



# Implementando Banco de Dados

# Sumário

Apresentação	4
<b>Módulo 1</b>	<b>6</b>
Implementando o SQL Server	6
Implementando o SQL Server	7
SQL Server e Management Studio	7
Instalação do SQL Server	9
Instalação do Management Studio	14
Azure Data Studio - alternativa ao Management Studio	16
<b>Módulo 2</b>	<b>19</b>
Configuração do banco de dados	19
Configuração do banco de dados	20
Processo de configuração	21
TableTransportadora	25
TableCategorias	25
TableDetalhesdoPedido	26
TableFornecedores	27
TableFuncionarios	28
TablePedidos	29
TableProdutos	30
Relacionamentos entre tabelas	31
TableClientes e TablePedidos	31
Comandos de manipulação de dados DML	33
Um pouco mais do WHERE	36

# Sumário

Módulo 3	40
Projetando um banco de dados	40
Projetando um banco de dados	41
Estrutura e layout de projeto de banco de dados	41
Análise de requisitos: definição do objetivo do banco de dados	43
Estrutura de banco de dados: organizando os dados em tabelas	47
Criando relações	50
Relação uma a uma	51
Relação uma para muitas	52
Relações muitas para muitas	53
Relações recursivas	54
Relações redundantes	55
Banco de dados: normalização	57
Primeiro Formulário ou Forma Normal (1NF)	58
Segundo Formulário ou Forma Normal (2NF)	59
Terceiro formulário normal ou Forma Normal (3NF)	62
Integridade de dados	63
Inserir índices e exibições	65
Propriedades estendidas	66
Fechamento	69
Referências	70

# Apresentação

Bem-vindo(a) ao curso **Implementando Banco de Dados!**

O objetivo deste curso é proporcionar conhecimentos que o possibilite atuar na atividade de implementação de um banco de dados em empresas de qualquer porte. Para isso, abordaremos noções de técnicas de modelagem de dados, com destaque aos bancos de dados relacionais; e questões de modelagem de bases relacionais e aspectos conceituais, lógicos e físicos de banco de dados.

Desejamos a você um excelente aprendizado!



## Vídeo

Confira o [vídeo](#) de apresentação do curso.

Perdeu algum detalhe? Confira o que foi abordado no vídeo.

Olá! Bem-vindo(a) ao curso Implementando Banco de Dados.

Neste curso você verá uma introdução sobre a implementação de banco de dados, compreendendo sobre esta implementação e suas particularidades, como a utilização da base SQL (*Structured Query Language Server*) e as diversas ferramentas para o seu auxílio.

Além disso, verá as diversas terminologias que são aplicadas na parte de implementação de um banco de dados e diversos passos a passos para sua implementação, para você poder colocar em prática este conjunto indexado de informações.

Vamos começar esta jornada?



Módulo 1

# Implementando o SQL Server

---

# Implementando o SQL Server

A partir de agora, conhecerá um **processo de implementação de banco de dados** utilizando como base o SQL (*Structured Query Language Server*), a ferramenta *Management Studio* e as diferentes formas de acesso a um processo básico de criação de um banco de dados. Acompanhe!

## SQL Server e Management Studio

Terminar a criação de um banco de dados, independentemente da finalidade, não é o último passo para que ele funcione. Na verdade, ainda é preciso que ele seja implementado, sobretudo, a partir de uma linguagem de programação que promova a interação entre usuário e máquina.

Assim, com o intuito de que você saiba mais sobre o assunto, convidamos você a assistir ao vídeo que está disponível seguir. Ele trata de duas plataformas muito importantes que promovem a interação “usuário x máquina” por meio dessa linguagem: o *SQL Server* e o *Management Studio*.



### Vídeo

Confira o [vídeo](#) sobre as plataformas que promovem a interação “usuário x máquina”.

Perdeu algum detalhe? Confira o que foi abordado no vídeo.

Olá! Vamos falar sobre plataformas que promovem a interação “usuário x máquina”?

Quando o assunto é a implementação de um banco de dados, o mercado dispõe de muitas alternativas de ferramentas. Uma delas é a linguagem SQL.

Ela é de conhecimento popular entre os Administradores de Banco de Dados (DBAs) e os desenvolvedores na execução de comandos em bancos de dados de caráter relacional.

Pois, possibilita a criação e a manipulação de tabelas, de colunas e de índices, a atribuição de permissões ao nível de usuários e a consulta aos dados do banco com o qual está trabalhando.

No dia a dia, a linguagem SQL permite o contato entre o usuário e o banco de dados, como uma espécie de diálogo entre eles.

Atualmente, por ser considerado uma porta de entrada para conseguir manusear outras modalidades de banco de dados, o SQL facilita o cotidiano de tarefas em SGBDs. Por isso, ele é uma excelente base para a implementação do banco de dados.

E por ser tão utilizado, tanto o *SQL Server* quanto o *SSMS (SQL Server Management Studio)* possuem um grupo expressivo de ferramentas gráficas e editores de códigos avançados para fornecer acesso fácil e intuitivo aos desenvolvedores e administradores.

Aqui você viu a importância da linguagem para a implementação de banco de dados e como seus recursos possibilitam a interação entre o usuário e a máquina.

Conforme você viu no vídeo, os desenvolvedores e Administradores de Banco de Dados (DBAs) necessitam de uma linguagem e softwares específicos, que possam implementar os bancos de dados de maneira eficaz e funcional. Dentre as várias alternativas atuais, há o *SQL Server* e o *SSMS (SQL Server Management Studio)*, que, além de muito usados, têm recursos gráficos e de edição de linhas de códigos complexas.

Sabendo da importância que possuem, no próximo tópico, você verá os primeiros passos para o download deles, a fim de utilizá-los ao longo das suas atividades envolvendo a implementação dos bancos de dados.

## Instalação do SQL Server

Depois de abordar a necessidade de ter uma linguagem e programas específicos para a implementação de banco de dados, você viu as características básicas que o *SQL Server* tem. Agora, daremos os primeiros passos rumo à sua utilização, começando pelo processo de instalá-lo. Confira!

Tomaremos como base a **instalação do SQL Server** em uma máquina com sistema operacional Windows 10. O processo no Windows Server ocorrerá da mesma maneira e em ambos os casos serão utilizadas versões a partir do ano de 2016.

O **download** está disponível no site da [Microsoft](#).

Ou faça download de uma edição especializada gratuita



Desenvolvedor

O SQL Server 2019 Developer é uma edição gratuita completa, licenciada para uso como banco de dados de desenvolvimento e teste em um ambiente de não produção.

[Fazer download agora >](#)



Express

O SQL Server 2019 Express é uma edição gratuita do SQL Server, ideal para desenvolvimento e produção de aplicações de área de trabalho, Web e pequenos servidores.

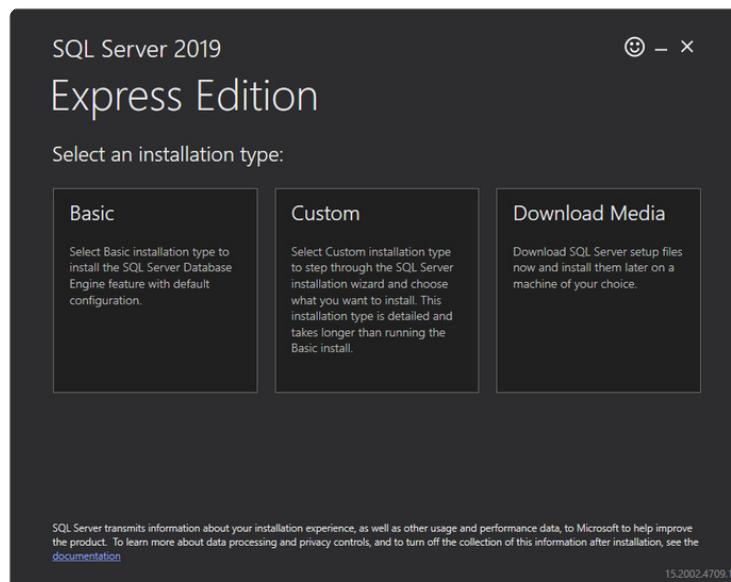
[Fazer download agora >](#)

### #PraCegoVer

Na imagem, há uma captura de tela com dois ícones, um computador e outro com um tablet e um smartphone, representando as opções de download do programa. Da esquerda para a direita, tem escrito desenvolvedor no ícone do computador, e a explicação: O SQL Server 2019 Developer é uma edição gratuita completa, licenciada para uso como banco de dados de desenvolvimento e teste em um ambiente de não produção. E Express no ícone do tablet e um smartphone, com a explicação: O SQL Server 2019 Express é uma edição gratuita do SQL Server, ideal para desenvolvimento e produção de aplicações de área de trabalho, Web e pequenos servidores.

Observe que duas versões estarão disponíveis, mas para o nosso exemplo optaremos pela versão do **SQL Express**, por ser gratuita e destinada ao desenvolvimento de aplicativos de plataforma web, operações com pequenos servidores e *websites*. Basta você selecionar a opção de download para o início da operação.

Um aplicativo será baixado e algumas **opções de instalação** serão apresentadas, conforme você pode observar na imagem abaixo:



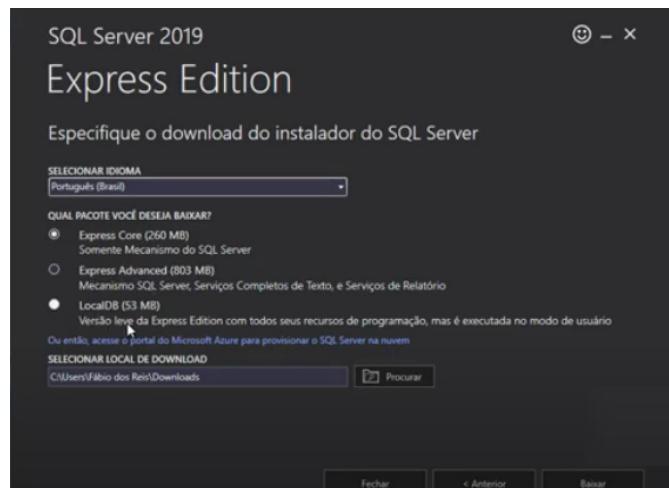
### #PraCegoVer

Na imagem, existe uma captura de tela do site referente à instalação do SQL Server 2019. No topo, há o título “SQL SERVER 2019 EXPRESS EDITION”. Abaixo do título, existem os três tipos de instalação: a *basic*, a *custom* e a download media. Na parte inferior, existe uma informação sobre a transmissão de informação sobre a instalação do SQL Server ao servidor desse programa. Já o fundo da captura de tela é escuro.

- **Básico:** É uma instalação de configurações padrão do mecanismo de dados SQL, e pode ser realizada remotamente.
- **Personalizado:** É uma instalação também remota, mas com o diferencial da personalização, ou seja, possui itens a mais, e por ser mais detalhado, leva mais tempo para concluir a instalação.
- **Baixar Mídia:** Nessa modalidade os arquivos são baixados localmente e a instalação é direto na própria máquina. No caso de mais computadores em rede com necessidade de instalação do SQL, é a opção mais simplificada e rápida.

Na simulação, a opção a ser levada em consideração será a de **Baixar Mídia** (Download Media). No entanto, o processo ocorrerá de maneira similar em todos os casos.

