_query("SELECT * FROM funcionario WHERE login='$usuario'AND senha='$senha'");
10 or die("<meta charset=utf-8> <script>
11 window.location=\"login.php\"
12 alert('Erro ao executar a instrução SQL')
13 </script>"); 
14
15 //LINHAS AFETADAS PELA CONSULTA
16 $linha = mysql_num_rows($querysql);
17 //VERIFICA SE RETORNOU ALGO
18 if ($linha == 0) {
19     echo"<meta charset=utf-8>
20 <script> window.location=\"login.php\"
21 alert('Usuario ou senha não encontrado');
22 </script>";
23 } else {
24
25     echo" <script> window.location=\"index.php\" </script>";
26 }
27 ?>

```

**Na linha 5 e 6:** As variáveis **\$usuario** e **\$senha** recebem os valores correspondentes através do POST, vindos do formulário da página de login do nosso template.

**Linha 9:** Passamos a instrução SQL select, esta vai selecionar todos os dados da tabela onde o campo usuário e senha forem iguais as variáveis **\$usuario** e **\$senha**.

Caso ocorra um erro, por exemplo; a tabela não exista será disparado uma mensagem de erro.

**Linha 16:** Nesta linha temos a variável **\$linha** que vai receber as linhas afetadas pela consulta, através da função **mysql\_query()**.

**Linhas 18 a 26:** É verificado se foi retornado algum registro na consulta, se o numero de linhas retornadas for igual à zero, é chamado à página de login novamente e uma mensagem é exibida. Se não a página de **index.php** é chamada, esta página de index é a que corresponde a área administrativa do site.

**2º passo:** Agora vamos alterar a nossa página de **login.php** do template sbfilmes, temos que localizar no corpo da página a div **content**, como mostrado na imagem abaixo;

```

86 <div id="content" align="center">
87     <div class="cleaner">
88         <input type="image" src="../images/cadeado.jpg"/>
89     </div>
90     <h2 class="style1">Autenticação de Usuário</h2>
91 
92     <div class="cleaner h50">
93         <form id="form1" name="form1" method="post" action="validarusuario.php">
94             <label>Usuário
95             <input type="text" name="usuario" id="usuario" />
96         </label>
97         <div class="cleaner"></div>
98             <label><br />
99                 Senha
100                <input type="password" name="senha" id="senha" />
101            </label>
102            <p></p>
103            <input name="" type="submit" value="Entrar" />
104        <label>
105            <input type="button" name="Button" id="button" value="Sair" />
106        </label>
107    </form>
108 </div>
109 
110 <div class="cleaner"></div>
111 </div> <!-- END of content -->

```

No formulário desta página vamos definir o **action** chamando o script **validarusuario.php**.

Com estes passos executados a nossa validação de usuário pelo banco já estar funcionando. Vamos testar!

Agora vamos gerenciar as sessões do nosso projeto, para esta tarefa vamos alterar o nosso script **validarusuario.php**

**3º passo:** Na instrução else, vamos acrescentar as linhas abaixo;

```

23 } else {
24     $id_user= mysql_result($querysql,0,"idfuncionario");
25     $login_user= mysql_result($querysql,0,"login");
26     $nome_user= mysql_result($querysql,0,"nome");
27     //INICIALIZA A SESSÃO
28     session_start();
29     //GRAVA AS VARIÁVEIS NA SESSÃO
30     $_SESSION[id]=$id_user;
31     $_SESSION[user]=$login_user;
32     $_SESSION[nomeUser]=$nome_user;
33     //REDIRECIONA PARA A PÁGINA
34     echo" <script> window.location=\"index.php\" </script>";
35 }
36 ?>

```

**Linhas 24 a 26:** Definimos três variáveis, onde cada variável recebe seu valor correspondente ao campo na tabela do banco de dados, usamos a função **mysql\_result()**, que retorna o valor do campo especificado.

**Linha 28:** Inicializar a sessão.

**Linhas 30 a 32:** Grava as variáveis na sessão.

**4º passo:** Neste passo vamos inserir um treco de código PHP na página principal da área administrativa, chamada **index.php**. Desta forma o usuário só terá acesso a esta página se estiver logado.

```

1 <?php require '../conect_db.php';?>
2 <?php
3 //INICIANDO UMA NOVA SESSÃO
4 session_start();
5
6 //SE NÃO TIVER VARIÁVEIS REGISTRADAS
7 if(!isset($_SESSION[id])){
8     echo" <script>
9     alert('Você precisa estar logado para acessar esta página');
10    window.location=\"login.php\"
11    </script>";
12 }
13 ?>
14 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
15 <html xmlns="http://www.w3.org/1999/xhtml">
16 <head>
17 <meta http-equiv="Content-Type" content="text/html; charset=
18 <title>SBFilmes - Área restrita</title>

```

**Linha 4:** Iniciamos uma nova sessão.

**Linha 7 a 12:** Se a variável `$_SESSION[id]` estiver vazia exibir um `alert()` e redirecionar para a pagina de login.

Nesta mesma página vamos inserir o nome do usuário logado no sistema, neste exemplo vamos inserir o nome do usuário logado, temos que localizar a **div content**, para colocarmos o nome do lado do titulo **Painel Administrativo**.

```

125         </div> <!-- END of sidebar -->
126
127     <div id="content" class="faq">
128         <h2>Painel Administrador - <?php echo $_SESSION[nomeUser];?></h2>
129         <h3>&ampnbsp</h3>
130     </div> <!-- END of content -->
131     <div class="cleaner"></div>
132 </div> <!-- END of main -->
133

```

**Na linha 128:** Exibimos o valor da variável de sessão `$_SESSION[nomeUser]`.

**5º Passo:** Agora vamos criar o script PHP para realizar o logout do sistema que se chamará `logout.php`.

```

1 <?php
2 //INICIALIZA A SESSÃO
3 session_start();
4 //DESTRÓI AS VARIÁVEIS
5 unset($_SESSION[id]);
6 unset($_SESSION[user]);
7 unset($_SESSION[nomeUser]);
8
9 session_destroy();
10 //REDIRECIONA PARA A TELA DE LOGIN
11 echo"
12 <script> window.location=\"login.php\"
13 </script>";
14 ?>

```

Sempre que formos trabalhar com sessão temos que iniciar a sessão, neste exemplo apagamos as variáveis específicas e depois destruímos todas as sessões.

**Linha 5 a 7:** Destruindo as variáveis de sessão criadas.

**Linha 9:** Destroi todas as sessões que foram criadas, nas linhas de código seguintes redireciona para a página de login.

Na página `index.php` da área restrita, vamos localizar o item de menu sair, na linha 100 chamamos o script de logout criado.

```
89 |     <li><a href="#">Locações</a>
90 |         <ul>
91 |             <li><a href="#">Sub menu 1</a></li>
92 |             <li><a href="#">Sub menu 2</a></li>
93 |             <li><a href="#">Sub menu 3</a></li>
94 |             <li><a href="#">Sub menu 4</a></li>
95 |             <li><a href="#">Sub menu 5</a></li>
96 |         </ul>
97 |     </li>
98 |     <li><a href="#">Reservas</a></li>
99 |     <li><a href="#">Clientes</a></li>
100|     <li><a href="logout.php">Sair</a></li>
```

## 5.6. CRUD

Nesta aula vamos realizar o desenvolvimento do nosso **CRUD** (Create, read, update e delete ), vamos fazer um exemplo mais simples antes de implementar em nosso template do projeto SBFilmes.

Teremos dois projetos um será o sbfilmes e o outro **cursophp\_mysql**.

### 5.6.1. Inserção

Nesta aula vamos aprender duas maneiras de realizar uma inserção de dados na tabela, estes exemplos serão feitos no projeto **cursophp\_mysql**.

**1ª Forma de inserção:** Nesta teremos a seguinte estrutura de arquivos;

- Uma página web com a extensão PHP chamada de **cadastro\_1.php**.
- Um arquivo PHP chamado **conect\_db.php**, que será responsável pelo gerenciamento da conexão com nossa base de dados.
- Um arquivo PHP chamado **insere\_cidade.php**, que será responsável de inserir as informações em nossa base de dados.
- Para estes exemplos usaremos nossa base de dados criada na aula anterior, onde criamos o script de geração do banco de dados.

Vamos criar o script de conexão com o banco de dados.

```

1 <?php
2
3 $hostname_db = "localhost";
4 $database_db = "db_php_mysql";
5 $username_db = "root";
6 $password_db = "270184";
7
8 $dbconn = mysql_connect($hostname_db,$username_db,$password_db) or die("Erro na conexão");
9
10 mysql_select_db($database_db, $dbconn);
11 ?>

```

**conect\_db.php:** Conforme observamos na imagem, foram criadas as variáveis abaixo;

**\$hostname\_db = "localhost";** => Nesta variável armazenamos o nome do servidor de banco de dados, que é a maquina onde o seu banco de dados esta instalado, em nosso exemplo estamos acessando a nossa própria maquina, meu servidor local, que pode ser acessado pelo nome **localhost** ou pelo IP **127.0.0.1**.

**\$database\_db = "db\_php\_mysql";** => Nesta variável determino o nome da nossa base de dados, no nosso caso "**db\_php\_mysql**".

**\$username\_db = "root";** => Esta por sua vez define o nome do usuário de acesso ao banco de dados, em nosso exemplo usaremos o usuário root, porem pode ser criado um usuário específico para o banco de dados do seu projeto web, que serão definidos por você ou pelo DBA da empresa.

**\$password\_db = "270184";** => Nesta é armazenada a senha de acesso ao banco de dados do seu projeto.



**ATENÇÃO:** O usuário e senha do banco de dados normalmente não é o mesmo de acesso ao sistema operacional, caso esteja usando o Wamp Server por padrão não é definida a senha do banco de dados.

Continuando a analise deste script;

