PHP COM LINUX E MySQL PARA INICIANTES

MÓDULO: INICIANDO COM MySQL:

https://www.diegobrocanelli.com.br/mysql/comandos-basicos-mysql-no-terminal/,

https://www.fosslinux.com/96135/how-to-properly-uninstall-mysql-server-in-ubuntu.htm

https://www.digitalocean.com/community/tutorials/how-to-install-mysql-on-ubuntu-20-04

MySQL:

É o banco de dados open source mais popular do mundo, ele possui vários fatores que o tornam um poderoso banco de dados para ser utilizado em vários projetos.

→ Fácil de usar
→ Utiliza a linguagem SQL
→ Alta compatibilidade
→ Multiusuários;
→ Fácil manutenção
→ Open source

O QUE É UM BANCO DE DADOS:

Um local onde podemos armazenar informações para consultas e utilização quando necessário

O db vai gerenciar os registros armazenados nele, por isso também poderá ser chamado de Sistema Gerenciador de db ou SGDB.

BANCO DE DADOS RELACIONAL:

Os bancos de dados relacionais armazenam os dados em linha e colunas, em tabelas, e são baseados na álgebra relacional, um campo da teoria dos conjuntos algébricos. O banco de dados relacional é constituído de: **campos, colunas, ou tuplas** e **tabelas.**

As linhas se relacionam com as colunas, por isso a denominação **relacional.**

A estrutura dos bancos relacionais permite associar informações de diferentes tabelas, utilizando **chaves estrangeiras**, também chamadas de índices.

BANCO DE DADOS NÃO RELACIONAL:

Já os bancos de dados não relacionais não é estruturado por tabelas, logo, não são necessários esquemas para adicionar um dado relacionando uma tabela com uma linha

INSTALANDO O MySQL:

sudo apt install mysql-server

ALTERAR A SENHA DO ROOT:

https://linuxhint.com/change-mysql-root-password-ubuntu/

Checar a versão do MySQL:
mysql –version

Parar o MySQL server:
sudo systemctl stop mysql.service

Checar se o MySQL parou:
sudo systemctl status mysql.service

Iniciar o MySQL sem conceder as tabelas e verificação de rede:
sudo systemctl set-environment MYSQLD_OPTS="--skip-networking –skip-grant-tables"

Agora podemos efetuar o login sem fornecer nenhuma senha.

Iniciar o server MySQL novamente:
sudo systemctl start mysql.service

Confirmar se o server está rodando:
sudo systemctl status mysql.service

Fazendo login no usuário root sem fazer login:
sudo mysql -u root

Liberar os privilégios:
flush privileges;

Selecionar o banco de dados:
use mysql;

Mudar a senha:
ALTER USER 'root'@'localhost' IDENTIFIED BY 'the-new-password';

Sair do Mysql
quit;

Reverter o servidor para as configurações normais:
sudo systemctl unset-environment MYSQLD_OPTS

Depois disso, remover a configuração do sistema modificado revertendo MySQL

sudo systemctl reverter mysql

Agora, mate todos os processos MySQL
sudo killall -u mysql

Reiniciar o servidor
sudo systemctl restart mysql.service

Faça login com a senha definida:
sudo mysql -u root -p

COMANDOS BÁSICOS:

show databases → Mostra as tabelas
use nomeDatabase → Utilizar a database
desc nomeTabela → Lista as tabelas criadas

CRIAR UMA TABELA:

Logar no MySQL com root
sudo mysql -u root -p

Criar a tabela:
CREATE DATABASE loja;

Selecionar a loja:
USE loja;

Criando uma tabela:

```
CREATE TABLE produtos (
    p_id INT NOT NULL AUTO_INCREMENT,
    p_nome VARCHAR (100) NOT NULL,
    p_autor VARCHAR(100) NOT NULL,
    p_tipo INT NOT NULL,
    p_genero VARCHAR (50),
    p_data DATE,
    PRIMARY KEY(p_id));
```

ALTER TABLE:

https://cursos.alura.com.br/forum/topico-como-deletar-uma-coluna-204858

Adiciona um campo na tabela:

```
ALTER TABLE produtos
ADD p_gravadora VARCHAR (50);
```

Para apagar uma coluna:
ALTER TABLE nomeTabela

DROP COLUMN nomeColuna;

Para alterar o nome na coluna:

ALTER TABLE nomeTabela
CHANGE nomeColuna novoNomeColuna **VARCHAR**(50);

CRIAR INDICE:

https://www.devmedia.com.br/indices-no-sql-server/18353

CREATE INDEX nomeIndice
ON nomeTabela (nomeColuna **ASC**);

