

Banco de Dados I

Material de Aula

Modelagem e implementação de bancos de dados

**Com dedicação e estudo, você pode
aprender a construir bancos de dados
relacionais eficazes.**

Prof. Enildo

E-mail: professor.enildo@hotmail.com

Sumário

Plano de Curso	3
Bases Tecnológicas	3
Conceitos de bancos de dados	5
Modelo relacional.....	6
Sistemas gerenciadores de bancos de dados (SGBDR)	6
SGBDR vs SGBD	7
Arquitetura cliente/servidor.....	8
Linguagem estruturada de consulta (SQL).	8
Bancos de dados relacionais	9
Modelo conceitual	10
Símbolos de entidade de diagramas ER.....	10
Símbolos de atributos de diagramas ER	11
Entidades, atributos e relacionamentos.....	11
Entidade	12
Atributos.....	12
Entidades.....	12
Relacionamentos:	12
Grau de relacionamento (binário/ ternário).....	13
Grau de cardinalidade.....	14
Auto relacionamento	14
Especialização e generalização.....	16
Dicionário de dados em banco de dados.....	18
Integridade referencial.....	20
Modelagem E-R e Normalização	21
Modelagem E-R	21
Normalização	21
Formas normais (1FN, 2FN e 3FN);.....	21
Ferramenta CASE para criação de Diagramas E-R;	22
Conceito de tabelas.....	26
Construção de projeto lógico de banco de dados	26
Tipos de dados e NULL;	27
Modelo lógico	28
Desnormalização	28
Linguagem SQL Padrão ANSI.....	29
Sublinguagens SQL.....	29
Linguagem de definição de dados (DDL) com SGBDR.....	30
Criação de banco de dados	31
Símbolos em sintaxe.....	31

Criação de objetos (tabelas, colunas, chaves e índices)	31
Criando tabelas.....	31
Adicionando colunas	31
Criando chaves.....	32
Criando índices.....	32
Alteração e exclusão de objetos.....	32
Introdução ao SGBD SQL (Server)	33
Links para ajudar nos estudos.....	33
Referências para estudos de construção de banco de dados relacional	34

Plano de Curso

Banco de dados I

Função:

- Modelagem e implementação de bancos de dados

Classificação:

- Planejamento

Atribuições e Responsabilidades

- Modelar bancos de dados relacionais.
- Implementar modelos de bancos de dados relacionais.

Valores e Atitudes

- Estimular a organização.
- Estimular o interesse na resolução de situações-problema.
- Promover ações que considerem o respeito às normas estabelecidas.

Competências

1. Desenvolver modelo de banco de dados relacional.
2. Implementar modelo com a utilização de sistema gerenciador de bancos de dados relacionais.

Habilidades

- 1.1 Pesquisar as necessidades de informações do sistema.
- 1.2 Normalizar tabelas de bancos de dados.
- 1.3 Correlacionar tabelas.
- 2.1 Criar objetos no banco de dados com a utilização de linguagem de definição de dados.

Bases Tecnológicas

Conceitos de bancos de dados

- Modelo relacional;
- Sistemas gerenciadores de bancos de dados (SGBDR);
- Arquitetura cliente/servidor;
- Linguagem estruturada de consulta (SQL).

Bancos de dados relacionais

- Entidades, atributos e relacionamentos;
- Integridade referencial.

Modelagem E-R e Normalização

- • Ferramenta CASE para criação de Diagramas E-R;
- • Tipos de dados e NULL;
- • Formas normais (1FN, 2FN e 3FN);
- • Desnormalização;
- • Especialização e generalização.

Linguagem de definição de dados (DDL) com SGBDR

- • Criação de banco de dados;
- • Criação de objetos (tabelas, colunas, chaves e índices);
- • Alteração e exclusão de objetos.