```

1 <?php
2
3 $hostname_db = "localhost";
4 $database_db = "db_php_mysql";
5 $username_db = "root";
6 $password_db = "270184";
7
8 $dbconn = mysql_connect($hostname_db,$username_db,$password_db) or die("Erro na conexão");
9
10 mysql_select_db($database_db, $dbconn);
11 ?>

```

**Linha 8:** Nesta linha temos a declaração da variável **\$dbconn** que irá receber a função **mysql\_connect()**, nesta função passamos três parâmetros o nome do servidor de banco de dados, o usuário e a senha de acesso ao banco de dados. Ambos definidos nas linhas superiores. A função **die()** vai abortar a execução da aplicação, é utilizado para tratamento de erros, neste exemplo caso a conexão com o banco falhe ela será chamada.

**Linha 10:** A função **mysql\_select\_db()** é responsável por selecionar a base de dados, temos que passar dois parâmetros, um é o nome da base de dados o outro é os dados da conexão.

Agora vamos criar o **cadastro\_1.php**, conforme imagem abaixo, no **action** do formulário chamaremos o script **insere\_cidade.php**, onde enviaremos os dados da pagina **cadastro\_1.php** para este script, o script irá executar a inserção dos dados.

## Cadastro de Cidades

Nome da Cidade

Enviar

```

12 <body>
13   <h1>Cadastro de Cidades</h1>
14   <label>Nome da Cidade</label>
15   <form name="form_cad" method="POST" action="insere_cidade.php">
16     <input type="text" name="nome" value="" size="50" />
17
18     <p><input type="submit" value="Enviar" name="enviar"/></p>
19
20   </form>
21 </body>

```

**Vamos codificar o script insere\_cidade.php**

```

1 <?php
2
3 require_once 'conect_db.php';
4
5 //Campos recebidos pelo formulário
6 $field_nome = $_POST['nome'];
7
8 //Enviar uma query
9 mysql_query("INSERT INTO t_cidade (nome_cidade) VALUES ('$field_nome')", $dbconn);
10 //Fecha a conexão
11 mysql_close($dbconn);
12 //Após a inserção retornar a página cadastro.php
13 echo"
14 <script>
15 window.location=\"cadastro.php\"
16 </script>
17 "
18 ?>

```

**Linha 3:** Fazemos a inclusão do script conect\_db.php, este script contem os dados da conexão.

**Linha 6:** Nesta linha temos o valor do campo nome recebido pelo formulário.

**Linha 9:** Utilizamos a instrução SQL **insert**, que realizar a inserção de registros no banco de dados.

**Linha 11:** A função **mysql\_close** é chamada para encerrar a conexão.

**Linhas 14 a 17:** Após a inserção redirecionar para a página de cadastro.

**2ª Forma de inserção:** Neste exemplo faremos a mesma inserção, porem a diferença é que iremos criar uma função para inserção na própria página de formulário, usaremos o mesmo script de conexão do exemplo anterior.

Vamos dividir em alguns passos este exemplo.

**1º Passo:** Criar uma página com o nome **cadastro.php**.

**2º Passo:** Criar uma função para salvar os dados em nossa base de dados.

```

7 <?php
8 function salvar(){
9     require_once 'conect_db.php';
10    if (isset($_POST['nome'])){
11
12        //Campos recebidos pelo formulário
13        $field_nome = $_POST['nome'];
14
15        //Enviar uma query
16        mysql_query("INSERT INTO t_cidade (nome_cidade) VALUES ('$field_nome')", $dbconn);
17        //Fecha a conexão
18        mysql_close($dbconn);
19    }
20    //fim do IF
21 ?>

```

**Linha 10:** Para evitar que dados sejam enviados a tabela quando a página for atualizada, inserimos o **if** com a função **isset()**, se a variável do **post** não estiver vazia as linhas posteriores serão utilizadas.

**3º Passo:** Na estrutura da página no **input submit**, vamos inserir o **onclik** e chamar a função salvar.

```

28 <body>
29     <h1>Cadastro de Cidades</h1>
30     <label>Nome da Cidade</label>
31     <form name="form_cad" method="POST">
32         <input type="text" name="nome" value="" size="50" />
33
34         <p><input type="submit" value="Enviar" name="enviar" onclick="<?php salvar() ?>" /></p>
35
36     </form>
37 </body>

```



### Exercício Prático

- Crie um formulário de cadastro de frutas com os seguintes campos **id\_fruta**, **nome\_fruta**, peso, você deverá criar este banco de dados assim como a tabela para armazenar os dados.

Neste exercício teremos que criar;

- ✓ Um banco de dados.
- ✓ Uma tabela conforme especificações.
- ✓ Um formulário de cadastro.
- ✓ Um script de conexão com o banco.
- ✓ Um script para inserção dos dados do formulário.



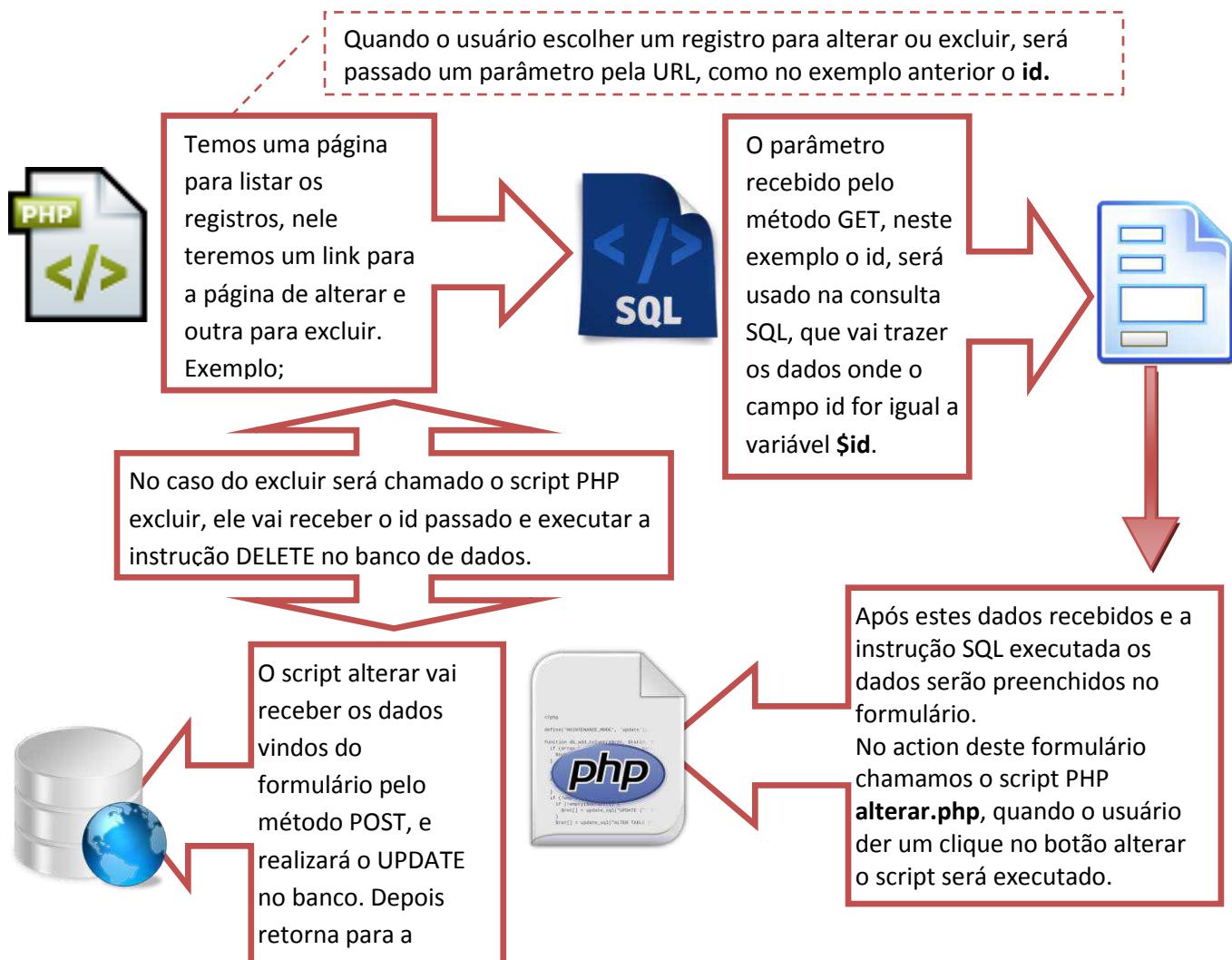
#### Projeto SBFilmes

- ✓ Para exercitarmos um pouco vamos realizar o cadastro de categoria de filmes da área administrativa.
- ✓ Agora temos que cadastrar as cidades em nosso banco de dados sbfilmes.

#### 5.6.2. Alteração

Para realizarmos uma alteração nos dados teremos que listar, e depois escolher o que será alterado.

Vamos entender como funcionará.



Vamos usar a tabela de cidades que já inserimos alguns registros nela, utilizaremos ela para listar os dados cadastrados, vamos criar uma página chamada de **manutenção\_dados.php**, criaremos também um estilo para a nossa página de manutenção.

O resultado final desta aula pode ser visto na imagem abaixo;



The screenshot shows a table titled "Listando dados da Tabela" with 12 rows. The columns are labeled "ID", "nome", "Alterar", and "Excluir". The data is as follows:

ID	nome	Alterar	Excluir
3	Itaitinga	<a href="#">Alterar</a>	<a href="#">Excluir</a>
7	Juazeiro	<a href="#">Alterar</a>	<a href="#">Excluir</a>
9	Quixeramobim	<a href="#">Alterar</a>	<a href="#">Excluir</a>
10	Caucaia	<a href="#">Alterar</a>	<a href="#">Excluir</a>
11	Pacajus	<a href="#">Alterar</a>	<a href="#">Excluir</a>
13	Fortaleza	<a href="#">Alterar</a>	<a href="#">Excluir</a>
15	Horizonte	<a href="#">Alterar</a>	<a href="#">Excluir</a>
16	Quixadá	<a href="#">Alterar</a>	<a href="#">Excluir</a>
19	Sobral	<a href="#">Alterar</a>	<a href="#">Excluir</a>
21	Acopiara	<a href="#">Alterar</a>	<a href="#">Excluir</a>
22	Paraipaba	<a href="#">Alterar</a>	<a href="#">Excluir</a>

Vamos criar primeiro nossa folha de estilo, e salvar em um diretório chamado css no nosso projeto cursophp\_mysql, você pode fazer o download desta folha de estilo no link <https://mega.co.nz/#!jYRUGJTZ!EWsQUWfUjXfb9l9vfoNqehHz9eQ2MIU35thCPINxx6c>

Em nossa página de manutenção vamos chamar a folha de estilo.

