



Curso Técnico em Desenvolvimento de Sistemas Online

BANCO DE DADOS II

GEEaD - Grupo de Estudo de Educação a Distância

Centro de Educação Tecnológica Paula Souza

Expediente

GEEaD – CETEC
GOVERNO DO ESTADO DE SÃO PAULO

EIXO TECNOLÓGICO DE INFORMAÇÃO E COMUNICAÇÃO
CURSO TÉCNICO EM DESENVOLVIMENTO DE SISTEMAS

FUNDAMENTOS DE INFORMÁTICA

Autor: José Mendes da Silva Neto

Revisão Técnica: Eliana C. Nogueira Barion / Lilian Aparecida Bertini

Revisão Gramatical: Juçara Maria Montenegro Simonsen Santos Editoração e

Diagramação: Flávio Biazim

APRESENTAÇÃO

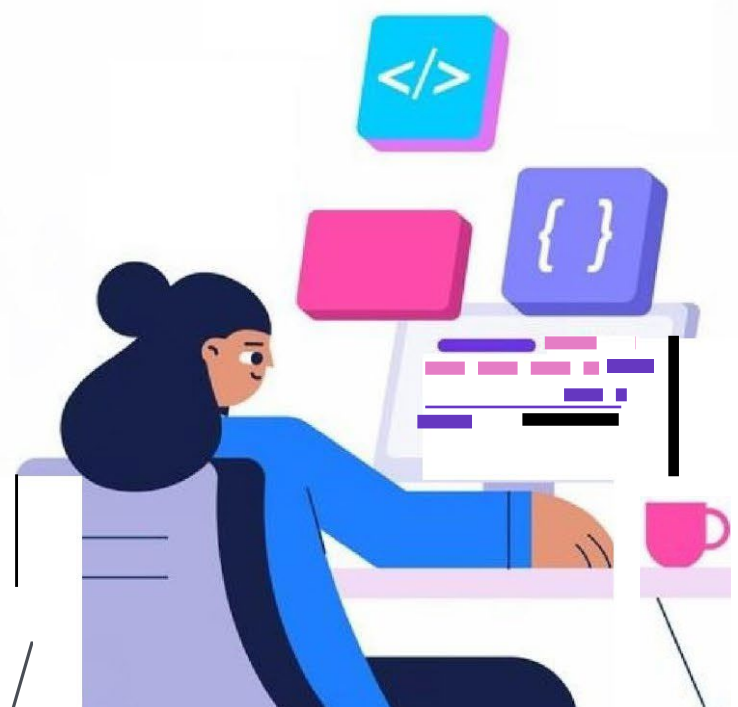
Este material didático do Curso Técnico em Desenvolvimento de Sistemas modalidade EaD foi elaborado especialmente por professores do Centro Paula Souza para as Escolas Técnicas Estaduais – ETECs.

O material foi elaborado para servir de apoio aos estudos dos discentes para que estes atinjam as competências e as habilidades profissionais necessárias para a sua plena formação como Técnicos em Desenvolvimento de Sistemas.

Esperamos que este livro possa contribuir para uma melhor formação e aperfeiçoamento dos futuros Técnicos.

AGENDA 4

APRESENTAÇÃO DA LINGUAGEM SQL ANSI





MERGULHANDO NO TEMA...

Sabemos que o Modelo Entidade-Relacionamento é um modelo conceitual usado para identificar como as entidades (pessoas, objetos ou conceitos) com suas propriedades e características (atributos) se relacionam entre si dentro de um sistema (relacionamento). Por meio de um conjunto definido de símbolos, representamos no Diagrama Entidade Relacionamento (DER), a estrutura do banco de dados do sistema que se pretende desenvolver.

Sabemos também que o Modelo Lógico parte para a construção de um modelo fortemente dependente do ambiente em que será implementado o Sistema Gerenciador de Banco de Dados. Sendo assim, os modelos são bastante dependentes da tecnologia a ser adotada (relacional, redes, orientado a objetos) e que para obtenção de um modelo lógico, a partir de um modelo conceitual, é preciso aplicar determinadas regras de derivação.

Nosso próximo passo é implementar o Banco de Dados, ou seja, desenvolver o Projeto Físico, trabalhando com o modelo relacional e o Sistema Gerenciador de Banco de dados MySQL. Para isso, utilizaremos a linguagem SQL (Structured Query Language - Linguagem de Consulta Estruturada) para implementar o projeto.