LINGUAGENS E FERRAMENTAS DE APOIO

brModelo

Microsoft SQL Server

AVALIAÇÕES

Participação nas discussões em sala de aula, colaboração com os colegas

Comprometimento com as tarefas e datas de entregas

Interesse e envolvimento na execução de tarefas em duplas e grupos

Avaliações teste

Avaliações prática

Trabalhos e exercícios

MENÇÃO

Menção	Conceito	Definição Operacional
MB	Muito Bom	O aluno obteve excelente desempenho no desenvolvimento das competências do componente curricular no período.
B	Bom	O aluno obteve bom desempenho no desenvolvimento das competências do componente curricular no período.
R	Regular	O aluno obteve desempenho regular no desenvolvimento das competências do componente curricular no período.
I	Insatisfatório	O aluno obteve desempenho insatisfatório no desenvolvimento das competências do componente curricular no período.

REPRESENTANTES

Devem ter dois por grupos

MATERIAL DE APOIO

Material em PDF desenvolvidos pelo professor

Referências de autoridades no assunto

Links de apoio

Conceitos de bancos de dados

Um banco de dados é uma coleção organizada de informações ou dados estruturados, geralmente armazenados eletronicamente em um sistema de computador.

Características

- **Organização:** Os dados são armazenados em um formato específico, facilitando a busca e a manipulação.
- **Estrutura:** Os dados são organizados em tabelas, linhas e colunas, com cada coluna representando um tipo específico de informação.
- **Relações:** As tabelas podem ser relacionadas entre si, permitindo a criação de consultas complexas.

Um banco de dados bem planejado dá aos usuários acesso a informações essenciais.

Um banco de dados bem-estruturado.¹

- Economiza espaço em disco ao eliminar dados redundantes.
- Mantém a exatidão e a integridade dos dados.
- Oferece acesso aos dados de maneiras úteis.

Componentes

- **Dados:** A informação em si, como nomes, datas, números etc.
- **Hardware:** O equipamento físico que armazena o banco de dados, como discos rígidos ou servidores.
- **Software:** O sistema de gerenciamento de banco de dados (SGBD), que controla o acesso e a manipulação dos dados.
- **Usuários:** As pessoas que utilizam o banco de dados, como administradores, desenvolvedores e usuários finais.

Tipos de Bancos de Dados:

- **Relacionais:** O tipo mais comum, armazena dados em tabelas relacionadas entre si.
- **NoSQL:** Mais flexíveis que os bancos de dados relacionais, são ideais para armazenar grandes volumes de dados não estruturados.
- **Orientados a objetos:** Armazenam dados em objetos, que são unidades de informação que contêm dados e métodos.

Vantagens de Usar Bancos de Dados:

- **Organização:** Os dados são armazenados de forma organizada e eficiente, facilitando a busca e a manipulação.
- **Compartilhamento:** Os dados podem ser facilmente compartilhados entre diferentes usuários e aplicativos.
- **Segurança:** Os dados podem ser protegidos contra acesso não autorizado.
- **Integridade:** Os dados podem ser mantidos consistentes e atualizados.
- **Escalabilidade:** Os bancos de dados podem ser dimensionados para atender às necessidades crescentes de uma organização.

Hoje os bancos de dados são essenciais para muitas empresas e estão no coração de muitos sistemas computacionais. Ter acesso rápido às informações é muito importante para a correta tomada de decisões em um negócio.

Se você pretende trabalhar com desenvolvimento de softwares, com certeza precisará trabalhar com bancos de dados em algum momento. Conhecer a história dos bancos de dados e como eles evoluíram é muito importante para entender como os bancos de dados mais comuns são organizados. Então como aconteceu a evolução dos bancos de dados?²

¹ <https://www.advancedit.com.br/tipos-de-banco-de-dados-conheca-os-principais-e-descubra-qual-deles-e-o-melhor/>

² <https://dicasdeprogramacao.com.br/a-historia-dos-bancos-de-dados/>

Modelo relacional

O modelo relacional em bancos de dados é um modelo de dados para organizar e gerenciar informações em bancos de dados. Ele se baseia na matemática e na teoria dos conjuntos para representar os dados de maneira simples e eficiente.