```

10  <head>
11      <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
12      <title></title>
13      <link href="css/estilo_form.css" rel="stylesheet">
14  </head>

```

Realizar a inclusão do **conect\_db.php** => **<?php require\_once 'conect\_db.php'; ?>**, que pode ser inserido no inicio da página.

 **<?php require\_once 'conect\_db.php'; ?>**

Abaixo temos a estrutura HTML da tabela:

```

15  <body>
16      <p></p>
17      <h1>Listando dados da Tabela</h1>
18      <table border="1" id="">
19          <thead>
20              <tr>
21                  <th>ID</th>
22                  <th>nome</th>
23                  <th>Alterar</th>
24                  <th>Excluir</th>
25              </tr>
26          </thead>

```

Agora vamos inserindo o código PHP para preencher o conteúdo da tabela

```

26         </thead>
27         <?php
28         $resultado = mysql_query("SELECT * FROM t_cidade", $dbconn);
29         if ($resultado) {
30             while ($row = mysql_fetch_assoc($resultado)) {
31                 ?>
32                 <tbody>
33                     <tr>
34                         <td>
35                             <?php echo "<p></p>", $row['idt_cidade']; ?>
36                         </td>
37                         <td><?php echo "<p></p>", $row['nome_cidade']; ?></td>
38                         <td><a href="alterar.php?id=<?php echo $row['idt_cidade']; ?>">Alterar</a></td>
39                         <td><a href="excluir.php?id=<?php echo $row['idt_cidade']; ?>">Excluir</a></td>
40                     </tr>
41                 </tbody>
42                 <?php
43             }//Fim do While
44             // Fim do IF.
45             mysql_close($dbconn); //Fecha a conexão
46         ?>
47     </table>

```

**Linha 28:** Nesta linha utilizamos a função **mysql\_query()** para listar os registros da tabela, o resultado da execução da query será armazenada na variável **\$resultado**.

**Linha 29:** Se a variável **\$resultado** estiver preenchida, ou seja, com registros o conteúdo da instrução **if** é executado.

**Linha 30:** Nesta linha usamos o **while** para povoar as linhas com os registros da tabela, dentro do parêntese no **while** temos, a variável **\$row** que recebe o resultado da função **mysql\_fetch\_assoc()**, que obtém uma linha do resultado como uma matriz associativa.

**Linha 35 e 37:** Nestas linhas obtemos o valor de cada campo da tabela, a variável **\$row** foi preenchida com os registros, então dentro dos **[ ]** passo o nome do campo da minha tabela.

**Linha 38 e 39:** Aqui criamos os links para chamar a página alterar, passando como parâmetro da URL o id, neste exemplo estaremos trabalhando com o método **get** na página de alterar. Um ponto muito importante desta etapa é a passagem de parâmetro pela URL, para isso temos que ter a página que iremos chamar, por exemplo; **alterar.php + ( ?id= )**.

Agora vamos criar a página para alterar os dados, chamaremos de **alterar.php**, não podemos esquecer de fazer o include do script de conexão com o banco.

```

10     <form name="form_alterar" method="post" action="alterar_cidade.php" >
11         <?php
12         $idAlt = $_GET['id'];
13         $resultado = mysql_query("SELECT * FROM t_cidade where idt_cidade=$idAlt", $dbconn);
14         if ($resultado) {
15             while ($row = mysql_fetch_assoc($resultado)) {
16                 ?>
17                 <a>ID</a>
18                 <input readonly="true" type="text" id="id_cidade" name="id_cidade" value="<?php echo $row['idt_cidade']; ?>" />
19                 <p></p>
20                 <a>Nome</a>
21                 <input type="text" name="nome_cidade" id="nome_cidade" value="<?php echo $row['nome_cidade']; ?>" />
22             <?php
23             }
24         ?>
25         <input type="submit" value="Alterar" name="alterar"/>
26     </form>

```

**Linha 12:** Criamos uma variável que vai receber o valor do método **get**.

**Linha 13:** Agora iremos fazer uma consulta baseado no id recebido do **GET**, selecionar todos os campos onde o **id\_cidade** for igual a variável **\$idAlt**.

Vamos criar o script **alterar\_cidade.php**, este script é responsável por receber os dados do formulário **alterar.php**, e executar a instrução SQL Update.

```

1 <?php
2
3 $nome_cidadde = $_POST['nome_cidade'];
4 $id = $_POST['id_cidade'];
5 include 'conect_db.php';
6
7 //Query de atualização (UPDATE)
8 $resultado = mysql_query("UPDATE t_cidade SET nome_cidade='".$nome_cidadde' WHERE idt_cidade='".$id."");
9
10 mysql_free_result($resultado);
11 mysql_close($dbconn);
12
13 //redireciona para a página de manutenção de dados
14 echo"
15 <script>
16 window.location=\"manutencao_dados.php\"
17 </script>
18 ";
19 ?>
```

**Linha 5:** Faço a inclusão do script de conexão com o banco de dados.

**Linha 8:** Escrevemos a instrução SQL para fazer a atualização dos dados, neste exemplo alteramos apenas um campo da tabela, porem pode ser feito com todos os campos da tabela, por exemplo; **mysql\_query("UPDATE Tabela SET Campo1='valor1', Campo2='Valor2', Campo3='Valor3'")**.

No exemplo acima fazemos a atualização do campo **nome\_cidade** onde **idt\_cidade** for igual a **\$id** a variável recebida.

**Linha 10:** Libera um resultado da memória associada ao identificador de resultado, apenas precisa ser chamada se você esta preocupado sobre quanta memória esta sendo usada, em consultas que retornam grandes conjuntos de resultados. Toda a memória associada a um resultado é automaticamente liberada ao final da execução do script.

### 5.6.3. Exclusão

Agora vamos para exclusão de registros, seguindo da página de manutenção, vamos criar um script PHP chamado **excluir.php**, vamos receber uma variável da URL para determinar que registro será excluído.

```
1 <?php
2 require_once 'conect_db.php';
3
4 $id = $_GET['id'];
5 mysql_query("DELETE FROM t_cidade where idt_cidade=$id");
6
7 mysql_close($dbconn);
8 echo '<meta charset=UTF-8>
9     <script>alert("Registro excluido")</script>';
10
11 echo" <script>
12 window.location=\"manutencao_dados.php\"
13 </script>
14 "
15 ?>
```

**Linha 4:** A variável **\$id** recebe o valor passado pelo método GET. Este id vem do formulário de manutenção e foi passado pela URL (**excluir.php?id=**).

**Linha 5:** Nesta linha executamos a instrução SQL delete, que realiza a exclusão do registro onde o **idt\_cidade** for igual a variável **\$id**, depois na linha 7 fechamos a conexão.



### Projeto SBFilmes

Em nosso projeto SBFilmes vamos fazer o cadastro de cliente, este cadastro é para o cliente realizar a locação dos filmes, Nesta situação vamos inserir os dados do cliente e depois realizar a autenticação dele no sistema área do cliente. Vamos seguir os passos do exercício de aprendizagem.



### Exercício de Aprendizagem

Neste exercício vamos desenvolver a página de cadastro de cliente, primeiro vamos entender os campos da tabela de cliente.

t_cliente	
idcliente	INT(11)
nome	VARCHAR(100)
data_nasc	DATE
rg	VARCHAR(45)
cpf	VARCHAR(45)
email	VARCHAR(45)
id_cidade	INT(11)
telefone	VARCHAR(45)
observacao	BLOB
rua	VARCHAR(100)
numero	INT(11)
senha_cli	VARCHAR(8)
Status	CHAR(1)
Indexes	

Esta tabela também será usada na validação da área do cliente, os dados inseridos no cadastro são; nome, data de nascimento, RG, CPF, email, o id da cidade, telefone, rua, numero e sua senha.

Para campo **id\_cidade** teremos uma lista de nomes de cidades no formulário, e o que será enviado para o banco é apenas o id da mesma.

Os campos para a validação do cliente no próximo exercício são; o email e a **senha\_cli**.

O campo Status serve para criamos um bloqueio, caso o cliente tenha alguma pendência financeira, setamos este campo com **B**(bloqueado) para ele não ter acesso a fazer novas locações, e **L** (liberado) ou vazio quando estiver tudo ok.

**1º Passo:** Vamos escolher uma página do template, nela criaremos um formulário chamado **cad\_cliente.php**.

 <?php require\_once 'conect\_db.php';?>

Depois de criado devemos fazer a inclusão do script **conect\_db.php**, que tem os dados da conexão com o banco de dados sbfilmes.

Na **div content** do formulário **cad\_cliente**, criaremos os campos para o cliente inserir seus dados.

```

110      <div id="content" class="faq">
111          <h2>Cadastro de cliente</h2>
112          <form name="form" method="POST" action="salvar_cliente.php">
113              Nome<input type="text" name="nome" size="70" />
114              <p></p>
115              Data de Nascimento<input type="text" name="dt_nasc" />
116              <p></p>
117              RG<input type="text" name="rg" size="40" />
118              <p></p>
119              CPF<input type="text" name="cpf" size="40" />
120              <p></p>
121              Email<input type="text" name="email" size="50" />
122              <p></p>
123              <label>Cidade</label>
124              <select name="cidade">
125                  <?php
126                      $result = mysql_query("SELECT * FROM t_cidades", $dbconn);
127                      while ($linha = mysql_fetch_assoc($result)) {
128                          ?
129                      <option value=<?php echo $linha['id_cidade']; ?> > <?php echo $linha['cidade']; ?></option>
130                  <?php } ?>
131                  </select>
132                  <p></p>
133                  Telefone<input type="text" name="telefone" />

```

Na linha 124 do nosso exemplo, vamos criar uma lista com os nomes das cidades, estes registros de cidades vem da tabela **t\_cidades**, o nome desta tag **select** é cidade.

**Linha 126:** Nesta linha fazemos uma consulta pegando todos os campos da tabela **t\_cidades**.

**Linha 127:** Agora com um **loop (while)** varremos todas as posições da matriz associativa, que foi obtida pela função **mysql\_fetch\_assoc ()**.

**Linha 129:** Aqui passamos o id da cidade no atributo **value** da tag **option**, depois o nome da cidade, os campos **id\_cidade** e cidade são da tabela **t\_cidade**.

Abaixo a visualização do formulário pronto.

The screenshot shows a web-based client registration form titled "Cadastro de cliente". On the left, there is a sidebar with a tree view labeled "Categorias" under "Categorias". The main form area contains the following fields:

- Nome: [Text input]
- Data de Nascimento: [Text input]
- RG: [Text input]
- CPF: [Text input]
- Email: [Text input]
- Cidade: [Dropdown menu] set to "Pacajus"
- Telefone: [Text input]
- Rua: [Text input]
- Numero: [Text input]
- Senha: [Text input]
-

**2º Passo:** Codificar o script **salvar\_cliente.php**, inicialmente fazemos a inclusão do script **conect\_db.php**, e depois criamos as variáveis que receberam os dados do formulário linha 5 a 14.