Mas antes, vamos estudar um pouco sobre a origem do Modelo Relacional e do SQL, retirado do Livro *Aprendendo SQL: Dominando os Fundamentos de SQL*, BEAULIEU, ALAN, Novatec Editora, 2010:

Em 1970, o Dr. E.F. Codd, do laboratório de pesquisas da IBM, publicou o artigo intitulado “A Relational Model of Data for Large Shared Data Banks” (“Um modelo relacional de dados para banco de dados volumosos compartilhados”) que propunha a representação dos dados como conjuntos de tabelas. Em vez de usar ponteiros para navegar entre entidades relacionadas, dados redundantes seriam usados para conectar registros em diferentes tabelas. A imagem a seguir mostra como as informações das contas de George e Sue seriam representadas nesse contexto.

Cliente

cliente_id	pnome	unome
1	George	Blake
2	Sue	Smith

Conta

conta_id	produto_id	cliente_id	saldo
103	CHK	1	75,00
104	SAV	1	250,00
105	CHK	2	783,64
106	MM	2	500,00
107	LOC	2	0,00

Produto

produto_id	nome
CHK	Emissão de cheques
SAV	Poupança
MM	Aplicações
LOC	Linha de crédito

Transacao

trs_id	trs_tipo_id	conta_id	valor	data
978	DBT	103	100,00	22/01/2004
979	CDT	103	25,00	05/02/2004
980	DBT	104	250,00	09/03/2004
981	DBT	105	1000,00	25/03/2004
982	CDT	105	138,50	02/04/2004
983	CDT	105	77,86	04/04/2004
984	DBT	106	500,00	27/03/2004

Imagem 2 – Exibição relacional dos dados de conta. Adaptado do livro *Aprendendo SQL: Dominando os Fundamentos de SQL*, BEAUL- IEU, ALAN, Novatec Editora, 2010, página 71.

Junto com a definição de Codd do modelo relacional, é proposta uma linguagem chamada DSL/Alpha para manipular os dados em tabelas relacionais. Logo após a publicação do artigo de Codd, a IBM comissionou um grupo para construir um protótipo de suas ideias. Esse grupo criou uma versão simplificada da DSL/Alpha, que foi chamada de SQUARE. Refinamentos da SQUARE levaram a uma linguagem denominada SEQUEL, que foi, finalmente, renomeada para SQL.

A SQL passou por muitas mudanças ao longo do caminho. Em meados da década de 1980, o American National Standards Institute (ANSI) começou a trabalhar no primeiro padrão da linguagem SQL, que foi publicado em 1986. Refinamentos subsequentes levaram a novos lançamentos do padrão SQL em 1989, 1992, 1999, 2003 e 2006. Junto com os refinamentos do núcleo da linguagem, novas funções foram adicionadas a ela para incorporar funcionalidades orientadas a objetos, entre outras coisas. O padrão SQL de 2006 trouxe importantes avanços, como a integração com XML e a definição da linguagem XQuery, voltada para consultas em documentos XML. Desde então, a linguagem SQL continuou a evoluir, com novos padrões publicados em 2011, 2016 e 2019, incorporando funcionalidades como suporte ampliado a dados temporais, recursos analíticos e melhorias em segurança. Embora muitos SGBDs ainda adotem funcionalidades baseadas em padrões anteriores, é importante saber que a linguagem SQL segue em constante atualização.

A SQL anda de mãos dadas com o modelo relacional porque o resultado de uma consulta SQL é uma tabela (também chamada, nesse contexto, de conjunto-resultado ou, em inglês, result set). Portanto, uma nova tabela permanente pode ser criada em um banco de dados relacional simplesmente armazenando o conjunto-resultado de uma consulta.

Uma última nota: SQL não é um acrônimo de qualquer coisa (apesar de muitas pessoas insistirem que significa “Structure Query Language” - Linguagem Estruturada de Consulta. Ao se referir à linguagem, é

igualmente aceitável dizer as letras individualmente (ou seja, S.Q.L.) ou usar a palavra sequel (pronunciase “síquel”).

Retirado do livro Aprendendo SQL: Dominando os Fundamentos de SQL, BEAULIEU, ALAN, Novatec Editora, 2010).

Para dar sequência ao objetivo dessa agenda, é preciso agora instalar o Sistema Gerenciador de Banco de Dados MYSQL.



O mesmo processo de instalação foi apresentado no componente Desenvolvimento de Sistemas II, caso já tenha instalado o Xampp por lá, não é necessário realizar a instalação novamente, passando para o passo 2.