Características do Modelo Relacional

- **Dados estruturados em tabelas:** As informações são organizadas em linhas (tuplas) e colunas (atributos), similar a uma planilha.
- **Chaves primárias:** Cada registro em uma tabela é identificado de forma única por uma chave primária.
- **Chaves estrangeiras:** As relações entre as tabelas são estabelecidas por meio de chaves estrangeiras, que referenciam as chaves primárias de outras tabelas.
- **Normalização:** O processo de dividir as tabelas em unidades menores para evitar redundância e inconsistências nos dados.

Vantagens do Modelo Relacional

- **Simplicidade:** Fácil de entender e usar, tanto para usuários técnicos quanto para não técnicos.
- **Flexibilidade:** Permite modelar uma grande variedade de tipos de dados.
- **Eficiência:** Oferece bom desempenho para operações de consulta e manipulação de dados.
- **Integridade:** As regras de negócio podem ser facilmente implementadas para garantir a qualidade dos dados.

Exemplos de Implementação do Modelo Relacional

- **MySQL:** Um SGBD relacional popular e gratuito.
- **Microsoft SQL Server:** Um SGBD relacional robusto e escalável.
- **Oracle Database:** Um SGBD relacional de alto desempenho e confiabilidade.

O modelo relacional é um modelo clássico e fundamental para bancos de dados, mas pode não ser adequado para todos os tipos de dados e aplicações.

Novas tecnologias, como bancos de dados NoSQL, oferecem alternativas para lidar com grandes volumes de dados não estruturados.³

Sistemas gerenciadores de bancos de dados (SGBDR)

Um SGBDR é um software que gerencia e controla o acesso a um banco de dados. Ele fornece uma interface para que os usuários possam criar, ler, atualizar e excluir dados.

Funções do SGBDR

- **Armazenamento:** Armazena os dados em um formato estruturado e seguro.
- **Organização:** Organiza os dados em tabelas, linhas e colunas.
- **Acesso:** Controla o acesso aos dados, garantindo a segurança e a integridade.
- **Manipulação:** Permite a manipulação dos dados por meio de comandos SQL.
- **Otimização:** Otimiza o desempenho das consultas e operações de banco de dados.
- **Recuperação:** Permite a recuperação de dados em caso de falhas.

³ <https://www.advancedit.com.br/tipos-de-banco-de-dados-conheca-os-principais-e-descubra-qual-deles-e-o-melhor/>

Tipos de SGBDR

- **Relacionais:** O tipo mais comum, como MySQL, SQL Server e Oracle.
- **NoSQL:** Para grandes volumes de dados não estruturados, como MongoDB, Cassandra e HBase.

Vantagens de Usar um SGBDR

- **Eficiência:** Melhora o desempenho e a eficiência do acesso aos dados.
- **Segurança:** Protege os dados contra acesso não autorizado.
- **Integridade:** Garante a consistência e a qualidade dos dados.
- **Facilidade de uso:** Oferece uma interface amigável para os usuários.
- **Escalabilidade:** Permite o crescimento do banco de dados de acordo com as necessidades.

Exemplos de SGBDR Populares

- **MySQL:** Gratuito, de código aberto e muito utilizado.
- **Microsoft SQL Server:** Robusto, escalável e com bom desempenho.
- **Oracle Database:** Alto desempenho, confiabilidade e segurança.
- **MongoDB:** NoSQL popular para documentos JSON.
- **Cassandra:** NoSQL escalável para grandes volumes de dados.

SGBDR vs SGBD

SGBDR e SGBD são termos frequentemente utilizados em informática, mas com significados distintos.

SGBDR: Sigla para Sistema Gerenciador de Banco de Dados Relacional.

É um software que gerencia e controla o acesso a um banco de dados relacional.

O modelo relacional organiza os dados em tabelas, com linhas e colunas, similar a uma planilha.

Exemplos de SGBDRs: MySQL, SQL Server, Oracle Database.

SGBD: Sigla para Sistema Gerenciador de Banco de Dados.

É um termo mais genérico que se refere a qualquer tipo de software que gerencia um banco de dados, independentemente do modelo de dados utilizado.