Ao selecionar a opção desejada, uma nova tela vai abrir para que você escolha o instalador. Observe as informações na imagem abaixo para conhecer as opções:



### #PraCegoVer

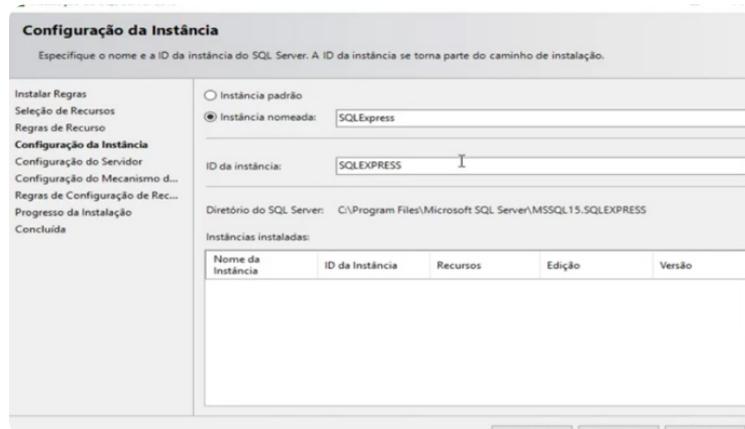
Na imagem, existe uma captura de tela do site referente à instalação do SQL Server 2019. Nela, existe no topo escrito "SQL SERVER 2019 EXPRESS EDITION". Abaixo do título, existe um recurso de escolha do idioma, bem como o local para download ser extraído, com três opções de pacote para fazer o download. Na parte inferior da captura de tela no canto direito, há outros três botões de fechar, anterior e baixar.

- **Express Core:** é a mais recomendada, com **mecanismos padronizados** do SQL. Mas caso você precise de mais recursos, eles podem ser instalados mais tarde e de acordo com a necessidade (eles costumam aparecer selecionados como padrão de download).
- **Express Advanced:** **versão completa** com serviços de texto e relatório.
- **Local DB:** **versão leve** do DB, com recursos em modo usuário.

Uma vez baixado o arquivo, ele deverá ser executado em **modo administrador** e a execução iniciará. Para compreender melhor o passo a passo da execução e quais configurações devem ser feitas, acompanhe com atenção a seguir:

- Quando der início à extração de dados, abrirá a **janela da central** para ser instalado o SQL Server, com as **alternativas** sobre a instalação do SQL Server, manutenção de aplicações e a instalação ou desinstalação de componentes adicionais. No caso de uma **primeira vez**, o recurso '**nova instalação autônoma de SQL Server ou adicionar recursos a uma instalação existente**' deve ser ativado, o que iniciará essa etapa do processo.

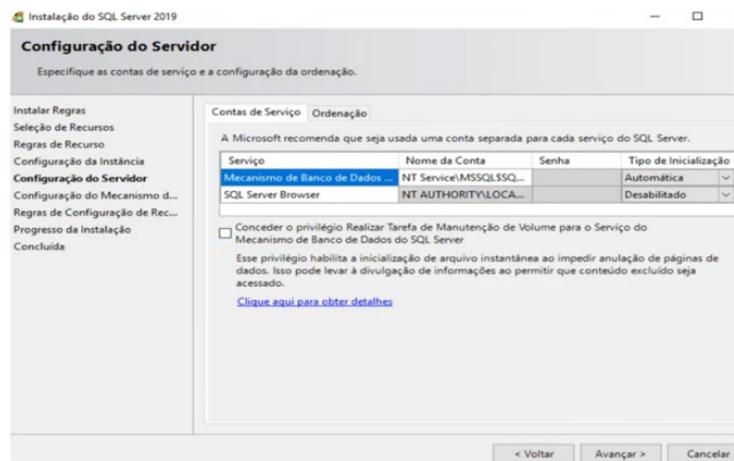
- As **licenças** que surgirão na tela **deverão ser aceitas** e, após uma verificação de regras, outra janela surgirá, recomendando o uso do Microsoft Update. Assim, será recomendável **deixar essa janela marcada** para facilitar os futuros processos de atualização. Os arquivos restantes serão executados e mais uma verificação de regras será realizada.
- Após essa etapa, **uma nova janela com recursos** a serem instalados surgirá; neste caso, trata-se de **configurações padrões**, e basta apenas **continuar**.



### #PraCegoVer

Na imagem, existe uma captura de tela de uma janela referente à instalação do SQL Server 2019. No topo, está escrito configuração da instância. Abaixo dela, existem as opções de configuração com as opções de "instância padrão" e "instância nomeada". Abaixo, há uma caixa de edição denominada de "instâncias instaladas".

- Na **tela de configuração da instância**, você poderá nomear como desejar ou manter as configurações apresentadas. É importante que o **nome de instância não se perca**, pois será necessário para futuros acessos e utilização de aplicações.



### #PraCegoVer

Na imagem, existe uma captura de tela de uma janela referente à instalação do SQL Server 2019. No topo, está escrito configuração do servidor. Abaixo dela, existem as opções de configuração.

- Após a configuração de instância, selecione o campo '**Configurações do servidor**'. A recomendação deverá ser a mesma, mantendo as configurações-padrão.

- A próxima aba requer atenção, pois é referente à '**Configuração do mecanismo de banco de dados**'. É nesta tela que são determinados o **modo de autenticação e os administradores**. Nesse caso, as opções são as de **manter o padrão de autenticação do Windows**, que utiliza o usuário que está logado. No modo misto, ele requer o cadastramento de uma **senha de acesso**. Ambas as configurações podem ser alteradas posteriormente. Ao clicar em **avançar**, essa etapa é iniciada e a instalação do SQL Server estará concluída.

### Atenção

Sempre observe os passos para que não pule nenhuma das etapas.

Vejamos agora outra plataforma que integra as infraestruturas de SQL, neste caso estamos falando do **Management Studio**.

## Instalação do Management Studio

O SSMS (SQL Server Management Studio) é uma **plataforma de integração** que gerencia infraestruturas de SQL. Inclusive, ela apresenta **ferramentas visuais** de configuração, monitoramento, administração de instâncias no SQL Server e para bancos de dados. É um **recurso visual** que facilita no processo de implantação e administração de banco de dados e está separado do download do SQL Server. Por isso, deverá ser **baixado separadamente**.

Para realizar o download, acesse o site da [Microsoft](#), conforme você pode ver abaixo:

### Baixar o SQL Server Management Studio (SSMS)

01/11/2021 • 7 minutos para o fim da leitura • 

Aplica-se a:  SQL Server (todas as versões compatíveis)  Banco de Dados SQL do Azure  Instância Gerenciada do Azure SQL  Azure Synapse Analytics

O SSMS (SQL Server Management Studio) é um ambiente integrado para gerenciar qualquer infraestrutura de SQL, do SQL Server para o Banco de Dados SQL do Azure. O SSMS fornece ferramentas para configurar, monitorar e administrar instâncias do SQL Server e bancos de dados. Use o SSMS para implantar, monitorar e atualizar os componentes da camada de dados usados pelos seus aplicativos, além de criar consultas e scripts.

Use o SSMS para consultar, criar e gerenciar seus bancos de dados e data warehouses, independentemente de onde estiverem – no computador local ou na nuvem.

### Baixar o SSMS

 [Download gratuito do SSMS \(SQL Server Management Studio\) 18.10](#) ↗

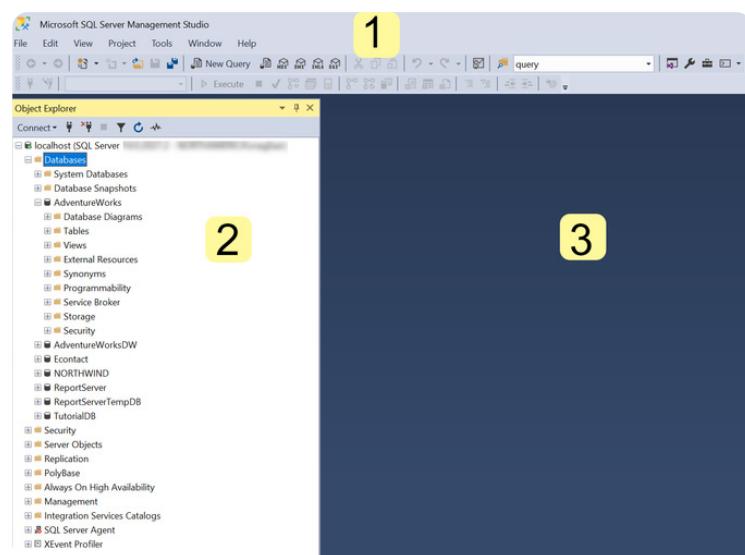
#### #PraCegoVer

Na imagem, há uma captura de tela do site em que o SQL Server Management Studio é baixado. No topo, há o título “Baixar o SQL Server Management Studio (SSMS)”. Abaixo dele, há uma descrição da ferramenta. Mais abaixo, há um outro título “Baixar o SSMS” em que, logo após ele, há o botão para o download gratuito do SSMS.

- Uma vez realizado o download, o arquivo deve ser **executado**, e a tela inicial do instalador abrirá.

- Desta forma, recomenda-se manter as **configurações-padrão**.
- Após a instalação, abra o aplicativo para realizar as **configurações**. O nome de instância do servidor deverá aparecer com a forma de autenticação configurada na instalação do *SQL Server*.
- Deve-se clicar no botão “**conectar**” e aguardar a conexão com o banco de dados. Uma vez realizada a conexão, será possível a criação e o gerenciamento de bancos de dados a partir do *SQL Server*.

A janela do *Management Studio* divide-se em três seções, as quais podem ser observadas adiante:



#### #PraCegoVer

A imagem mostra os três passos, começando pelo número 1, que está na barra superior. O número dois está em uma janela na lateral denominada *Object Explorer*. E o número 3 está ao lado da janela de número 2, que é o campo para exibir os resultados que são obtidos.

- A **barra superior** (1) é o local onde se encontram as principais opções de controle, tais como: abrir, salvar e executar código.
- A janela ***Object Explorer*** (2) concede o acesso ao banco de dados e às tabelas.
- A área **Output** (3) apresenta os resultados obtidos na execução dos códigos.

Neste tópico, você entendeu como fazer o download do *SQL Server* e do SSMS (*SQL Server Management Studio*), os quais são importantes recursos para a implementação

de banco de dados, proporcionando maior interação entre usuário e máquina, independentemente do projeto.

No entanto, é interessante ressaltar que existe uma alternativa ao *Management Studio*, que é o **Azure Data Studio**. No próximo tópico, você entenderá as suas principais características. Vamos lá!

## Azure Data Studio - alternativa ao *Management Studio*

No *podcast* a seguir, entenderemos um pouco mais sobre o Azure Data Studio, uma das alternativas ao *Management Studio*. Confira!



### **Podcast**

Confira o [podcast](#) sobre o Azure Data Studio.

Perdeu algum detalhe? Confira o que foi abordado no *podcast*.

O Azure *Data Studio* é uma ferramenta lançada recentemente pela Microsoft e é mais leve que o *Management Studio*.

Ela foi criada para atuar com bancos de dados SQL Azure, SQL Server e banco de dados em nuvem e está disponível para as plataformas Windows, MacOS, Linux e Docker.

Cabe lembrar que sua intenção não visa à substituição do SQL Server *Management Studio*, mas sim, ser mais uma opção para os desenvolvedores.