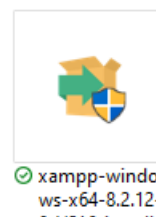
Passo 1: Instalação do ambiente XAMPP

Acesse o site oficial: <https://www.apachefriends.org/index.html>

Faça o download da versão recomendada de acordo com o seu Sistema Operacional:



Após o download, clique no instalador para iniciar a execução:



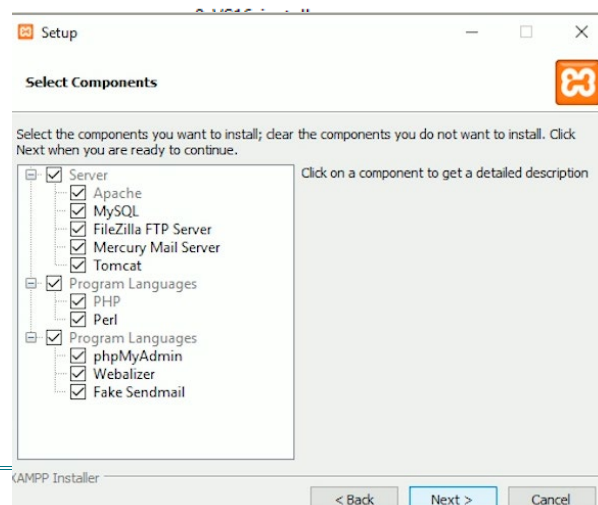
Durante a instalação, selecione os componentes:

Apache (servidor web).

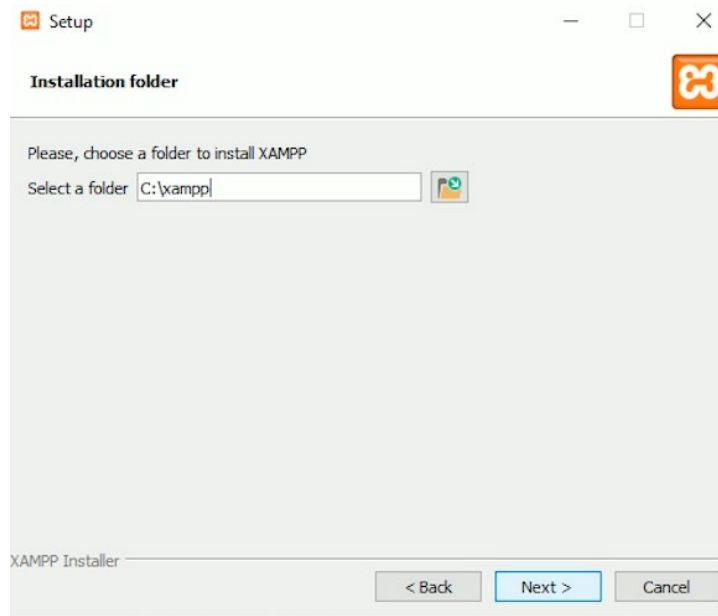
MySQL (banco de dados).

PHP (linguagem de programação).

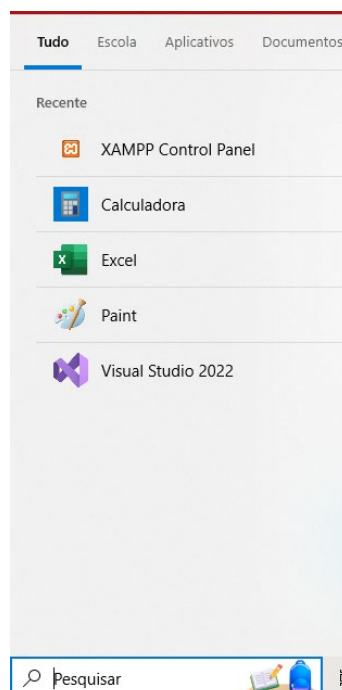
phpMyAdmin (interface gráfica para gerenciar bancos de dados).



Normalmente, instala-se programas na pasta Arquivos de Programas, mas o ideal é manter o Xampp com o caminho mais curto possível – na maioria dos casos, C:\xampp.



Após concluir instalação, abra o Painel de Controle do XAMPP.

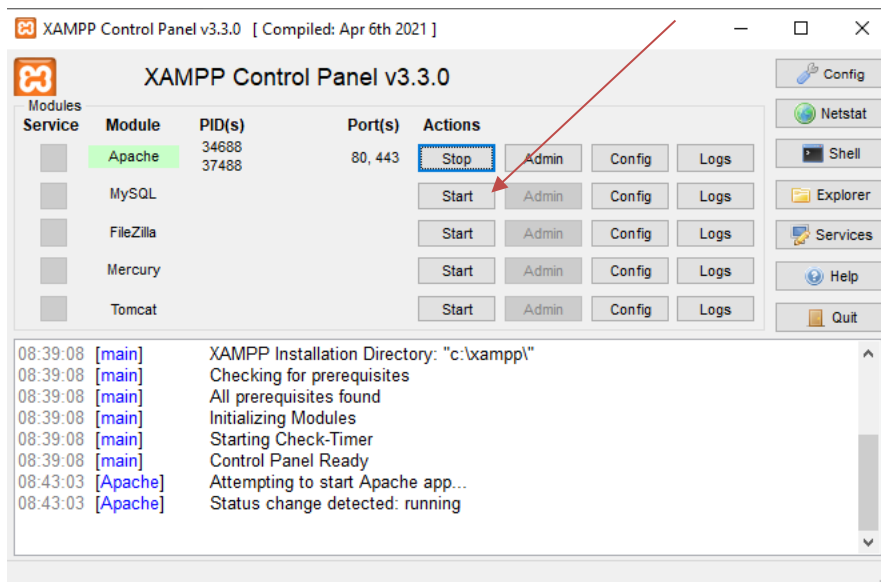


Dica:

