



PROGRAMAÇÃO PHP-VERSÃO 2

Programação PHP - Versão 2



Nome: _____

Sobre o curso

A linguagem de programação em PHP, criada por Rasmus Lerdorf em 1994, foi projetada apenas para alguns scripts de monitoramento, mas hoje em meio ao crescimento e popularidade da linguagem, ela vem se tornando um dos principais meios de programação web. Tendo uma arquitetura simples e intuitiva, o PHP é uma ferramenta vital para o Desenvolvedor Web que possui sede de conhecimento e um desejo de se destacar no mercado de trabalho.

O que aprender com este curso?

Neste curso você aprenderá conceitos sobre esta linguagem, regras, comandos e funções. Aprenderemos sobre banco de dados e como integrá-lo no PHP. Durante as aulas desenvolveremos um projeto colocando em prática tudo o que foi aprendido, os exercícios complementarão as aulas.



**PROGRAMAÇÃO
PHP-VERSÃO 2**



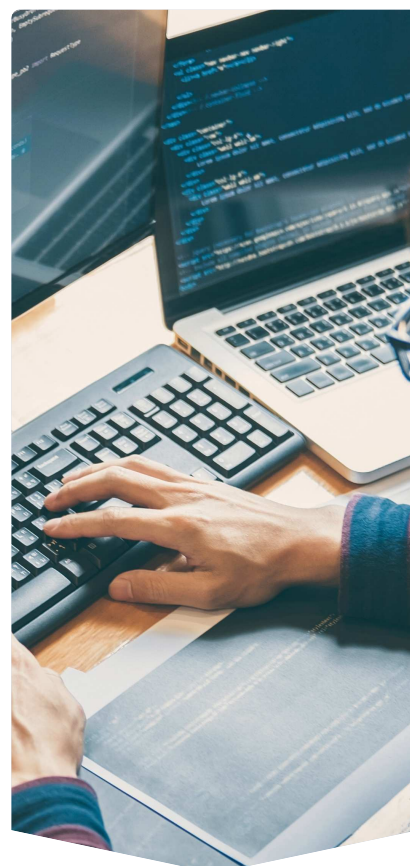
Quantidade de Aulas
20 aulas



Carga horária
30 horas



Programas Utilizados
Notepad++ e XAMPP.



Sumário

1 - Introdução ao PHP

- 1.1 - História do PHP
- 1.2 - Como funciona o PHP
- 1.3 - Os programas
- 1.4 - Exercícios Passo a Passo
- 1.5 - Exercícios de Fixação

2 - Notepad++ e Conceitos Básicos de Programação

- 2.1 - Veja uma ilustração do Notepad++.
- 2.2 - Organização das pastas
- 2.3 - Conceitos básicos de programação

2.3.1 - Variáveis

2.3.2 - Operadores Aritméticos

2.3.3 - Operador de Incremento

2.3.4 - Operador de Decremento

2.3.5 - Operadores de Atribuição

2.4 - Exercícios Passo a Passo

2.5 - Exercícios de Fixação

3 - Operadores de Comparação, Lógicos e Estrutura Condicional.

- 3.1 - Operador de Comparação
- 3.2 - Operadores lógicos
- 3.3 - Estruturas de decisão
- 3.4 - Exercícios Passo a Passo
- 3.5 - Exercícios de Fixação

4 - Estrutura Condicional e Estrutura de Repetição.

- 4.1 - Estrutura de repetição
- 4.2 - Switch...case
- 4.3 - Exercícios Passo a Passo
- 4.4 - Exercícios de Fixação

5 - Estrutura de Repetição, Strings e Funções

- 5.1 - Manipulação de Strings
- 5.2 - Funções em PHP
- 5.3 - Exercícios Passo a Passo
- 5.4 - Exercícios de Fixação

6 - Variáveis Compostas

- 6.1 - Variáveis Compostas
 - 6.1.1 - Como acessar os itens de um array?
 - 6.1.2 - Arrays com indexação numérica.
 - 6.1.3 - Arrays Associativos
 - 6.1.4 - Iteração com Arrays.
- 6.2 - Foreach

6.3 - Exercícios Passo a Passo

6.4 - Exercícios de Fixação

7 - Hospedagem de Site

- 7.1 - Hospedagem
 - 7.1.1 - Como hospedar um site
 - 7.1.2 - Espaço em disco
- 7.2 - Transferência de dados
 - 7.2.1 - Escolha uma hospedagem
- 7.3 - Meus Sites
 - 7.3.1 - Gerenciador de sites
- 7.4 - Serviço de FTP
- 7.5 - Exercícios Passo a Passo
- 7.6 - Exercícios de Fixação

8 - Cookies e Sessões

- 8.1 - Cookies
- 8.2 - Criando uma sessão:
- 8.3 - Exercícios Passo a Passo
- 8.4 - Exercícios de Fixação

9 - Integração PHP com HTML

- 9.1 - MVC
- 9.2 - Smarty
 - 9.2.1 - Método Assign()
 - 9.2.2 - Método display()
- 9.3 - Exercícios Passo a Passo
- 9.4 - Exercícios de Fixação

10 - Banco de Dados - Parte 1

- 10.1 - Banco de Dados
- 10.2 - Ativando o MySQL
 - 10.2.1 - APLICATIVO
 - 10.2.2 - ORGANIZAR BANCO DE DADOS
 - 10.2.3 - Dados do tipo String:
 - 10.2.4 - Dados do tipo numérico.
 - 10.2.5 - Dados do tipo data.
 - 10.2.6 - COMANDO PARA CRIAR UMA TABELA.
 - 10.2.7 - COMANDO PARA INSERIR DADOS NA TABELA
 - 10.2.8 - COMANDO PARA RECUPERAR DADOS DE UMA TABELA
- 10.3 - Exercícios Passo a Passo
- 10.4 - Exercícios de Fixação

11 - Banco de Dados - Parte 2

- 11.1 - UPDATE
- 11.2 - DELETE
- 11.3 - Diagrama Entidade Relacional

11.3.1 - CHAVE PRIMÁRIA

11.3.2 - CHAVE ESTRANGEIRA

11.4 - Exercícios Passo a Passo

11.5 - Exercícios de Fixação

12 - Projeto etapa1: Estrutura, conexão, exibir categorias e produtos

12.1 - Gerenciamento do site

12.1.1 - Estrutura de pastas

12.1.2 - Try/Catch

12.1.3 - PDO

12.2 - Exercícios Passo a Passo

12.3 - Exercícios de Fixação

13 - Projeto etapa2: Detalhes do produto e Área Administrativa

13.1 - \$_REQUEST

13.2 - FETCH_OBJ

13.3 - bindParam

13.4 - Exercícios Passo a Passo

13.5 - Exercícios de Fixação

14 - Projeto etapa3: Excluir categoria e Cadastrar Produtos

14.1 - Evento ONCLICK

14.2 - Input Tipo FILE

14.3 - Função MAX()

14.4 - Exercícios Passo a Passo

14.5 - Exercícios de Fixação

15 - Projeto etapa4: Editar e Atualizar Produtos

15.1 - Atualizar imagem

15.2 - Atualizar página

15.3 - Exercícios Passo a Passo

15.4 - Exercícios de Fixação

16 - Projeto etapa5: Excluir Produto e Área de Pedidos.

16.1 - Excluindo uma imagem

16.2 - Função date()

16.3 - Função time()

16.4 - Função rand()

16.5 - Função agregada Sum()

16.6 - Exercícios Passo a Passo

16.7 - Exercícios de Fixação

17 - Projeto etapa6: Excluir Pedido e Cadastrar Cliente

17.1 - Unset

17.2 - Exercícios Passo a Passo

17.3 - Exercícios de Fixação

18 - Projeto etapa7: Listar Pedidos dos Clientes.

18.1 - Comando Null

18.2 - Exercícios Passo a Passo

18.3 - Exercícios de Fixação

19 - Projeto etapa8: Editar e Atualizar

19.1 - ALTER TABLE

19.1.1 - COMANDO ADD

19.1.2 - COMANDO DROP

19.2 - Exercícios Passo a Passo

19.3 - Exercícios de Fixação

20 - Ativar/Desativar Cliente, Login e Hospedagem

20.1 - Exportar dados

20.2 - Importar

20.3 - Exercícios Passo a Passo

20.4 - Exercícios de Fixação

O PHP é uma linguagem de programação de código aberto que ganhou forças ao passar dos anos na criação de scripts. Encontraremos uma grande diferença com outras linguagens: o PHP não roda na máquina do usuário, ele processa tudo o que é necessário no servidor e apenas o HTML é enviado para o usuário.

Para programar em PHP não é necessário um grande editor de textos, se você tiver no seu sistema operacional, o aplicativo Bloco de Notas já basta para programar. O que acontece é que hoje podemos instalar editores que auxiliam na digitação do código, passando dicas. Podemos citar o Notepad++.

O PHP roda nos sistemas operacionais Windows, Linux e Mac. É necessário instalar um conjunto de programas para rodar o PHP. No caso do sistema operacional Windows, devemos instalar o WampServer.

1.1. História do PHP

Nos anos 90 a Internet já possuía diversos sites, podemos dizer que em formato institucional, ou seja, apenas mostrava informações no geral sobre um tipo de negócio, explicando o funcionamento, o que oferecia, como fazer contato. Um certo usuário chamado Rasmus Lerdorf, expôs sua necessidade de saber o total de visitantes na sua página, e como era um conhecedor em programação “C”, desenvolveu uma interface para contabilizar o número de visitas na sua página.



Rasmus batizou sua interface de Personal Home Page Tools, novos recursos foram sendo implementados, como formulários personalizados, aumentando assim a interatividade com os usuários. A interface foi renomeada para Personal Home Page Tools/FI. Mas o nome que ficou marcado e mantido foi PHP.

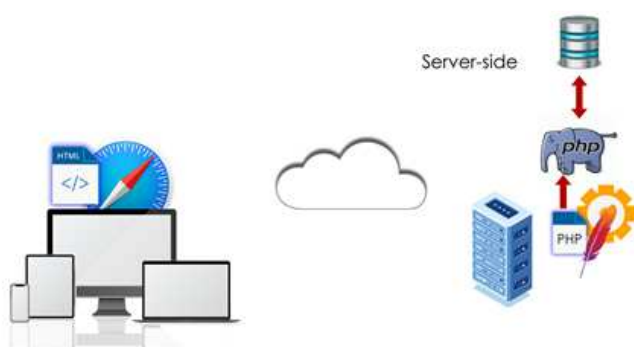
Ao passar do tempo, o PHP recebeu contribuições de outros programadores, assim, ganhou outras funcionalidades tornando-se cada vez mais profissional. Foi lançado neste período a versão 3 da linguagem, e foi tomando proporções maiores em pouco tempo.

1.2. Como funciona o PHP

Antes, vamos entender como funciona a relação entre o cliente e o servidor. Quando criamos uma página no formato HTML, precisamos de uma hospedagem. Agora, quando o cliente faz uma requisição desta página, o servidor envia uma cópia deste arquivo para o cliente, e apenas o seu navegador vai interpretá-lo. O client-side (lado do cliente) é a linguagem que não é processada no servidor, mas no seu browser.



Uma página criada no formato PHP necessita de algumas ferramentas para interpretação do código, essas ferramentas são instaladas no servidor Web e não no navegador do cliente. Quando o cliente faz uma requisição de uma página, o servidor processa o código e apenas envia o resultado no formato HTML para o navegador do cliente, dessa maneira ganhamos na agilidade. Esse procedimento é chamado de server-side (onde o trabalho pesado fica no lado do servidor).



1.3. Os programas

Para estudar PHP é necessário que o seu computador seja transformado em um servidor Web, para isso utilizaremos a ferramenta XAMPP, basta acessar o Google Chrome e pesquisar por XAMPP.

XAMPP é a sigla para X(usado em vários sistemas operacionais, Window, Linus, entre outros), A(Apache HTTP Server, mais conhecido como Apache), P(PHP, linguagem de programação mais popular utilizada no backend), P(Perl, uma linguagem de programação de alto nível) . Vamos falar um pouco sobre cada um deles.

Apache



É um servidor web gratuito, que tem como finalidade disponibilizar arquivos, banco de dados e e-mail. Essas são aplicações que podem acessar arquivos do servidor, o Apache age como intermediador entre o servidor e a máquina do cliente.

MySql



É um sistema gerenciador de banco de dados relacional, permitindo inserir e gerenciar o conteúdo armazenado num banco de dados. O SQL é uma linguagem de consulta estruturada.

PHP



É uma linguagem de código aberto para desenvolvimento das aplicações.

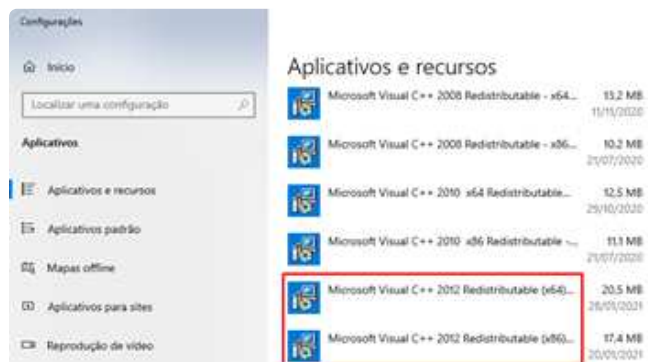
Lembre-se: Todos os programas usados durante o curso são de livre acesso, por isso o curso já possui um link para fazer o download. Procedimentos para baixar e instalar:

Primeiramente é necessário saber para que serve a DLL Visual C++ 2012 Redistributable.

Ela é responsável por instalar os componentes de tempo de execução do Visual

C++, que são bibliotecas fundamentais para executar aplicativos, programas e jogos que pedem a versão 2012. Quando instalamos um programa, por exemplo, ele já vai apresentar erro, caso o computador não tenha a versão exigida. Neste caso, para rodar o WampServer, foi necessário o Visual C++ 2012.

Para conferir se o seu computador possui essa versão, basta acessar o Painel de Controle/ Adicionar ou Remover Programas.



Caso não esteja instalado no seu computador, acesse o site da *Microsoft*, na seção *download*, e baixe conforme a versão do seu sistema operacional.

XAMPP

Para fazer o download, acesse o site https://www.apachefriends.org/pt_br/index.html. Lembre-se que os programas estão nos arquivos auxiliares, peça ajuda para seu instrutor.

1.4. Exercícios Passo a Passo

1. Abra o Google Chrome;
2. Na caixa de pesquisa digite xampp e, em seguida, pressione a tecla Enter;
3. Clique no primeiro resultado www.apachefriends.org;
4. O site foi acessado, agora clique no link XAMPP para Windows;
5. Assim que o download for concluído, abra a pasta Download e confira o arquivo de instalação.

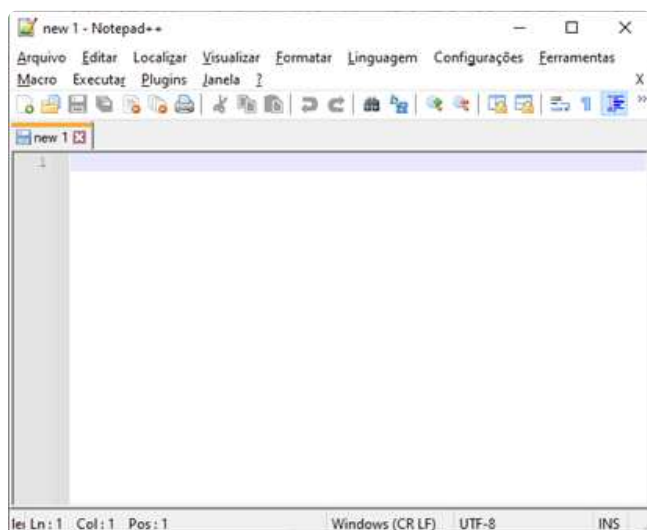
1.5. Exercícios de Fixação

1. Abra o Google Chrome, pesquise e acesse o site do Notepad++, e então baixe o aplicativo.

Quando falamos em programação, além dos códigos, pensamos qual editor de textos usar para programar, podemos citar o tradicional Bloco de Notas do Windows, ou o Dreamweaver, entre outros, mas durante o curso estaremos usando o Notepad++.