INSERINDO DADOS NA TABELA:

Exemplo:

INSERT INTO produtos
VALUES
(1, 'nevermind', 'nirvana', 1, 'grunge', '1991-02-01', 'records');

ALTERANDO DADOS DA TABELA:

Exemplo:

UPDATE produtos
SET p_data = '1996-06-14'
WHERE p_id = 2;

DELETANDO DADOS DA TABELA:

Exemplo:

DELETE FROM produtos
WHERE p_id = 1;

Para complemento do material de MySQL, utilizar a apostila anotacoes.odt de 7-postgreSQL

PHP:

MÓDULO: INICIANDO COM O PHP 7.2

INTRODUÇÃO:

É uma linguagem de programação utilizada para construir sites dinâmicos, extensões de integração de aplicações e agilizar no desenvolvimento de um sistema.

→ Linguagem mais utilizada na web.
→ Desenvolvimento focado para web.
→ Frameworks ideias para todo tipo de projeto.
→ Suporte constante (está sempre sendo atualizado).
→ Utilizado por sites grandiosos:
- Wikipedia
- Pinterest
- Facebook

INSTALANDO O APACHE E PHP

sudo apt install apache2

sudo apt install php

INSTALAR MÓDULOS DO PHP: (IMPORTANTE!)

sudo apt install **php-mysql** ou sudo apt install **php-pgsql**

Para ver todos os módulos instalados no php:
apt-cache search --names-only ^php

ABRIR UM SERVIDOR NO PHP:

php -S localhost:8080

DECLARAR UMA VARIÁVEL:

$nomeVariavel = valor;

var_dump () :
Mostra o tipo da variável, function, array. Etc

CONSTANTE:
define (“MAXIMO_VALOR”, 100);

ALGORITMOS E LÓGICA DE PROGRAMAÇÃO:

Algoritmos → É uma sequência logica que visa obter uma solução para um determinado tipo de problema.

Lógica de programação → É a organização coesa de uma sequencia de instruções voltadas à resolução de um problema, ou a criação de um software.

→ Variáveis e constantes
→ Tipos de dados
→ Estrutura de seleção e repetição

TIPOS DE DADOS:

→ string
→ number
→ boolean
→ float
→ array
→ objeto

ESTRUTURAS DE DECISÃO:

→ if
→ else if
→ while
→ case
→ for
→ foreach

OPERADORES DE COMPARAÇÃO:

=
!=
==
===
< >
<
<=
>
>=
<=>

OPERADORES DE ATRIBUIÇÃO:

+=	→ adição
-=	→ subtração
***=**	→ multiplicação
/=	→ divisão
%=	→ módulo (resto)
.=	→ concatenação

OPERADORES LÓGICOS:

&& → AND
|| → OU
XOR → Exclusivo
!= → NÃO (Não é verdadeiro).

REQUISIÇÃO DE ARQUIVOS:

INCLUDE E REQUIRE:

Utiliza outros arquivos como base de código. Um exemplo seria: criar um header em um arquivo e fazer um include em outro para reutilizar o código.

Caso o arquivo não for encontrado, **require** pausa a execução do programa, enquanto o **include** permite que continue.

CONFIGURANDO O PHP.INI:

No terminal:

cd /etc/php/8.1/apache2

Após:

sudo nano php.ini

Após:

display_errors = **off** mudar para → display_errors = **on**

Ativar o driver de pdo, iremos utilizar para ter acesso ao banco de dados: (basta retirar a virgula da frente para ativá-lo).

extension=pdo_mysql
extension=pdo_pgsql
extension=pdo_curl
extension=openssl

Configurar o date.timezone para PtRue

date.timezone = America/Sao_Paulo

IMPORTANTE!!!!

EXEMPLOS DE ESTUDOS PHP ATÉ O MOMENTO, IR ATÉ A PASTA
8-php/Udemy/1-iniciandoComPHP

MÓDULO: INICIANDO ORIENTAÇÃO A OBJETOS COM PHP

https://balta.io/blog/orientacao-a-objetos

CLASSES:

Estrutura estática que define um tipo de dado. Essa **classe** pode conter **atributos** que também podemos chamar de variáveis.

Dentro das classes, também podemos ter **funções**, que chamamos de **métodos**. (são usados para manipular os atributos, variáveis).

__CONSTRUCT:

Definir **__construct ($parametros),** faz com que ao criar um **new conta (class)** faz com que ele seja executado como primeiro código, assim, colocando certas condições no código.