```

1 <?php
2 require_once 'conect_db.php';
3
4 //Campos recebidos pelo formulário
5 $nome = $_POST['nome'];
6 $dt_nasc = $_POST['dt_nasc'];
7 $rg = $_POST['rg'];
8 $cpf = $_POST['cpf'];
9 $email = $_POST['email'];
10 $cidade = $_POST['cidade'];
11 $telefone = $_POST['telefone'];
12 $rua = $_POST['rua'];
13 $numero = $_POST['numero'];
14 $senha = $_POST['senha'];
15
16 function converter_dt($dt) {
17     //AGORA VAMOS EXPLODIR ELA PELAS / E SERÁ CRIADO UM ARRAY COM AS PARTES
18     $partes_da_data = explode('/', $dt);
19
20     //AGORA REMONTAMOS A DATA NO FORMATO BRASILEIRO, OU SEJA,
21     //INVERTENDO AS POSIÇÕES E COLOCANDO AS BARRAS
22     $data_brasileiro = $partes_da_data[2] . '/' . $partes_da_data[1] . '/' . $partes_da_data[0];
23     return $data_brasileiro;
24 }
25 //tratar a data antes de enviar ao banco de dados
26 $dt_nasc_novo = converter_dt($dt_nasc);

```

Em nosso formulário de cadastro temos o campo data de nascimento, o tipo de dados data do SGBD MySQL trabalha com o padrão de data americano(**ano – mês - dia**), diferente do padrão brasileiro (**dia-mês -ano**), por conta dessa diferença vamos criar uma função de conversão de data.

**Linha 16:** Aqui criamos uma função **converter\_dt**, esta função recebe como parâmetro uma data, na linha 18 explodimos o valor recebido pelas barras (/), para criar um array das partes, usamos a função **explode()** do PHP. Na linha 22 fazemos a inversão das posições e incluímos as barras, estamos pegando as partes da data e concatenando. Ao fim desta função retornamos a data modificada.

**Na linha 26:** Antes de passar a data para o banco de dados, temos que declarar a variável **\$dt\_nasc\_novo** que vai receber o retorno da função **converter\_dt()**, esta recebe a data vinda do formulário no padrão brasileiro e converte para o americano.

Com esta função conseguimos resolver o problema de datas, no formulário podemos usar o padrão brasileiro, que antes de ir para o banco esta mesma data é convertida para o padrão americano, posso utilizar a mesma função para exibir os dados para o usuário no padrão brasileiro, é só modificar a posição do array.

Lembrando que esta função não altera o formato de data do banco, a modificação é feita apenas para interação com o usuário.

**3º Passo:** Continuando o nosso exemplo, Agora vamos criar a nossa query de inserção, neste exemplo de **insert** após o nome da tabela entre parênteses, passamos os campos que queremos inserir, como podemos observar colocamos apenas os campos os quais queremos inserir dados nele, com suas respectivas variáveis. Após os dados inseridos e não havendo erros, a página de cadastro de cliente será chamada para que possa ser feito outro cadastro, ou direcionar para o **index**.

```

28 //Enviar uma query
29 mysql_query("INSERT INTO t_cliente (idcliente, nome, data_nasc, rg, cpf, email,
30     id_cidade, telefone, rua, numero, senha_cli)
31     VALUES (null,'$nome','$dt_nasc_novo','$rg','$cpf',
32         '$email','$cidade','$telefone','$rua','$numero','$senha')", $dbcconn)
33     or die(mysql_error());
34 //Fecha a conexão
35 mysql_close($dbcconn);
36 //Após a inserção retornar a página
37 echo"
38 <script>
39 window.location=\"cad_cliente.php\"
40 </script>
41 "
42 ?>
```

---



### Projeto SBFilmes

- ✓ Agora para testarmos o cadastro vamos desenvolver o login da área de cliente, através desta área é que o cliente poderá realizar as locações. O nome de usuário do cliente é o seu email e a senha será armazenada no campo **senha\_cli**, da tabela **t\_cliente**.
- ✓ Crie o cadastro de filmes da área administrativa, nesta tarefa teremos que criar um script de upload para enviar ao servidor as imagens das capas dos filmes.



### Exercício Prático

Faça um programa web que realize um cadastro de contas bancárias com as seguintes informações: **número da conta, nome do cliente, CPF e saldo**. O banco permitirá o cadastramento de contas com o mesmo CPF e não poderá haver mais que uma conta com o mesmo número da conta. Crie o menu de opções a seguir.

Menu de opções:

1. Cadastro de contas de determinado cliente.
2. Login do cliente.
3. Visualizar contas de determinado cliente, somente se estiver logado.
  - a. O cliente pode excluir uma conta.
4. Alterar dados pessoais do Cliente.
5. Saque onde o valor é subtraído do seu saldo.

6. Deposito onde o valor será acrescentado ao seu saldo.
7. Sair (logout).

#### 5.6.4. Usando SQL no PHP

Nestas aulas vamos trabalhar com algumas query SQL usando o PHP, veremos algumas consultas com relacionamento de tabelas e um exemplo de paginação de uma consulta, podemos utilizar todas as instruções SQL do MySql em seu código PHP. Abaixo temos algumas cláusulas SQL que podem ser úteis.

**Cláusula like** – verifica se o conteúdo do dado contém a string que está antes do % (José%), após o % (%Ana) ou entre o % (%usa%). Este comando é utilizado com a cláusula where.

**Sintaxe:** select [coluna] from [tabela] where [coluna] LIKE 'string';

**Ex.** select \* from pessoa where nm\_pessoa like 'J%';

**Cláusula between** – verifica se o conteúdo do dado está entre um limite mínimo e máximo.

**Sintaxe:** select [coluna] from [tabela] where [coluna] BETWEEN 'limite inferior' and 'limite superior'

**Ex.** select \* from pessoa where salario between 2600 and 3000;

**Cláusula in** – verifica se o conteúdo do dado é igual a uma das opções oferecidas.

**Sintaxe:** select [coluna] from [tabela] where [coluna] IN ('opção1', 'opção 2');

**Ex.** select \* from pessoa where sexo in ('M', 'F');

**Cláusula order by** – ordena o resultado da seleção de uma coluna.

**Sintaxe:** select [coluna] from [tabela] ORDER BY [coluna];

**Ex.** select \* from pessoa order by nm\_pessoa;

**Cláusula group by** – agrupa o resultado da seleção eliminando as repetições contidas em uma coluna.

**Sintaxe:** select [coluna] from [tabela] GROUP BY [coluna];

**Ex.** select cpf, cd\_pessoa from pessoa group by cpf, cd\_pessoa;

## Consultas SQL - Filmes de uma categoria

Neste exemplo vamos criar uma pagina chamada **consultas\_sql.php**, abaixo temos a visualização desta página.



## Consultas SQL - Filmes de uma categoria

Categoria

ID: 2 Título: Todo mundo em Pânico Categoria: infantil

ID: 3 Título: Na estrada Categoria: infantil

ID: 4 Título: As aventuras de Tadeo Categoria: infantil

ID: 5 Título: O rei Leão Categoria: infantil

A proposta deste exemplo é criar uma consulta de filmes listando-os pela categoria, iremos selecionar uma categoria e serão mostrados abaixo os filmes que pertencem a ela.

Vamos iniciar codificando a página, nas primeiras linhas temos a inclusão do script `conect_db.php`, e na linha 2 vamos setar o charset, para resolver problemas com acentuação.

```

1 <?php require_once '../conect_db.php'; ?>
2 <?php header("Content-Type: text/html; charset=ISO-8859-1", true); ?>
3 <!DOCTYPE html>
4 <html>
5   <head>
6     <meta http-equiv="content-Type" content="text/html; charset=iso-8859-1" />
7     <title></title>
8   </head>
9   <body>
10     <h1>Consultas SQL - Filmes de uma categoria</h1>
11     <form name="form1" method="POST" action="consultas_sql.php">
12       Categoria <select name="sel_cat">
13         <?php
14           $resultado = mysql_query("SELECT * FROM t_categoria", $dbconn);
15           while ($linha = mysql_fetch_assoc($resultado)) {
16             ?
17             <option value=<?php echo $linha['idcategoria'];?>> <?php echo $linha['nome'];?></option>
18             <?php } ?>
19         </select>
20         <input type="submit" value="Enviar" name="bnt" />
21     </form>

```

**Linhas 13 a 18:** Temos uma consulta que lista todos os campos da tabela `t_categoria`, na linha 17 passamos no value do `<option>` o `idcategoria` este valor será usado para fazer outra consulta.

Continuando o script na linha 24, declaramos a variável **\$idcat** ela vai receber o valor (value) da tag <select> da linha 17, através do método POST.