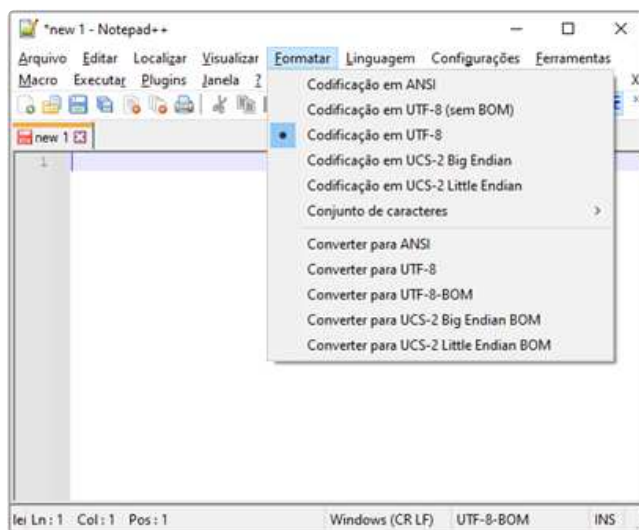
O Notepad++ é muito usado para programar, devido ao fato que tem suporte para diversas linguagens de programação.

2.1. Veja uma ilustração do Notepad++.

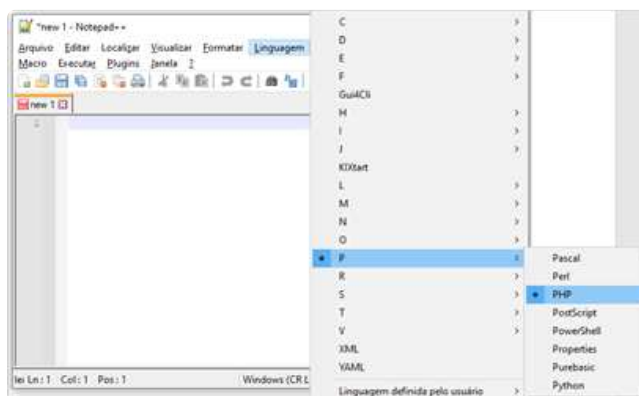


Quando o Notepad++ for aberto, uma página em branco é exibida, porém, é importante fazer os ajustes antes de iniciar a programação.

O primeiro ajuste é referente a codificação dos caracteres, com ele podemos representar qualquer caractere de qualquer idioma. Acesse o menu **Formatar** e selecione a opção **Codificação em UTF-8**.

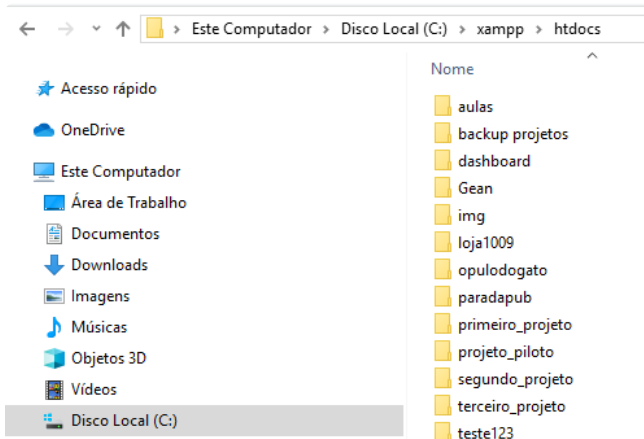


O segundo ajuste é referente a linguagem que iremos usar. Ao ativar, os códigos digitados serão validados. Acesse o menu **Linguagem** e escolha a categoria “P” e opção **PHP**.



2.2. Organização das pastas

No momento de planejar seus projetos, ou mesmo se for praticar, é importante criar as pastas dentro da pasta **xampp/htdocs**, somente neste local o servidor consegue ler e executar os arquivos.

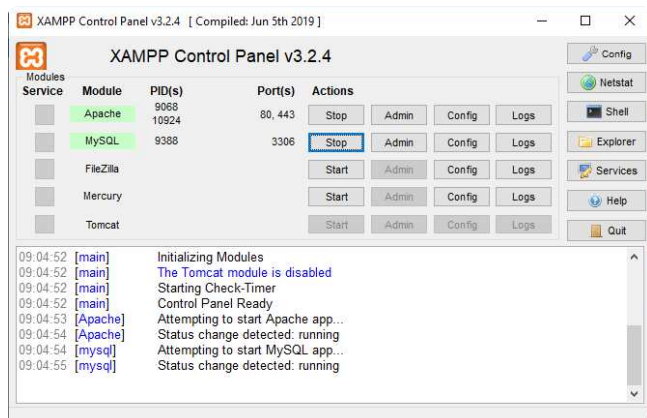


XAMPP

Quando iniciamos o Xampp, logo o ícone será exibido na barra de tarefas do Windows.



É necessário ativar os serviços, Apache e o MySQL, veja a imagem.



2.3. Conceitos básicos de programação

O PHP utiliza as seguintes tags para delimitar o código, tudo que estiver escrito dentro delas será processado no servidor.

No fim de cada linha de código devemos usar o ponto e vírgula, sinalizando fim de instrução.

```
1 <?php
2     $codigo=14;
3     $produto="Bolsa";
4     $preco=65.90;
5 ?>
```

2.3.1. Variáveis

Uma variável é um objeto capaz de reter e representar um valor ou expressão. Para funcionar corretamente, as variáveis precisam ser definidas por nomes e tipos.

Nome: toda variável começa com o cifrão (\$) e uma string. O PHP é case sensitive, ou seja, diferencia os nomes. Exemplo: \$aula e \$AULA. Tenha cuidado ao definir um nome. Evite nomes em maiúsculas, apenas utilize quando for nome composto, exemplo, \$nomeCompleto.

Tipos de dados

STRING: São delimitados por aspas. Exemplo:

```
$curso="Windows";
```

```
?>
```

INTEGER: São números inteiros, pode ser positivo ou negativo. Exemplo:

```
$codigo=22;
```

```
?>
```

FLOAT OU DOUBLE: São valores em ponto flutuante, onde podemos definir casas decimais. Exemplo:

```
$preco=450.90;
```

```
?>
```

BOOLEANO: São os tipos que avaliam se o valor é verdadeiro ou falso.

ARRAY: É uma variável para armazenar vários valores. Funciona como um índice.

```
$carro[1]="Audi";  
$carro[2]="BMW";  
$carro[3]="Honda";  
echo $carro[2];  
?>
```

2.3.2. Operadores Aritméticos

Para realizar as operações matemáticas dentro do nosso projeto, utilizamos esses operadores. Confira:

OPERADOR	NOME	EXEMPLO
+	Soma	\$x + \$y
-	Subtração	\$x - \$y
/	Divisão	\$x / \$y
*	Multiplicação	\$x * \$y

2.3.3. Operador de Incremento

O operador de incremento é representado pelo sinal de soma, duplo, assim: “++”. A finalidade deste operador é de incrementar o valor da variável toda vez que fizermos o uso dela. Exemplo:

Preciso listar 5 números, começando com o número 1.

```
$num=1;  
Echo $num++;  
Echo $num++;  
Echo $num++;  
Echo $num++;  
Echo $num++;  
?>
```

Com esse exemplo vamos visualizar os números 1, 2, 3, 4 e 5.

2.3.4. Operador de Decremento

O operador de decremento é representado pelo sinal de subtração, duplo, assim: “--”. A finalidade deste operador é de ir subtraindo o valor da variável toda vez que fizermos o uso dela. Exemplo:

Preciso listar 5 números, começando com o número 9, o último número da lista vai ser o 5.

```
$num=9;  
Echo $num--;  
Echo $num--;  
Echo $num--;  
Echo $num--;  
Echo $num--;  
?>
```

2.3.5. Operadores de Atribuição

Os operadores de atribuição retornam um valor atribuído a uma variável. Exemplo: Tenho um valor de venda inicial dos meus produtos, e preciso atribuir mais 2.50.

```
$valorVenda=50;  
$valorVenda+=2.50;  
echo $valorVenda;  
?>
```

O resultado deste exemplo vai ser de 52.5.

Veja a tabela abaixo. O mesmo exemplo mostrado acima pode ser usado com qualquer um destes operadores de atribuição.

OPERADOR	NOME
=	Atribuição simples
+=	Atribuição com adição
-=	Atribuição com subtração
*=	Atribuição com multiplicação
/=	Atribuição com divisão

2.4. Exercícios Passo a Passo

1. Este exercício tem como objetivo criar uma lista de clientes, deve aparecer as

seguintes informações: Código do cliente, Nome, Endereço, Cidade, Telefone e e-mail. Fique a vontade para preencher os dados como desejar. Clique em Iniciar, selecione XAMPP Control Panel;

2. Ative a opção Start e, em seguida, feche esta janela;

3. Clique em Iniciar, abra o Notepad++. Se tiver algum arquivo aberto, feche;

4. Clique no menu Formatar, selecione Codificação em UTF-8;

5. Clique no menu Linguagem, selecione a categoria "P", e a opção "PHP";

6. Digite o código conforme o indicado;

```
1 <?php
2 $codigoCli=202;
3 $nomeCli="Fábio Kühn";
4 $enderecoCli="Av. 7 de Setembro";
5 $cidadeCli="São Leopoldo";
6 $telefoneCli="(51) 99800-0000";
7 $emailCli="fabiok@gmail.com";
8
9 echo "Dados do Cliente";
10 echo "<hr/>";
11 echo "Código: ".$codigoCli;
12 echo "<br/>";
13 echo "Nome: ".$nomeCli;
14 echo "<br/>";
15 echo "Endereço: ".$enderecoCli;
16 echo "<br/>";
17 echo "Cidade: ".$cidadeCli;
18 echo "<br/>";
19 echo "Telefone: ".$telefoneCli;
20 echo "<br/>";
21 echo "E-mail: ".$emailCli;
22 ?>
```

7. Clique no menu Arquivo, selecione Salvar, digite exercicio1;

8. Abra o navegador, digite na barra de endereço: localhost/aulas/aula2/exercicio1.php

2.5. Exercícios de Fixação

1. Este exercício tem como objetivo calcular uma lista de compras. Deve aparecer as seguintes informações: Código do produto, Descrição, total em estoque, quantidade vendida, quantidade atual, preço e total de vendas. Fique a vontade para preencher os dados como desejar.

2. -Calcular a quantidade atual com base no total em estoque e na quantidade vendida.

-Calcular o total de vendas com base na quantidade vendida e no preço.

-Salve com o nome fixacao1



Existem situações onde precisaremos comparar valores entre variáveis, para isso usamos os operadores de comparação. Os valores serão comparados, retornando verdadeiro (TRUE) ou falso (FALSE).

3.1. Operador de Comparação

OPERADOR	NOME	SIGNIFICADO	EXEMPLO	RESULTADO
==	Igual a	Verifica igualdade entre dois valores.	\$x == \$y	Verdadeiro se \$x for igual a \$y
===	Idêntico	Além de verificar a igualdade verifica o tipo de dados entre dois valores.	\$x === \$y	Verdadeiro se \$x for igual a \$y e for do mesmo tipo.
!=	Diferente	Verifica a diferença entre dois valores.	\$x != \$y	Verdadeiro se \$x não for igual a \$y
<>	Diferente	Idem ao anterior.	\$x <> \$y	Verdadeiro se \$x não for igual a \$y
>	Maior que	Verifica se um valor é maior que o outro.	\$x > \$y	Verdadeiro se \$x for maior que \$y
>=	Maior ou igual a que	Verifica se um valor é maior ou igual ao outro.	\$x >= \$y	Verdadeiro se \$x for maior ou igual a \$y
<	Menor que	Verifica se um valor é menor que o outro.	\$x < \$y	Verdadeiro se \$x for menor que \$y
<=	Menor ou igual a que	Verifica se um valor é menor ou igual ao outro.	\$x <= \$y	Verdadeiro se \$x for menor ou igual a \$y

3.2. Operadores lógicos

Existem situações onde precisaremos comparar diversas expressões, retornando apenas um resultado, Verdadeiro (True) ou Falso (False).

Veja a tabela abaixo:

OPERADOR	NOME	EXEMPLO	RESULTADO
AND	E	(22 > 18) AND (123 == 123)	Verdadeiro se 22 for maior que 18 e 123 for igual a 123.
&&	E	(22 > 18) && (123 == 123)	Verdadeiro se 22 for maior que 18 e 123 for igual a 123.
OR	OU	(22 > 18) OR (123 == 123)	Verdadeiro se 22 for maior que 18 ou 123 for igual a 123.
	OU	(22 > 18) (123 == 123)	Verdadeiro se 22 for maior que 18 ou 123 for igual a 123.

Vamos entender como os operadores funcionam:

E = AND - &&

OU = OR - ||

Os operadores fazem a mesma coisa, o que acontece é que o && e || tem maior prioridade na avaliação do que AND e OR.

Quando usamos os operadores AND ou && todas as expressões verificadas precisam ser verdadeiras para que o resultado apareça.

Quando usamos os operadores OR ou ||, se apenas uma expressão for verdadeira o resultado será exibido como verdadeiro.

Exemplo1:

Se o programa analisar a profissão e a cidade de um usuário, e estipular que somente ele terá acesso aos produtos do catálogo, caso seja vendedor e resida na cidade de Porto Alegre.

Para essa situação devemos utilizar o operador AND ou &&, ficando da seguinte maneira:

Profissão=="vendedor" AND
cidade=="Porto Alegre".

Exemplo2:

Para ter acesso ao relatório de vendas de uma empresa, o programa deve fazer a seguinte análise: somente será permitido acesso para o gerente Paulo ou para o vendedor Pedro.

Gerente=="Paulo" OR vendedor=="Pedro"

Somente será liberado mediante a uma das alternativas propostas.

Sobre o método POST:

É comum acessar um site e passar por uma tela de login, ou até mesmo acessar o nosso e-mail, ou realizar um cadastro, essas informações devem ser sigilosas. Para isso acontecer devemos usar o método de envio POST, que envia os parâmetros pela URL, escondendo eles.

Propriedade Action:

É nesta propriedade que definimos o script de destino, no caso para onde enviamos os dados do método.

Recebendo dados do formulário:

\$_POST: Usado para coletar dados de um formulário.

Exemplo:

Como serão criadas as variáveis para receber os dados deste formulário.

```
$nome=$_POST["nome"];
```

```
$email=$_POST["email"];
```

3.3. Estruturas de decisão

Controle de fluxo é a habilidade de ajustar a maneira como um programa realiza suas tarefas. Por meio de instruções especiais, chamadas, comandos, essas tarefas podem ser executadas seletivamente, repetidamente ou excepcionalmente. Não fosse o controle de fluxo, um programa poderia executar apenas uma única sequência de tarefas, perdendo completamente uma das características mais interessantes da programação: a dinâmica.

O comando IF é comum em muitas linguagens de programação, sua função é verificar se uma condição é verdadeira ou falsa.

A forma mais simples de controle de fluxo é o IF/ELSE. Ele é empregado para executar seletivamente ou condicionalmente um outro

comando, mediante um critério de seleção. Esse critério é dado por uma expressão, cujo valor resultante deve ser um dado do tipo booleano, isto é, true ou false. Se esse valor for true, então o outro comando é executado; se for false, a execução do programa segue adiante. A sintaxe para esse comando é:

```
if ([condição]){  
  
    [comando]; // Executado se a condição for true  
  
}
```

Uma variação do IF/ELSE permite escolher alternadamente entre dois outros comandos à executar. Nesse caso, se o valor da expressão condicional que define o critério de seleção for true, então o primeiro dos outros dois comandos é executado, do contrário, o segundo.

```
if([condição]){  
  
    [comando 1]; // Executado se a condição for true  
  
}else{  
  
    [comando 2]; // Executado se a condição for false  
  
}
```

Exemplo 1:

Vamos classificar uma pessoa como sendo criança.

```
$idade = 10;  
  
if($idade<=12){  
  
    echo "Criança";  
  
}
```

No exemplo acima, se a idade for maior que 12, nenhuma mensagem será exibida.