Isso porque enquanto o SQL Server *Management Studio* é indicado na administração de bancos de dados e atividades de caráter complexo (como SQL Server), o Azure *Data Studio* é indicado para o trabalho de criação de *queries*.

Visto que o Azure *Data Studio* consiste em uma área de trabalho multiplataforma de software livre, destinada a profissionais que utilizam tanto plataformas de dados locais quanto em nuvem nos ambientes Windows, MacOS e Linux. Ele entrega ao usuário uma experiência de manipulação de dados moderna, intuitiva, fluida e rápida, com painéis personalizáveis, oferecendo mais conforto ao personalizar seu fluxo de atividades.

Sendo assim, o Azure *Data Studio* é uma opção que compensa ser estudada como um complemento à ferramenta de SQL Server *Management Studio*.

Conforme você ouviu no *podcast*, o Azure *Data Studio* é um programa voltado aos administradores de banco de dados com uma plataforma dinâmica e funcional, podendo ser usado tanto nas plataformas de dados locais quanto nos sistemas Windos, MacOS e Linux.



## Saiba mais

Você pode conhecer mais sobre essa ferramenta, assistindo ao vídeo [criando um banco de dados SQL | Microsoft Azure](#).

Parabéns! Você chegou ao fim do Módulo 1.

Aqui, você viu sobre a importância de ter um programa eficaz voltado à implementação de um banco de dados. Para isso, não só apresentamos as características essenciais, como também mostramos como fazer o download e a instalação do *SQL Server Management Studio* e de seu software alternativo, o *Azure Data Studio*.

Contudo, é importante não se limitar somente a isso, pois devemos aprender a configurar adequadamente os bancos de dados relacionais durante a sua implementação.

No próximo módulo, vamos explorar mais a fundo esse assunto. Vamos lá?



Módulo 2

## Configuração do banco de dados

---

# Configuração do banco de dados

Neste módulo, você terá a oportunidade de criar um banco de dados para um estabelecimento de suprimentos utilizando o **T-SQL**. Nele, você verá como são armazenadas e manipuladas as informações.



## #PraCegoVer

Na imagem, um notebook aberto está sendo enquadrado em um fundo escuro e, de modo sobreposto, está escrito a palavra T-SQL com vários códigos de programação em volta dela.

Na criação de um **banco de dados relacional**, as informações são organizadas em tabelas que podem estar relacionadas umas às outras. Assim sendo, os conhecimentos sobre bancos de dados constituem uma base importante para os profissionais de diversas áreas de TI. Desta forma, as bases de dados manipulam e gerenciam os dados e as informações vitais na vida empresarial e pessoal, como dados bancários, transações via internet, bases de dados sociais etc.

Existem alguns **comandos de criação** do banco de dados que precisamos conhecer. Eles dizem respeito às funções básicas de criação de um banco de dados, conforme podemos observar no quadro abaixo.

<b>CREATE</b>	Comando utilizado para efetuar a criação de um banco de dados ou uma Tabela. Sintaxe para criação do banco de dados: CREATE DATABASE nome_banco Sintaxe para criar a tabela: CREATE TABLE nome_tabela
<b>USE</b>	Comando utilizado para escolha e acesso, mais precisamente para a utilização do banco de dados. Sintaxe: USE nome_banco
<b>DROP</b>	Comando cuja função é remover um banco de dados, eliminando consequentemente todas as tabelas que o compõem. Sintaxe: DROP DATABASE nome_banco
<b>GO</b>	Comando que indica o encerramento de uma instrução. Sintaxe: USE nome_banco; GO SELECT * FROM nome_tabela

Após relembrar essas características fundamentais, bem como os papéis relevantes desempenhados pelos bancos de dados relacionais e os principais comandos de criação deles, vamos conhecer o processo de configuração.

## Processo de configuração

Você viu que os bancos de dados desempenham um papel importante nas organizações, sobretudo, por conta do armazenamento e da manipulação das diversas informações.

Porém, vale ressaltar que, para desempenhar esse importante papel, é necessário que uma série de ações sejam empregadas, começando pelo processo de configuração. Sobre isso, observe as informações utilizadas para a composição de um banco de dados:

Arquivo de dados
Nome do banco de dados: Logística
Nome lógico do arquivo de dados: Logística, DAT
Caminho do arquivo: pasta BDLogistica, unidade C
Nome do arquivo: Logistica_Dados.mdf
Tamanho: 5 MB
Tamanho máximo do arquivo: 50 MB
Incremento de crescimento automático do arquivo: 5 MB

O **arquivo de dados** é o local que **armazena informações** do banco de dados. Tal arquivo constitui o **ponto de partida** do banco de dados e aponta para os outros arquivos no interior do banco. Assim, qualquer banco de dados possui um arquivo de dados primário, cuja extensão indicada para nomeá-lo é **o formato de .mdf**.

### Arquivo de Log

Nome lógico do arquivo de log: Logistica\_LOG

Caminho do arquivo: Pasta BDLogistica, unidade C

Nome do arquivo: Logistica\_Dados.Idf

Tamanho: 5 MB

Tamanho máximo do arquivo: 25 MB

Incremento de crescimento automático do arquivo: 5 MB

Um arquivo de *log* mantém todas as informações utilizadas na recuperação do banco de dados. Para cada banco de dados, deverá existir um arquivo de *log*, havendo a possibilidade de existir mais de um. A extensão de nome de arquivo indicada para arquivos de *log* é **.Idf**. Para saber como criar esse arquivo, acesse: [\*\*Criação do arquivo de dados e de log\*\*](#).

O conjunto de tabelas que comporá o banco de dados pode ser dividido conforme a disposição abaixo:



Adiante, conheceremos os detalhes de cada tabela, começando pela **TableClientes**.

Campos	Tipos de Dados	Permitir nulos
CodigoDoCliente	Char (10)	
NomeDaEmpresa	VARCHAR (70)	
NomeDoContato	VARCHAR (70)	
CargoDoContato	VARCHAR (40)	
Endereco	VARCHAR (50)	
Cidade	VARCHAR (25)	
Regiao	VARCHAR (25)	
CEP	Char (15)	
Pais	VARCHAR (25)	
Telefone	Char (20)	
Fax	Char (20)	

A sintaxe de criação da tabela baseada no **T-SQL** utilizará a **instrução Create Table**, a qual é composta de diversas instruções. Observe-a, de modo resumido, e não se esqueça de selecionar os destaques para saber a função de cada instrução utilizada:

```
CREATE TABLE NomeDaTabela
(
    NomeDaColuna TipoDeDados Restrições,
    NomeDaColuna TipoDeDados Restrições,
)
```

A instrução **NomeDaColuna TipoDeDados: Restrições** deverá ser repetida até que o número de colunas seja criado.

Instrução	Função
CREATE TABLE	Instrução que cria a tabela
NomeDaTabela	Nome que será dado à tabela
NomeDaColuna	Nome da coluna da tabela
TipoDeDados	Dado que pode ser <i>Varchar, Char, Int, Null,</i>
Restrições	Define a <i>Primary Key, Unique, Clustered NonClustered</i> e <i>Foreign Key References</i>

A criação das outras tabelas seguirá essa lógica. Por isso, será utilizado o SSMS (*SQL Server Management Studio*). Veja, no link, o processo [Criação de tabelas pelo SSMS - TableClientes](#) e acompanhe o processo detalhadamente.

Desta forma, você acompanhou ao longo deste tópico os elementos essenciais voltados à configuração dos bancos de dados, principalmente, no que diz respeito ao arquivo de dados, ao arquivo de *log* e à sintaxe de criação da tabela baseada no T-SQL.

Em função disso, para que você se aprofunde no tema, veja nos tópicos seguintes as características básicas e as configurações de cada tipo de tabela que comporão o banco de dados que será implementado, a começar pela TableTransportadora. Pronto para começar?

## TableTransportadora

A tabela que vamos conhecer agora será a TableTransportadora, cujas características serão vistas a seguir.

Campos	Tipo de Dados	Permitir nulos
CodigoDaTransportadora	Int	
NomeDaEmpresa	Varchar (30)	
Telefone	Char (20)	Sim

Veja, no link, a [Criação de tabela TableTransportadora](#) e saiba como é simples criá-la.

## TableCategorias

Outra tabela que compõe o banco de dados é a tabela de categorias, observe suas informações.

Campos	Tipo de Dados	Permitir nulos
CodigoDaCategoria	Int	
NomeDaCategoria	Varchar (30)	
Descricao	Varchar (100)	Sim

Para saber como criá-la, convidamos você a ver no link a [Criação da tabela: TableCategorias.](#)

## TableDetalhesdoPedido

Depois de conhecer a TableTransportadora e a TableCategorias, é hora de conhecer a tabela TableDetalhesdoPedido. Vamos entender seus detalhes? Acompanhe a seguir.

Campos	Tipo de Dados	Permitir nulos
NumeroDoPedido	Int	
CodigoDoProduto	Int	
PrecoUnitario	Money	Sim
Quantidade	Int	Sim
Desconto	Float	Sim

No link [Criação da tabela: TableDetalhesdoPedido](#), você verá como pode criá-la sem restar dúvidas!

## TableFornecedores

Agora você entenderá como funciona a TableFornecedores. Veja seus detalhes na tabela abaixo.

Campos	Tipo de Dados	Permitir nulos
CodigoDoFornecedor	Int	
NomeDaEmpresa	Varchar (70)	
NomeDoContato	Varchar (70)	
CargoDoContato	Varchar (40)	Sim
Endereco	Varchar (50)	Sim
Cidade	Varchar (25)	Sim
Regiao	Varchar (25)	Sim
CEP	Char (15)	Sim
Pais	Varchar (25)	Sim
Telefone	Char (20)	Sim
Fax	Char (20)	Sim

Confira como criar essa tabela vendo o link [Criação da tabela: TableFornecedores.](#)

## TableFuncionarios

Chegou a vez de descrever a TableFuncionarios. A seguir, confira quais são as suas principais características.

Campos	Tipo de Dados	Permitir nulos
CodigoDoFuncionario	Int	
Sobrenome	Varchar (30)	Sim
Nome	Varchar (30)	
Cargo	Varchar (40)	Sim
Tratamento	Char (10)	Sim
DataDeNascimento	Date	Sim
DataDeContratacao	Date	Sim
Endereco	Varchar (50)	Sim
Cidade	Varchar (25)	Sim
Regiao	Varchar (25)	Sim
CEP	Char (15)	Sim
Pais	Varchar (25)	Sim
TelefoneResidencial	Char (20)	Sim
Ramal	Char (5)	Sim
Observacoes	Varchar (200)	Sim

Veja como é simples criar essa tabela vendo o link [Criação da tabela: TableFuncionarios](#).

## TablePedidos

Depois de compreender a TableFuncionarios, vamos abordar a TablePedidos. Acompanhe a seguir.

Campos	Tipo de Dados	Permitir nulos
NumeroDoPedido	Int	
CodigoDoCliente	Char (10)	
CodigoDoFuncionario	Int	
DataDoPedido	Date	Sim
DataDeEntrega	Date	Sim
DataDeEnvio	Date	Sim
CodigoDaTransportadora	Int	
Frete	Money	Sim
NomeDoDestinatario	Varchar (50)	Sim
EnderecoDoDestinatario	Varchar (50)	Sim
CEPdeDestino	Char (15)	Sim
PaisDeDestino	Varchar (25)	Sim
CidadeDeDestino	Varchar (25)	Sim
RegiaoDeDestino	Varchar (25)	Sim

No link [Criação da tabela: TablePedidos](#), você pode conferir como ela é criada.