```

22     <p></p>
23     <?php
24     $idcat=$_POST['sel_cat'];
25     $querysql = "SELECT idfilme,titulo,nome
26             FROM t_filme,t_categoria
27             WHERE t_categoria.idcategoria= t_filme.categoria_idcategoria
28             AND t_categoria.idcategoria=' $idcat ' ";
29
30     $result_query = mysql_query($querysql, $dbconn) or die(mysql_error());
31     while ($linha_filmes = mysql_fetch_assoc($result_query)) {
32         echo 'ID: ', $linha_filmes['idfilme'];
33         echo ' Titulo: ', $linha_filmes['titulo'];
34         echo ' Categoria: ', $linha_filmes['nome'], '<p></p>';
35     }
36     ?>
37   </body>
```

**Linhas 25 a 28:** Nestas linhas criamos a query da consulta, com a instrução SQL SELECT selecionamos os campos idfilme, titulo, nome estes campos são da tabela t\_filme e t\_categoria, onde o id da tabela t\_categoria(idcategoria) for igual a o idcategoria da tabela t\_filme (categoria\_idcategoria), desta forma garanto que só serão listados o idcategoria que existir na tabela t\_filme, evitando assim registros duplicados, agora passo o idcategoria da tabela t\_categoria for igual a variável **\$idcat**, como podemos observar neste exemplo usamos aspas simples, podendo também usar a concatenação de strings.

**Linhas 30 a 35:** Agora é só executar a query e dentro do loop while passar os campos que serão exibidos na página.

### Consulta paginada

Neste exemplo vamos criar uma consulta, para listar apenas uma faixa de registro em cada página usaremos a instrução SQL limit, na imagem abaixo temos o exemplo desta aula. Vamos utilizar a tabela t\_filme do banco de dados sbfilmes, vamos usar os campos titulo e fotocapa neste exemplo.

Movie	Price	Rental Option
Vai que da certo	R\$ 6,90	Alugar
Todo mundo em Pânico	R\$ 6,90	Alugar
Na estrada	R\$ 6,90	Alugar
As aventuras de Tadeo	R\$ 6,90	Alugar

## Estrutura inicial da página

Abaixo temos o include da conexão da base de dados sbfilmes, e o link dos estilos.

```

1  <?php require_once 'conect_db.php'; ?>
2  <?php header("Content-Type: text/html; charset=ISO-8859-1", true); ?>
3  <!DOCTYPE html>
4  <html>
5      <head>
6          <meta http-equiv="content-Type" content="text/html; charset=iso-8859-1" />
7          <title></title>
8      </head>
9      <link href="css/template_style.css" rel="stylesheet" type="text/css" />
10     <script type="text/javascript" src="js/jquery.min.js"></script>
11     <style>
12         #rodape{
13             height: 50px;
14             clear:both;
15             border-top: 1px;
16         }
17         #topo {
18             height: 60px;
19             padding-left: 2%;
20             padding-top: 1%;
21         }
22         #geral{
23             width: 90%;
24             height: 90%;
25             margin-bottom: 2%;
26             text-align: center;
27         }
28     </style>

```

Dentro da body na linha 35, declaramos uma variável chamada **\$maximo**, através dela vamos determinar quantos registros serão mostrados por página.

```

29 <body>
30     <div id="topo">
31         <h1>Lista de filmes</h1>
32     </div>
33     <?php
34     // Maximo de registros por pagina
35     $maximo = 4;
36     // Declaração da pagina inicial
37     $pagina = $_GET["pagina"];
38     if ($pagina == "") {
39         $pagina = "1";
40     }
41     // Calculando o registro inicial
42     $inicio = $pagina - 1;
43     $inicio = $maximo * $inicio;
44
45     $query = mysql_query("SELECT * FROM t_filme", $dbconn);
46     // Conta os resultados no total da query
47     $total = mysql_num_rows($query);

```

**Linha 37:** Declaramos a variável **\$pagina** que vai receber o valor do método GET, abaixo temos uma estrutura condicional, se a variável **\$pagina** estiver vazia o valor um é atribuído à página.

**Linhas 42 a 43:** Nestas linhas fazemos um cálculo para determinar o registro inicial.

**Linha 45:** Nesta linha fazemos uma consulta para listar todos os registros da tabela **t\_filme**, para utilizar esta query na linha abaixo.

**Linha 47:** Armazenamos na variável **\$total** o numero de linhas de um conjunto de resultados, que foi obtido pela função `mysql_num_rows()`.

### Conteúdo da página

Neste trecho do código dando continuidade ao trecho anterior, iniciamos a geração do conteúdo que será apresentado na página.

**Linha 52:** Nesta linha criamos a query para obter os registros da tabela `t_filme`, usamos a cláusula `limit` que retorna o limite entre dois valores, onde passamos a variável **\$inicio** que teve seu valor calculado nas linhas anteriores, e a variável **\$maximo** que definimos o valor da mesma no inicio do código.

**Linha 54:** Agora passamos a query SQL que esta armazenada na variável **\$sql**, para a função `mysql_fetch_object()`, que retorna um objeto com propriedades que correspondem a linha obtida, sendo armazenado na variável **\$linha**.

```

49     ##### // INICIO DO CONTEÚDO #####
50
51 // Realizamos a query
52 $sql = mysql_query("SELECT * FROM t_filme LIMIT $inicio,$maximo", $dbconn);
53 // Exibimos os nomes dos produtos e seus respectivos valores
54 while ($linha = mysql_fetch_object($sql)) {
55     ?>
56
57     <div id="geral">
58         <div class="col col_14 product_gallery">
59             <a href="filmedetai.php"> 1) {
79         echo "<br />";
80     // Exibição de pagina
81     if ($menos > 0) {
82         echo "<a href=\"" . $_SERVER['PHP_SELF'] . "?pagina=$menos>anterior</a> ";
83     } // Listando as paginas
84     for ($i = 1; $i <= $pgs; $i++) {
85         if ($i != $pagina) {
86             echo "<a href=\"" . $_SERVER['PHP_SELF'] . "?pagina=" . ($i) . ">$i</a> | ";
87         } else {
88             echo "<strong> " . $i . " </strong>| ";
89         }
90     } //fim for
91     if ($mais <= $pgs) {
92         echo "<a href=\"" . $_SERVER['PHP_SELF'] . "?pagina=$mais>proxima</a> ";
93     }
94     }
95     ?>
96     </div>
97     </body>
```

**Nas linhas 78 a 82:** Temos estruturas condicionais para determinar se **\$pgs** (páginas) é maior ou menor, se **\$menos** maior que zero criar o link com a localização atual do script através da variável global **\$\_SERVER**.

**Linha 84:** Nesta linha temos uma estrutura de repetição “for”, para criar a numeração das páginas e acrescentar o link das mesmas.

**Linha 91:** Nesta estrutura condicional temos se **\$mais** menor ou igual a **\$pgs**, criar o link com a localização da página.

### 5.6.5. Consulta

Nesta aula vamos desenvolver um formulário de pesquisa, iremos acrescentar nele o link para alterar e excluir.

Vamos fazer uma consulta de cidades pelo nome, usaremos a base de dados **db\_php\_mysql**, este exemplo é bem prático e pode ser implementado com qualquer tabela.

Abaixo temos o resultado final deste exemplo.

Pesquisa de Cidades			
Resultado da pesquisa			
ID	nome	Alterar	Excluir
3	Itaitinga	<a href="#">Alterar</a>	<a href="#">Excluir</a>
7	Juazeiro	<a href="#">Alterar</a>	<a href="#">Excluir</a>
9	Quixeramobim	<a href="#">Alterar</a>	<a href="#">Excluir</a>
10	Caucaia	<a href="#">Alterar</a>	<a href="#">Excluir</a>
11	Pacajus	<a href="#">Alterar</a>	<a href="#">Excluir</a>
13	Fortaleza	<a href="#">Alterar</a>	<a href="#">Excluir</a>
15	Horizonte	<a href="#">Alterar</a>	<a href="#">Excluir</a>
16	Quixadá	<a href="#">Alterar</a>	<a href="#">Excluir</a>
19	Sobral	<a href="#">Alterar</a>	<a href="#">Excluir</a>

Vamos criar uma nova página e chamaremos ela de **consultar.php**, abaixo temos a codificação da página.

```

1 <!DOCTYPE html>
2 <?php require_once 'conect_db.php'; ?>
3 <?php $tx_pesq = $_POST['tx_pesquisa'];?>
4 <html>
5   <head>
6     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
7     <title></title>
8     <link href="css/estilo_form.css" rel="stylesheet">
9   </head>
```

**Na linha 3:** Temos a variável de pesquisa, que vai receber pelo método POST o valor digitado pelo usuário no input.

Abaixo, na linha 30 temos a construção da nossa query, que vai selecionar todos os campos da tabela `t_cidade` onde o nome da cidade iniciar com o valor digitado pelo usuário no input.

```

20         <table border="1">
21             <thead>
22                 <tr>
23                     <th>ID</th>
24                     <th>nome</th>
25                     <th>Alterar</th>
26                     <th>Excluir</th>
27                 </tr>
28             </thead>
29             <?php
30             $resultado = mysql_query("SELECT * FROM t_cidade WHERE nome_cidade like '$tx_pesq'", $dbconn);
31             if ($resultado) {
32                 while ($row = mysql_fetch_assoc($resultado)) {
33             ?>
34                 <tbody>
35                     <tr>
36                         <td> <?php echo "<p></p>", $row['idt_cidade'];?></td>
37                         <td><?php echo "<p></p>", $row['nome_cidade'];?></td>
38                         <td><a href="alterar.php?id=<?php echo $row['idt_cidade']; ?>">Alterar</a></td>
39                         <td><a href="excluir.php?id=<?php echo $row['idt_cidade']; ?>">Excluir</a></td>
40                     </tr>
41                 </tbody>
42                 <?php
43             //Fim do loop while
44             //Fim do IF
45         ?>
46     <?php mysql_close($dbconn); ?>
47     </table>

```

## Consulta com filtros

A proposta é fazer aqueles formulários em que temos 2 ou 3 ou mais inputs, que servirão de filtro para a consulta que faremos na base de dados. Porém, o usuário pode querer preencher apenas um desses campos, dois deles, ou todos, e a nossa query, deve se adequar a esta realidade, mandando para o servidor apenas a consulta correta. Abaixo está uma solução para o problema. Usando um array, e alguns **if's**,

## Estrutura HTML

```

1     <?php require_once 'conect_db.php'; ?>
2     <!DOCTYPE html>
3     <html>
4         <head>
5             <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
6             <title></title>
7             <style type="text/css">
8                 label { display: block; }
9             </style>
10            </head>
11            <body>
12                <form name="form1" method="post">
13                    <h1>Pesquisa com filtros</h1>
14                    Nome: <input type="text" name="nome_cli" value="" /><p></p>
15                    RG: <input type="text" name="rg" value="" /><p></p>
16                    CPF: <input type="text" name="cpf" value="" /><p></p>
17                    <input type="submit" value="Pesquisar" name="pesq_filtro" />
18                </form>

```

Continuando a codificação da nossa página, na linha 21 temos uma estrutura condicional, se o método submetido for igual ao POST, será criando um array para a cláusula where.

**Nas linhas 25 a 27:** Temos as variáveis recebendo o valor de retorno da função getpost(), nesta função passamos um valor como parâmetro.

**Nas linhas 30 e 32:** Utilizamos uma estrutura condicional para, se **\$nome** for verdadeiro, estiver preenchido o array **\$where[ ]** receber o trecho da consulta.