Exemplo 2:

Vamos classificar uma pessoa como adolescente ou adulta.

```
$idade = 16;

if($idade<18){

echo "Adolescente";

}else{

echo "Adulto";

}
```

No exemplo acima, verificamos se a pessoa é adolescente ou adulta.

Exemplo 3:

Vamos classificar uma pessoa como criança, adolescente ou adulta.

```
$idade = 16;

if($idade<12){

echo "Criança";

}else if($idade<18){

echo "Adolescente";

}

echo "Adulto";

}
```

Exemplo 4:

Vamos verificar se uma pessoa é maior de idade e do sexo masculino. Para preencher a vaga de emprego, somente com esses dois critérios ela segue adiante na seleção.

```
$idade = 20;

$sexo = "Masculino";

if($idade > 18 AND sexo=="Masculino"){

echo "Próxima etapa";

}else{
```

```
echo "Etapa encerrada";
```

```
}
```

Exemplo 5:

Vamos verificar se uma pessoa é maior de idade ou do sexo feminino. Para preencher a vaga de emprego, com qualquer um destes critérios ela segue adiante na seleção.

```
$idade = 15;

$sexo = "Feminino";

if($idade > 18 OR $sexo == "Feminino"){

echo "Próxima etapa";

}else{

echo "Etapa encerrada";

}
```

Exemplo 6:

Vamos verificar se um determinado usuário preencheu corretamente seus dados de acesso, login e senha para prosseguir a etapa.

```
$login = "adm";

$senha = "a123m";

if($login == "adm" && $senha == "a123m"){

echo "Próxima etapa";

}else{

echo "Confira os dados";

}.
```

3.4. Exercícios Passo a Passo

1. Este exercício tem como objetivo calcular um desconto de R\$ 3,00 (três reais) para compras acima de 5 unidades, caso contrário será calculado o preço normal. O leite será o produto de exemplo. Se necessário ative o Xampp.

2. Abra o Notepad++;

3. Clique em Formatar, Codificação em UTF-8;

4. Clique em Linguagem, na categoria "H" escolha HTML;

5. Crie um formulário conforme o indicado;

Informe o Produto

Informe o Preço do Produto

Informe o Valor de Desconto

Informe a Quantidade

Enviar

6. Salve como formulario-desconto;

7. Crie um novo documento. Clique em Formatar, Codificação em UTF-8;

8. Clique em Linguagem, na categoria "P" escolha PHP;

9. Digite o código conforme o indicado;

```
1 <?php
2 $produto=$_POST["produto"];
3 $preco=$_POST["preco"];
4 $desconto=$_POST["desconto"];
5 $quantidade=$_POST["quantidade"];
6
7
8 if($quantidade>5){
9     $total=($quantidade*$preco)-$desconto;
10    $msg="Total Com desconto: ".$total;
11 }else{
12     $total=($quantidade*$preco);
13     $msg="Total Sem desconto: ".$total;
14 }
15 echo "Produto: ".$produto."<br/>";
16 echo "Preço: ".$preco."<br/>";
17 echo "Quantidade: ".$quantidade."<br/>";
18 echo "Desconto: ".$desconto."<br/>";
19 echo $msg;
20 ?>
```

10. Salve como resolve-formulario-desconto;

11. Abra o navegador e digite localhost/aulas, faça os testes.

3.5. Exercícios de Fixação

1. Este exercício tem como objetivo avaliar as permissões dos usuários, se a permissão for igual a 1, exibir Sistema Completo Liberado, se a permissão for igual a 2, exibir Apenas Cadastro e Consulta estão liberados.

2. Será necessário um formulário para preencher o nome de usuário e outro para informar o nível de acesso. Salvar como formulario-acesso;

3. Criar um php para avaliar os dados de acesso e exibir as mensagens conforme o nível. Salvar como resultado-formulario-acesso.

Estudaremos a estrutura de repetição `for`, a estrutura mais utilizada pelos desenvolvedores na construção de laços de repetição que possuem uma quantidade de ciclos pré-definidos. Os "Loops", conhecidos como laços, são estruturas de repetição, utilizados para executar repetidamente uma instrução ou bloco de instrução, enquanto determinada condição `for` satisfatória.

As estruturas de repetições, possuem quatro elementos fundamentais: inicialização, condição, corpo e iteração. A inicialização compõe-se de todo código que determina a condição inicial da repetição. A condição é uma expressão booleana avaliada após cada leitura do corpo e determina se uma nova leitura deve ser feita ou se a estrutura de repetição deve ser encerrada. O corpo compõe-se de todas as instruções que são executadas repetidamente. A iteração é a instrução que deve ser executada depois do corpo e antes de uma nova repetição.

4.1. Estrutura de repetição

FOR: A sintaxe do laço `for` é a seguinte:

```
for(condição inicial; teste condicional; após iteração){
```

```
    //bloco de comandos;
```

```
}
```

O laço funciona da seguinte maneira:

O laço se inicia pela condição inicial, geralmente se inicia o contador. Esse primeiro estágio SEMPRE acontece.

O segundo estágio é o teste da condição do laço, um simples teste condicional. Caso seja verdade, o código é executado.

Ao término de execução do código, ocorre o fator de mudança, que geralmente é um incremento ou decremento, sempre após cada iteração do looping.

Depois, a condição é testada novamente. Caso retorne 'true', o código é executado.

Ao término de execução do código, sempre ocorre o fator de mudança... e assim sucessivamente. Veja o exemplo prático:

```
for($i=1; $i<=5; $i++){  
  
    echo "Número: ".$i;  
  
}
```

O exemplo acima tem como propósito criar uma lista numerada de 1 a 5, para isso a variável `$i` inicia com o valor 1, a condição avalia o próximo número até que alcance 5, através do incremento, entenda:

`$i = 1`

`1 = 1 + 1`

`2 = 2 + 1`

`3 = 3 + 1`

`4 = 4 + 1`

`5`

Quando a condição `for` satisfatória, conforme a condição exige, o programa lista dos e encerra, sai `for` do laço.

Nota: O `FOR()` é usado em outras situações que certamente utilizaremos mais tarde.

4.2. Switch...case

O comando Switch é um comando de tomada de decisão, controla o fluxo do programa e permite que seja definido um código diferente em várias condições. Através deste comando conseguimos comparar o valor de uma variável aos valores em cada case.

Se o valor de um determinado case for igual ao da variável, o comando é executado.

Break

O comando break interrompe a funcionalidade do switch no momento que o valor de case for igual ao da expressão.

Estrutura do Switch:

```
switch(variável){
```

```
case valor1:
```

```
comando;
```

```
break;
```

```
case valor2:
```

```
comando;
```

```
break;
```

```
case valor3:
```

```
comando;
```

```
break;
```

```
default:
```

```
comando;
```

```
}
```

4.3. Exercícios Passo a Passo

1. Este exercício tem como objetivo criar uma lista com todas as séries do ensino fundamental, de 1 a 9, onde o aluno vai selecionar a série que começará a frequentar. Deve aparecer na tela como resposta o nome e a série do aluno. Inicie os serviços do Xampp.

2. Abra o Notepad++. No menu formatar, escolha Codificação em UTF-8. No menu Linguagem, na categoria P, escolha PHP;

3. Salve o documento como exercicio4 na pasta "xampp\htdocs\aulas\aula4";

4. Digite o código conforme o indicado, salve e faça os testes no localhost.



```
1 <?DOCTYPE HTML>
2 <html>
3 <head>
4 <title>Formulário de Alunos</title>
5 <meta charset="utf-8" />
6 </head>
7 <body>
8 <div>
9 <form name="meu_form" method="POST" action="exercicio4.php">
10 <p><input type="text" placeholder="Nome" required="required" name="nome"/></p>
11 <p>
12 <select name="serie">
13 <option value="0">Série</option>
14 <?php for($s=1;$s<=9;$s++) { ?>
15 <option value="<?php echo $s;?>"><?php echo $s;?></option>
16 <?php } ?>
17 </select>
18 </p>
19 <p><input type="submit" value="Enviar"/></p>
20 </form>
21 </div>
22 <div>
23 <?php
24 if(!empty($_POST)) {
25 $nome=$_POST["nome"];
26 $serie=$_POST["serie"]; ?>
27 <p><?php echo "Aluno: ",$nome;?></p>
28 <p><?php echo "Série: ",$serie;?></p>
29 <?php } ?>
30 </div>
31 </body>
32 </html>
```

4.4. Exercícios de Fixação

1. Este exercício tem como objetivo utilizar o switch e criar uma lista que mostre o percentual de desconto conforme a faixa de salário escolhida. Veja as opções:

1. até um salário mínimo (R\$ 1.045)
2. para quem ganha entre R\$ 1.045,01 R\$ e 2.089,60.
3. para quem ganha entre R\$ 2.089,61 e R\$ 3.134,40.
4. para quem ganha entre R\$ 3.134,41 e R\$ 6.101,06.

Se escolher a opção 1: deve aparecer 7,5% de desconto sobre o salário | opção 2: deve aparecer 9% de desconto sobre o salário | opção 3: deve aparecer 12% de desconto sobre o salário | opção 4: deve aparecer 14% de desconto sobre o salário.

O `while` é um comando que manda um bloco de código ser executado enquanto uma condição não for satisfatória. Assim, permite que sejam criados loops de execução. O `while` é um comando muito útil, mas pode ser perigoso, pois, se não tratarmos o critério de parada corretamente, o laço pode não ter fim e o programa não faz o que deveria, podendo entrar em loop infinito, como é chamado.

Frequentemente, em nossas aplicações, precisamos repetir a execução de um bloco de códigos do programa até que determinada condição seja verdadeira, ou até que uma quantidade de vezes seja satisfeita.

While – Esta instrução é usada quando não sabemos quantas vezes um determinado bloco de instruções precisa ser repetido. Com ele, a execução das instruções vai continuar até que uma condição seja verdadeira. A condição a ser analisada para a execução do laço de repetição deverá retornar um valor booleano.

Veja a sintaxe:

```
While(teste condicional){

//comandos;

// serão executados enquanto o teste
condicional for igual a verdadeiro (true)

}
```

Perceba que, somente se a condição for verdadeira, o corpo do laço de repetição com seus respectivos comandos será executado. Portanto, o conteúdo será repetido até que esta condição não seja mais verdadeira.

Veja o exemplo:

Será calculado R\$ 10,00 sobre o valor do salário, enquanto ele for menor que R\$ 1400,00.

```
<?php
    $salario=1100;
    while($salario<=1400){
        $salario+=10;
        echo $salario."<br/>";
    }
?>
```

No exemplo acima, o salário é iniciado com o valor de R\$ 1100, quando entra no laço a condição é avaliada, e enquanto for menor ou igual a R\$ 1400, será atribuído R\$ 10,00 sobre o valor, quando chegar em R\$ 1400, será atribuído o valor de R\$ 10,00 e o laço é encerrado.

5.1. Manipulação de Strings

A string é um dos tipos de dados que usamos frequentemente, podemos citar nomes, e-mails, senhas, frases, entre outros casos.

Vamos conhecer algumas funções para a manipulação de strings.

strtolower()

O `strtolower` recebe uma string e retorna o mesmo valor com todas as letras convertidas em minúsculas.

Exemplo 1:

```
<?php
    $nome="ANA MARIA";
    $resultado=strtolower($nome);
    //exibe: ana maria.
?>
```

strtoupper()

O `strtoupper` recebe uma string e retorna o mesmo valor com todas as letras convertidas em maiúsculas.

Exemplo 1:

```
<?php
    $nome="ana maria";
    $resultado=strtoupper($nome);
    // exibe: ANA MARIA
?>
```

substr()

O substr recebe uma string e retorna uma parte, a partir de uma posição e um comprimento.

Exemplo 1:

```
<?php
    $frase="Receita da Semana";
    $parte=substr($frase,10);
    // exibe: Semana
?>
```

Exemplo 2:

```
<?php
    $frase="Receita da Semana";
    $parte=substr($frase,0,7);
    // exibe: Receita
?>
```

strlen()

O strlen recebe uma string e retorna a quantidade de caracteres.

Exemplo 1:

```
<?php
    $frase="Receita da Semana";
    $tamanho=strlen($frase);
    // exibe: 17
?>
```

str_replace()

O str_replace serve para substituir um trecho de texto por outro.

str_replace (valor a ser alterado, novo valor, texto que contém o valor a ser alterado)

Exemplo 1:

```
<?php
    $frase="Receita da Semana";
    $novoTexto=str_replace("da Semana","do Mês",$frase);
    echo $novoTexto;
    // exibe: Receita do Mês
?>
```

trim()

O trim tem como finalidade retirar o espaço no início e no final de uma string.

Exemplo 1:

```
<?php
    $frase="  Receita da Semana  ";
    $espaco=trim($frase);
    echo "Hoje tem: ".$espaco.", acompanhe.";
    // exibe: Hoje tem: Receita da Semana, acompanhe.
?>
```

5.2. Funções em PHP

Uma função, auxilia muito no dia a dia da programação, consiste em um bloco de código nomeado, com um objetivo específico.

Para criar uma função devemos entender a forma de declará-la.

Iniciamos com a palavra reservada function, seguida do nome da função e de sua lista de argumentos, o corpo da função é delimitado por chaves.

Estrutura básica:

```
function      nome_função($parâmetro1,
$parâmetro2...$parâmetroN)
```

```
{
    Comandos;
}
```

É importante saber que o function é padrão, o "nome_função" deve ser associado ao que ela vai executar.

As funções podem ser classificadas com retorno ou sem retorno, as sem retorno apenas executam os comandos sem devolver um valor específico como resultado. Quando uma função possui retorno, resultam diretamente em um determinado valor.

Função sem parâmetro e sem retorno:

```
<?php
function msg(){
    echo "Olá Mundo!";
}
msg();
?>
```

Resultado:

Olá Mundo!

Função com parâmetro e sem retorno:

```
<?php
function msg($nome){
    echo "Olá Mundo!";
    echo $nome;
}
msg("Ana Maria");
?>
```

Resultado:

Olá Mundo! Ana Maria

Função com parâmetro e com retorno:

```
<?php
function somar($v1r1, $v1r2){
    return $v1r1 + $v1r2;
}
$res=somar(10, 35);
echo $res;
?>
```

Resultado:

Total: 35

Utilizamos o return para definir o resultado da função e sinalizar o fim da execução desta.

5.3. Exercícios Passo a Passo

1. Este exercício tem como objetivo criar uma lista de 30 números em ordem decrescente. Inicie os serviços do Xampp.

2. Abra o Notepad++;

3. Digite o código conforme o indicado;

```
1 <?php
2 $num=30;
3 while($num >= 1){
4     echo $num."<br/>";
5     $num--;
6 }
7 ?>
```

4. Salve como exercício5_1

5. Abra o Google Chrome e acesse localhost/aulas/aula5 e confira o resultado.

5.4. Exercícios de Fixação

1. Este exercício tem como finalidade converter duas strings, um nome e uma frase. Utilize as funções strtoupper e strtolower.

2. Na variável nome, digite: DAIANA DA SILVA e na variável frase, digite: O medo cega os nossos sonhos.

3. Converta o nome para minúsculas e a frase para maiúsculas.

Um Array é uma Estrutura de Dados, isto significa que é uma forma de representar, manipular e armazenar dados em um computador. Um Array também é chamado de Variável Composta Homogênea, isto significa que é um tipo de variável que consegue armazenar mais de um dado de um único tipo. Composta: mais de um dado; Homogênea: um único tipo. Exemplos: armazenar 10 salários, armazenar 15 nomes, armazenar 20 notas, etc. Existem vários "tipos" de Arrays, hoje veremos o Array denominado de "Variáveis Compostas Homogêneas Unidimensionais".

6.1. Variáveis Compostas

Variáveis Compostas Homogêneas Unidimensionais (Array de uma dimensão).

São variáveis compostas que necessitam de apenas um índice para individualizar um elemento do conjunto. Essas variáveis também são chamadas de Vetores.

Índice	i	i+1	i+2	i+3	i+4	i+5
Vetor	Laranja	Mamão	Goiaba	Abacaxi	Morango	Manga
Posição	0	1	2	3	4	5

A Figura acima ilustra um Array de uma dimensão com seis posições. Cada quadrado desse Array é correspondente a um espaço de memória que armazena um dado. Como o Array tem seis posições, então, são seis espaços de memória que serão utilizados para armazenar esses dados.