## TableProdutos

Por fim, você verá a TableProdutos, a última que forma o conjunto de tabelas que comporá o banco de dados. Observe as suas características.

Campos	Tipo de Dados	Permitir nulos
CodigoDoProduto	Int	
NomeDoProduto	Varchar (50)	
CodigoDoFornecedor	Int	
CodigoDaCategoria	Int	
QuantidadePorUnidade	Char (30)	Sim
PrecoUnitario	Money	Sim
UnidadesEmEstoque	Int	Sim
UnidadesPedidas	Int	Sim
NivelDeReposicao	Int	Sim
Descontinuado	Int	Sim

No link [Criação da tabela: TableProdutos](#), você pode ver, em detalhes, o processo para criar essa tabela.

Desta forma, vimos a **configuração de banco de dados** sendo feita via script, após essa estrutura ser definida com a ajuda do SSMS (SQL Server Management Studio).



### Saiba mais

Caso queira saber mais, confira a videoaula sobre [Criação de Tabelas com scripts, conceitos de chaves primária e estrangeira](#). Vale a pena conferir!

Até aqui, você entendeu as características, bem como a criação do conjunto de tabelas que comporão o banco de dados. Porém, não basta criá-las, é importante destacar que deve haver um relacionamento entre as tabelas, a fim de que o banco de dados seja adequado. A seguir, abordaremos esse tema com mais detalhes. Vamos lá!

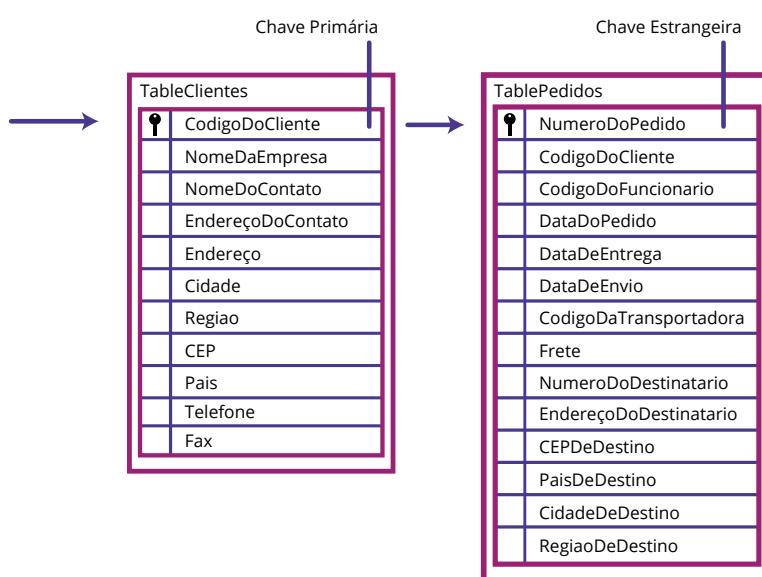
## Relacionamentos entre tabelas

Conforme você estudou, criar as tabelas dos bancos de dados não é o suficiente para que eles funcionem. Além disso, é fundamental estabelecer os relacionamentos entre elas.

Acompanhe agora como trabalhar com **relacionamentos entre tabelas**, lembrando que sempre haverá uma **chave primária** em uma tabela, a qual ela se repetirá como **chave estrangeira** na tabela com a qual se relaciona.

### TableClientes e TablePedidos

Para começar a assimilar os relacionamentos entre tabelas, veja adiante o caso entre a TableClientes e a TablePedidos, sobretudo, percebendo o comportamento da chave primária e da chave estrangeira.

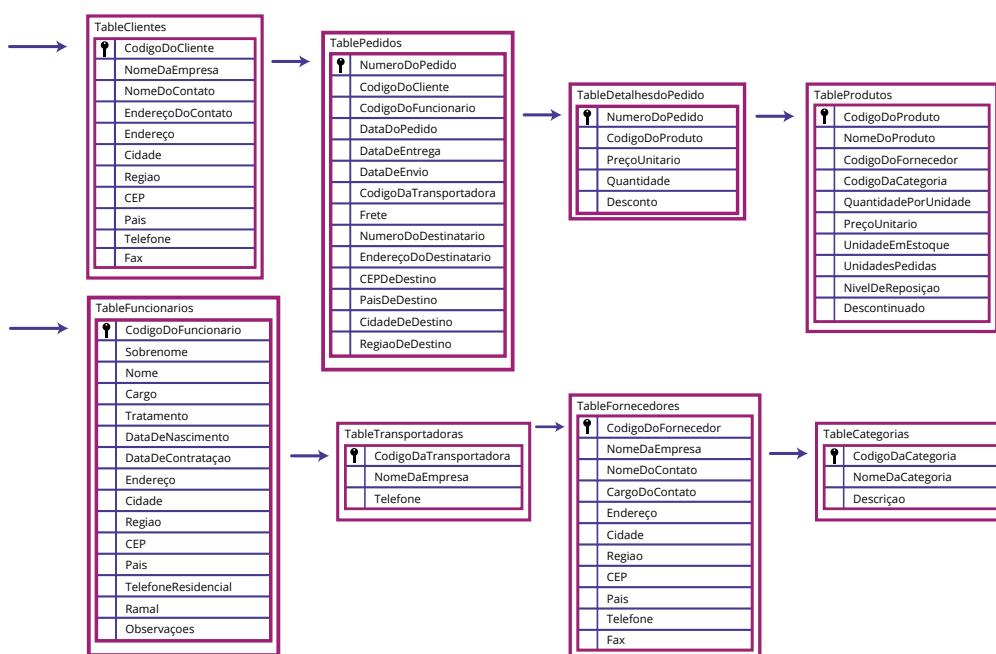


## #PraCegoVer

A figura mostra o relacionamento entre duas tabelas: a TableClientes e a TablePedidos (tabela clientes e a tabela pedidos). Da esquerda para a direita, na tabela clientes, há uma coluna com doze linhas, sendo que, na primeira de cima para baixo, está localizada a chave primária de onde parte uma linha que leva a uma legenda de mesmo nome. No restante das linhas, existem os dados dos clientes. Já na segunda tabela, existe uma coluna e quinze linhas, sendo que, na primeira de cima para baixo, está localizada a chave estrangeira de onde parte uma linha que leva a uma legenda de mesmo nome. No restante das linhas abaixo dessa, existem os dados dos pedidos.

Acesse [Como relacionar TablePedidos e TableClientes](#) para entender o processo detalhadamente.

Após perceber o comportamento da chave primária e da chave estrangeira a partir da TableClientes e da TablePedidos, veja o relacionamento entre as outras tabelas que formam o conjunto que compõe o banco de dados.



## #PraCegoVer

Imagem mostra oito tabelas (TableClientes; TableTransportadora; TableCategorias; TableDetalhesDoPedido; TableProdutos; TablePedidos; TableFuncionarios; e TableFornecedores). Da esquerda para a direita, da tabela Clientes e Funcionários, partem duas linhas de suas respectivas primeiras linhas que se ligam à tabela Pedidos. Desta última, partem duas linhas que se ligam à tabela Transportadoras e Detalhes do Pedido. Desta última, parte uma linha que se liga à tabela Produtos. Desta última, partem duas linhas que se ligam à tabela Fornecedores e Categorias.

Com base nas tabelas apresentadas na imagem acima, veja agora a demonstração do **relacionamento** entre elas, ou seja, a correspondência entre os dados contidos nas colunas de ambas as tabelas.

	<a href="#">TablePedidos e TableFuncionarios</a>
	<a href="#">TablePedidos e TableDetalhesdosPedidos</a>
	<a href="#">TableDetalhesdoPedido e Produtos</a>
	<a href="#">TableProdutos e TableCategorias</a>
	<a href="#">TableProdutos e TableFornecedores</a>
	<a href="#">TablePedidos e TableTransportadora</a>

Viu como é simples entender a maneira pela qual ocorre a articulação? Ou seja, o relacionamento entre as tabelas que comporão os bancos de dados é, principalmente, em relação à dinâmica associada à chave primária e à chave estrangeira de cada uma.

Entretanto, para que isso seja viável, é fundamental que os comandos de manipulação de dados feitos por meio de uma linguagem específica sejam aplicados. Dessa forma, você estudará detalhadamente sobre esse tema a seguir. Fique ligado!

## Comandos de manipulação de dados DML

É imprescindível que haja uma série de comandos e de programações associadas à manipulação do conteúdo dos bancos de dados relacionais. Assim, para que isso seja feito de modo consistente, existe o *Data Manipulation Language* ou Linguagem de Manipulação de Dados.

*Data Manipulation Language* são **comandos** que abarcam as **principais operações realizadas em um banco de dados**, como: incluir, editar, excluir e consultar.

Observe, a seguir, esses comandos e suas sintaxes.



### #PraCegoVer

Na imagem, há uma programadora trabalhando em seu notebook, o qual está em cima da mesa de escritório. Ao lado do aparelho, há um papel com um gráfico impresso e um porta lápis. Atrás dela, há prateleiras com várias pastas de documentos.

## INSERT

O comando **INSERT** é usado para a inserção de dados em uma tabela.

Esse comando é uma instrução T-SQL.

### Sintaxe:

```
INSERT INTO Nome_da_Tabela (Coluna_1, Coluna_2, Coluna_3, Coluna_4,  
VALUES ('VALOR 1', 'VALOR 2', 'VALOR 3', 'VALOR 4');
```

No link [Cadastro de novos clientes – COMANDO INSERT](#), você confere como construir esse comando.

## SELECT

O comando **SELECT** é utilizado para visualizar os dados cadastrados ou ainda recuperar dados.

### Sintaxe:

```
SELECT Coluna FROM NomeDaTabela;
```

Acesse [Tabela de clientes – COMANDO SELECT](#) para entender como construir esse comando

## WHERE

O comando **WHERE**, quando **associado** ao comando **SELECT**, proporciona uma **condição específica de execução** a uma consulta de dados, limitando os resultados e exibindo apenas linhas compatíveis com a condição estabelecida.

### Sintaxe:

```
SELECT FROM * TableClientes
```

```
WHERE País = 'Brasil';
```

Para entender como construir esse comando, convidamos você a ver o link: [Seleção de clientes do Brasil – Comando WHERE](#).

## UPDATE

No caso de **UPDATE**, referimos a atualizações de registros.

### Sintaxe:

```
UPDATE nome_tabela
```

```
SET nome_coluna = novo_valor
```

```
WHERE nome_coluna = valor;
```

No link [Atualização do código de QUEENB para KING – COMANDO UPDATE](#), você confere como construir esse comando.

## DELETE

O comando **DELETE** é utilizado para deletar registros.

### Sintaxe:

```
DELETE FROM nome_tabela
```

```
WHERE coluna = valor;
```

Para entender como construir esse comando, convidamos você a acessar o material: [Inserção e exclusão de registro – COMANDO DELETE](#).

Até este ponto, você conheceu os principais comandos empregados na *Data Manipulation Language*, que são: *INSERT*, *SELECT*, *UPDATE*, *DELETE* e *WHERE*.

Agora, é preciso fazer a inclusão de dados em uma tabela por meio do comando *INSERT*. Entretanto, será possível realizar a importação de dados de um banco existente. Veja essa demonstração aqui: [Importando dados da Tabela Clientes](#).