Caso o Painel de Controle não apareça automaticamente, o ícone fica disponível ao lado do relógio.



Inicie os serviços Apache e MySQL:



Após finalização da instalação do Xampp vamos iniciar o processo para criação de banco de dados SQL utilizando a ferramenta phpmyadmin.

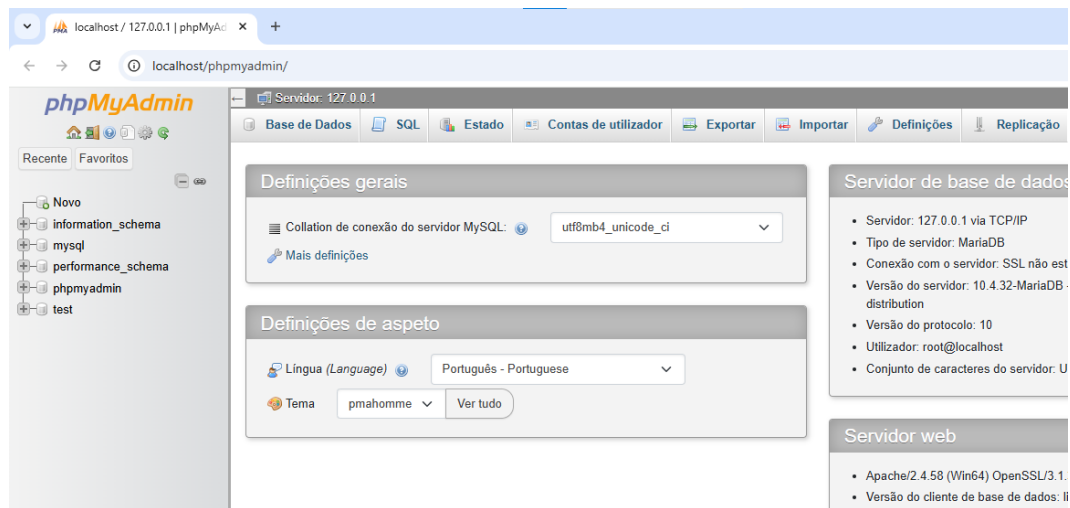
Passo 2: Executando o phpMyAdmin

O phpMyAdmin é uma ferramenta gráfica incluída no XAMPP que facilita o gerenciamento de bancos de dados MySQL. Com ele, você pode criar, modificar e excluir bancos de dados, tabelas e registros sem a necessidade de executar comandos SQL manualmente.

É importante esclarecer que existem vários SGBD's disponíveis, assim como interfaces gráficas que facilitam sua utilização, como é o caso do MySQL e da ferramenta Workbench que utilizaremos nas próximas agendas. Nosso foco agora é o conhecimento da linguagem SQL, utilizada independente do SGBD ou ferramenta gráfica. A ferramenta gráfica facilita muito a implementação de um banco de dados, mas os SGBD's também possuem interface de comando, possibilitando ao desenvolvedor implementar o banco por meio de códigos em SQL.

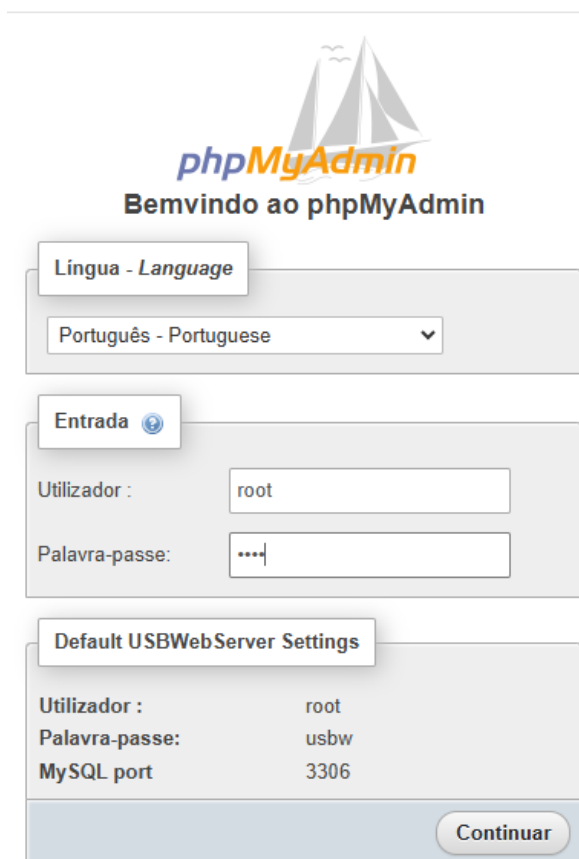
Após a configuração do XAMPP, os comandos SQL podem ser executados diretamente no phpMyAdmin.