O índice nos ajuda a percorrer o Array, permitindo que o elemento daquela determinada posição possa ser armazenado, acessado, atualizado, excluído, enfim, manipulado. O índice pode "andar" da esquerda pra direita, por isso vocês veem escrito "i", "i +

1", e assim por diante. O índice "i" é a variável de incremento (contador) e a cada iteração que fizermos, ela é somada de um, para podermos caminhar no array.

Exemplo 1:

```
$frutas=array("Laranja","Mamão","Goiaba",,
```

Exemplo 2:

```
$frutas=[  
"Laranja","Mamão","Goiaba","Abacaxi","Morango"
```

As duas formas estão corretas, o segundo exemplo possui um código mais limpo.

6.1.1. Como acessar os itens de um array?

A forma mais simples é indicar a posição do vetor conforme mostra abaixo.

```
Echo $frutas[2];
```

Resultado: Goiaba

6.1.2. Arrays com indexação numérica.

```
$frutas[0]="Laranja";
```

```
$frutas[1]="Mamão";
```

```
$frutas[2]="Goiaba";
```

6.1.3. Arrays Associativos

Esse tipo de array utiliza caracteres para indexar o conteúdo.

```
$frutas["Laranja"]=1.50;
```

```
$frutas["Mamão"]=2.60;
```

```
$frutas["Goiaba"]=3.80;
```

6.1.4. Iteração com Arrays.

Exemplo 1:

Para percorrer um array, podemos utilizar o comando "for".

```
1 <?php
2 $frutas=["Laranja","Mamão","Goiaba","Abacaxi","Morango","Manga"];
3
4 for($i=0;$i<count($frutas);$i++){
5     echo $frutas[$i]."<br/>";
6 }
7 ?>
```

A função **count()** auxilia contando quantas posições há em um determinado array.

Exemplo 2:

A função **sort()** classifica o conteúdo de um array em ordem crescente.

```
1 <?php
2 $frutas=["Laranja","Mamão","Goiaba","Abacaxi","Morango","Manga"];
3 sort($frutas);
4 for($i=0;$i<count($frutas);$i++){
5     echo $frutas[$i]."<br/>";
6 }
7 ?>
```

Exemplo 3:

A função **rsort()** classifica o conteúdo de um array em ordem decrescente.

```
1 <?php
2 $frutas=["Laranja","Mamão","Goiaba","Abacaxi","Morango","Manga"];
3 rsort($frutas);
4 for($i=0;$i<count($frutas);$i++){
5     echo $frutas[$i]."<br/>";
6 }
7 ?>
```

Exemplo 4:

A função **array_sum()** tem como finalidade calcular a soma dos elementos de um array.

```
1 <?php
2 $frutas["Laranja"]=1.20;
3 $frutas["Mamão"]=2.50;
4 $frutas["Goiaba"]=5.20;
5 $frutas["Abacaxi"]=3.60;
6 $frutas["Morango"]=1.90;
7 $frutas["Manga"]=2.40;
8
9 $total=array_sum($frutas);
10 echo $total;
11
12 ?>
```

Sizeof()

Retorna o número de elementos de um array.

```
1 <?php
2 $produtos=["café","açúcar","farinha","pão","queijo"];
3 $tamanho=sizeof($produtos);
4 echo $tamanho;
5 ?>
```

Resultado: 5

6.2. Foreach

A utilização é bem simples e direta. Vamos usar essa variante do for, que percorre sempre, do começo ao fim, todos os elementos de um Array.

É bem útil, também, em termos de precaução e organização, pois alguns programadores não gostam de usar o índice 0, usam direto o índice 1 do array, ou às vezes nos confundimos e passamos (durante as iterações) do limite do array. Isso pode ser facilmente evitado usando o laço for modificado para percorrer os elementos de um array.

Sintaxe:

```
<?php
foreach ($array as $value) {
    //código a ser executado;
}
```

Exemplo:

```
<?php
$carros = array("fusca","gol","uno","chevete");
sort($carros);
foreach($carros as $lista){
    echo $lista."<br/>";
}
```

6.3. Exercícios Passo a Passo

1. Este exercício tem como finalidade criar um array de 12 produtos com preço. Exibir a quantidade de produtos na lista e calcular o total dos preços. Se necessário, ativar o Xampp;

2. Abrir o Notepad++ e fazer as devidas configurações;

3. Digite o código conforme o indicado;

```

1 <?php
2 $produtos["bolsa"]=60.00;
3 $produtos["pulseira"]=5.50;
4 $produtos["corrente"]=7.50;
5 $produtos["brincos"]=12.00;
6 $produtos["mochila"]=23.00;
7 $produtos["caderno"]=17.90;
8 $produtos["estojo"]=5.50;
9 $produtos["pendrive"]=32.00;
10 $produtos["fone de ouvido"]=45.00;
11 $produtos["cadeira plástica"]=55.00;
12 $produtos["caneta"]=2.40;
13 $produtos["teclado 102 teclas"]=65.00;
14
15 echo count($produtos). " produtos <br/>";
16
17 echo "<br/>";
18
19 foreach($produtos as $indice=>$conteudo){
20     echo "Preço do ".$indice." : ".$conteudo."<br/>";
21 }
22
23 $total=array_sum($produtos);
24
25 echo "<br/>";
26
27 echo "Total: ".number_format($total,2);
28
29 ?>

```

4. Salvar como `exercicio_array` na pasta `aula6`;

5. Abrir o `localhost/aula6` e executar o arquivo `exercicio_array`.

6.4. Exercícios de Fixação

1. Este exercício tem como finalidade criar um array com uma lista de 10 clientes. Exibir a quantidade de clientes e classificar em ordem alfabética.

A hospedagem de sites é um serviço que funciona online, como um aluguel, disponível 24 horas, permitindo armazenar seus sites e/ou aplicações na web. As vantagens de anunciar um serviço ou divulgar qualquer tipo de negócio cresce todos os dias na internet, e ter um site é estar presente no mundo digital. O que acontece é que ao criar um site ou uma aplicação, vários arquivos são gerados como, imagens, textos, vídeos e banco de dados.

7.1. Hospedagem



Nesta aula iremos recordar alguns assuntos já vistos no curso de HTML e CSS.

Para que o seu projeto seja visualizado na internet é necessário a hospedagem em um servidor, que é um computador ligado 24 horas. Além de manter o site funcionando, ainda tem como função proteger de ataques maliciosos.

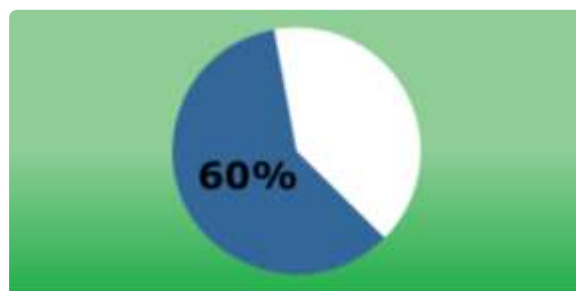
7.1.1. Como hospedar um site

O primeiro passo é registrar o domínio que será o seu endereço na web.

Quando hospedamos um site, devemos saber quais os recursos que estão sendo oferecidos por cada empresa. Vamos conhecer alguns dos principais recursos: Espaço em disco, transferência, domínios e e-mail.

7.1.2. Espaço em disco

É importante verificar se o espaço que é disponível no servidor será suficiente para armazenar os arquivos do seu site.



Observação: Quando uma empresa oferece espaço em disco ilimitado, está informando que não há um espaço definido para armazenamento do seu site. Tenha cuidado.

7.2. Transferência de dados

Quando uma pessoa acessa ou navega em seu site ou quando você o atualiza, a transferência ou tráfego diz respeito à quantidade de dados que serão transferidos.

Assim como acontece com o espaço em disco, também acontece com a transferência de dados. Uma delas é o número de acessos simultâneos no seu site. Quando um site é pequeno, não costuma ter problemas, mas se o seu tráfego for crescente, é necessário contratar um plano de hospedagem mais robusto.



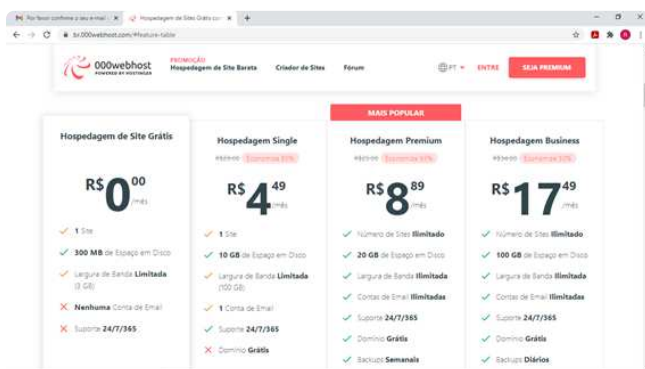
7.2.1. Escolha uma hospedagem

Para praticar, podemos fazer uso de servidores de hospedagem gratuitos, onde conseguiremos testar nossos sites antes de entregar para o cliente, navegando em todas as páginas e conferindo todos os itens.

Um dos servidores de hospedagem que podemos sugerir, entre diversos que existem, é o <https://br.000webhost.com>

Funcionamento da hospedagem:

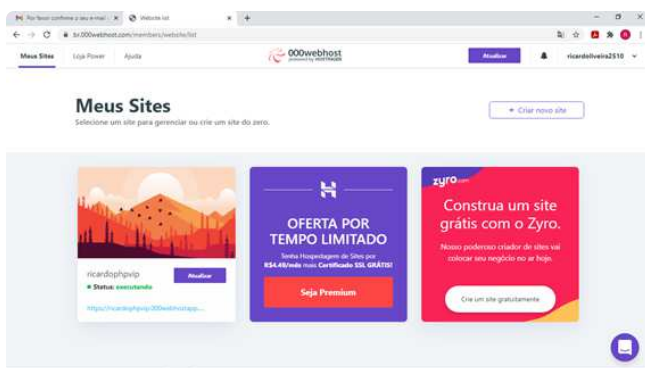
A revenda de hospedagem webhost, que estamos utilizando no curso, possui os seguintes planos, conforme imagem abaixo. Os valores informados variam de acordo com a revenda.



Plano	Preço (mês)	Benefícios
Hospedagem de Site Grátis	R\$ 0,00	1 Site, 300 MB de Espaço em Disco, Largura de Banda Limitada (3 GB), Nenhuma Conta de E-mail, Suporte 24/7/365, Domínio Grátis
Hospedagem Single	R\$ 4,49	1 Site, 10 GB de Espaço em Disco, Largura de Banda Limitada (100 GB), 1 Conta de E-mail, Suporte 24/7/365, Domínio Grátis, Backups Semanais
Hospedagem Premium	R\$ 8,89	Número de Sites Ilimitado, 20 GB de Espaço em Disco, Largura de Banda Ilimitada, Contas de E-mail Ilimitadas, Suporte 24/7/365, Domínio Grátis, Backups Diários
Hospedagem Business	R\$ 17,49	Número de Sites Ilimitado, 100 GB de Espaço em Disco, Largura de Banda Ilimitada, Contas de E-mail Ilimitadas, Suporte 24/7/365, Domínio Grátis, Backups Diários

A revenda oferece a seus usuários a possibilidade de adquirir uma hospedagem e um domínio gratuito com intensão de realizar seus testes e no futuro contratar para seu próprio uso ou para indicar a um cliente. Assim como a webhost existem outros serviços gratuitos de hospedagem.

Quando acessamos o painel, surge uma área que precisamos explorar.



7.3. Meus Sites

Esta página apresenta, na primeira imagem, o site que será desenvolvido pelo usuário. Na segunda imagem, a revenda oferece oferta para cliente premium. Na última imagem, oferece um recurso que permite ao usuário criar um site seguindo passo a passo do próprio sistema da revenda.

Vamos focar na primeira área.



A imagem nos apresenta algumas opções, confira:

Ao passar o ponteiro do mouse na imagem surgem as opções Gerenciador de sites e Ações Rápidas.

O Gerenciador de sites acessa uma nova página com uma série de recursos.

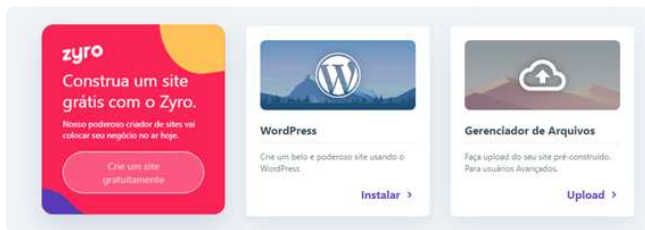
Em Ações Rápidas podemos visualizar o site em uma nova aba ou exibir uma caixa de diálogo com informações do site.

7.3.1. Gerenciador de sites

Acessando esse recurso são apresentadas algumas categorias à esquerda.

Crie um site

Permite criar um site através do construtor oferecido pela revenda, instalar o WordPress ou acessar o gerenciador de Arquivos.



Ferramentas

Permite registrar um domínio, acesso ao gerenciador de arquivos, permite criar e editar um banco de dados e ainda gerenciar seus e-mails.

Configurações de site

Geral: permite alterar informações, como: nome do site, senha, entre outros recursos.

Estatísticas: exibe um gráfico que demonstra um panorama quanto à capacidade de transferência de dados, espaço em disco, quantidade de arquivos, cota de e-mails. São estatísticas para melhor controle do seu site.

Comunidade de Ajuda

Apresenta o Discord, o Blog e a Comunidade Do Fórum como uma forma de buscar auxílio sobre alguma dúvida que surgir quanto ao uso do painel.

Aprenda a programar

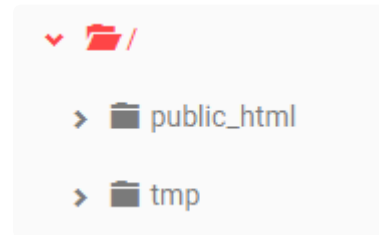
Acessa uma página onde você aprenderá HTML e CSS.

7.4. Serviço de FTP

FTP é um protocolo de transferência de arquivos. Basicamente é um serviço que permite ao usuário enviar ou receber documentos por meio de um endereço no navegador ou um software instalado.

A transferência é realizada entre um servidor e um cliente.

O gerenciador de arquivos apresenta a seguinte estrutura:



A pasta public_html é o local onde todos os nossos arquivos devem permanecer para o funcionamento do site.

Quando abrimos a pasta public_html, automaticamente os arquivos salvos serão listados. Neste caso temos o index.php e o estilo1.css.

<input type="checkbox"/>	Name ▼	Size	Date
<input type="checkbox"/>	.htaccess	0.2 kB	2021-02-25 06:41:00
<input type="checkbox"/>	estilo1.css	1.5 kB	2021-02-25 07:04:00
<input type="checkbox"/>	index.php	1.8 kB	2021-02-25 07:04:00

O cabeçalho do painel apresenta uma série de recursos que estaremos listando abaixo.



New File: Permite criar um novo documento.



New Folder: Permite criar uma nova pasta.



Search: Permite pesquisar um determinado arquivo dentro da estrutura.



Upload Files: Permite localizar os arquivos que serão adicionados no site.



Refresh: Atualiza a lista de arquivos.



Ícones: Altera a forma de exibição dos nossos ícones.



Language: permite alterar o idioma do gerenciador.



Logout: Encerrar as atividades no gerenciador.