No entanto, esse último comando, o *WHERE*, contém algumas características específicas, importantes para sua atuação profissional. Por esse motivo, vamos estudá-lo mais detalhadamente no próximo tópico.

## Um pouco mais do *WHERE*

Conheça um pouco mais sobre a sintaxe da cláusula ***WHERE***. A seguir, veja algumas formas de utilizá-la em seu banco de dados.

### ***WHERE BETWEEN***

Ela permite a definição de um **intervalo** de valores a ser selecionado.

#### Sintaxe

*BETWEEN VALOR\_INICIAL AND VALOR\_FINAL.*

Para saber o que deve ser feito para realizar esse comando, veja o link:

[WHERE BETWEEN](#).

### ***WHERE IN***

Cria uma **lista** de dados que será retornada na consulta.

#### Sintaxe

*/N (Valor 1, Valor 2, Valor 3, ...).*

Veja [WHERE IN](#) para conhecer os passos de como realizar esse comando.

## WHERE NOT

Esse comando é utilizado quando **não** se deseja **visualizar** algum dado em uma pesquisa.

### Sintaxe

*NOT IN ('Valor1', 'Valor2', 'Valor3');*

No link [\*\*WHERE NOT\*\*](#), você verá como pode realizar esse comando.

## WHERE LIKE

O **LIKE** **verifica** se uma série de caracteres corresponde a um **padrão especificado**.

### Sintaxe

*LIKE (cadeia de caracteres).*

Para complementar seu aprendizado sobre assunto, sugerimos que veja o conteúdo [\*\*WHERE LIKE\*\*](#).

Como você pôde observar, na sintaxe desse comando é possível utilizar **caracteres-coringa**. No quadro abaixo, apresentamos os tipos, bem como sua descrição e exemplos práticos de como aplicá-los. Confira!

Caractere Coringa	Descrição	Exemplo	Função
%	Qualquer cadeia de zero ou mais caracteres.	<i>WHERE NomeDaEmpresa LIKE '%Gua%'</i>	Localiza todos os nomes de empresa com <b>Gua</b> em qualquer lugar no campo.
_(sublinhado)	Qualquer caractere único.	<i>WHERE NomeDoContato LIKE '_ean'</i>	Localiza todos os nomes de quatro letras que terminam com <b>ean</b> (Jean, Dean, Sean e assim por diante).

[ ]	Qualquer caractere único no intervalo ([a-f]) ou conjunto ([abcdef]) especificado.	<i>WHERE</i> NomeDoContato <i>LIKE</i> '[D-X]ia'	Localiza os nomes de contato que terminem com <b>ia</b> e que comecem com qualquer caractere único entre <b>D</b> e <b>X</b> , por exemplo, Sofia, Lia, Olivia, e assim por diante. Em pesquisas de intervalo, os caracteres incluídos no intervalo podem variar de acordo com as regras de classificação do agrupamento.
[^]	Qualquer caractere único que não esteja no intervalo (^[a-f]) ou conjunto (^[abcdef]) especificado.	<i>WHERE</i> NomeDoContato <i>LIKE</i> 'Mi[^A]%'	Localiza todos os nomes de contato que comecem com <b>Mi</b> e cuja letra seguinte não seja <b>A</b> .

## Saiba mais



Acompanhe um processo de configuração de banco de dados na nuvem, assistindo ao vídeo [SQL Azure - Conectando e administrando um banco de dados na nuvem](#). Vale a pena conferir!

Neste tópico, você não só estudou, como também acompanhou o passo a passo utilizado para a aplicação da sintaxe da cláusula *WHERE*, uma importante operação realizada nos bancos de dados.

Parabéns! Você chegou ao final do Módulo 2!

Aqui, você compreendeu como criar um banco de dados, além de entender a maneira pela qual as informações são armazenadas e manipuladas nele, usando o T-SQL. Vimos o modo adequado de configuração do banco de dados, sobretudo, empregando as tabelas que os comporão. Abordamos, também, os diferentes tipos de relacionamentos entre essas tabelas e os principais comandos de manipulação de dados DML (*INSERT, SELECT, UPDATE, DELETE* e *WHERE*).

Apesar disso, é essencial que todas essas ações sejam articuladas e projetadas. Para entender melhor sobre isso, convidamos você a realizar o terceiro e último módulo do curso, que abordará o planejamento do banco de dados.

Preparado? Vamos lá!



Módulo 3

## Projetando um banco de dados

---

# Projetando um banco de dados

No módulo anterior, você pôde acompanhar como são criadas as tabelas em um banco de dados como elas se relacionam. No entanto, a criação de um banco de dados exige um projeto e uma preparação em fases distintas. E é sobre este assunto que veremos neste módulo. O primeiro passo diz respeito à estrutura e ao layout de banco de dados.

## Estrutura e layout de projeto de banco de dados

O passo inicial voltado ao projeto de um banco de dados é sua estrutura e layout.

Para ajudá-lo a compreender os pontos que vamos abordar ao longo deste tópico, preparamos um podcast a respeito desse tema. Acompanhe!



### Podcast

Confira o [podcast](#) sobre a criação de um projeto de banco de dados.

Perdeu algum detalhe? Confira o que foi abordado no *podcast*.

Olá! Vamos conhecer as fases de criação de um projeto de banco de dados?

Antes disso, vale lembrar que um banco de dados construído de maneira estruturada fornece, aos usuários e administradores; facilidades de acesso e manutenção, economia de espaço em disco, eliminação de dados redundantes, fornecimento de proteção e exatidão, integridade de dados e aumento da performance de processos corporativos.

Com isso, segundo William Pereira Alves, analista de sistemas com mais de 20 anos de experiência, a criação de um banco de dados deve ser organizada e bem planejada.

Então, para sua criação, é importante seguir algumas fases que demonstram uma ordem, auxiliando no planejamento de banco de dados.

A primeira delas é a análise de requisitos e delimitação dos objetivos do banco de dados. A segunda é a organização dos dados em tabelas. A terceira, a especificação de chaves primárias e relacionamentos. E a quarta é a padronização de tabelas.

Agora que você já sabe as fases dessa criação, no próximo tópico confira cada uma delas detalhadamente.

No próximo tópico, detalharemos cada uma das **fases** de criação desse projeto de banco de dados, que citamos no *podcast*, indicadas por Willian Pereira Alves, Analista de Sistemas brasileiro, começando pela análise de requisitos. Vamos lá!

## Análise de requisitos: definição do objetivo do banco de dados

Um projeto de banco de dados com estrutura e layout adequados facilita a sua execução pela equipe de desenvolvedores, economizando recursos e tempo.

Assim, para elaborar um projeto dessa natureza, Willian Pereira Alves evidencia que são necessárias quatro fases. A primeira delas é a análise de requisitos e delimitação dos objetivos do banco de dados.

Preparamos um vídeo que detalha essa primeira fase que auxilia no planejamento, a análise de requisitos. Acompanhe!



### Vídeo

Confira o [vídeo](#) sobre a análise de requisitos: definição do objetivo do banco de dados.

Perdeu algum detalhe? Confira o que foi abordado no vídeo.

Olá! Aqui vamos falar sobre a análise de requisitos: definição do objetivo do banco de dados.

Compreender as finalidades do banco de dados é muito importante, visto que constitui uma base conceitual que direciona o processo de criação do banco de dados. Neste caso, o desenvolvedor ou o administrador de banco de dados deve considerar as perspectivas envolvidas referentes à criação de um banco de dados para uma entidade qualquer.

Diante deste aspecto, ainda é necessário ficar atento às formas pelas quais os usuários e administradores vão acessar os dados disponíveis.

Mas, antes da criação do banco de dados, é possível fazer a coleta de informações que podem ser úteis nesse processo, da seguinte maneira:

- Realizar entrevistas com potenciais usuários para entender possíveis necessidades.
- Pesquisar bancos semelhantes para entender seu funcionamento e prover melhores soluções.
- Analisar, se for o caso, o banco de dados antigo e verificar o que pode ser aproveitado.

Já com os dados disponíveis, é interessante que eles sejam reunidos, analisados e organizados. Assim, uma lista conceitual pode ser redigida para orientar sobre quais dados se desejam armazenar, bem como as características, as pessoas, os locais ou os eventos descritos por esses dados. Por exemplo:

Os dados de clientes são:

- Nome.
- Endereço.
- Cidade, Estado, CEP.
- E-mail.

Já os de produtos são:

- Nome.
- Preço.
- Quantidade em estoque.
- Quantidade em pedidos.

E os dados de Pedidos são:

- ID do pedido.
- Representante de vendas.
- Data.
- Produto(s).
- Quantidade.
- Preço.
- Total.

As informações levantadas construirão um inventário de dados, que descreverá as tabelas e os campos contidos no interior do banco de dados.

É interessante que as informações sejam arranjadas em partes, tornando mais fácil e útil suas interpretações.

No caso de um banco de dados que pretende abrigar informações de usuários de diversas partes do mundo, por exemplo, deve-se considerar a separação por países para conseguir realizar um filtro de maneira mais inteligente no futuro.

É preciso, ainda, evitar o uso de um mesmo ponto de dados em duas ou mais tabelas, visto que isso geraria uma complexidade desnecessária.

Após essa triagem sobre os tipos de dados que poderão ser armazenados, inicia-se a etapa do planejamento de um banco de dados.

Após ver o vídeo, você compreendeu a importância de se definir o objetivo do banco de dados a ser cumprido e de se coletar o máximo de informações ligadas ao objetivo traçado. Para isso, apresentamos algumas formas de coletar essas informações, principalmente, por meio de questionários.

Concluindo essa primeira etapa, é necessário que esse conjunto de informações coletadas e definidas seja organizado, sobretudo, a partir de um esquema gráfico, ou seja, em uma estrutura.

No próximo, tópico veremos como realizar essa estrutura gráfica, aprendendo sobre a segunda etapa da elaboração de um projeto de construção de banco de dados. Confira!

## Estrutura de banco de dados: organizando os dados em tabelas

Para a evolução do projeto, será necessária uma **representação visual** do banco de dados. Para isso, é preciso entender como os dados relacionais serão estruturados.

Os dados relacionais serão agrupados em tabelas formadas de linhas e colunas, como uma planilha. A conversão de sua lista de dados em tabelas deverá considerar a elaboração de **uma tabela para cada tipo de entidade**. Podemos citar como exemplo: produtos, vendas, clientes ou pedidos.

Cada **linha** constituinte da tabela será um **registro**, e os registros incluem dados sobre algo ou alguém, como, por exemplo, um cliente qualquer. As **colunas**, que são campos ou atributos, conterão um **único tipo de informação** e aparecerão em cada registro.

Por exemplo, observe os endereços ou telefones de clientes que estão listados na tabela a seguir.

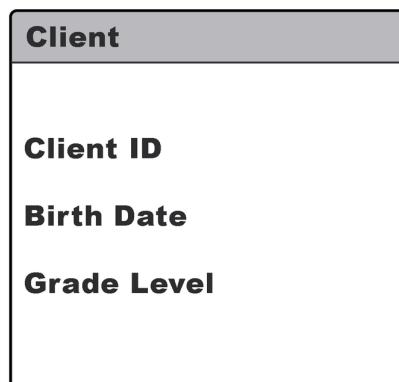
Nome	Sobrenome	Idade	Telefone
João	Silva	42	(11) 99856-4042
Andressa	Costa	40	(41) 9654-1322
Roberto	Paim	53	(31) 9658-1243

Os **dados** entre um registro e o outro devem **permanecer consistentes**. Dessa forma, deve-se atribuir o tipo de dado apropriado para cada coluna. Os tipos de dados comuns incluem os seguintes atributos:

Atributos	Definição
<b>CHAR</b>	Define tamanho específico de texto.
<b>VARCHAR</b>	Define texto de tamanhos diversos.
<b>TEXT</b>	Define grandes quantidades de texto.
<b>INT</b>	Define os números inteiros, sejam eles positivos ou negativos.
<b>FLOAT, DOUBLE</b>	Define que também é possível armazenar no banco de dados os números de pontos flutuantes.
<b>BLOB</b>	Define o registro de dados binários.