```

20      <?php
21          if( $_SERVER['REQUEST_METHOD']=='POST' )
22          {
23              $where = Array();
24
25              $nome = getPost('nome_cli');
26              $rg = getPost('rg');
27              $cpf = getPost('cpf');
28
29
30          if( $nome ){ $where[] = " `nome` like '%{$nome}%'"; }
31          if( $rg ){ $where[] = " `rg` like '%{$rg}%'"; }
32          if( $cpf ){ $where[] = " `cpf` = '{$cpf}'"; }
33
34          $sql = "SELECT * FROM t_cliente ";
35          if( sizeof( $where ) )
36              $sql .= ' WHERE '.implode( ' AND ', $where );
37
38          //filtro anti injection
39          function filter( $str ){
40              return addslashes( $str );
41          }
42          function getPost( $key ){
43              return isset( $_POST[ $key ] ) ? filter( $_POST[ $key ] ) : null;
44          }

```

**Linha 34:** Aqui temos a query que lista todos os campos da tabela t\_cliente, que é o inicio da nossa query.

**Na linha 35:** Temos uma condição que conta os elementos de um array.

**Na linha 36:** fazemos a concatenação da query, e a utilização da função implode() que junta os elemento da matriz **\$where** na string \$sql.

**Linha 39:** Aqui criamos uma função de filtro anti injection, é uma forma de proteger seu código contra ataques de injeções de código PHP na sua consulta.

Agora vamos executar a nossa query, passando a variável **\$sql** para a função **mysql\_fetch\_assoc()**.

```

46          //execução da query
47          $resultado = mysql_query($sql) ;
48          while ( $linha = mysql_fetch_assoc($resultado) ){
49              echo '<p></P>ID: '.$linha['idcliente'];
50              echo '<p></P>Nome: '.$linha['nome'];
51              echo '<p></P>RG: '.$linha['rg'];
52              echo '<p></P>CPF: '.$linha['cpf'];
53
54          }
55      ?>
</body>

```

## 5.7. Relatórios.

Nesta aula vamos aprender a criar relatório em PDF usando PHP, para esta prática vamos utilizar a classe FPDF, é uma classe para PHP de grande utilidade no processo de criação de relatórios em formato PDF, utilizando funções simples e poderosas. O FPDF é um Software Livre (e gratuito), o que significa que você poderá utilizá-lo livremente para uso pessoal ou

comercial, além de poder modificá-lo e estudar seu código-fonte, ao contrário da **PDFlib**, que exige o pagamento de licenças para uso comercial.

O FPDF possui suporte às principais funções para geração de relatórios, como por exemplo:

- ✓ Formatação de cabeçalho (Header) e rodapé (Footer);
- ✓ Quebra de página automática;
- ✓ Quebra de linha e justificação de texto automática;
- ✓ Suporte a imagens (JPEG e PNG);
- ✓ Suporte a fontes TrueType e Type1;

## Instalando o FPDF

Para utilizarmos precisamos fazer o download no site <http://www.fpdf.org> e baixar a última versão do FPDF, depois, vamos descompacta-la no diretório raiz do nosso site, como podemos observar sua instalação é bem simples.

No site desta classe podemos encontrar vários tutoriais sobre sua utilização, neste exemplo vamos criar um modelo para ser utilizado, depois podemos visitar o site para nos aprofundarmos.

Vamos criar um formulário com uma lista **drop-down (<select>)**, nela vamos listar os nomes de clientes cadastrados na base de dados sbfilmes, conforme imagem abaixo, quando o usuário der um clique no botão imprimir, será enviado via **POST** o id do cliente selecionado, o script **modelo\_pdf2.php** recebe o valor da variável e lista os dados do cliente selecionado no formulário.

Abaixo temos o formulário e o resultado da geração do PDF.

The screenshot shows a Firefox browser window. The address bar displays 'localhost/curso\_phpmysql/fase5/form\_pdf.php'. The main content area shows a form titled 'SBFilmes - Dados do Cliente' with fields for ID cliente (7), Nome (Limas Tecnologia), Data de Nascimento (1983-03-13), RG (432343), CPF (003.574.623-86), and E-mail (jplimamaster@gmail.com). To the right, a separate window titled 'Imprimir dados do Cliente' shows the same form data, with 'Nome do Cliente' set to 'Limas Tecnologia' and an 'Imprimir' button.

Vamos criar o formulário, a pagina web será chamada de **form\_pdf.php**, inicialmente iremos incluir o script **conect\_db.php** com os dados da conexão da base de dados sbfilmes.

```

1      <?php require_once '../conect_db.php'; ?>
2      <!DOCTYPE html>
3      <html>
4          <head>
5              <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
6              <title></title>
7          </head>
8          <body>
9              <form name="form" method="POST" action="modelo_pdf2.php">
10                 <h1>Imprimir dados do Cliente</h1>
11                 Nome do Cliente <select name="sel_clientes">
12                     <?php
13                         $resultado= mysql_query("SELECT * FROM t_cliente", $dbconn) or die(mysql_error());
14
15                         while ($linha= mysql_fetch_assoc($resultado)) {
16
17                             ?>
18                             <option value="<?php echo $linha['idcliente'];?>"> <?php echo $linha['nome'];?></option>
19                             <?php ?>
20                         </select>
21                         <input type="submit" value="Imprimir" name="enviar" />
22                     </form>
23
24             </body>
25         </html>

```

No action do formulário chamaremos o script **modelo\_pdf2.php**.

Agora vamos criar o script **modelo\_pdf2.php**, inicialmente iremos incluir a classe **fpdf.php**, e depois o script **conect\_db.php**.

```

1      <?php
2      require('fpdf17/fpdf.php');
3      require_once '../conect_db.php';
4
5      class PDF extends FPDF {
6          // topo da página
7          function Header() {
8              // Logotipo
9              $this->Image('../images/templatemo_logo.png', 10, 6, 30);
10             // Arial bold 15
11             $this->SetFont('Arial', 'B', 15);
12             // Mover para a direita
13             $this->Cell(80);
14             // Título
15             $this->Cell(20, 10, 'SBFilmes - Dados do Cliente', 6, 0, 'C');
16             // quebra de linha
17             $this->Ln(20);
18         }
19         // rodapé da página
20         function Footer() {
21             // Posição em 1.5 cm de baixo
22             $this->SetY(-15);
23             // Arial italic 8
24             $this->SetFont('Arial', 'I', 8);
25             // Número de página
26             $this->Cell(0, 10, 'Page ' . $this->PageNo() . '/{nb}', 0, 0, 'C');
27         }
28     }

```

**Linha 5:** Criamos a classe pdf que vai herdar os métodos e objetos da classe FPDF.

**Linha 7:** Vamos criar uma função chamada Header, que será o topo da nossa página em pdf.

**Linhas 9 a 17:** Nestas linhas da função Header, temos o método Image onde definimos a imagem que irá aparecer em nosso arquivo pdf, onde (10) e (6) são as coordenadas da

imagem no documento, (30) representa o tamanho da imagem em milímetros, posso passar a altura e a largura utilizando dois valores (30), (60), (se omitido é utilizado o tamanho original da imagem).

Outro parâmetro que pode ser passado é o formato da imagem (os seguintes formatos são válidos: **jpg**, **jpeg**, **png**). Se não especificado, esse valor será determinado a partir da extensão do arquivo. Abaixo temos um exemplo com todos os parâmetros:

```
$pdf->Image('logo.jpg',4,5,150,70,jpg);
```

Na linha 11 estamos setando a fonte pelo método **SetFont()**, passamos três parâmetros básicos, o nome da fonte (Arial), o atributo de formatação **B** é para deixar em negrito e o **I** para itálico, o valor (15) representa o tamanho da fonte.

Na linha 13 temos o método **cell()** que será bastante utilizado, então vamos entender como este método funciona.

### Método Cell()

Cria uma célula de tamanho, texto, borda e alinhamento configurados pelo programador.

#### Sintaxe:

```
Cell(float width [, float height [, string texto [, borda [, int In [, string alinhamento [, int fill [, mixed link]]]]]]])
```

Onde **width** é a largura e **height**, a altura da célula. Além destes parâmetros, em texto informa-se o conteúdo da **célula(string texto)**. A opção borda deve ser preenchida com **0**(zero), para retirá-la, ou 1 para colocar a borda. **In** indica para onde vai o cursor após a chamada da função **Cell()**, e seus valores podem ser 0 para a direita, 1 para o começo da próxima linha e 2 para baixo.

Alinhamento pode ser preenchido com **L** para a esquerda (**default**), **C** para centralizado e **R** para a direita.

Em **fill**, devem-se informar os valores **0** para transparente (**default**) e **1** para preenchimento de fundo da célula.

Em link, poderia se utilizar a URL ou identificador retornado pela função **AddLink()**.

**Na linha 20:** Temos a função **Footer** que é o nosso rodapé, na linha 22 temos função **SetY**, que muda o cursor para a coordenada em y desejada. **SetY(float coordenada)**:

Continuando o nosso script, na linha 31 cria-se um objeto. Com os valores default para página são: tamanho A4 e no modo Retrato, com as medidas em milímetros (mm).