Acesse o phpMyAdmin em <http://localhost/phpmyadmin/>.



Importante:

Se estiver utilizando o usbserver para acesso, será solicitado usuário e senha:

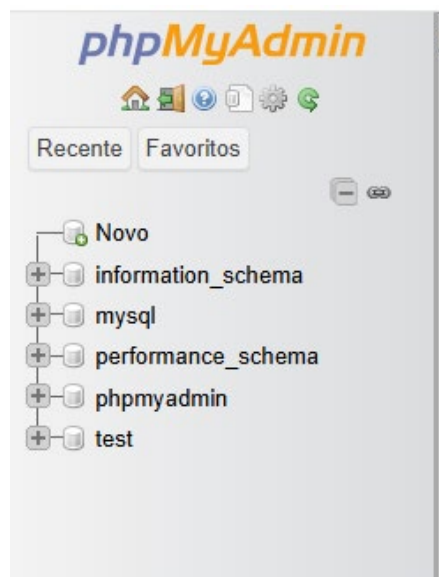


The image shows the phpMyAdmin login interface. At the top, there is a logo with a sailboat and the text "phpMyAdmin Bemvindo ao phpMyAdmin". Below this, there is a section for language selection labeled "Lingua - Language" with a dropdown menu set to "Português - Portuguese". Underneath is a section labeled "Entrada" with fields for "Utilizador :" (username) containing "root" and "Palavra-passe:" (password) containing "usbw". Below that is a section labeled "Default USBWebServer Settings" with fields for "Utilizador :" (username) containing "root", "Palavra-passe:" (password) containing "usbw", and "MySQL port" containing "3306". At the bottom right of this section is a button labeled "Continuar".

Digite para usuário: **root**

Senha: **usbw**

Na tela inicial do phpMyAdmin é possível visualizar a lista de bancos de dados existentes.



Na barra superior é possível visualizar o menu de opções para gerenciamento e criação de base de dados.



Agora, vamos criar um banco de dados para nossos testes, vamos fazer isso primeiramente digitando os comandos SQL de forma manual. Para isso selecione a aba **SQL** disponível no topo da tela.



Em linguagem de programação, quando falamos de sintaxe nos referimos à forma de escrever o código fonte (palavras reservadas, comandos, recursos diversos). Os conteúdos entre os símbolos < > ou [], encontrados na sintaxe, significam que os mesmos devem ser substituídos ou são opcionais, respectivamente.

Vamos em frente!!!

Para criar um banco de dados, utilize o comando **create database** ou **create schema**:

Sintaxe:

```
create database <nome_do_banco_de_dados>;
```

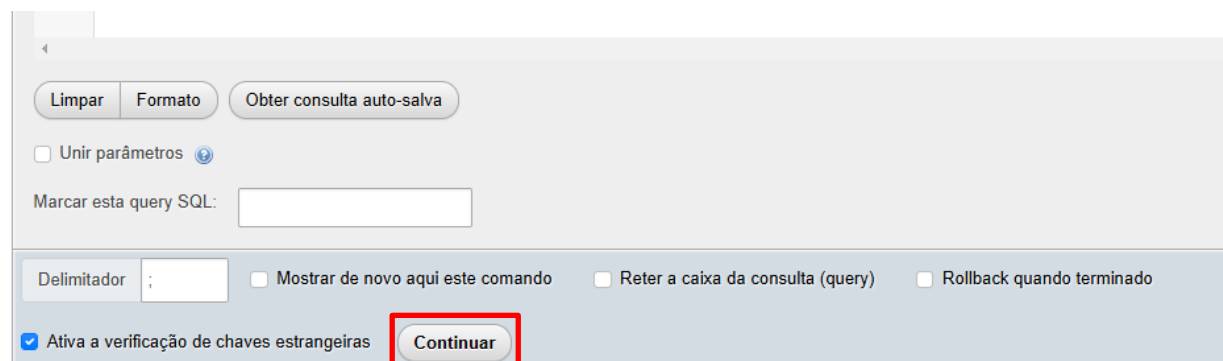
```
create schema <nome_do_banco_de_dados>;
```

Exemplo:

```
create database escola;
```



Clique no botão **Continuar** para executar o comando.