7.5. Exercícios Passo a Passo

1. Este exercício tem como objetivo criar sua conta no webhost. Acesse o Google Chrome e digite: <https://br.000webhost.com/>

2. É importante criar um e-mail que será usado para hospedar seus projetos e exercícios (no Gmail ou usar uma conta existente). Clique em Comece Agora e, em seguida, clique em Registre-se grátis;

3. Informe o seu e-mail e crie uma senha forte usando maiúsculas, símbolos como arroba, exclamação e números e, em seguida, clique em cadastre-se;

4. Abra o seu e-mail e confirme a conta para liberação de acesso;

5. Clique em Get Started;

6. Clique em Outro;

7. Clique em Pular;

8. Este passo é importante, pois define o nome do seu site, que será usado no curso e para praticar durante o dia. Crie um nome, exemplo: fulanophpvip e tente usar a mesma senha para acesso ao painel criado anteriormente;

9. Na área "Mais Popular" clique em Selecionar;

10. Acessando a outra área de controle, digite: <https://br.000webhost.com/login-cpanel?from=panel> e pressione enter;

11. Agora Informe Seus Dados

7.6. Exercícios de Fixação

1. Este exercício tem como objetivo fazer upload de um arquivo dentro do servidor que acabamos de contratar;

2. Dentro do gerenciador de arquivos clique na pasta "public_html", que está no lado esquerdo do painel;

3. Irá aparecer apenas o documento chamado .htaccess. Clique no botão Upload Files (seta apontando para nuvem), depois da lupa;

4. Acesse a pasta Arquivos auxiliares/Aula7/ e selecione o index.php e o estilo1.css e abra-os;

5. Acesse o seu site e confira o formulário.

Os cookies são pacotes de dados incorporados pelo servidor na máquina do usuário. A sua principal função é buscar informações do visitante. Podemos citar como exemplo aquele cookie que um site cria para guardar as últimas buscas do internauta, ou suas preferências, como a cor de fundo da página, entre outros.

8.1. Cookies

É importante saber que o cookie fica disponível de acordo com o tempo determinado pelo desenvolvedor, somente será apagado quando a data de validade expirar.

Para criar um cookie devemos observar a estrutura abaixo.

Setcookie(name, value, expira, caminho, domínio, segurança, httponly)

Name (nome): Permite definir o nome do cookie.

Value (valor): Permite definir o conteúdo do cookie.

Expire (expira): Opcional. Permite definir o tempo de vida do cookie. Utilizamos a função `time() + 3600` que retorna a hora atual + uma hora, este seria o tempo de duração de um determinado cookie, 1 hora.

Path (caminho): Opcional. Permite definir um diretório que o cookie estará disponível, se definir `"/aula/"`, o cookie só estará disponível neste diretório.

Domain (domínio): Opcional. Permite definir o domínio do cookie. Se não definir, ativa o default.

Secure (segurança): Opcional. Permite especificar se um determinado cookie só deve

ser transmitido através de uma conexão segura HTTPS.

Httponly: Opcional. Permitir que scripts alterem as configurações abrindo brechas no sistema, definir como `true` para bloquear scripts.

Criando um cookie para armazenar o nome de um produto por 2 minutos.

```
Setcookie("produto","Bermuda",time() + 120);
```

Como recuperar o valor de um cookie:

Para recuperar o valor de um cookie utilizamos a variável superglobal `$_COOKIE[]`, conforme o exemplo abaixo:

```
$_COOKIE["produto"];
```

SUPERGLOBAL

As superglobais são variáveis pré-definidas pelo PHP, e estão disponíveis para todo o script. Confira algumas na lista abaixo:

`$_SERVER`, `$_GET`, `$_POST`, `$_FILES`, `$_COOKIE`, `$_SESSION` e `$_REQUEST`.

Sessões

As sessões permitem que os dados de um usuário possam ser utilizados durante todo o tempo em que acessa uma determinada área e navega dentro de um website ou uma aplicação. Podemos concluir que a sessão tem como objetivo autenticar os dados dos usuários, garantindo assim que cada um acesse somente a área que lhe é permitido.

8.2. Criando uma sessão:

A função `session_start()` permite iniciar uma nova sessão.

É importante que a função `session_start()` seja acrescentada no início do código.

`session_start()`

?>

Gravando valores na sessão:

Para armazenar conteúdo em uma sessão, utilizamos a super global `$_SESSION[]`.

Veja o exemplo armazenando o nome de usuário:

```
$_SESSION["login"]="carol";
```

Recuperar valores de uma sessão:

Para acessar o conteúdo de uma variável de sessão basta fazer uso da super global `$_SESSION[]`. Veja o exemplo:

```
Echo $_SESSION["login"];
```

Eliminar o conteúdo de uma sessão:

Podemos remover uma variável da sessão, utilizamos a função `unset()`. Veja o exemplo:

```
Unset($_SESSION["login"]);
```

Destruindo toda a sessão:

Utilize a função `session_destroy()` para destruir todas as sessões.

8.3. Exercícios Passo a Passo

1. Este exercício tem como objetivo criar um cookie para armazenar o nome do visitante e sua cidade. Ative o XAMPP se necessário;

2. Abra o Notepad++ e faça as devidas configurações;

3. Digite o código conforme o indicado;

```
1 <!DOCTYPE HTML>
2 <html>
3 <head>
4 <title></title>
5 <meta charset="utf-8"/>
6 </head>
7 <body>
8 <form name="new_form" method="POST" action="gravar-cookies.php">
9 <h1>CADASTRO</h1>
10 <table>
11 <tr>
12 <td><input type="text" placeholder="Nome" required="required" name="nome"/></td>
13 </tr>
14 <tr>
15 <td><input type="text" placeholder="Cidade" required="required" name="cidade"/></td>
16 </tr>
17 <tr>
18 <td><input type="submit" value="Enviar"/></td>
19 </tr>
20 </table>
21 </form>
22 </body>
23 </html>
```

4. Salvar como `formulario-cookies.php` na pasta `exercicio-aula8`. Clique no menu Arquivo, Salvar. Se necessário, chame seu instrutor para auxiliar;

5. Crie um novo documento e faça as devidas configurações;

6. Digite o código conforme o indicado;

```
1 <?php
2 $nome=$_POST["nome"];
3 $cidade=$_POST["cidade"];
4 setcookie("resNome",$nome,time()+60);
5 setcookie("resCidade",$cidade,time()+60);
6 >
7 <!DOCTYPE HTML>
8 <html>
9 <head>
10 <title></title>
11 <meta charset="utf-8"/>
12 </head>
13 <body>
14 <h3>Os cookies de pesquisa foram gravados no navegador. Clique no link abaixo para visualiza-los.</h3>
15 <a href="ler-cookies.php">VISUALIZAR COOKIES</a>
16 </body>
17 </html>
```

7. Salvar como `gravar-cookies.php` na pasta `exercicio-aula8`;

8. Crie um novo documento, configure a codificação para UTF8 e a linguagem para PHP.

9. Digite o código conforme o indicado;

```
1 <?php
2 $visNome=$_COOKIE["resNome"];
3 $visCidade=$_COOKIE["resCidade"];
4 >
5
6 <!DOCTYPE HTML>
7 <html>
8 <head>
9 <meta charset="utf-8"/>
10 </head>
11 <body>
12
13 <h3>Os cookies gravados foram:</h3>
14 <?php
15 echo "Nome: " . $visNome . " Cidade: " . $visCidade;
16 >
17
18 </body>
19 </html>
```

10. Salve como `ler-cookies`

11. Abra o localhost em aulas, exercício-aula8, `formulario-cookies.php`

8.4. Exercícios de Fixação

1. Este exercício tem como objetivo criar uma sessão utilizando o login e senha do

usuário, caso estiver correto, será exibido uma lista de produtos. Caso contrário, exibir a mensagem "Confira seus dados".

Iniciamos o nosso curso de PHP aprendendo uma série de comandos, porém acabamos misturando com HTML, e essa sem dúvida não é uma boa prática. Nesta aula vamos falar sobre como separar um projeto em camadas.

9.1. MVC

Para dividir um projeto em camadas, utilizamos o MVC.

O MVC é um padrão de arquitetura de softwares, uma forma de separar o projeto, dividir em camadas, são elas: Model, View e Controller.

Model: é a camada responsável pela manipulação dos dados, leitura, escrita e validações.

View: é a camada de interação do usuário, ou seja, é a exibição dos dados por html ou xml.

Controller: a camada responsável por receber as requisições do usuário, faz o controle de qual model usar e qual view será exibida ao usuário.

9.2. Smarty

Utilizaremos a biblioteca Smarty para manter o PHP e o HTML separados, mas interagindo entre eles.

Uma das vantagens de separá-los é o reaproveitamento do código, a facilidade de realizar uma manutenção e a compreensão do código.

Funções básicas do Smarty:

New Smarty

Esta função cria um novo objeto que permite abrir um arquivo html para que possamos trabalhar paralelamente.

Estrutura:

```
$variavel = new Smarty;
```

9.2.1. Método Assign()

Este método atribui um valor a uma variável. Passa informações para o html.

Estrutura:

```
$variavel->assign(variável,valor);
```

Exemplo:

```
$variavel->assign("cor","preto");
```

9.2.2. Método display()

Este método faz a mesclagem entre o php e o html com a finalidade de exibir o resultado dentro do html.

Estrutura:

```
$variavel->display("arquivo.html");
```

Como adicionar as variáveis do PHP no HTML.

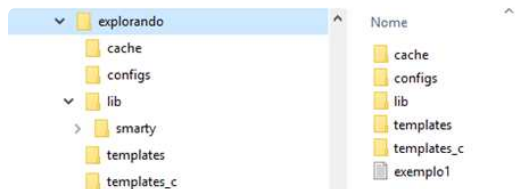
Para definir o local onde a variável será inserida, devemos utilizar um sinalizador. Use as chaves seguido do nome da variável.

Estrutura:

```
{nome-da-variavel}
```

Estrutura de pastas

É de extrema importância a organização das pastas e arquivos, acompanhe o modelo abaixo.



O exemplo acima possui a pasta principal, chamada “explorando”, dentro estão as seguintes subpastas:

Cache, configs, lib, templates e templates_c.

Devemos criar estas pastas para o bom funcionamento da biblioteca.

Na pasta “lib” iremos copiar a biblioteca “smarty”.

Nossos arquivos php:

Na pasta principal, neste caso “explorando”, devemos salvar os arquivos.php, como “exemplo1.php”.

Na pasta “templates” devemos salvar os arquivos.html, como no caso, o “exemplo1.html”.

9.3. Exercícios Passo a Passo

1. Este exercício tem como objetivo criar um projeto organizando as pastas. Crie dentro da pasta “XAMPP” uma pasta chamada “primeiro_projeto”;

2. Abra esta pasta e crie as seguintes pastas: cache, configs, lib, templates e templates_c;

3. Dentro da pasta “lib” copie a pasta “smarty” que se encontra dentro da pasta “Programas”. Qualquer dúvida chame o seu instrutor.

4. Abra o Notepad++;

5. Faça as configurações necessárias, ative o sistema UTF8 e a linguagem para PHP;

6. Digite o código conforme o indicado;

```
1 <?php
2
3 include("lib/smarty/libs/Smarty.class.php");
4 $smarty = new Smarty;
5
6 $smarty->cache_dir="cache";
7 $smarty->config_dir="configs";
8 $smarty->compile_dir="templates_c";
9 $smarty->template_dir="templates";
10
11 ?>
```

7. Salve na pasta “configs” com o nome “config.ini.php”;

8. Crie um novo documento, e faça as configurações necessárias, ative o sistema UTF8 e a linguagem para PHP;

9. Digite o código conforme o indicado;

```
1 <?php
2
3 include("configs/config.ini.php");
4
5 $smarty->assign('texto', 'Programação PHP com MVC');
6
7 $smarty->display('exemplo1.html');
8
9
10 ?>
```

10. Salve na pasta “primeiro_projeto” com o nome “exemplo1.php”;

11. Crie um novo documento, e faça as configurações necessárias, ative o sistema UTF8 e a linguagem para HTML;

12. Digite o código conforme o indicado;

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset="utf-8" />
5     <title></title>
6 </head>
7
8 <body>
9
10     <h2>{$texto}</h2>
11
12 </body>
13 </html>
```

13. Salve na pasta “templates” com o nome “exemplo1.html”;

14. Ative o XAMPP se necessário, e verifique o resultado.

9.4. Exercícios de Fixação

1. Este exercício tem como objetivo criar um formulário com os campos nome, cidade e email. O arquivo html será salvo como formulário1.html na pasta "templates" e o arquivo php como formulário1.php na pasta primeiro_projeto.

2. Deve ser mostrado o conteúdo de cada campo no html. Confira o resultado no localhost.

Existem vários tipos de banco de dados e eles estão presentes na nossa vida há muito tempo. A lista telefônica pode ser considerada um bom exemplo.

Antigamente as empresas armazenavam informações em arquivos físicos, mas o surgimento e evolução dos computadores possibilitaram o armazenamento de dados de modo digital. Assim os bancos de dados evoluíram e se tornaram o coração de muitos sistemas de informação. A definição de Banco de dados encontrada na internet é essa:

10.1. Banco de Dados

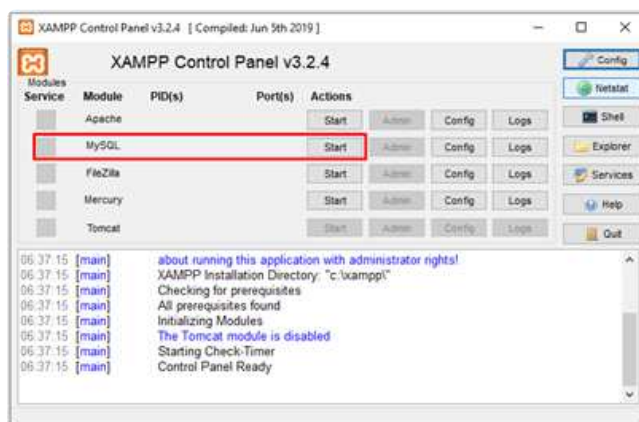
“Banco de dados são coleções de informações que se relacionam de forma que crie um sentido. São de vital importância para empresas e há duas décadas se tornaram a principal peça dos sistemas de informação.”

O MySQL é um banco de dados conhecido por sua facilidade de uso, sua interface é simples, e roda em vários sistemas operacionais. É protegido por uma licença de software livre, desenvolvida pela GNU.

O MySQL é um SGDB, Sistema de Gerenciamento de Banco de Dados, utiliza a linguagem SQL (Structured Query Language, linguagem de consulta estruturada).

10.2. Ativando o MySQL

Quando ativamos o servidor, podemos ativar o MySQL no painel de controle do XAMPP.



10.2.1. APLICATIVO

O phpMyAdmin é um aplicativo de código aberto, desenvolvido para trabalhar com MySQL, possibilitando ser usado no servidor local e em um servidor web.

ACESSO

No seu navegador basta digitar <http://localhost/phpmyadmin/> e pressionar a tecla enter.



O phpMyAdmin exibe, no lado esquerdo, todos os bancos de dados no formato de uma árvore de diretórios.

No lado direito, temos as abas: Base de Dados, SQL, Estado, Contas de utilizador, Exportar, Importar, Configurações, Replicação e Mais.

Base de Dados: Permite criar um banco de dados definindo o nome e a codificação, normalmente utilizamos utf8_bin para o reconhecimento dos caracteres, permite eliminar uma ou mais base de dados.

Base de Dados



SQL: Permite executar os códigos diretamente no aplicativo.

Exportar: Permite exportar uma ou mais tabelas ou o banco de dados. Selecione uma ou mais tabelas e clique no botão Executar, ou selecione o banco de dados. O arquivo será gerado no formato SQL, por padrão.

Importar: Permite importar uma ou mais tabelas ou o banco de dados.

Operações: Permite realizar algumas configurações, se o banco de dados estiver selecionado. Opções como: Criar tabela, Renomear banco de dados, Remover o banco de dados estarão disponíveis. Se uma tabela estiver selecionada, opções como: Renomear tabela, Copiar tabela para outra base de dados, Esvaziar dados da tabela ou Remover tabela estarão disponíveis.

10.2.2. ORGANIZAR BANCO DE DADOS

Quando criamos um banco de dados é importante criar um esquema conforme as necessidades do cliente.

Agenda de contatos

Um cliente possui um estabelecimento comercial chamado Bazar do Porto, precisa

coletar informações de seus clientes e novos clientes, para isso precisa que seu site disponha de uma área onde todos possam se cadastrar e posteriormente o empresário possa realizar consultas e alterações nos dados cadastrais.

Veja o exemplo:



Antes de criar o modelo acima dentro do site do cliente, é importante planejar como serão lançadas essas informações no banco de dados. A seguir iremos criar o banco de dados e uma tabela. Para realizar testes, serão cadastrados alguns clientes e realizará algumas buscas.

Nome do banco de dados: bazarporto

Nome da tabela: clientes

Entendendo os atributos de uma tabela: quando criamos uma tabela, precisamos definir os campos em cada coluna e com eles o tipo de dados que vai armazenar.

10.2.3. Dados do tipo String:

Char – tamanho fixo, completado com espaço em branco.

Varchar – tamanho variável.

Text – tamanho variável, até 2GB de dados.

O varchar é usado para armazenar até 255 caracteres. Exemplo: Nome, endereço, cidade.

10.2.4. Dados do tipo numérico.

Int – permite números inteiros, como código, quantidade e idade.

Decimal – é ideal para cálculo de valores.

Double – é o mais adequado para cálculos científicos gerais.

Float – é o double com menos bytes para representação.

10.2.5. Dados do tipo data.

Date – armazena apenas uma data. De 1 de janeiro de 0001 a 31 de dezembro de 9999.

Time – armazena um tempo apenas para uma precisão de 100 nanosegundos.

Depois das informações que uma tabela precisa conter, vamos entender como criar uma:

10.2.6. COMANDO PARA CRIAR UMA TABELA.

`CREATE TABLE nome_da_tabela (campos e tipos de dados).`

CRIANDO A TABELA DE CLIENTES