```

30 // Instanciação da classe herdada
31 $pdf = new PDF();
32 $pdf->AliasNbPages();
33 $pdf->AddPage();
34 $pdf->SetFont('Times', '', 12);
35
36
37 $id_cliente = $_POST['sel_clientes'];
38 // Conteúdo do PDF
39 $resultado = mysql_query("SELECT * FROM t_cliente where idcliente=$id_cliente", $dbconn);
40 while ($linha = mysql_fetch_assoc($resultado)) {
41     $pdf->Cell(0, 10, 'ID cliente: ' . $linha['idcliente'], 0, 1);
42     $pdf->Cell(0, 10, 'Nome: ' . $linha['nome'], 0, 1);
43     $pdf->Cell(0, 10, 'Data de Nascimento: ' . $linha['data_nasc'], 0, 1);
44     $pdf->Cell(0, 10, 'RG: ' . $linha['rg'], 0, 1);
45     $pdf->Cell(0, 10, 'CPF: ' . $linha['cpf'], 0, 1);
46     $pdf->Cell(0, 10, 'E-mail: ' . $linha['email'], 0, 1);
47 }
48 // Geração da saída no arquivo PDF
49 $pdf->Output();
?>

```

Para alterar os valores padrões deste objeto preciso passar os parâmetros deste objeto.

Vejamos a sintaxe abaixo **PDF()** é:

**PDF([String Orientação [,String Unidade [,mixed formato]]]);**

O parâmetro Orientação pode ser preenchido com:

- P: Portrait (Retrato);
- L: Landscape (Paisagem);

Em Unidade podemos informar os valores:

- pt: para pontos;
- mm: para milímetros;
- cm: para centímetros;
- in: para polegadas;

E em Formato:

- A3;
- A4;
- A5;
- Letter;
- Legal;

**Na linha 32:** Defini um alias para o número de páginas, se não for informado nenhum assume o default.

**Na linha 33:** Adiciona uma nova página.

**Linha 37:** Nesta linha recebemos o valor da tag **<select>** que será armazenado na variável **\$id\_cliente**.

**Linha 39:** Agora faremos a consulta para obter os dados do cliente, através do valor selecionado no formulário, vamos trazer todos os dados do cliente onde o **idcliente** que é o nome do campo da tabela, for igual a **\$id\_cliente**, que obteve o valor através do método **POST** do formulário.

**Linhas 40 a 46:** Temos o laço onde irá adicionar as **cell()** com os valores obtidos da consulta, concatenando o nome do campo para mostrar no pdf e o valor do registro da tabela.

**Linha 49:** Nesta linha é criado o arquivo PDF através do método **output()**. Se o navegador tiver o plugin Adobe Acrobat Reader instalado, o documento será aberto pelo próprio navegador. Caso contrário, ele pedirá para fazer o download do PDF gerado.

## 6. Frameworks para desenvolvimento em PHP

Os frameworks facilitam o desenvolvimento de software, permitindo que os programadores se ocupem mais com os requisitos do software do que com os detalhes tediosos, de baixo nível do sistema.

Com o uso de frameworks, os programadores tem o controle de seu tempo e de seus códigos-fonte, as tarefas repetitivas são minimizadas, os projetos são concluídos em menos tempo, os padrões são seguidos.

Os diversos frameworks de desenvolvimento agilizam o processo de criação e manutenção de aplicativos web. Cada framework tem suas particularidades, nestas aulas vamos conhecer um pouco de algumas dos frameworks mais utilizados, dependendo da sua necessidade escolherem qual melhor se encaixa em seu projeto.

### 6.1. Soluções de framework do PHP

Abaixo serão apresentados alguns frameworks que vamos conhecer que são: CakePHP, Symfony, Zend Framework e Solar Framework.

Os frameworks são bastante utilizados em projetos complexos, e que utilizam algum padrão de desenvolvimento como MVC, que é a separação da estrutura da aplicação em três partes distintas: Modelo, Visão e Controle:

- **Modelo:** gerencia os dados da aplicação.
- **Visão:** gerencia a saída gráfica e textual da parte da aplicação visível ao usuário.
- **Controle:** interpreta as entradas de mouse e teclado do usuário, comandando a Visão e o Modelo para se alterarem de forma apropriada.

Os frameworks, sejam elas escritas em PHP ou em qualquer outra linguagem, oferecem ao programador um conjunto de códigos prontos que permitem realizar as tarefas mais básicas no desenvolvimento de um aplicativo. Por oferecer essa estrutura básica, os frameworks tornam o desenvolvimento mais rápido e reduzem o volume de código repetitivo escrito pelo programador.

## 6.2. O CakePHP Framework

O *CakePHP* é um *Framework* para desenvolvimento de aplicações web no padrão MVC (*Models-Views-Controllers*) e ActiveRecord.

A idéia principal é ser um framework estruturado, que permita a usuários PHP de todos os níveis desenvolverem aplicações web robustas sem perda da flexibilidade.

Softwares que atendem os requisitos e que são de rápida manutenção/alteração, e também aqueles que são implementados usando boas práticas de desenvolvimento e padrões, são os chamados Bons Softwares.

Com o *CakePHP*, a construção de um software assim se torna fácil. Com este *Framework*, também é possível implementar usando Test-Driven-Development (TDD), ou seja, desenvolvimento orientado a testes. Este fato permite que tudo na aplicação possa ser testado e torna essa ferramenta muito poderosa.

O site oficial deste framework é <http://cakephp.org/>, abaixo temos as principais características deste framework;

- ✓ **Construir rapidamente**

Use os recursos de geração e estrutura de código para criar rapidamente protótipos.

- ✓ **Não Configuração**

Nada de XML complicado ou arquivos YAML. Basta configurar seu banco de dados e você estará pronto para utilizar.

- ✓ **Licença amigável**

CakePHP é licenciado sob a licença MIT que o torna perfeito para o uso em aplicações comerciais.

- ✓ **Baterias incluídas (Blocos)**

As coisas que você precisa são built-in. Traduções, acesso de banco de dados, caching, validação, autenticação e muito mais são todos construídos em um dos quadros originais PHP MVC.

- ✓ **Limpe Convenções MVC**

Em vez de ter de planejar aonde as coisas vão, o CakePHP vem com um conjunto de convenções para orientá-lo no desenvolvimento de sua aplicação.

- ✓ **Proteger**

CakePHP vem com built-in ferramentas para validação de entrada, proteção CSRF, proteção adulteração Form, prevenção de injeção SQL, prevenção XSS, ajudando você a manter seu aplicativo seguro.



### Why use CakePHP

#### Build Quickly

Use code generation and scaffolding features to rapidly build prototypes.

#### Batteries Included

The things you need are built-in. Translations, database access, caching, validation, authentication, and much more are all built into one of the original PHP MVC frameworks.

#### No Configuration

No complicated XML or YAML files. Just setup your database and you're ready to bake.

#### Clean MVC Conventions

Instead of having to plan where things go, CakePHP comes with a set of conventions to guide you in developing your application.

#### Friendly License

CakePHP is licensed under the MIT license which makes it perfect for use in commercial applications.

#### Secure

CakePHP comes with built-in tools for input validation, CSRF protection, Form tampering protection, SQL injection prevention, and XSS prevention, helping you keep your application safe & secure.

## 6.3. O Solar Framework

Solar é um framework de aplicações web para PHP 5. É a ideia de Paul M. Jones. Outros desenvolvedores estão trabalhando em componentes adicionais para o pacote de distribuição de projeto padrão, tudo sob a nova licença BSD, o download dela pode ser encontrado em <http://solarphp.com/>.

Este framework permite Integração completa dos padrões de desenvolvimento, como: MVC, Query Objetos, Lazy Load. Abaixo temos algumas características deste framework.

- ✓ **Elegante e consistente:** A própria base de código é fácil de compreender, adere a convenções de nomenclatura bem documentadas, e apresenta forte integridade conceitual.
- ✓ **Nome Completo E-spacing:** As classes solares têm o seu próprio PHP 5.2, tornado mais fácil para misturar e combinar componentes de outras bibliotecas e frameworks.
- ✓ **Configurações herdadas de classe:** Defina um valor no arquivo de configuração para uma classe, e todos os seus filhos herdam esses valores por padrão.
- ✓ **Localização herdada por classe:** Defina as strings de locale para uma classe, e todos os seus filhos herdam.
- ✓ Segurança fácil de usar: Segurança de profundidade contra injeção de SQL, cross-site scripting, cross-site request forgery e outros exploits comuns.
- ✓ **Modelo de sistema robusto:** Que permite criar formulários automaticamente a partir de objetos de registro.
- ✓ **Filtragem de dados:** Extensível para validar e limpar entradas (input) do usuário.

The screenshot shows the homepage of the Solar Framework for PHP 5. At the top, there's a banner with the Solar logo and the text "SOLAR Framework for PHP 5". Below the banner is a navigation bar with links for Home, Project, Packages, Classes, Manual, and Bugs. The main content area is divided into three columns: "Discover", "Download", and "Explore". The "Discover" column contains a brief introduction to Solar and a link to "Read more about Solar...". The "Download" column features a large "Download Now" button and some text about the latest release. The "Explore" column lists several items for users to interact with, such as getting started, examining packages, consulting the API reference, and viewing source code. A sidebar on the right contains a "Discuss" section with links for reporting bugs, joining the mailing list, and reading more about Solar. At the bottom, there's a copyright notice and a "STAT COUNTER".

## 6.4. O symfony Framework

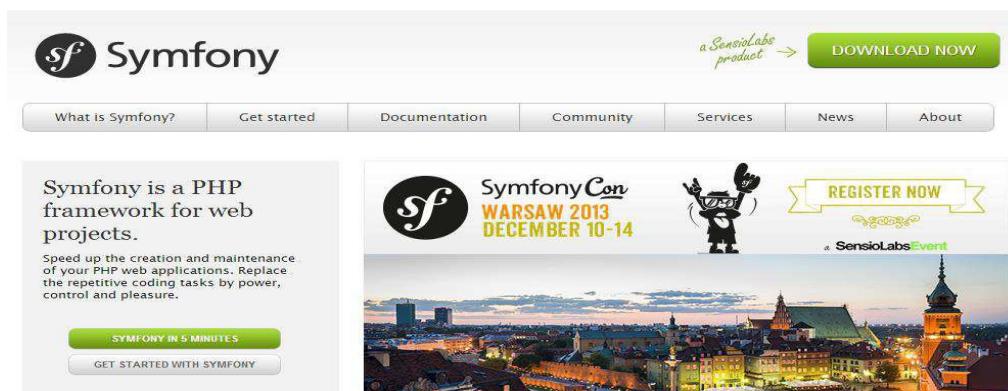