É importante que você saiba que os sistemas de gestão de banco de dados proporcionam registros de dados do tipo '**'Autonumeração'**', que gera uma numeração exclusiva a cada linha de forma automática.

Uma visão mais geral do banco de dados pode ser obtida com o uso de **diagramas de entidade-relacionamento**. Eles não tratam ainda de tabelas reais, mas de '**caixas**' no diagrama. Dessa forma, cada caixa deverá ser titulada para indicar o descriptivo de dados e os atributos que serão listados, conforme você pode observar na imagem abaixo.

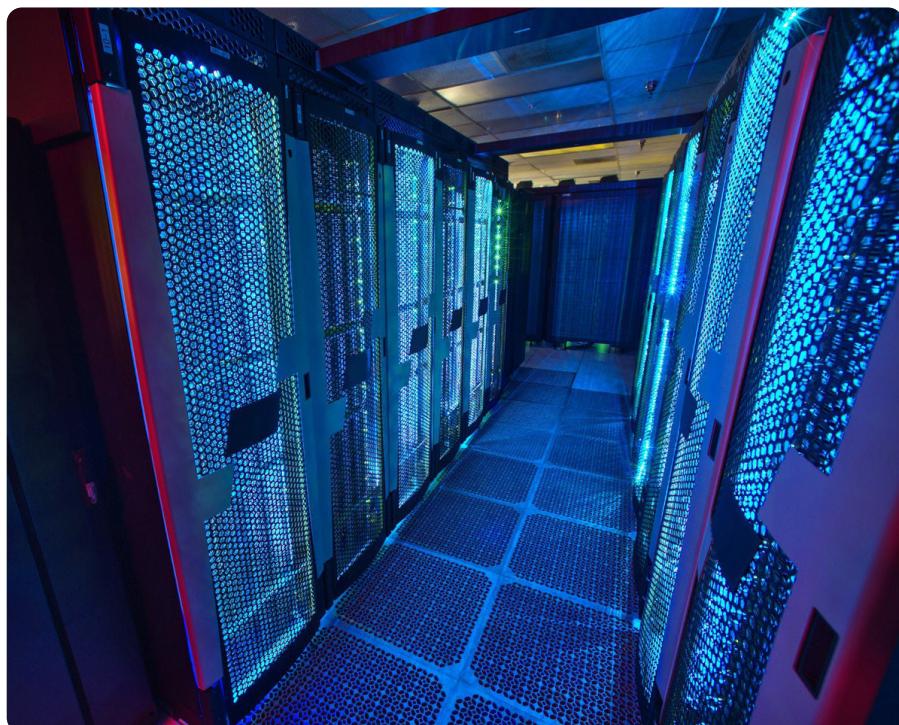


Chegará o momento da definição de qual atributo ou quais atributos serão utilizados como chave primária para cada tabela, caso exista essa necessidade. Uma chave primária consiste no identificador para uma certa entidade, o que significa que uma escolha deverá ser realizada.

Os atributos escolhidos como chaves primárias devem ser únicos, não podem ser modificados e precisam estar presentes. Igualmente, não podem ser nulos ou vazios. Por essa razão, normalmente, os nomes de usuários são boas chaves primárias. Ao contrário das informações mutáveis, como os números de telefones ou endereços.

**Lembre-se!** É possível utilizar vários campos em conjunto como chave primária, o que será chamado de **chave composta**.

Ao criar um banco de dados, as estruturas de dados lógicos e a estrutura de dados físicos devem estar estimadas na **capacidade do tamanho** do banco de dados, principalmente quando estamos nos referindo à questão de conseguir suportar certa quantidade de registros. Essa previsão deve ocorrer para que sejam determinados o **nível de desempenho** e o **espaço de armazenamento necessário** na instauração do banco de dados.



#### #Pracegover

Na imagem, há um corredor escuro em que vários servidores estão enfileirados, emitindo uma luz clara.

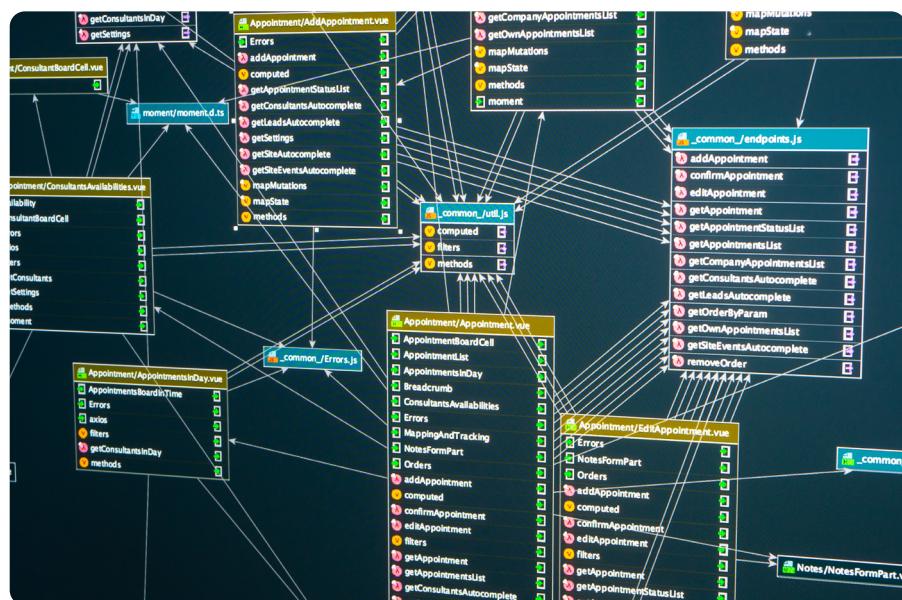
Neste tópico, você conheceu a segunda etapa do planejamento de um banco de dados, a estruturação. Nela, é possível não só esquematizar a sua estrutura por meio de uma representação gráfica, como também desenvolver as tabelas em que os dados relacionais serão agrupados em linhas e colunas. Por fim, compreendeu a relevância de determinar o nível de desempenho e o espaço de armazenamento necessário (estrutura de dados lógicos e físicos) na instauração do banco de dados.

Ainda nessa estruturação, é preciso delimitar os tipos de relações entre as diferentes tabelas. Esta é a terceira etapa do projeto criação de banco de dados, chamada de especificação de chaves primárias e relacionamentos. Vamos saber mais sobre ela?

## Criando relações

Após a delimitação das tabelas do banco de dados, será o momento de analisar as relações estabelecidas entre elas.

Nesse momento, o fator **cardinalidade** aparece e se refere à quantidade de elementos que interagem entre duas ou mais tabelas relacionadas.



#Pracegover

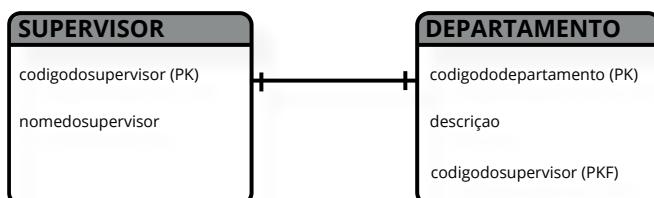
Na imagem, existem doze tabelas, que são ligadas por setas que se entrecruzam sob um fundo escuro.

A identificação da cardinalidade auxilia o desenvolvedor ou o administrador de banco de dados na **divisão de dados em tabelas de forma eficiente**. Cada entidade possui uma relação com as outras, que normalmente acontece de **cinco** maneiras. Vamos conhecer cada uma das relações existentes.

## Relação uma a uma

Quando existe apenas uma instância de entidade A para cada instância de entidade B, temos uma relação de uma para uma, descrita como **1:1**. Dessa forma, essa relação é representada em um diagrama de Entidade Relacionamento com um traço partindo de uma extremidade para outra.

No que diz respeito a isso, observe as informações na imagem a seguir:



### #PraCegoVer

Na imagem, existem dois quadros cujos títulos são supervisor e departamento, sendo que, dentro do primeiro quadro da esquerda para a direita, está escrito codigodosupervisor (PK) e nomedosupervisor. No segundo quadro da esquerda para a direita, está escrito codigododepartamento (PK), descrição e nomedosupervisor(PKF).

- Uma relação 1:1 é conveniente para indicar que as relações são **unitárias**, ou seja, uma tabela está ligada a outra em **relação direta**.
- Neste exemplo, podemos interpretar que um supervisor será o responsável por um departamento e essa relação está descrita no relacionamento.
- Nesse caso, uma coluna deverá ser repetida em ambas as tabelas. Normalmente, isso será realizado com a chave primária.

## Relação uma para muitas

Outra relação é a chamada relação de uma para muitas. Esse tipo de relação acontece frequentemente em transações de negócios, quando um registro contido em uma tabela se associa a diversas entradas existentes em outra tabela.

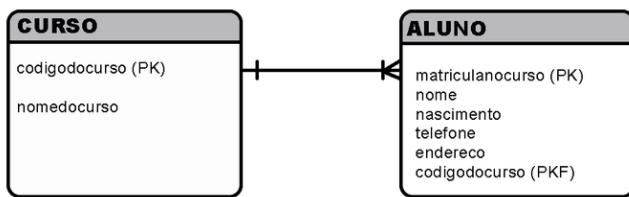
Um exemplo que podemos citar é o de casos em que um cliente solicita diversos pedidos em um e-commerce, que ocorrerá em formato de árvore ou de hierarquia.



### #PraCegoVer

Na imagem, as mãos de um programador são enfocadas e estão digitando em um notebook, que está em cima de uma mesa de escritório. Por meio de uma projeção de um holograma, há um banco de dados em formato de árvore.

Outro caso são os cursos on-line, observe o exemplo e as informações abaixo:



### #PraCegoVer

Na imagem, existem dois quadros cujos títulos são curso e aluno, sendo que está escrito neles, respectivamente, codigodocurso (PK) e nomedocurso, já no outro quadro está escrito matriculancurso (PK), nome, nascimento, telefone, endereco e codigodocurso (PKF).

- Aqui, temos o campo de 'codigodocurso' como chave primária na tabela entidade 'curso', na 'tabela' temos como chave primária a 'matriculancurso', e 'codigodocurso' que atuará como chave estrangeira.
- A relação uma para muitas (**1:M**) é indicada entre os quadros Curso para Aluno, indicada por esse símbolo.