Observe que o banco de dados já foi criado.

Para consultar utilize a sintaxe **show databases;**

Servidor: 127.0.0.1

Base de Dados | SQL | Estado | Contas de utilizador | Exportar

Executar consulta(s) SQL no servidor "127.0.0.1":

```
1 show DATABASES;
```

Limpar | Formato | Obter consulta auto-salva

☐ Unir parâmetros

Marcar esta query SQL:

Delimitador: ; ☐ Mostrar de novo aqui este comando ☐ Reter a caixa da con

☒ Ativa a verificação de chaves estrangeiras **Continuar**

Ao executar todos os bancos são listados, já incluindo o recém-criado.

Mostrar Caixa do query

⚠ A seleção atual não contém uma coluna exclusiva. Os re

A sua consulta SQL foi executada com êxito.

```
show DATABASES;
```

☐ Perfil [[Editar em linha](#)] [[Editar](#)] [[Criar código PHP](#)] [[Ac](#)

Opções extra

Database

- escola
- information_schema
- mysql
- performance_schema
- phpmyadmin
- test

Operações resultantes das consultas

[Imprimir](#) [Copiar para área de transferência](#)

[Marcar este comando SQL](#)

Consola

Para voltar a área de edição SQL basta clicar no botão **Mostrar Caixa do query**.

Observe que no final é digitado “;” (ponto e vírgula), você irá utilizar muito, assim como em outras linguagens. Sua função no MYSQL é a de finalizar a linha de comando, informando que deverá ser executada após clicar no botão **Continuar**.

Vamos conhecer alguns outros comandos disponíveis.

Para selecionar um banco de dados utilize o comando:

Sintaxe: use<nome_banco_de_dados>;

Exemplo: use escola;

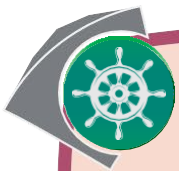
Para listar todas as tabelas de um banco de dados utilize o comando: **show tables;**

Para listar todas as tabelas de um banco de dados utilize o comando: **show tables;**

Importante: Você já deve ter percebido que estamos utilizando somente caracteres em minúsculo, isso é importante por causa do recurso CASE SENSITIVE, que difere letras maiúsculas de minúsculas. Nos computadores com S.O. WINDOWS por padrão não é diferenciado, já em computadores com S.O. LINUX, você poderá ter alguns problemas de adaptação, pois normalmente seus filesystems são CASE SENSITIVE, ou seja, essa diferenciação está presente. Existe uma forma de ajustar isso, mas o melhor é você se adaptar ao tipo de plataforma que está utilizando, portanto, se digitar as letras em minúsculo não terá problema!

Não se preocupe com tanta informação!

Você verá que à medida em que for aplicando os comandos, estes ficarão cada vez mais naturais, além de ter disponível a ferramenta gráfica Workbench que o auxiliará, com mais clareza, no desenvolvimento de códigos.



VOCÊ NO COMANDO

Como você pôde perceber, o entendimento do objetivo do negócio para implementação de qualquer projeto relacionado ao desenvolvimento de sistemas ou de um banco de dados é muito importante. Vimos ainda que esse entendimento deve ser detalhado de maneira organizada, ou seja, não pode ser realizado de qualquer jeito, devemos seguir uma metodologia, conceitos e regras para conseguirmos abstrair o mundo real para um banco de dados, entregando um projeto com que o cliente consiga tomar decisões importantes para seu negócio, a partir da extração dos dados armazenados. Nesse trajeto, passamos, então, pelos Modelos conceitual e lógico, até chegar ao modelo físico, que é o banco de dados propriamente dito, pronto para armazenar informações e receber requisições de consulta, que será a base dos estudos das próximas agendas.

A seguir temos um exemplo de um contexto:

Uma empresa bancária mantém um cadastro com os dados de seus clientes: CPF, nome, endereço (logradouro, número, bairro, cidade, estado e CEP) e telefone, o cliente pode ter mais de um e de suas contas com número e saldo. Um cliente pode ter várias contas no banco e uma conta pode ser de vários clientes (conta conjunta). O banco possui também um cadastro com suas agências (código e nome). Sabe-se ainda que uma agência pode ter várias contas e uma conta pode ser somente de uma agência.



Imagem 04 - freepik

A partir do contexto acima, revise seus conhecimentos adquiridos nas últimas agendas do módulo 1 e desenvolva os projetos conceitual e lógico para a empresa bancária.

Utilize a ferramenta brModelo que você já conhece para auxiliá-lo nesta tarefa. Vamos lá?