A primeira versão do symfony foi lançada em Outubro de 2005 pelo fundador do projeto Fabien Potencier. Fabien é o CEO da Sensio (<http://www.sensio.com/>), uma agência web francesa muito conhecida pelos seus pontos de vista inovadores sobre desenvolvimento web. **symfony** é um framework completo projetado para aperfeiçoar o desenvolvimento de aplicações web, através de várias características. Para começar ele separa um aplicativo web das regras do negócio, lógica e apresentação. Ele contém diversas ferramentas e classes que visam reduzir o tempo de desenvolvimento de uma complexa aplicação web. Além disso, ele automatiza tarefas comuns, para que o desenvolvedor possa se concentrar inteiramente nas especificidades da aplicação. O resultado final dessas vantagens é que não há necessidade de reinventar a roda a cada vez que um novo aplicativo web é construído.

O symfony foi escrito inteiramente em PHP 5. Ele foi testado em vários projetos do mundo real, é na verdade, no uso de alta demanda em sites de negócios. Ele é compatível com a maioria das bases de dados disponíveis, incluindo o MySQL, PostgreSQL, Oracle e Microsoft SQL Server. No site do projeto <http://symfony.com/>, podemos fazer o download deste framework. Ele roda em plataformas Windows e Unix. Abaixo temos às suas características.

### Características symfony

O symfony foi construído a fim de cumprir os seguintes requisitos:

- ✓ Fácil instalação e configuração em mais plataformas (garantido para trabalhar no padrão Unix e Windows).
- ✓ Mecanismo de banco de dados independente.
- ✓ Simples de usar, na maior parte dos casos e suficientemente flexível para se adaptar aos casos complexos.
- ✓ Baseado na premissa de convenção sobre a configuração, o desenvolvedor precisa configurar apenas o convencional.
- ✓ Compatível com a maioria das melhores práticas web e padrões de design.
- ✓ Código legível, documentação e fácil manutenção.
- ✓ Fácil de estender, permitindo a integração com outras bibliotecas.



## 6.5. O Zend Framework

A ZEND é uma empresa fundada em 1999 por Andi Gutmans e Zeev Suraski e é a responsável pela manutenção e desenvolvimento de produtos e serviços para a linguagem PHP. O Zend Framework foi lançado no dia 04 de março de 2006 e inclui diferentes componentes desenvolvidos em PHP5 para prover alta qualidade para desenvolvimento de aplicações web e web services.

O Zend Framework fornece código limpo estável, é baseada em PHP e orientada a objetos, usa o paradigma MVC, possui contribuidores de software livre, contribuidores que assumem responsabilidade pelo fato de seu código não ser propriedade intelectual de terceiros, podemos fazer o download do framework no site <http://framework.zend.com/>.



Este framework tem como meta facilitar sua vida de programação, não apenas em geral, instituindo o padrão MVC, mas também para coisas específicas que você tende a fazer o tempo todo, como acessar bancos de dados e geração de arquivo PDF.

A estrutura componente do Zend Framework 2 é único, cada componente é projetado com poucas dependências em outros componentes. Segue o princípio de projeto orientado a objetos sólidos.

Essa arquitetura flexível permite aos desenvolvedores usar quaisquer componentes, na tabela abaixo vamos conhecer estes componentes.

Componentes da Zend Framework incluem:

Zend_Controller	Esse módulo fornece o controle geral para o aplicativo. Converte pedidos em ações específicas e assegura que sejam executados.

Zend_Db	Esse é baseado em PHP Data Objects (PDO) e fornece acesso a bancos e dados de forma genérica.
Zend_Feed	Esse facilita o consumo de alimentações RSS e Atom.
Zend_Filter	Esse fornece funções de filtragem de string, como isEmail() e getAlpha().
Zend_InputFilter	Para Zend_Filter, esse é projetado para trabalhar com arrays como entradas de formulário.
Zend_HttpClient	Esse possibilita executar pedidos HTTP facilmente.
Zend_Json	Esse possibilita converter objetos PHP facilmente em JavaScript Object Notation e vice-versa.
Zend_Log	Esse fornece funcionalidade de criação de log de propósito geral.
Zend_Mail	Esse possibilita que você envie texto e e-mail MIME com diversas partes.
Zend_Mime	Esse é usado pelo Zend_Mail para ajudar a decodificar mensagens MIME.
Zend_Pdf	Esse possibilita criar novos documentos PDF e carregar e editar documentos PDF existentes.
Zend_Search	Esse possibilita executar procura sofisticadas em seu próprio texto. Por exemplo, é possível construir um mecanismo de procura que retorne resultados baseados na relevância ou em outros fatores.
Zend_Service_Amazon, Zend_Service_Flickr e Zend_Service_Yahoo	Esses fornecem fácil acesso a essas APIs de serviço da Web.
Zend_View	Esse manipula a parte "view" do padrão MVC.
Zend_XmlRpc	Esse possibilita criar facilmente um cliente XML-RPC. (Os recursos do servidor são planejados para o futuro).

Tabela 10 - Componentes da Zend Framework

# Referências Bibliográficas

*Construindo Aplicações Web com Php e Mysql*Novatec

*Desenvolvimento de Aplicações Em Php - Biblioteca Software Livre*Fca

*Dominando PHP e MYSQL Do iniciante ao profissional*2011Alta Books

*Faça um Site - Php 5.2 Com Mysql 5.0 - Comércio Eletrônico - Orientado Por Projeto - Para Windows*Erica

[http://www.php.net/manual/pt\\_BR](http://www.php.net/manual/pt_BR)

*Mysql - Guia do Programador* Novatec

*Php - Programando com Orientação a Objetos - 2<sup>a</sup> Ed. 2009*NOVATEC

*Php 6 e Mysql 5 para Web Sites Dinâmicos*Ciencia Moderna

*Php And Mysql Web Development*Sams Publishing Digital

*Php Para Iniciantes*CIENCIA MODERNA

*Php5 And Mysql® bible*John Wiley & Sons

*Web Interativa com Ajax e PHP*Novatec

# Índice de Tabelas

Tabela 1 - Variáveis Globais – Fonte: <a href="http://php.net/manual/pt_BR/language.variables.superglobals.php">http://php.net/manual/pt_BR/language.variables.superglobals.php</a> .....	6
Tabela 2 - Funções para Array .....	20
Tabela 3 – Funções de manipulação de arquivos .....	23
Tabela 4 – Modos de Arquivos .....	23
Tabela 5 - Opções de configuração de e-mail .....	37
Tabela 6 - Principais Bancos de Dados suportados em PHP - Fonte <a href="http://www.php.net">http://www.php.net</a> .....	43
Tabela 7 - Funções PHP para SGBD MySql.....	44
Tabela 8 – Funções para tratamento de exceções.....	47
Tabela 9 – Funções para sessão.....	50
Tabela 10 - Componentes da Zend Framework .....	91



## Hino Nacional

Ouviram do Ipiranga as margens plácidas  
De um povo heróico o brado retumbante,  
E o sol da liberdade, em raios fúlgidos,  
Brilhou no céu da pátria nesse instante.

Se o penhor dessa igualdade  
Conseguimos conquistar com braço forte,  
Em teu seio, ó liberdade,  
Desafia o nosso peito a própria morte!

Ó Pátria amada,  
Idolatrada,  
Salve! Salve!

Brasil, um sonho intenso, um raio vívido  
De amor e de esperança à terra desce,  
Se em teu formoso céu, risonho e límpido,  
A imagem do Cruzeiro resplandece.

Gigante pela própria natureza,  
És belo, és forte, impávido colosso,  
E o teu futuro espelha essa grandeza.

Terra adorada,  
Entre outras mil,  
És tu, Brasil,  
Ó Pátria amada!  
Dos filhos deste solo és mãe gentil,  
Pátria amada, Brasil!

Deitado eternamente em berço esplêndido,  
Ao som do mar e à luz do céu profundo,  
Fulguras, ó Brasil, florão da América,  
Iluminado ao sol do Novo Mundo!

Do que a terra, mais garrida,  
Teus risonhos, lindos campos têm mais flores;  
"Nossos bosques têm mais vida",  
"Nossa vida" no teu seio "mais amores."

Ó Pátria amada,  
Idolatrada,  
Salve! Salve!

Brasil, de amor eterno seja símbolo  
O lábaro que ostentas estrelado,  
E diga o verde-louro dessa flâmula  
- "Paz no futuro e glória no passado."

Mas, se ergues da justiça a clava forte,  
Verás que um filho teu não foge à luta,  
Nem teme, quem te adora, a própria morte.

Terra adorada,  
Entre outras mil,  
És tu, Brasil,  
Ó Pátria amada!  
Dos filhos deste solo és mãe gentil,  
Pátria amada, Brasil!

## Hino do Estado do Ceará

Poesia de Thomaz Lopes  
Música de Alberto Nepomuceno  
Terra do sol, do amor, terra da luz!  
Soa o clarim que tua glória conta!  
Terra, o teu nome a fama aos céus remonta  
Em clarão que seduz!  
Nome que brilha esplêndido luzeiro!  
Nos fulvos braços de ouro do cruzeiro!

Mudem-se em flor as pedras dos caminhos!  
Chuvas de prata rolem das estrelas...  
E despertando, deslumbrada, ao vê-las  
Ressoa a voz dos ninhos...  
Há de florar nas rosas e nos cravos  
Rubros o sangue ardente dos escravos.  
Seja teu verbo a voz do coração,  
Verbo de paz e amor do Sul ao Norte!  
Ruja teu peito em luta contra a morte,  
Acordando a amplidão.  
Peito que deu alívio a quem sofria  
E foi o sol iluminando o dia!

Tua jangada afoita enfune o pano!  
Vento feliz conduza a vela ousada!  
Que importa que no seu barco seja um nada  
Na vastidão do oceano,  
Se à proa vão heróis e marinheiros  
E vão no peito corações guerreiros?

Se, nós te amamos, em aventuras e mágoas!  
Porque esse chão que embebe a água dos rios  
Há de florar em meses, nos estios  
E bosques, pelas águas!  
Selvas e rios, serras e florestas  
Brotam no solo em rumorosas festas!  
Abra-se ao vento o teu pendão natal  
Sobre as revoltas águas dos teus mares!  
E desfraldado diga aos céus e aos mares  
A vitória imortal!  
Que foi de sangue, em guerras leais e francas,  
E foi na paz da cor das hóstias brancas!



# GOVERNO DO ESTADO DO CEARÁ

*Secretaria da Educação*