Exemplo:

```
public function __construct ($cpfTitular, $nomeTitular) {
        $this → cpfTitular = $cpfTitular;
        $this → nomeTitular = $nomeTitular;
}
```

Para chamar o construct:

```
$primeiraConta = new Conta (“123456789”, “Elbert”);
```

__DESTRUCT:

É chamado quando não há mais nenhuma referência para o objeto em questão.

Destrói um atributo que não está mais endo utilizado na memória/que não foi atribuído a uma variável.

Exemplo:

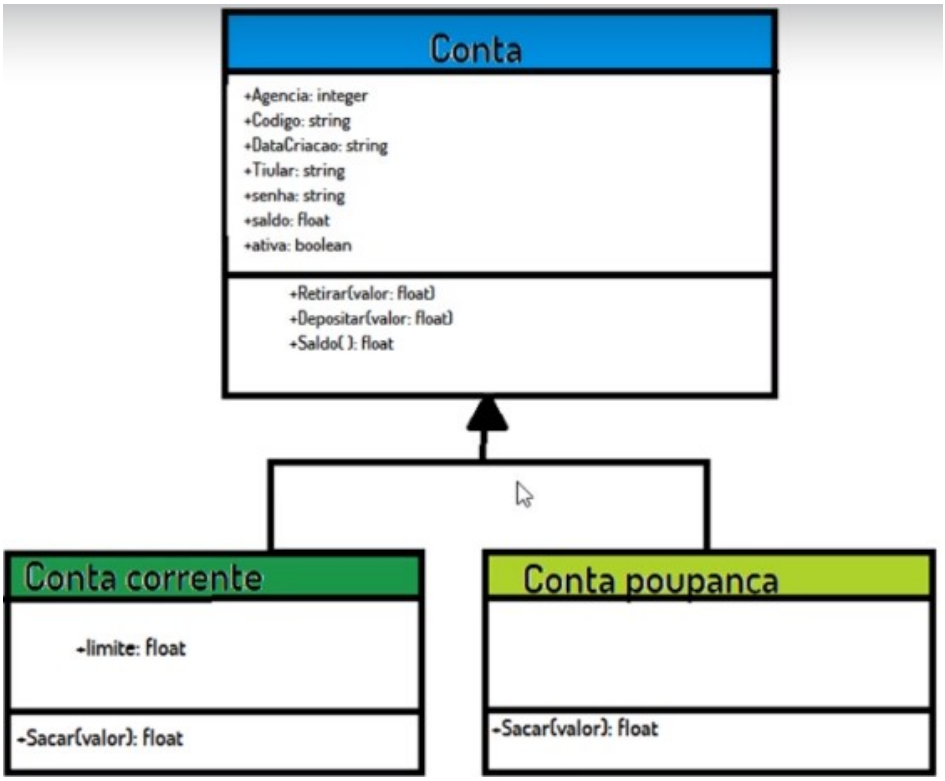
```
function __destruct () {
        echo "O objeto" . $this->descricao . " foi finalizado";
}
```

MÓDULO: AVANÇADO EM ORIENTAÇÃO A OBJETOS COM PHP

HERANÇA:

Pode auxiliar com a duplicação de código. Para acessar um membro da classe base, pode ser utilizar **PARENT.**

Um dos benefícios da herança é evitar a duplicações de códigos.



Para utilizar a classe pai herdada na nova classe, utilizar o **extends**

Exemplo:

class ContaCorrente **extends** Conta **(Definir os parâmetros do pai e filho)**{}

Para herdar os atributos da classe pai, basta utilizar **parent::**

Exemplo:

parent::__construct($agencia, $conta, $titular, $saldo, $ativa, $limite);

POLIMORFISMO:

É o princípio que permite que classes derivadas de uma mesma superclasse tenham métodos iguais (com a mesma nomenclatura, mesmos parâmetros) porém com comportamentos diferentes redefinida em cada uma da classe filha.

Exemplo:

```
function Sacar($quantidade) {
        if (($this->saldo + $this->limite) >= $quantidade){
                parent::Sacar($quantidade);
        } else {
                echo "Saldo insuficiente";
        }
}
```

No **parent::** acima, ele está se referindo a função sacar do arquivo pai.

INTERFACE:https://www.fosslinux.com/96135/how-to-properly-uninstall-mysql-server-in-ubuntu.htm

Permitem a criação de códigos que **especificam** quais métodos uma classe deve implementar, sem definir como esses métodos serão tratados.