```
CREATE TABLE clientes
(
    codigo INT AUTO_INCREMENT NOT NULL PRIMARY KEY,
    nome VARCHAR(50),
    email VARCHAR(50),
    telefone VARCHAR(15),
    cidade VARCHAR(50),
    cep NUMERIC,
    dtNasc date,
    sexo char(1)
)
```

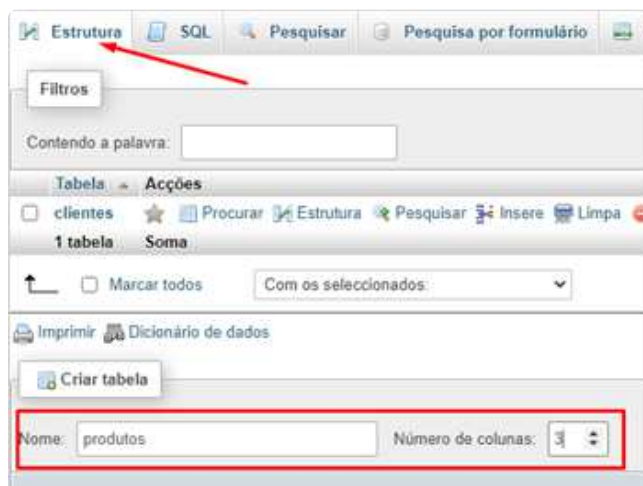
O comando **AUTO_INCREMENT** define que o registro será automaticamente numerado em sequência.

O comando **NOT NULL** define que o campo não pode ser nulo.

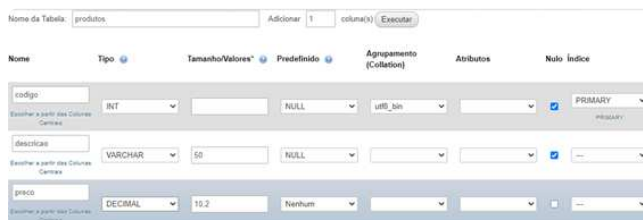
O comando **PRIMARY KEY** define o campo como chave primária, ou seja, este campo não será duplicado, e uma referência para pesquisa, atualizar e excluir registros.

Podemos realizar todas as operações de forma visual, sem código, somente através dos

recursos do phpMyAdmin, a seguir vamos ver como criar uma tabela no design.



Iremos criar três campos para demonstrar o funcionamento.



O campo código é do tipo INT, NOT NULL (Nulo) Índice (Primary) A_I (Auto_Increment)

O campo descrição é do tipo VARCHAR (50)

O campo preco é do tipo double (10,2)

10.2.7. COMANDO PARA INSERIR DADOS NA TABELA

```
INSERT INTO nome_tabela (lista_de_campos) VALUES (lista_dados)
```

INSERINDO CLIENTES NA TABELA

```
INSERT INTO clientes (nome, email, telefone, cidade, cep, dtNasc, sexo)
VALUES
("André", "andre@ig.ig", "90000-0001", "Poa", "95780000", "1980-12-25", "m")
```

10.2.8. COMANDO PARA RECUPERAR DADOS DE UMA TABELA

```
SELECT <lista_de_campos> FROM <nome_da_tabela>
```

LISTAR TODOS OS CLIENTES

```
SELECT * FROM clientes
```

LISTAR APENAS O NOME E A CIDADE

```
SELECT nome, cidade FROM clientes
```

A cláusula WHERE permite ao comando SQL passar condições de filtragem.

```
SELECT campo1, campo2, ...  
FROM nome_tabela  
WHERE condição;
```

LISTAR APENAS CLIENTES DE PORTO ALEGRE

```
SELECT * FROM clientes WHERE cidade="Poa"
```

Operador LIKE é usado com o comando WHERE para buscas aproximadas.

```
SELECT campos FROM tabela WHERE campo LIKE 'valor'
```

Utilize a porcentagem para buscar todas as ocorrências antes e/ou depois da palavra chave.

LISTAR TODOS OS CLIENTES COM SOBRENOME "SILVA".

```
SELECT campos FROM tabela WHERE campo LIKE "%Silva%"
```

ORDER BY: Permite classificar os registros, o termo "asc" define ordem ascendente e o termo "desc" descendente. Podemos classificar por um determinado campo.

LISTAR CLIENTES EM ORDEM ALFABÉTICA

```
SELECT * FROM clientes ORDER BY nome
```

LISTAR CLIENTES EM ORDEM DECRESCENTE

```
SELECT * FROM clientes ORDER BY nome DESC
```

10.3. Exercícios Passo a Passo

1. Este exercício tem como objetivo criar um banco de dados chamado "pubdoporto". O cliente precisa registrar seus colaboradores com os seguintes dados: nome, email, telefone, cep, dataNasc, sexo, e salario. Se necessário ative o XAMPP e inicialize os serviços do Apache e MySQL.

2. Abra o navegador, digite o endereço entre aspas "localhost/phpmyadmin/", pressione Enter.

3. Clique no botão Início e na aba Base de Dados;

4. Digite o nome entre aspas "pubdoporto", e ao lado selecione utf8_bin, clique em Criar;

5. Clique na aba SQL e digite o código conforme o informado;

```
CREATE TABLE colaboradores
```

```
(  
    codigo INT AUTO_INCREMENT NOT NULL PRIMARY KEY,  
    nome VARCHAR(50),  
    email VARCHAR(50),  
    telefone VARCHAR(15),  
    cep NUMERIC,  
    dataNasc date,  
    sexo char(1),  
    salario double(10,2)  
);
```

6. Cadastre 5 colaboradores, clique na tabela colaboradores e na aba SQL. Digite o código conforme o indicado;

```
INSERT INTO colaboradores (nome, email, telefone, cep, dataNasc, sexo, salario)  
VALUES  
("Sabrina Silva","sabrina@ig.ig","90000-0001","95780000","1987-04-28","f",1200.00),  
("Camila Silva","camila@ig.ig","90000-0003","95760000","1990-11-09","f",950.90),  
("Guilherme Nascimento","gui@ig.ig","90000-0004","95760000","1994-09-11","m",1100.00),  
("Amilton Peixoto","amilton@ig.ig","90000-0005","95760000","1998-03-11","m",950.90),  
("Danilson Peixoto","denilson@ig.ig","90000-0006","95780000","1996-11-21","m",700.00)
```

7. Clique na tabela para conferir o cadastro.

10.4. Exercícios de Fixação

1. Este exercício tem como objetivo realizar algumas buscas na tabela colaboradores;

2. Faça uma busca por nome, telefone e salário.

4. Faça uma busca por colaboradores, cujo sobrenome é Silva.

3. Faça uma busca classificando a coluna nome em ordem crescente.

Aprenderemos alguns comandos para manipular dados nas tabelas. Na aula anterior, aprendemos os comandos CREATE TABLE, INSERT E SELECT, nesta aula aprenderemos os comandos UPDATE e DELETE. Outro recurso são as entidades relacionais que permitem criar vínculo entre as tabelas.

11.1. UPDATE

Mudar dados em uma tabela é algo bem comum, podemos dar um exemplo de um cadastro, o cliente mudou de endereço, de telefone, de e-mail e assim por diante, aí entra a alteração dos dados de um registro e para realizar esta tarefa utilizamos o UPDATE.

Estrutura:

```
UPDATE nome_tabela
SET CAMPO = "novo_valor"
WHERE CONDIÇÃO
```

Em *nome_tabela* será digitada a tabela que será modificada.

Em *campo* informaremos qual o campo que será alterado o valor.

Em *novo_valor* será digitado o valor que substituirá o antigo.

WHERE: se não for informado, a tabela inteira será atualizada. **Condição:** uma regra que determina a execução do comando.

Exemplo:

	A	B	C
1	id	nome	endereço
2	1	Amanda Dias	Rua 1º de Março, 1000
3	2	Suelen Silva	Av. 7 de Setembro, 12
4	3	Guilherme Dias	Rua Castro Alves, 332
5	4	Tiago Silva	Rua Imigração, 143
6	5	Melissa Santos	Av. 7 de Setembro, 24

A tabela clientes possui alguns registros. Precisamos alterar o endereço do Tiago Silva para Rua 1º de Março, 1000. Veja como devemos proceder.

```
UPDATE clientes
SET endereço = "Rua 1º de Março, 1000"
WHERE id=4
```

Outro exemplo:

A tabela de clientes possui uma coluna desconto, para alterar o desconto para 5% à todos os clientes, devemos proceder da seguinte forma.

	A	B	C	D
1	id	nome	endereço	desconto
2	1	Amanda Dias	Rua 1º de Março, 1000	3%
3	2	Suelen Silva	Av. 7 de Setembro, 12	2%
4	3	Guilherme Dias	Rua Castro Alves, 332	2%
5	4	Tiago Silva	Rua Imigração, 143	3%
6	5	Melissa Santos	Av. 7 de Setembro, 24	2%

```
UPDATE clientes
SET desconto=5
```

11.2. DELETE

Apagar registros é uma outra possibilidade dentro de um banco de dados, devemos ter o máximo de atenção quando realizar este processo.

O comando usado é o DELETE. Estrutura:

```
DELETE FROM nome_da_tabela
WHERE condição
```

É necessário definir o nome da tabela que será modificada e com a cláusula WHERE criamos a condição para que o comando seja executado. *Exemplo:*

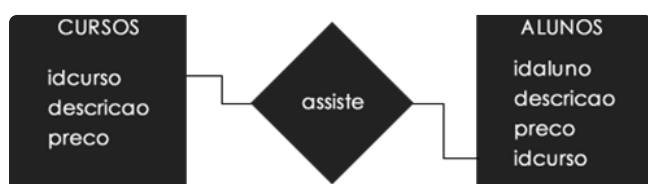
Seguindo o exemplo da tabela de clientes acima, vamos sugerir eliminar a cliente Suelen Silva.

```
DELETE FROM clientes WHERE id=2
```

11.3. Diagrama Entidade Relacional

O diagrama ER permite demonstrar, de forma gráfica, como um fluxograma como as entidades (tabelas) se relacionam entre si dentro de um sistema. Esses diagramas utilizam algumas formas para sua representação.

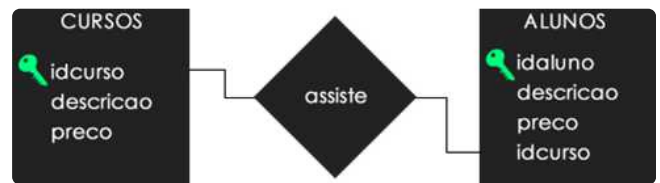
As entidades são nossas tabelas e normalmente representadas como um retângulo. O losango representa graficamente a associação (relacionamento) entre as tabelas, a palavra-chave que faz ligação entre elas.



11.3.1. CHAVE PRIMÁRIA

Quando um campo é definido como sendo chave primária, o registro passa a ser único. Esta é uma característica, não haverá duplicação de dados, de uma forma mais simples, é como um CPF, não existem dois. Outra característica é que esse registro não seja nulo e ainda que esse registro seja usado como índice para buscas, atualização e exclusão.

Ela tem como finalidade organizar um banco de dados, cada tabela possuirá apenas uma chave primária.



Para definir um campo como sendo do tipo chave primária, usamos o seguinte comando.

```
id int PRIMARY KEY
```

Outros atributos fazem parte da construção deste campo.

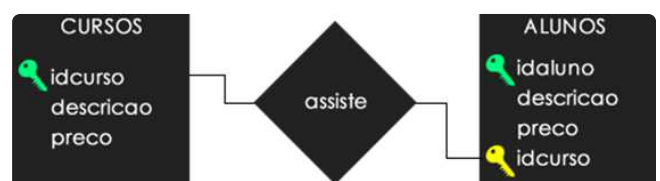
```
id int PRIMARY KEY NOT NULL AUTO_INCREMENT
```

NOT NULL: define que o registro não será nulo.

AUTO_INCREMENT: define que o registro será automaticamente numerado.

11.3.2. CHAVE ESTRANGEIRA

Uma chave estrangeira é usada para criar relacionamento entre duas tabelas. A chave estrangeira nada mais é do que a chave primária dentro de outra tabela.



Ela vai permitir que sejam cadastrados vários alunos em um determinado curso, por exemplo.

As diferenças são:

A chave primária identifica o registro.

A chave estrangeira permite o relacionamento entre duas tabelas.

Como criar uma tabela para permitir que o relacionamento funcione corretamente?

```
CREATE TABLE livros
(
    idlivro INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(50),
    descricao VARCHAR(50),
    genero VARCHAR(50)
) ENGINE=INNODB DEFAULT CHARSET=UTF8;
```

ENGINE=INNODB: é um mecanismo que serve para que as chaves primárias e estrangeiras possam ser utilizadas.

ALTER TABLE

O comando ALTER TABLE permite alterar a estrutura de uma tabela, adicionando ou excluindo atributos.

Adicionando um atributo(campo):

```
ALTER TABLE nome_da_tabela
ADD campo tipo_de_dados
```

FOREIGN KEY

Define um campo como chave estrangeira, usada para vincular uma ou mais tabelas. A chave estrangeira nada mais é que a chave primária de uma tabela em outra.

REFERENCES

Este comando indica a entidade (tabela) que tem origem numa determinada chave.

JOIN

O comando JOIN ou INNER JOIN é o método que permite a junção de duas tabelas. Estrutura:

```
SELECT campos
FROM tabela1
INNER JOIN tabela2
ON tabela1.campo=tabela2.campo
```

Exemplo:

Temos um banco de dados chamado centralbiblioteca que possui duas tabelas, livros

e leitores. Para gerar uma consulta que retorne quais os livros que cada leitor retirou, devemos utilizar a seguinte linha de comando.