1. Começando pelo modelo conceitual, identifique as entidades que a empresa precisa armazenar e os dados que necessitam ser armazenados (atributos). Logo após, identifique os relacionamentos entre as entidades e suas cardinalidades. Feito isso você tem o Projeto Conceitual.

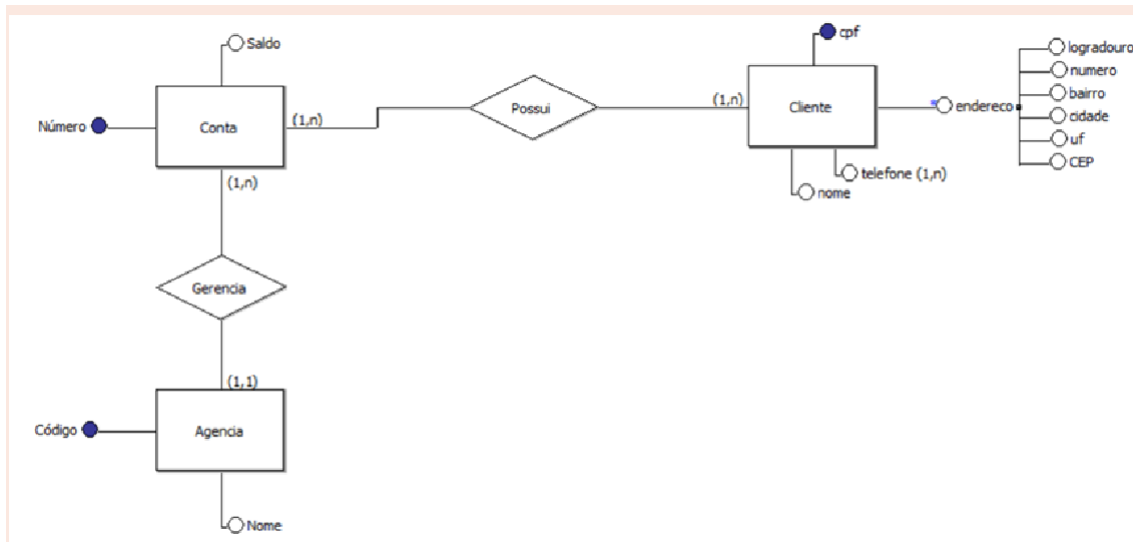
Confira se você acertou:

Imagem 05 - Projeto Conceitual, desenvolvido no aplicativo brModelo

2. Aplique as regras de derivação e de Mapeamento para obter o Projeto Lógico. 7

Confira o seu modelo com o exemplo a seguir:

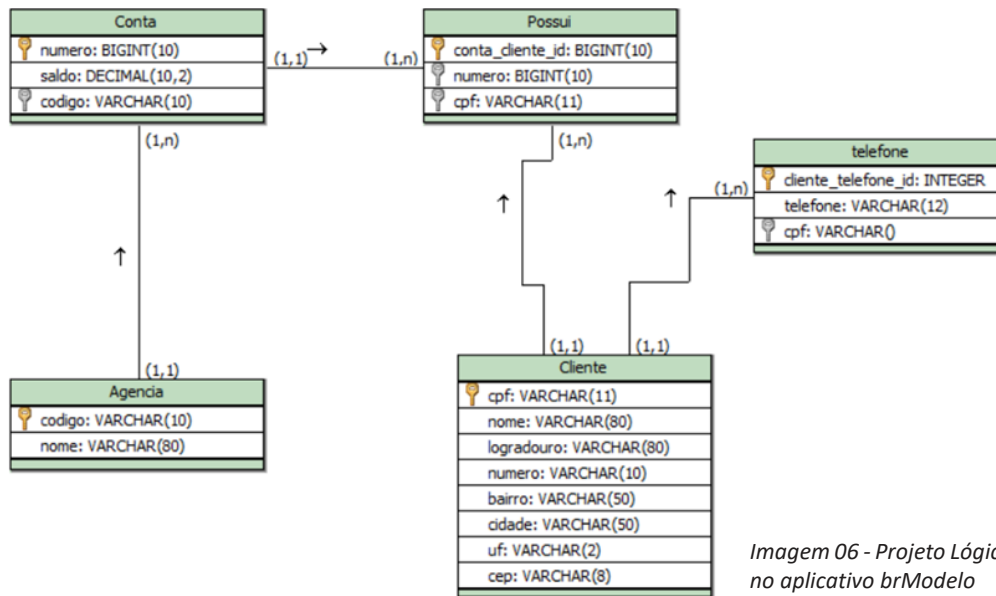


Imagem 06 - Projeto Lógico, desenvolvido no aplicativo brModelo

Para obter o Modelo Lógico foram aplicadas as seguintes regras:

1) Mapear as entidades regulares ou não fracas e seus atributos.