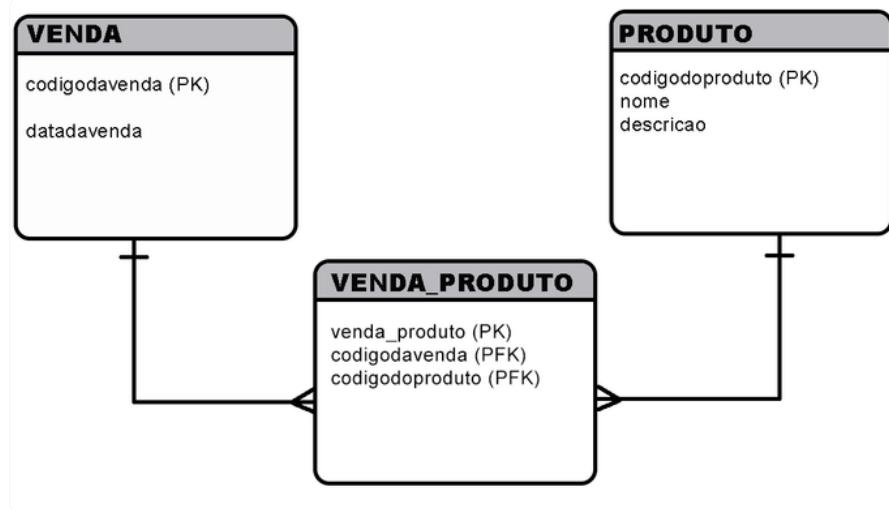
Note que a implementação de uma 1:M procede à medida que a configuração do banco de dados ocorre. Nessa situação, a chave primária de uma tabela se transforma em chave estrangeira na outra.

## Relações muitas para muitas

Outra relação que se destaca é a de muitas para muitas. Nela há diversas entidades de uma tabela associadas a diversas entidades de outra tabela, criando, assim, uma relação muitas para muitas (**M:M**).

No exemplo abaixo, como uma venda de vários itens, as entidades VENDA e PRODUTO são rapidamente identificadas, pois uma venda integra diversos itens e um item será encontrado em diversas vendas.

Dessa forma, o caso de relacionamento M:M comporta duas ou mais entidades. Mas no exemplo apresentado abaixo, utilizamos a entidade VENDA\_PRODUTO que associaria as outras entidades. Observe com atenção!



### #PraCegoVer

Na imagem, existem três quadros cujos títulos são venda, venda\_produto e produto, sendo que está escrito neles, respectivamente, codigodavenda (PK) e datadavenda, já no segundo quadro está escrito venda\_produto (PK), codigodavenda (PKF), codigodoproduto (PKF), já no último quadro está escrito codigodoproduto (PK), nome e descricao.

Note que os registros contidos na tabela de ligação (VENDA\_PRODUTO) correspondem às duas entidades nas tabelas que se avizinham.

## Relações recursivas

Já nas relações recursivas é possível que as relações em uma tabela **apontem para si mesma**. Uma tabela com os dados de colaboradores de uma empresa, por exemplo, que possua o atributo 'diretor', apontará para um indivíduo nessa tabela. E assim estabelece uma relação recursiva, igualmente denominada de **autorrelacionamento**, que cconstitui casos especiais e por isso é pouco recorrente.

## Relações redundantes

No caso de relações redundantes, elas **são expressas mais de uma vez**, em que se removerá uma dessas relações sem que nenhuma informação relevante se perca.

Digamos que uma entidade “alunos” possua uma relação direta com outra entidade denominada “professores” e, ainda, possua uma relação com “professores” de maneira indireta por meio de “aulas”. Assim, se a relação “alunos” e “professores” for excluída não, prejudicará os registros. Isso porque eles continuariam a relacionar os alunos e professores por meio da atribuição de aulas.

Para que você entenda melhor **como os diagramas são aplicados**, convidamos você a ouvir o seguinte *podcast*. Confira!



### Podcast

Confira o [podcast](#) sobre como os diagramas são aplicados.

Perdeu algum detalhe? Confira o que foi abordado no *podcast*.

Olá! Você sabe como os diagramas são aplicados?

Os diagramas entidades-relacionamento são recursos utilizados em projetos para bancos de dados relacionais nos segmentos da engenharia de software, sistemas de informações empresariais, educação e pesquisa.

Além dessas áreas, a sua presença vem crescendo também nos segmentos da criação do próprio banco de dados, pois eles são importantes para uma visualização prévia do banco de dados que será construído.

De modo geral, os diagramas são úteis em processos de design de banco de dados, solução de problemas de lógica, implementação, análise de bancos de dados relacionais em processos de negócios e reengenharia de processos.

Com o crescimento do ensino a distância, igualmente, os diagramas servem para o desenvolvimento de mapas de fluxo de dados para fins educacionais, pois promovem mais profissionalismo e uma menor taxa de retrabalho na criação de bancos de dados.

Viu só como os diagramas podem auxiliar no seu trabalho? Na sequência, continue seus estudos com a padronização de tabelas.

## Saiba mais



Caso queira se aprofundar nesse assunto, indicamos o vídeo [Diagrama Entidade Relacionamento - modelo de banco de dados](#). Vale apena conferir!

Desta forma, você estudou sobre a terceira etapa do projeto de criação de bancos de dados, a especificação de chaves primárias e relacionamentos. Além disso, também viu os diferentes tipos de relacionamentos (relação uma a uma, uma para muitas, muitas para muitas, recursivas e redundantes).

Agora, veremos a última etapa do projeto de banco de dados: a padronização de tabelas. Vamos lá!

## Banco de dados: normalização

A última etapa do projeto de criação de banco de dados é a padronização de tabelas. Isso ocorre após a confecção do design conceitual do banco de dados, ou seja, o esquema de tabelas.

Depois disso, serão aplicadas as **regras de normalização**. Sua função é de atuar na **certificação da estrutura correta das tabelas**, que são regras similares aos padrões de qualidade aplicados na indústria.

Vale destacar que não são todas as estruturas de bancos de dados que utilizam processos de normalização. Trata-se de uma prática imposta com o surgimento de bancos de dados especializados em processamento de transações on-line (OLTP, abreviação em inglês).



### #PraCegoVer

Na imagem, estão enquadradas uma mão que segura um tablet em que está escrito OLAP junto com um gráfico e a outra mão interage com o aparelho. Elas estão sobre uma mesa de escritório que contém um notebook aberto, um bloco de anotações com um lápis e óculos, além de uma xícara de café.

Inclusive, nos bancos de dados de processamento analítico on-line (OLAP) existem os processos de **desnormalização**, uma vez que a ênfase se encontra nos processos de velocidade de cálculo. Nesses processos, estão incluídos os aplicativos de suporte a tomadas de decisão, os quais requerem análises de dados e necessitam

de disponibilidade rápida, mas não de alterações.

Em cada formulário e etapa de normalização, são incluídas as regras associadas com outros formulários em níveis inferiores.

Nos tópicos a seguir, você estudará as três regras que garantem a estruturação adequada das tabelas, as quais são representadas pelo primeiro, segundo e terceiro formulário. Vamos conferir!

## Primeiro Formulário ou Forma Normal (1NF)

A confecção do primeiro formulário, normalmente, procura **normatizar especificações em cada uma das células da tabela**, que somente comportam um valor, sendo que os atributos devem ser atômicos e isentos de grupos repetidos e de **valor único**.

Veja um exemplo de tabela com valores **incorrectos**, ou seja, que comportam mais de um valor em cada uma das células.

<b>idProduto</b>	<b>Tamanho</b>	<b>Preço</b>
1	Pequeno, médio	US\$ 18
2	Médio, grande	US\$ 16
3	Grande, extragrande	US\$ 13

Observe que na coluna ‘tamanho’ **mais de um valor** se encontra inserido. O correto, então, seria uma **divisão de dados em diversas tabelas e registros**, até que todos os dados estejam alocados unitariamente em cada célula. Ou seja, contendo um único valor, nessa situação são chamados de **atômicos**.

No caso da tabela sobre a Primeira Forma Normal (1NF), os dados compostos podem ser redistribuídos em tabelas adicionais, conforme o exemplo apresentado abaixo.

<b>idUsuario</b>	<b>nomeUsuario</b>	<b>enderecoUsuario</b>	<b> contatoUsuario</b>
1	Antonio	Rua da Lagoa 75	(21) 992568756
2	Camila	Rua Braga 23	(21) 992865987/ (21)33532145

Mais uma vez, os dados inseridos aparecem de forma **incorrecta** e necessitam de **ajustes**. Assim, a sua redistribuição ficaria conforme as próximas duas tabelas. Observe com atenção:

<b>idUsuario</b>	<b>nomeUsuario</b>	<b>enderecoUsuario</b>
2	Camila	Rua Braga 23

<b>idCcontato</b>	<b>Usuario_idUsuario</b>	<b>descreverContato</b>	<b>numeroContato</b>
1	2	celular	(21) 992865987
2	2	residencial	(21) 33532145

Sendo assim, as relações possuem uma chave primária e asseguram que o atributo seja atômico. Podemos citar o caso de um atributo de endereço, que deverá ser **subdividido em outros componentes**, tais como: a rua, número, complemento, bairro, cidade, estado e CEP. Desta forma, verificamos que **os atributos com multivalores devem ser separados**.

## Segundo Formulário ou Forma Normal (2NF)

Para conhecer detalhadamente sobre o segundo formulário (2NF), convidamos você a assistir ao vídeo que preparamos! Confira!



## Vídeo

Confira o [vídeo](#) sobre o aprendizado com o segundo formulário normal (2NF).

Perdeu algum detalhe? Confira o que foi abordado no vídeo.

Olá! Agora que você já sabe como confeccionar o primeiro formulário, vamos seguir o aprendizado com o segundo formulário normal (2NF)?

Esse formulário aponta para os atributos que devem fundamentalmente ser dependentes de uma chave primária.

Toda relação construída no 1NF (Primeira Forma Normal) deve eliminar as dependências parciais. Por exemplo, um atributo associado à idade estará ligado ao atributo de data de nascimento, que, por sua vez, estará ligado a algum outro atributo e assim sucessivamente.

Dessa forma, ocorre uma dependência funcional parcial, visto que uma tabela que contenha esses atributos, certamente, não cumpriria as exigências do segundo formulário normal.

A intenção da 2NF é de organizar os relacionamentos, principalmente, evitando as duplicações e ainda usando as tabelas com dados relevantes para a conservação da integridade das informações.

Neste exemplo, encontramos os dados dos alunos de um curso e a última coluna está o nome do curso. Entende-se que o curso existe independentemente de uma alteração ou exclusão de registro de qualquer um dos alunos.

<b>idAluno</b>	<b>nomeAluno</b>	<b>idadeAluno</b>	<b>codigoCurso</b>	<b>nomeCurso</b>
1	Carlos	18	3	Marketing
2	Mara	23	3	Marketing
3	Rubens	21	3	Marketing

Embora seja uma informação de identificação, o código do curso seria suficiente para a identificação do curso em que o aluno está matriculado.

A solução disso compete em separar a informação que não seja diretamente dependente da chave primária, criando, desta forma, uma nova tabela relacionada.

Assim, contamos com uma distribuição inteligente de dados e com sua integridade garantida.

Você conferiu, no vídeo anterior, como o segundo formulário normal (2NF) evidencia os atributos dependentes da chave primária, bem como a relação disso com o primeiro formulário. Tais fatores são essenciais para a estruturação adequada das tabelas de um banco de dados.

A seguir, veremos mais um formulário que você pode utilizar!

## Terceiro Formulário Normal ou Forma Normal (3NF)

Para que a normalização seja concluída, o Terceiro Formulário Normal (3NF) **adicionará**, aos passos anteriores, a premissa de que **toda coluna** com atributos não chave sejam independentes de outros atributos em outras colunas, mas que **continuem dependentes de uma chave primária**. Isso significa que uma eventual alteração em um atributo não chave não afeta outros atributos. Contudo, caso isso ocorra, a tabela não atenderá às condições do terceiro formulário normal.