```
SELECT leitores.nome, livros.nome AS livro, livros.descricao
FROM livros
JOIN leitores
ON livros.idlivro = leitores.idlivro
```

11.4. Exercícios Passo a Passo

1. Este exercício tem como objetivo criar um banco de dados chamado "escolaweb" com duas tabelas, "cursos e professores". Abra o Xampp e ative o Apache e o MySQL;

2. Abra o phpMyAdmin;

3. Clique no botão Início e na aba Base de Dados;

4. Digite escolaweb e altere o tipo para utf8_bin. Clique em Criar;

5. Clique na aba SQL e digite o código conforme o indicado;

```
CREATE TABLE cursos
(
    idcurso INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    curso VARCHAR(40),
    descricao VARCHAR(40)
) ENGINE=INNODB DEFAULT CHARSET=UTF8;

CREATE TABLE professores
(
    idprof INT NOT NULL AUTO_INCREMENT PRIMARY KEY,
    nome VARCHAR(40),
    email VARCHAR(40),
    idcurso INT
) ENGINE=INNODB DEFAULT CHARSET=UTF8;
```

6. Clique na tabela cursos e na aba SQL para cadastrar três cursos. Digite o código conforme o indicado e logo após clique no botão Executar;

```
INSERT INTO cursos (curso, descricao)
VALUES
("Excel", "Curso completo de Excel"),
("Java", "Curso Java Módulo 1"),
("PHP", "Curso PHP Módulo 2")
```

7. Clique na tabela professores e na aba SQL para cadastrar 6 professores. Digite o

código conforme o indicado e logo após clique no botão Executar;

```
INSERT INTO professores (nome, email, idcurso)
VALUES
("Julio", "julio@ig.ig", 1),
("Carina", "carina@ig.ig", 1),
("Marcelo", "marcelo@ig.ig", 3),
("Carlos", "carlos@ig.ig", 2),
("Daniel", "daniel@ig.ig", 2),
("Fernanda", "fernanda@ig.ig", 3)
```

11.5. Exercícios de Fixação

1. Clique no banco escolaweb. Liste os cursos usando o comando SELECT;

2. Liste os cursos por professores usando os comandos SELECT, JOIN E ON.

Planejar e executar um projeto demanda tempo, tirar do papel e colocar em prática vai depender de uma série de informações. É importante saber as necessidades do cliente, coletando o máximo de informações.

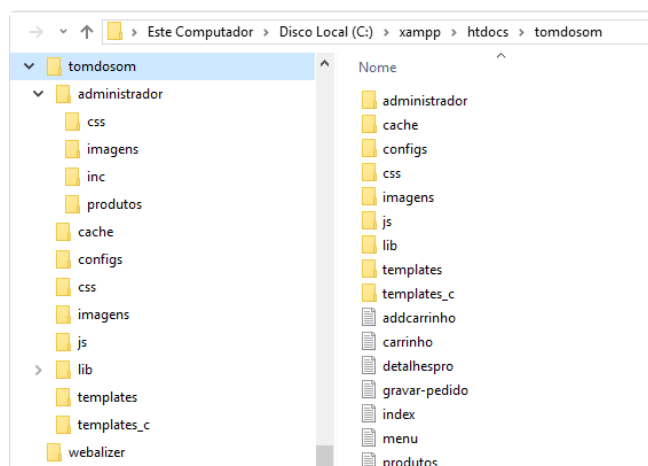
Para entender alguns procedimentos, vamos utilizar como base a nossa loja online. O primeiro passo é saber as necessidades da loja, neste caso o site vai oferecer instrumentos musicais e o internauta seleciona seus produtos e envia um orçamento. Somente o administrador do site tem acesso aos orçamentos.

12.1. Gerenciamento do site

O site permite que o administrador cadastre, altere e exclua categorias e produtos, e visualize os orçamentos.

12.1.1. Estrutura de pastas

A organização das pastas é fundamental, elas precisam estar dentro da pasta **htdocs** do xampp. Veja o exemplo abaixo.



A pasta **tomdosom**: é a principal, deve ser criada dentro da pasta **htdocs**.

A pasta **lib**: deve armazenar a biblioteca smarty.

A pasta **configs**: deve armazenar o arquivo `config.ini.php` que possui os caminhos para uso da biblioteca smarty e para uso das pastas padrão, `cache`, `configs`, `templates` e `templates_c`.

Nesta aula iniciaremos a integração do banco de dados com o php. Nesta etapa, definimos qual o servidor será contratado para hospedagem do site, o mesmo informará login, senha e o nome do banco de dados.

Veja uma forma simples de definição dos dados do servidor.

Servidor: será adicionado o caminho que o servidor web disponibilizará ou servidor local, que é o nosso caso. Exemplo:

```
$servidor = localhost
```

Usuário: será adicionado o nome disponibilizado pelo servidor web ou servidor local.

```
$usuario = root
```

Senha: será adicionada conforme disponibilizado pelo servidor web ou servidor local.

```
$senha = nenhuma
```

Banco: será adicionado conforme disponibilizado pelo servidor web ou servidor local.

```
$banco = dbtomdosom
```

Define

Para definir uma constante, usamos a função `define()`.

O que é uma constante?

Uma constante é como uma variável, porém o valor depois de definido não pode ser alterado, como sugere o nome. As constantes são sempre definidas em maiúsculas não necessita do cifrão.

As constantes são globais em todo o script.

Exemplo:

12.1.2. Try/Catch

As funções Try/Catch tem como finalidade tratar exceções que o programador não tem como prever, podemos citar erros de execução do código, perda de conexão com a internet, entre outros. Erros de comando ou falhas de conexão com a internet, tratamos como exceções.

O Try tem como objetivo tentar executar o código, caso nenhum erro ocorra.

O Catch tem como objetivo tratar o erro causado ao tentar executar um código.

12.1.3. PDO

Funções como o `mysql_connect` e `mysql_query` são consideradas obsoletas a partir da versão PHP 5.5, recomenda-se utilizar a classe PDO.

A classe PDO possui novas funções para integração do MySQL com o PHP para as novas versões do PHP.

A seguinte linha de código conecta o banco de dados em todo o script.

```
$PDO = new PDO( 'mysql:host=' .  
MYSQL_HOST . ';dbname=' .  
MYSQL_DB_NAME, MYSQL_USER,  
MYSQL_PASSWORD );
```

A variável \$PDO passa a herdar todas as funções da classe PDO. Os seguintes parâmetros são informados: o primeiro é o caminho do banco, o segundo é o nome do

banco, o terceiro é o nome de usuário e o último é a senha.

INCLUDE

A função include permite incluir um determinado arquivo.

PREPARE

O comando prepare permite preparar instruções SQL que serão executadas.

EXECUTE

Este comando executa uma instrução preparada.

FETCHALL

Esta função retorna um array com todos os registros.

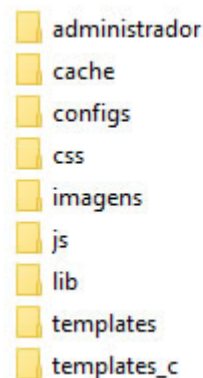
FETCH_ASSOC

Esta função retorna uma matriz associativa, que corresponde aos nomes das colunas.

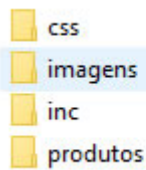
12.2. Exercícios Passo a Passo

1. Este exercício tem como objetivo criar um projeto chamado "projeto_piloto". Algumas pastas serão criadas e outras copiadas, desta forma teremos um padrão para outros projetos. Abra a pasta htdocs que se encontra no disco local, diretório xampp;

2. Crie a pasta "projeto_piloto" e seguintes pastas conforme o indicado;



3. Dentro da pasta "administrador" crie as seguintes pastas conforme o indicado;



4. Copie a pasta "smarty" que está dentro da pasta Arquivos auxiliares, para dentro da pasta "lib". Qualquer dúvida chame o seu instrutor.

5. Copie o arquivo "config.ini" que está dentro da pasta Arquivos auxiliares, para dentro da pasta "configs". Qualquer dúvida chame o seu instrutor.

12.3. Exercícios de Fixação

1. Este exercício tem como objetivo praticar criar um banco de dados, duas tabelas e

relacionar dados;

2. Crie um banco de dados chamado "carros_online", crie duas tabelas, marcas e modelos;

3. Crie a tabela marcas conforme o indicado;

4. Crie a tabela modelos conforme o indicado;

5. Cadastre três marcas. Fiat, Volkswagen e GM.

6. Cadastre 2 modelos por marca. Celta - GM, Corsa - GM, Voyage - Volks, Fox - Volks, Siena - Fiat e Palio - Fiat

7. Faça uma busca de modelos por marca.

Desenvolveremos nesta aula a listagem das categorias que foram cadastradas anteriormente, e atualizaremos as informações utilizando o comando SELECT para listar, e para atualizar, o comando UPDATE.

13.1. \$_REQUEST

\$_REQUEST serve para pegar o valor de uma variável do tipo \$_POST, \$_GET ou \$_COOKIE. *Exemplo:*

```
$variavel = $_REQUEST["variável"];
```

13.2. FETCH_OBJ

Retorna um objeto com nomes de propriedades que correspondem aos nomes das colunas em uma query. *Exemplo:*

```
$qry = $PDO->prepare("SELECT * FROM  
nome_da_tabela");
```

```
$qry->execute();
```

```
$resp=qry->fetch(PDO::FETCH_OBJ);
```

13.3. bindParam

O bindParam vincula uma variável à um espaço reservado com nome ou ponto de interrogação no SQL. *Exemplo:*

```
$qry->bindParam(':nome','valor');
```

Elementos do formulário recebendo valores

Os elementos, input, select, textarea, entre outros, podem receber um valor do banco de dados. Exemplo:

```
INPUT, SELECT, {$categoria}, TEXTAREA  
e {$categoria}
```

13.4. Exercícios Passo a Passo

1. Este exercício tem como objetivo atualizar os dados editados no formulário. Abra o Xampp e ative o Apache e o MySQL.

2. Abra o arquivo cad-categorias.php, qualquer dúvida chame o seu instrutor. O arquivo está na pasta Arquivos auxiliares/Conteudo/13.

3. Digite o código conforme o indicado e salve o documento dentro da pasta administrador. Qualquer dúvida chame o seu instrutor.

```
if($acao=="Atualizar"){  
    $queryUpdate = $PDO->prepare("UPDATE tbcategorias SET categoria = ?, linkcateg = ? WHERE idcat = ?");  
    $queryUpdate->bindParam(1,$categoria);  
    $queryUpdate->bindParam(2,$linkcateg);  
    $queryUpdate->bindParam(3,$idcat);  
    $queryUpdate->execute();  
  
    echo "<script>  
        alert('Dados alterados');  
        document.location.href='cad-categorias.php';  
    </script>";  
}
```

4. Abra o navegador e atualize a categoria PEDAIS para PEDAIS DUPLOS. Verifique se houve alteração.

13.5. Exercícios de Fixação

1. Este exercício tem como objetivo abrir a tabela carros_online e alterar via SQL a marca "GM" para "General Motors". Utilize o comando UPDATE.

Concluiremos, nesta aula, os procedimentos para excluir uma categoria. As instruções serão passadas no PHP com o comando delete, e ainda estaremos exibindo uma mensagem de alerta sobre a categoria que será apagada.

14.1. Evento ONCLICK

O evento onclick é definido quando o usuário clica no elemento, pode ser um link ou um botão, por exemplo.

Estrutura:



Este evento foi usado no link específico para excluir a categoria. Veja como ficou:

```
<a href="cad-produtos.php?idpro={ $p.idpro }&acao=excluir"
onclick="return verificar()"> </a>
```

Dentro do evento **onclick** passamos o comando **return** que é uma expressão que retorna um valor, no caso a função **verificar()**.

Outro detalhe é o uso do e-comercial "&", que tem como finalidade passar mais de uma variável na mesma requisição.

14.2. Input Tipo FILE

O input define um campo de entrada de dados, aqui estamos tratando do atributo type, definindo o comportamento do campo como sendo para upload de arquivos.

O tipo file define um campo de seleção de arquivos.

\$_FILES

É uma superglobal que tem a capacidade de armazenar todas as informações do arquivo enviado para upload.

```
$_FILES["variavel"]["name"]
```

O primeiro parâmetro, o "**name**", contém o nome original do arquivo selecionado.

```
$_FILES["variavel"]["tmp_name"]
```

O segundo parâmetro, o "**tmp_name**", contém o nome temporário do arquivo enviado, que foi armazenado no servidor.

```
$_FILES["variavel"]["type"]
```

O terceiro parâmetro, o "**type**", contém o tipo (formato) do arquivo.

```
$_FILES["variavel"]["size"]
```

O quarto parâmetro, o "**size**", contém o tamanho do arquivo, em bytes.

14.3. Função MAX()

A função MAX() é uma função agregada que tem como finalidade analisar um conjunto de valores e retornar o maior entre eles.

Função move_uploaded_file()

Esta função move um arquivo enviado através de um formulário para um novo destino.

Estrutura:

```
move_uploaded_file(string $filename, string $destination)
```

A variável **\$filename** recebe o arquivo enviado.

A variável **\$destination** define o local de destino do arquivo, passando informações necessárias como nome, id, extensão do arquivo, entre outros argumentos que acharem importantes.

14.4. Exercícios Passo a Passo

1. Este exercício tem como objetivo desenvolver a listagem de produtos. Abra o Xamp e ative o Apache e o MySQL;

2. Abra o arquivo cad-produtos.html no Notepad++. Qualquer dúvida chame o seu instrutor. O arquivo está na pasta Arquivos auxiliares/Conteudo/14.

3. Digite o código conforme o indicado;

```
75 <!-- BLOCO PRODUTOS -->
76 {foreach $itemProdotos as $p}
77 <div>
78 <div><span class="title">{$p.destinogre}</span></div>
79 <div><span class="title">{$p.categoria}</span></div>
80 <div>
81 <a href="cad-produtos.php?idpro={$p.idpro}&acao=editar">Clique aqui para editar</a>
82 </div>
83 <div>
84 <a href="cad-produtos.php?idpro={$p.idpro}&acao=excluir">Clique aqui para excluir</a>
85 </div>
86 </div>
87 {/foreach}
```

4. Salve o arquivo dentro da pasta administrador. Qualquer dúvida chame o seu instrutor.

5. Abra o localhost/tomdosom/administrador/, clique em Produtos e confira a lista.

14.5. Exercícios de Fixação

1. Este exercício tem como objetivo listar apenas os produtos da categoria tecas, com o id categoria=3. Deve aparecer o produto, o preço e a categoria.

A atualização de dados é uma das atividades que estaremos desenvolvendo nesta aula. São vários detalhes que fazem parte desta etapa. Além dos dados do formulário, ainda temos a possibilidade de alterar a imagem do produto. Sobre a atualização dos produtos: depois da imagem ser alterada e a página atualizada, a imagem ainda fica retida na memória, então entra a solução de um script no arquivo html.

15.1. Atualizar imagem

Algumas ações devem ser criadas para atualizar uma imagem, como definir o caminho, definir o tipo de arquivo (extensão). Outro detalhe é saber se a imagem será alterada, vamos conferir alguns procedimentos.

```
$caminhoPadrao="fotopadrao.jpg";
$extPro=".jpg";
if(empty($idpro)){
    $smarty->assign("foto",$caminhoPadrao);
}else{
    $smarty->assign("foto",$idpro.$extPro);
}
```

A imagem acima apresenta um teste que deve ser realizado quando o administrador da página acessar Produtos. Acontece que, ao entrar em produtos, não vai ter uma foto ilustrando. É necessário clicar em Editar para ela aparecer conforme o produto selecionado, por esse motivo criamos uma imagem padrão em branco, poderia ter o logo da empresa, mas definimos uma imagem em branco para teste.

```
$destino = "produtos/";
$ext=".jpg";
if(file_exists($destino.$ext)){
    echo "Produto existente";
}else{
    move_uploaded_file($foto, $destino.$idpro.$ext);
}
```

A imagem acima apresenta um segundo teste, neste caso se o administrador não modificar a imagem, permanece a atual, caso contrário é feito o upload.

15.2. Atualizar página

Quando o administrador atualizar os dados do produto, incluindo a imagem, certamente as modificações serão verificadas, acontece que a imagem fica na memória, mesmo atualizando no botão “Recarregar esta página” ou com as teclas CTRL + F5, muitas vezes não resolve, então devemos utilizar o comando abaixo para auxiliar.