Interfaces são definidas da mesma forma que classes, mas com a **palavra-chave interface substituindo class** e com nenhum dos métodos tendo seu conteúdo definido.

Todos os métodos declarados em uma interface devem ser públicos, essa é a natureza de uma interface.

Exemplo:

Criar uma interface:

```
interface iAluno {
        function getNome();
        function setNome($nome);
}
```

Implementando a interface com os métodos:

```
class Aluno implements iAluno {
        function setNome($nome){
                $this->nome = $nome;
         }

        function getNome (){
                return $this->nome;
        }
}
```

Utilizando a interface:

```
$pedro = new Aluno;
$pedro->setNome("Elbert");
```

```
echo $pedro->getNome();
```

ENCAPSULAMENTO:

É uma técnica em programação orientada a objetos que separa o programa em partes, o mais isoladas possível. Ele tem como função proteger o atributo se ele pode ou não ser modificado.

PUBLIC, PROTECTED E PRIVATE:

PUBLIC → Pode ser acessados livremente pela classe em que foi criado, por classes herdeiras e pelo próprio programa que faz uso da Classe. (**PUBLIC** é uma classe padrão do php)

PROTECTED → Somente poderão ser acessados pela classe onde foram criados ou por classes descendentes (mas não podem ser acessados a partir do programa que faz uso da classe)

PRIVATE → Somente podem ser acessados dentro da própria classe em que foram declarados.

SPL__AUTOLOAD:

O spl__autoload captura o nome de uma classe não declarada. Com o nome da classe, basta montar o "path" para carregá-la.

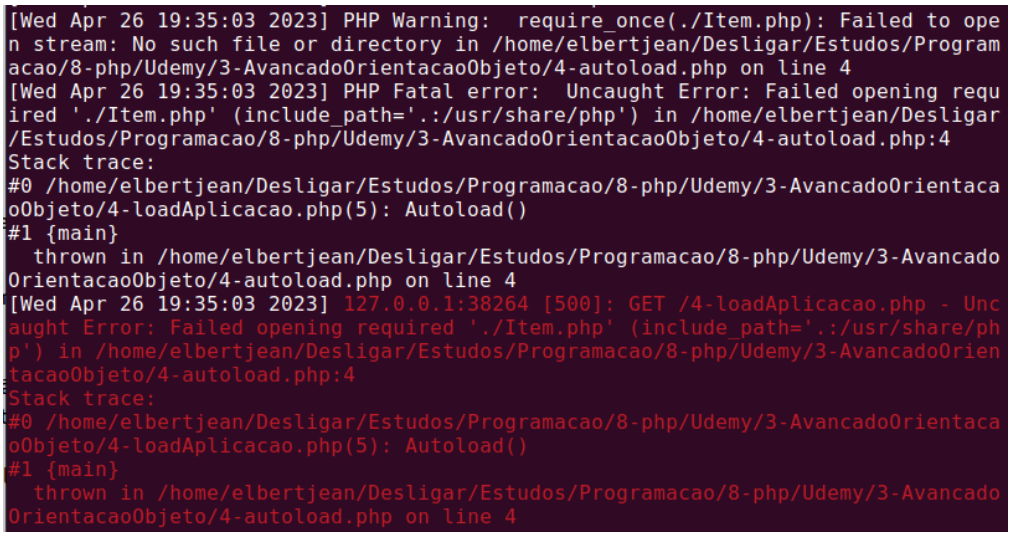
Exemplo:

```
function Autoload($nomeClasse){
        require_once("./class/" . $nomeClasse . ".php");
}

spl_autoload_register("Autoload");
```

OBSERVAÇÕES IMPORTANTES!!!!!!! - (ESTUDAR MAIS SOBRE O ASSUNTO)

Ao fazer os testes na aplicação, a funcao Autoload converte a primeira letra para maiúscula, fazendo assim com que o arquivo não seja encontrado e dando erro.



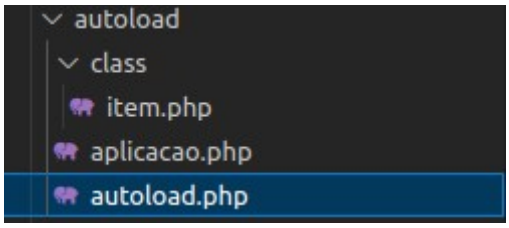
A mesma coisa para arquivos com nomes múltiplos (autoloadItem), o autoload não reconhece o nome do arquivo e capta apenas o nome do **Item** que a primeira letra que está em maiúsculo, ignorando o **autoload** do início do nome do arquivo.

```
[Wed Apr 26 19:35:03 2023] PHP Fatal error:  Uncaught Error: Failed opening requ
ired './Item.php' (include_path='.:/usr/share/php') in /home/elbertjean/Desligar
/Estudos/Programacao/8-php/Udemy/3-AvancadoOrientacaoObjeto/4-autoload.php:4
Stack trace:
```