Observe a seguinte tabela, que indica os relacionamentos interdependentes que não atendem ao 3NF.

Pedido	Preço	Dedução
1	US\$ 45,99	US\$ 4,12
2	US\$ 23,73	US\$ 2,87
3	US\$ 14,15	US\$ 0,11

Claramente, a tabela destaca a relação de **dependência** entre o preço e a dedução. Em que: se o preço sofrer alteração, a dedução seguirá o mesmo caminho. Das relações advindas da 2NF, é necessário realizar uma triagem e deletar quaisquer dependências funcionais de caráter transitivo.

Veja outro exemplo disposto no quadro abaixo e na tabela logo em seguida.

<b>idNota</b>	<b>primeira- Nota</b>	<b>segunda- Nota</b>	<b>terceira- Nota</b>	<b>quarta- Nota</b>	<b>media Nota</b>
1	6	6	6	6	6
2	8	8	8	8	8

<b>idNota</b>	<b>primeiraNota</b>	<b>segundaNota</b>	<b>terceiraNota</b>	<b>quartaNota</b>
1	6	6	6	6
2	8	8	8	8

Note que nos dois registros de notas dos alunos, aparecem a composição das notas e as médias registradas. Nesse caso, o campo ‘mediaNota’ será **dependente** dos resultados anteriores, para que seja calculado. No entanto, os outros registros não dependem desse campo.

Dessa forma, se o campo ‘mediaNota’ fosse excluído, **não alteraria** os registros anteriores. Ou seja, os atributos não chave devem ser mutuamente independentes.

Ainda considerando a última etapa do projeto (isto é, a padronização de tabelas), é imprescindível que, após a validação dos dados a partir das regras de 1NF, 2NF e 3NF, a integridade dos dados seja avaliada. Vamos entender mais a seguir!

## Integridade de dados

Considerando esse passo importante no processo de banco de dados, preparamos um *podcast* que fala, com mais detalhes, sobre a integridade dos dados. Acompanhe!



## Podcast

Confira o [podcast](#) sobre a validação de dados.

Perdeu algum detalhe? Confira o que foi abordado no *podcast*.

Olá! Você sabia que após a validação dos dados do projeto de banco de dados, ainda é necessário avaliar a integridade deles?

A validação de dados, a partir de regras apropriadas de um banco de dados, é configurada e os sistemas da gestão de banco de dados aplica essa validação de forma automática.

Estas regras, que são de integridade para entidades, apontam que uma chave primária não poderá ser NULA. Com isso, caso seja composta por diversas colunas, nenhuma delas poderá estar NULA, pois, do contrário, os registros não poderão ser identificados em sua individualidade.

Assim, se uma chave estrangeira constar em uma tabela, a regra de integridade referencial requer que ela esteja correspondida por uma chave primária em outra tabela que faz referência. Caso uma chave primária seja modificada ou deletada, as alterações necessitarão de implementação toda vez que essa chave tiver uma referência no banco de dados.

Viu só como as regras de integridade de dados garantem a identificação dos dados? Continue aprofundando seus estudos para saber mais sobre os conceitos de padronização de tabelas.

Ao ouvir o *podcast*, você compreendeu que as regras de integridade garantem que os registros presentes nas colunas das tabelas, referentes a um determinado banco, não sejam nulos, promovendo, assim, a identificação dos seus dados.

Contudo, além da integridade de dados, outros dois conceitos também são relevantes à etapa de padronização de tabelas: são os índices e as exibições. Vamos saber mais sobre eles.

## Inserir Índices e exibições

Um **índice** é uma **cópia classificada** de uma coluna, ou de mais de uma, cujos valores podem estar em ordem crescente ou descendente. A adição de um índice permite que usuários possam **localizar registros com eficiência e rapidez**.



### #Pracegover

Na imagem de perfil, dois administradores estão em frente a uma tela de computador em cima de uma mesa de escritório. O homem que está mais enquadrado, aponta para tela, enquanto digita no seu notebook. O segundo que está ao seu lado apoia a cabeça sobre as duas mãos que estão apoiadas sobre a mesa e olha fixamente para o apontamento do colega. Ao fundo, estão duas programadoras conversando.

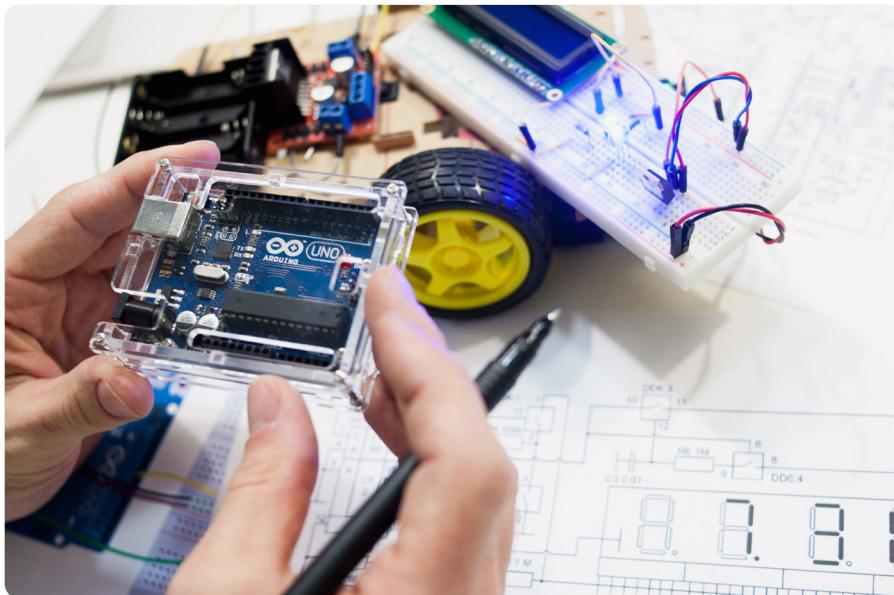
Já os índices podem acelerar a recuperação de dados, e isso é algo bom. Contudo, eles podem diminuir as taxas de inserção, atualização e exclusão de dados, pois um índice necessita ser reconstruído a cada vez que um registro passar por alterações.

Em relação a uma **exibição**, destacamos as **consultas que são salvas sobre os dados**, que reúnem variados dados relevantes de diversas tabelas, ou ainda, exibem uma tabela parcialmente.

Dessa forma, esta foi a última etapa do projeto de criação de um banco de dados. Mas existem outras medidas e processos que podem ser aplicados no recém-criado banco de dados, como é o caso das propriedades estendidas. Acompanhe!

## Propriedades estendidas

Após a conclusão do layout básico do banco de dados, pode-se realizar um processo de refinamento com algumas propriedades estendidas. Elas podem ser textos instrucionais, máscaras de entrada, normas de formatação aplicadas a um certo esquema, exibição ou coluna. Essas regras ficam armazenadas no banco de dados, enquanto que sua consulta e apresentação de dados terão consistência entre diversos programas que acessarem os dados armazenados.



### #PraCegoVer

A imagem mostra uma mão segurando um chip de um protótipo de robô em cima de uma planta de um projeto mecânico.

**A integração entre bancos de dados tradicionais e novas tecnologias de informação** proporciona infinitas possibilidades de mercado. Os bancos de dados SQL podem se conectar com operações em **Arduino**, uma **plataforma de prototipagem** que possibilita, entre outras funcionalidades, o acesso a processos de automação serial, capazes de automatizar digitalmente casas, indústrias e ambientes interativos. Inclusive, ele pode controlar as luzes, fechaduras, sons e afins. De modo geral, seria uma plataforma presente em projetos de Internet das Coisas.



### Saiba mais

Caso queria saber mais sobre esse assunto, indicamos o vídeo [Aprenda o que é SQL e como integrar com Arduino](#). Vale a pena conferir!

Neste último tópico, você estudou acerca da estrutura e layout de projetos de banco de dados. Aprendeu que elaborar um projeto dessa natureza proporciona uma otimização de recursos, além de potencializar a elaboração do banco de dados por parte do time de desenvolvedores.

Você também conheceu cada uma das quatro fases de elaboração e, por fim, estudou sobre as propriedades estendidas, uma série de processos que almejam refinar o banco de dados recém-criado.

# Fechamento

Parabéns! Você finalizou o curso **Implementando Banco de Dados**.

Neste curso, foi possível se aprofundar em diversos comandos e conhecer seus principais conceitos. Abordamos, também, como alguns de seus recursos são aplicados na implementação de banco de dados.

Inicialmente, abordamos o procedimento de implementação do banco de dados usando o *SQL Server* e o *Management Studio*. Em seguida, estudamos os processos da criação, do armazenamento e da manipulação das informações contidas no banco de dados.

Já no terceiro módulo, você se aprofundou no projeto associado ao banco de dados, principalmente, em relação à sua estrutura, ao seu layout e validação das tabelas que o compõem.

O importante é sempre praticar e se manter atualizado a respeito dos assuntos aqui apresentados, uma vez que os avanços tecnológicos e as alterações na área são constantes em nosso dia a dia!

Sempre que precisar, você pode retornar e rever o curso.

# Referências

ALVES, Willian Pereira. **Banco de Dados**. São Paulo: Érica, 2014. v. 1.

DEVMEDIA. **Guia Completo de SQL. Devmedia**: São Paulo, 2018. Disponível em: <https://www.devmedia.com.br/guia/guia-completo-de-sql/38314>. Acesso em: 26 jan. 2021.

FORMAS de Normalização de Banco de Dados. Direção de William Carvalho. Produção de Extraclasse. Curitiba: Extraclasse, 2016. Videoaula (10:50m). Disponível em: <https://www.youtube.com/watch?v=XPj82D6G3II>. Acesso em: 25 jan. 2021.

FREEPIK. **Recursos gráficos para todos**. Freepik, 2021. Disponível em: <https://br.freepik.com/>. Acesso em: 11 de nov. de 2021.

LUCIDCHART. **Criação e estruturação de banco de dados**. São Paulo, 2019. Disponível em: <https://www.lucidchart.com/pages/pt/tutorial-de-criacao-e-estruturacao-de-banco-de-dados>. Acesso em: 27 jan. 2021.

MICROSOFT. **Documentação técnica do SQL Server 2019**. São Paulo. Disponível em: <https://docs.microsoft.com/pt-br/sql/sql-server/?redirectedfrom=MSDN&view=sql-server-ver15>. Acesso em: 22 jan. 2021.

MICROSOFT. **Guia de introdução (SQL Server 2019)**. São Paulo. Disponível em: <http://technet.microsoft.com/pt-br/library/hh231622.aspx>. Acesso em: 23 jan. 2021.

ORACLE. **Modelos de Implementação de Banco de Dados em Nuvem. Oracle**. São Paulo, 2018. Disponível em: <https://www.oracle.com/br/database/what-is-a-cloud-database/>. Acesso em: 27 jan. 2021.

SHUTTERSTOCK. **Transforme ideias em conquistas**. ShutterStock, 2021. Disponível em: <https://www.shutterstock.com/pt/>. Acesso em: 11 de nov. de 2021.

SILVA, Eduardo. **Aula 10. Normalização de Dados**. 2010. Disponível em: <http://www.conekti.com/site/index.php/apostilas/>. Acesso em: 23 jan. 2021.