Agência = {codigo, nome}

Conta = {numero, saldo}

Cliente = {cpf, nome, logradouro, numero, bairro, cidade, uf, CEP}

Obs.: o atributo telefone não foi mapeado na entidade Cliente porque é um atributo multivalorado, será tratado em uma outra regra.

2) Mapear as entidades fracas.

Neste exemplo não existe entidades fracas.

3) Mapear os relacionamentos binários 1:1.

Neste exemplo não existe relacionamentos binários 1:1, o que existe é uma visão da entidade Conta em relação ao relacionamento com a entidade Agência, mas a visão da entidade Agência, 1:N, prevalece, por tratar um maior número de ocorrências em relação à visão da entidade Conta.

4) Mapear os relacionamentos binários 1:N.

Neste passo identificamos o relacionamento Gerencia, onde:

- Uma agência pode ter várias contas e uma conta pode pertencer a somente uma agência. Sendo assim, o atributo codigo, da entidade Agência, é adicionado à entidade Conta.

Conta = {numero, saldo, codigo}

5) Mapear relacionamentos binários N:M.

Neste passo identificamos o relacionamento Possui, onde:




- O cliente pode ter mais de uma conta no banco e uma conta pode ser de vários clientes.

Assim, devemos criar uma nova entidade, a partir do relacionamento Possui, adotando como chave primária a concatenação da chave primária de Conta e da chave primária de Cliente, criando-se uma nova entidade:

Possui = {numero, cpf}

Ou ainda, criar uma chave primária para identificar o relacionamento entre as entidades, Conta e Cliente, criando-se uma nova entidade:

Possui = {conta_cliente_id, numero, cpf}

Possui	
	conta_cliente_id: BIGINT(10)
	numero: BIGINT(10)
	cpf: VARCHAR(11)

Um atributo não precisa ser PK para ser único e obrigatório. É o que vai acontecer com numero e CPF, o conteúdo dos dois é obrigatório assim como a ocorrência dos dois é única, fazendo com que o Banco de Dados não permita a inclusão da ligação de um cliente com a mesma conta mais de uma vez, é o que chamamos de UK (Unique Key, Chave Única).

6) Mapear atributos multivalorados.

Neste passo identificamos o atributo telefone, criando assim, uma nova entidade, inserindo o atributo multivalorado e atributo chave da entidade à qual ele pertence com ambos os atributos como chave primária concatenada:

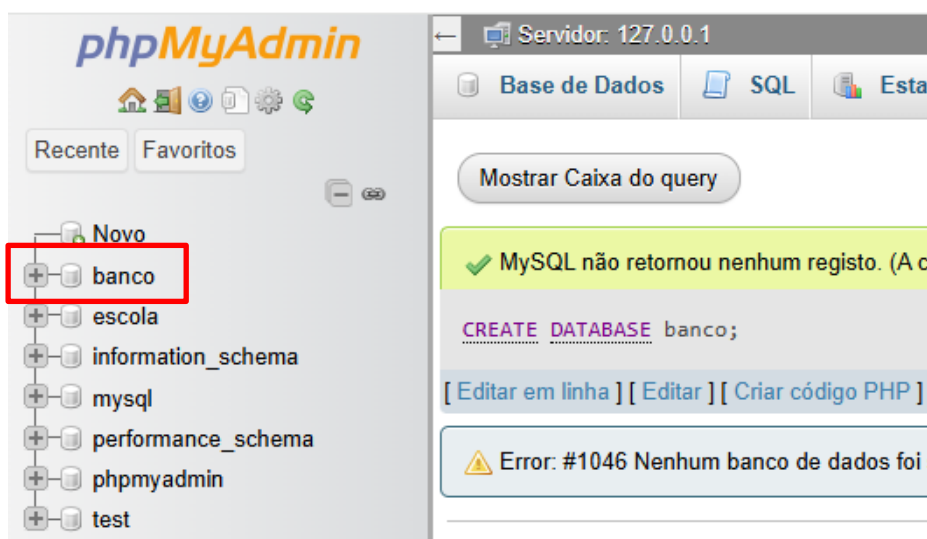
Telefone = {telefone, cpf}

Ou ainda, criar uma chave primária para identificar a ligação entre os atributos, telefone e cpf, assim como foi feito na estrutura Possui, criando-se uma nova entidade:

Telefone = {conta_telefone_id, telefone, cpf}

3. Aproveite e crie o Banco de Dados utilizando o comando **create database**:

create database banco;



Observação: o comando show databases foi utilizado apenas para evidenciar a criação do Banco de Dados.