```
<a href="cad-produtos.php?idpro={$p.idpro}&acao=editar"
onclick="document.location.reload(true)">
</a>
```

Com o método “**location.reload()**” a página atual é recarregada. O parâmetro “**true**” faz com que a página seja recarregada do servidor.

Empty()

A função empty() tem como finalidade verificar se uma variável é vazia, retorna verdadeiro ou falso.

Estrutura:

```
if(empty($variavel)){
    echo "Variável vazia";
}
```

File_exists()

A finalidade deste método é verificar se um arquivo ou diretório existe. Exemplo:

```
$caminho = "documentos/arquivo.txt";  
if(file_exists($caminho)){  
    echo "O arquivo existe";  
}
```

15.3. Exercícios Passo a Passo

1. Abra o Xampp e ative o Apache e o MySQL;
2. Abra o PhpMyAdmin e acesse o banco de dados "pubdoporto";
3. Pesquise os colaboradores do sexo feminino. Clique na tabela de Colaboradores e na aba SQL.
4. Digite o seguinte código: `SELECT * FROM colaboradores WHERE sexo='f'`. Em seguida, clique no botão Executar;

5. Atualize o salário da Camila Silva para 1500. Digite o seguinte código: `UPDATE colaboradores SET salario=1500 WHERE codigo=2`. Em seguida, clique no botão Executar;

6. Verifique se foi alterado clicando na tabela colaboradores.

15.4. Exercícios de Fixação

1. Insira três novos colaboradores, dois homens e uma mulher, defina o salário acima de 1000 para cada um deles.
2. Faça uma pesquisa dos colaboradores com salário abaixo de 1600 e outra dos que ganham acima de 1600.

Nesta apostila, continuamos nos produtos. Criaremos a etapa em que o administrador terá o controle de excluir o produto com suas informações e a imagem. Para excluir o registro, utilizaremos o comando delete.

16.1. Excluindo uma imagem

Para excluir a imagem da pasta, utilizaremos a função unlink(), basta definir o caminho completo.

Estrutura:

```
unlink(caminho_do_arquivo);
caminho_do_arquivo = "pasta/".campoid.extensao
```

Este exemplo abaixo vai excluir a imagem do produto com referência na pasta, no id do produto e no formato de arquivo.

```
unlink("produtos/". $idpro. ".jpg");
```

Nota: é importante saber que, primeiramente, devemos apagar a imagem e depois o registro, se for feito o contrário não teremos o id do produto como referência para rastrear a imagem e apagar.

16.2. Função date()

A função date() formata a data e a hora atual, e possui parâmetros.

Parâmetro	Descrição	Valores
d	Dia do mês com zero à esquerda.	01 até 31
D	Retorna um dia da semana com três letras.	Mon até Sun
m	Retorna de forma numérica o mês, com zero à esquerda.	01 até 12
Y	Retorna o ano com quatro dígitos.	Exemplo: 1990
H	Retorna a hora com zero à esquerda, formato 12 horas.	01 até 12
i	Retorna os minutos com zero à esquerda.	00 até 59
s	Retorna os segundos com zero à esquerda.	00 até 59

16.3. Função time()

A função time() retorna a hora atual.

16.4. Função rand()

A função rand() gera um número inteiro aleatório.

Podemos definir um valor inicial e final, a função vai mostrar de forma aleatória números nesta faixa indicada.

Exemplo:

```
$numero=rand(10,40);
echo $numero;
```

Toda vez que for solicitada, a página vai exibir um número aleatório dentro desta margem.

16.5. Função agregada Sum()

A função SUM() soma um conjunto de valores.

Exemplo:

Somar o preço de todos os produtos da tabela tbprodutos, campo precopro.

```
SELECT sum(precopro) as 'total' FROM tbprodutos
```

16.6. Exercícios Passo a Passo

1. Este exercício tem como objetivo somar o salário de todos os colaboradores. Criaremos um banco de dados chamado "rvtecidos" e copiaremos da pasta "Arquivos

auxiliares/Conteudo/16/Tabela Colaboradores" o código para gerar a tabela de colaboradores. Abra o Xampp e ative o Apache e MySQL;

2. Crie o banco de dados chamado "rvtecidos";

3. Selecione o banco de dados "rvtecidos" e clique na aba "SQL";

4. Abra o arquivo Tabela Colaboradores que está na pasta Arquivos auxiliares e copie o código;

5. Volta para a aba SQL, cole o código e clique em Executar;

6. Clique na aba SQL e digite o código conforme o indicado;

```
SELECT SUM(salario) as "salário" FROM colaboradores
```

7. Calcular o salário dos colaboradores com idade igual ou acima dos 25 anos. Clique na aba SQL e digite o código conforme o indicado;

```
SELECT SUM(salario) as "salário" FROM colaboradores WHERE idade >= 25
```

16.7. Exercícios de Fixação

1. Listar somente os colaboradores com cargo de vendedores;

2. Listar a relação de colaboradores em ordem alfabética;

3. Listar apenas os colaboradores abaixo de 25 anos.

A página permite que o usuário adicione seus produtos no carrinho, visualize os itens selecionados e permita excluir um item da lista. Um dos procedimentos que iremos realizar é gravar os dados do cliente e atualizar a tabela carrinho com o nome e o número da sessão.

Utilizaremos comandos como o INSERT e UPDATE para gravar os dados do cliente e atualizar a tabela carrinho, serão realizados os dois comandos na mesma ação.

17.1. Unset

O comando unset permite apagar a informação de uma variável específica. Com este comando podemos apagar o conteúdo de variáveis do tipo get, post, cookie, session, entre outros.

Estrutura:

Unset(variável).

Exemplo:

```
$login = "gerente";
```

```
Unset($login);
```

No exemplo acima, foi criada uma variável chamada \$login com valor igual à gerente. Usamos o unset para apagar o valor desta variável.

17.2. Exercícios Passo a Passo

1. Este exercício tem como objetivo criar um banco de dados chamado modelagem01. Criaremos três tabelas: fornecedores, produtos e estoque. Serão cadastradas algumas informações em cada tabela. Ative o Xampp e os serviços do Apache e MySQL.

2. Abra o PhpMyAdmin e crie o banco de dados chamado modelagem01;

3. Selecione o banco modelagem01, clique na aba SQL e crie as tabelas conforme o indicado.

```
CREATE TABLE tbfornecedores
(
    idfornecedor int not null AUTO_INCREMENT PRIMARY KEY,
    f_nome varchar(50),
    f_contato varchar(50)
);
CREATE TABLE tbprodutos
(
    idproduto int not null AUTO_INCREMENT PRIMARY KEY,
    p_nome varchar(50),
    p_precocusto float(10,2),
    p_precovenda float(10,2),
    idfornecedor int
);
CREATE TABLE tbestoque
(
    idestoque int not null AUTO_INCREMENT PRIMARY KEY,
    e_qtde int,
    e_unidade varchar(50),
    idproduto int
);
```

4. Clique na guia SQL e cadastre as informações conforme o indicado.

```
INSERT INTO tbfornecedores
(f_nome, f_contato)
VALUES
("Carlos", "carlos@ig.ig"),
("Juliana", "juliana@ig.ig");

INSERT INTO tbprodutos
(p_nome, p_precocusto, p_precovenda, idfornecedor)
VALUES
("Café", 7.50, 8.90, 1),
("Açúcar", 2.20, 3.70, 2),
("Arroz", 10.50, 17.80, 1);

INSERT INTO tbestoque
(e_qtde, e_unidade, idproduto)
VALUES
(20, "pacotes", 1),
(15, "kg", 2),
(23, "kg", 3)
```

17.3. Exercícios de Fixação

1. Este exercício tem como objetivo fazer um filtro entre as tabelas, exibir o nome do produto, a quantidade, o preço de venda e o nome do fornecedor.

Nesta etapa, o administrador vai ter a possibilidade de visualizar os pedidos de cada cliente, utilizaremos o comando “null” para verificar variáveis nulas, a função cycle para alternar cores na tabela e o number_format para definir o formato de moeda.

18.1. Comando Null

O comando NULL permite verificar se uma variável inicia com valor nulo, muitas vezes é necessário para a requisição de página, o php pode solicitar valores iniciais que ainda não foram processados, e por esse motivo pode gerar erros.

Exemplo:

```
$variavel = null;
```

Cycle

Esta função é usada para facilitar o uso de duas ou mais cores, através de um ciclo de valores. Veja o exemplo abaixo.

```
style="background:{cycle values='#DDD,#EEE'};">
```

Number_format

Utilizando a função number_format() formatamos números com separador de milhar. O exemplo abaixo mostra a variável valor e a atribuição da função separada com o símbolo do teclado, pipe “|” e os separadores de milhar. Exemplo:

```
{ $valor|number_format:2:",": "." }
```

18.2. Exercícios Passo a Passo

1. Este exercício tem como objetivo abrir o banco de dados modelagem01 e alterar o preço de venda de cada produto com acréscimo de R\$ 1,50. Abra o Xampp e ative o Apache e o MySQL;

2. Acesse localhost/phpmyadmin/

3. Clique em modelagem01 e selecione a tabela tbprodutos;

4. Clique na aba SQL e digite o código conforme o indicado.

```
UPDATE tbprodutos SET p_precovenda = 8.9+1.5 WHERE idproduto = 1  
UPDATE tbprodutos SET p_precovenda = 3.7+1.5 WHERE idproduto = 2  
UPDATE tbprodutos SET p_precovenda = 19.3+1.5 WHERE idproduto = 3
```

18.3. Exercícios de Fixação

1. Este exercício tem como objetivo abrir o banco de dados modelagem01 e criar a tabela de clientes;

2. Estes serão os campos: idcliente, cli_nome, cli_telefone, cli_dataNas, cli_email, cli_cidade e cli_parcelas.

3. O campo idcliente será do tipo INT. Os campos cli_nome, cli_email e cli_cidade, varchar 50. O campo cli_telefone varchar 15. O campo cli_dataNasc date. O campo cli_parcelas int.

Na aula 19 estaremos disponibilizando na área administrativa controles como editar e alterar dados do cliente. Para editar os dados do cliente passaremos o id do cliente na query para buscar apenas um cliente específico, da mesma forma o id do cliente será usado em outra query para atualizar os dados.

Acrescentaremos um novo campo na tabela de clientes, o campo status, que vai receber o valor “s” indicando que o cliente está ativo ou “n” indicando que o cliente está inativo. Para acrescentar um campo na tabela, utilizamos o comando ALTER.

19.1. ALTER TABLE

O comando ALTER TABLE permite modificar uma tabela existente, é possível adicionar ou remover uma coluna em uma tabela.

Para adicionar um campo na tabela devemos utilizar o comando ADD:

19.1.1. COMANDO ADD

Permite adicionar um campo em uma determinada tabela com seus atributos.

19.1.2. COMANDO DROP

Permite remover um campo de uma determinada tabela.

Vamos mostrar um exemplo. Criaremos uma tabela de clientes, veja o exemplo abaixo:

```
CREATE TABLE clientes
(
    idcliente int NOT null AUTO_INCREMENT PRIMARY key,
    nome varchar(40),
    email varchar(40)
)
```

Agora, adicionar alguns registros.

```
INSERT INTO clientes
(nome, email)
VALUES
("Carlos", "carlos@ig.ig"),
("Lais", "lais@ig.ig"),
("Lania", "lania@ig.ig"),
("Daniel", "daniel@ig.ig")
```

Vamos adicionar os campos telefone e cidade.

```
ALTER TABLE clientes
ADD telefone varchar(40),
ADD cidade varchar(40)
```

Para remover um campo da tabela:

```
ALTER TABLE clientes
DROP telefone
```

Quando executamos o comando drop uma caixa de diálogo pede confirmação se iremos ou não excluir o campo.



19.2. Exercícios Passo a Passo

1. Este exercício tem como objetivo criar um banco de dados chamado dbescola. Criaremos uma tabela chamada alunos com os campos idaluno, nome e email. Serão cadastrados 5 alunos. Abra o Xampp, ative o Apache e o MySQL.

2. Acesse o Chrome e digite localhost/phpmyadmin/

3. Clique em Início e na guia Base de Dados. Digite dbescola

4. Clique no banco dbescola e na guia SQL;

5. Digite o código conforme o indicado;

```
CREATE TABLE alunos
(
    idaluno int NOT null AUTO_INCREMENT PRIMARY KEY,
    nome varchar(50),
    email varchar(50)
)
```

6. Clique em Executar. Clique na tabela alunos.

7. Clique na guia SQL e digite o código conforme o indicado.

```
INSERT INTO alunos
(nome, email)
VALUES
("Aline", "aline@ig.ig"),
("Marcio", "marcio@ig.ig"),
("Carlos", "carlos@ig.ig"),
("Julio", "julio@ig.ig"),
("Guilherme", "guilherme@ig.ig")
```

8. Clique em Executar. Clique na tabela alunos.

19.3. Exercícios de Fixação

1. Este exercício tem como objetivo adicionar um campo para preencher a média de cada aluno.

Clique na tabela alunos. O campo media deve ser do tipo numeric(10,2).

Na aula 20 foram habilitados os controles para ativar e desativar um cliente. Criaremos a tela de login para acesso à área administrativa. Aprenderemos a Exportar e Importar o banco de dados.

Para realizar o procedimento de desativar um cliente, utilizamos o comando update e passamos o valor “n” para o campo status, para ativar, passamos o valor “s”.

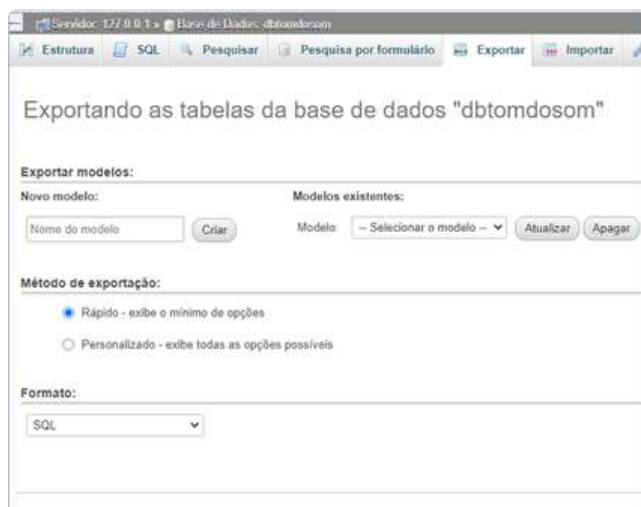
Utilizamos o comando UPDATE para ativar/desativar.

O login do administrador foi testado comparando os dados de entrada com o banco de dados, foi verificado se a variável foi preenchida, evitando acesso imediato.

Utilizamos a função EMPTY() para verificar se a variável está vazia, a superglobal \$_SESSION para criar uma sessão ao se logar no administrador. Usamos a função ISSET() para verificar se uma variável é existente ou se ela não possui valor nulo.

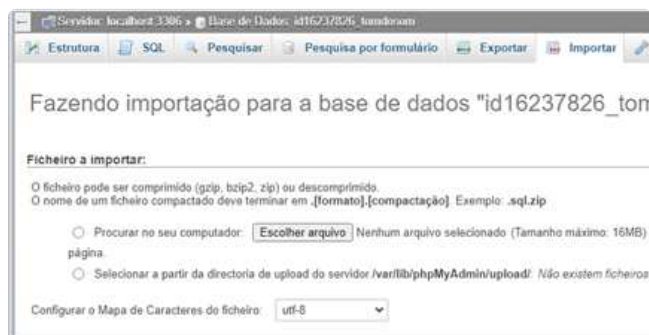
20.1. Exportar dados

Utilizamos a guia Exportar para fazer download do banco de dados para o computador, gerando um arquivo do tipo SQL, fácil e prático de manipular em qualquer programa.



20.2. Importar

No servidor 000webhost utilizamos a guia Importar para fazer upload do banco de dados.



20.3. Exercícios Passo a Passo

1. Este exercício tem como objetivo criar a tabela de cursos com os campos idcurso, curso, descricao e valor. Serão cadastrados 5 cursos. Abra o Xampp, ative o Apache e o MySQL.
2. Abra o Google Chrome, digite localhost/phpmyadmin/
3. Clique em dbescola e na guia SQL;
4. Digite o código conforme o indicado;


```
CREATE TABLE cursos
(
  idcurso int NOT null AUTO_INCREMENT PRIMARY KEY,
  curso varchar(50),
  descricao varchar(50),
  valor float(10,2)
)ENGINE=INNODB
```

5. Cadastre os seguintes cursos conforme o indicado;

```
INSERT INTO cursos
(curso, descricao, valor)
VALUES
("HTML e CSS", "Criação de Layouts", 650.00),
("InDesign", "Direção de Arte e Design de Logo", 700.00),
("Excel Avançado", "Planilhas Dinâmicas", 650.00),
("CoreDraw", "Criação de Logo", 550.00),
("Photoshop", "Edição de imagem", 720.00)
```

20.4. Exercícios de Fixação

1. Este exercício tem como objetivo atualizar o campo média de todos os alunos;

2. Determinaremos as médias para cada aluno conforme a lista. Aline 7.0, Marcio 6.5, Carlos 5.5, Julio 8.2 e Guilherme 7.0