Um modo de resolver esse problema foi utilizar o strtolower na variável para deixá-la minúscula e renomear o arquivo para letras minúsculas.

```php
function Autoload($nomeClasse){
    require_once("./class/" . strtolower($nomeClasse) . ".php");
}

spl_autoload_register("Autoload");
```



TRATAMENTO DE EXCEPTION:

São usadas para alterar o fluxo normal de um script se ocorrer um erro especificado.

Try {} e catch () {} → Utilizado para tratamento de exceções

Tratamentos dentro do **catch:**

getMessage() -> retorna uma mensagem de erro
getCode() -> retorna o código de erro
getFile() -> retorna o arquivo fonte de onde ocorreu o erro
getLine() -> retorna a linha onde ocorreu o erro
getTrace() -> retorna uma array até onde ocorreu o erro
getTraceAsString() -> retorna as ações em forma de strings

Exemplo:

```
$nome = null;

try{
	if(!$nome){
	throw new Exception("<br>Erro encontrado na variável $nome", 1);
	}
} catch(Exception $a) {
	echo $a->getFile() . "<br>";
	echo $a->getMessage() . "<br>";
	echo $a->getLine() . "<br>";
}
```

PROJETO – CADASTRO DE CDs (CRUD)

Projeto se encontra na pasta **./4-projetoCadastroDeCD**

Abaixo, um vídeo mostrando como ficou o projeto finalizado.

https://youtu.be/HYngmS6zS20

MÓDULO: SESSÕES

INTRODUÇÃO A SESSÕES:

As sessões permitem que o usuário possa efetuar **Login** em um sistema web. Estas informações de login são utilizadas para ter certeza de que o usuário está logado na página e de que ele pode estar usando o sistema.

Para alcançar esta proteção por usuário, nós usamos sessões.

PROTOCOLO HTTP:

Quando navegamos na internet usamos o protocolo HTTP (**protocolo de transferência de HiperTexto**).

O HTTP trata cada requisição de páginas como única, ou seja, toda vez que você tenta acessar uma página da web, mesmo que você já tenha acessado esta página a 1 minuto atrás, o protocolo HTTP vai abrir uma nova requisição para a página, como se você estivesse acessando pela primeira vez.

O protocolo HTTP não armazena as imagens ou textos da página web que você acessou, por tanto, a cada novo acesso, ele requisita os mesmos dados.

Para cada requisição do usuário, as Sessions criadas no PHP são apagadas. Para “reverter” esta situação o PHP utiliza os **Cookies**.

O PHP cria no HD do servidor, dentro de uma pasta temp um arquivo semelhante à:

sess_0cc195bbc0f2b6a331c349e326975964

Nete arquivo fica gravado as informações contidas na variável global **$_SESSION**.

Ao mesmo tempo, o PHP cria um COOKIE com o mesmo valor acima, porém sem o **sess_**

session_start() → Quando utilizado, o PHP acessa o seu cookie criado para saber o nome do arquivo e vai "alimentar" a variável global $_SESSION com os valores presentes neste arquivo.

Desta forma, o PHP consegue guardar as informações, mesmo trabalhando com várias requisições.

PREPARED STATEMENT:

É usada para executar a mesma instrução repetidamente com alta eficiência e proteger contra injeções de SQL.

A execução da instrução preparada consiste em duas etapas: preparar e executar. No estágio de preparação, um modelo de instrução é enviado ao servidor de banco de dados. O servidor executa uma verificação de sintaxe e inicializa os recursos internos do servidor para uso posterior.

O servidor MySQL oferece suporte ao uso de espaço reservado anônimo e posicional com **?**.

DICAS DE SITES:

https://br.freepik.com/

https://www.flaticon.com/br/

https://icons8.com.br/

https://bootsnipp.com/

https://fontawesome.com/icons?d=gallery&p=2&q=down

https://unsplash.com/pt-br

DICAS DE LIVROS:

PHP PROGRAMANDO COM ORIENTAÇÃO A OBJETOS
Pablo Dell'Ogilio

CONSTRUINDO APLICAÇÕES WEB COM PHP E MYSQL
André Milani

JAVASCRIPT O GUIA DEFINITIVO
David Flanagan