O SGBD pode ser relacional (SGBDR), NoSQL, orientado a objetos etc.

Exemplos de SGBDs que não são SGBDRs: MongoDB (NoSQL), POET (orientado a objetos).

Em resumo:

- SGBDR é um tipo específico de SGBD que utiliza o modelo relacional.
- SGBD é um termo mais amplo que abrange todos os tipos de softwares que gerenciam bancos de dados, independentemente do modelo.

Analogia

SGBDR é como um carro específico, como um Fiat Uno.

SGBD é como um veículo em geral, que pode ser um carro, moto, caminhão etc.

Qual usar?

Se você precisa de um banco de dados relacional, escolha um SGBDR.

Se você precisa de um banco de dados NoSQL, orientado a objetos etc., escolha um SGBD específico para esse modelo.

A escolha do tipo de SGBD depende das necessidades específicas da sua aplicação.

É importante ter conhecimentos básicos de SGBDs para gerenciar e utilizar um banco de dados de forma eficiente.

Arquitetura cliente/servidor

A arquitetura cliente/servidor é um modelo de computação distribuída que divide as responsabilidades entre dois tipos de programas, clientes e servidores. Essa divisão permite um melhor aproveitamento dos recursos e facilita a escalabilidade de sistemas.

Funcionamento

- **Cliente:** Solicita serviços ou recursos ao servidor.
- **Servidor:** Processa a requisição do cliente e envia uma resposta.
- **Comunicação:** A comunicação entre cliente e servidor é feita através de protocolos de rede, como TCP/IP e HTTP.

Exemplos de aplicações

- **Navegação na web:** O navegador é o cliente e o servidor web entrega as páginas.
- **E-mail:** O cliente de e-mail se conecta ao servidor de e-mail para enviar e receber mensagens.
- **Jogos online:** O cliente do jogo se conecta ao servidor do jogo para jogar com outras pessoas.

Tipos de Arquitetura Cliente/Servidor

- **Dois Níveis (Two-Tier):** Cliente se comunica diretamente com o servidor.
- **Três Níveis (Three-Tier):** Intermediário entre cliente e servidor, como um aplicativo web.
- **N-Níveis:** Arquiteturas mais complexas com vários níveis de intermediários.

Vantagens

- **Escalabilidade:** Facilidade de adicionar mais clientes ou servidores.
- **Desempenho:** Distribuição da carga de trabalho entre os servidores.
- **Segurança:** Centralização da segurança no servidor.
- **Manutenção:** Facilidade de gerenciar e atualizar o sistema.

Desvantagens

- **Complexidade:** Arquiteturas complexas podem ser mais difíceis de configurar e gerenciar.
- **Custo:** Servidores podem ser mais caros que computadores clientes.
- **Dependência:** O sistema depende da disponibilidade do servidor.

A escolha da arquitetura cliente/servidor depende das necessidades específicas da sua aplicação.

É importante ter conhecimentos básicos de redes e protocolos para configurar e gerenciar um sistema cliente/servidor.

Linguagem estruturada de consulta (SQL).

A Linguagem de Consulta Estruturada (SQL) é uma linguagem de programação declarativa usada para gerenciar dados em bancos de dados relacionais. Ela permite manipular, recuperar e modificar dados de forma eficiente e organizada.

Funcionalidades da SQL

- Criar, ler, atualizar e excluir dados (CRUD) em tabelas.
- Definir a estrutura das tabelas e seus relacionamentos.
- Controlar o acesso aos dados e garantir sua segurança.
- Realizar consultas complexas para obter informações específicas dos dados.
- Combinar dados de várias tabelas em uma única consulta.
- Analisar dados e gerar relatórios.

Bancos de dados relacionais

Um banco de dados relacional (SGBDR) é um sistema de armazenamento de dados que organiza os dados em tabelas relacionadas entre si. As tabelas são compostas por linhas (registros) e colunas (atributos). Essa estrutura facilita a organização, o acesso e a manipulação dos dados.

Características

- **Modelo relacional:** Baseado em matemática e teoria dos conjuntos.
- **Tabelas:** Estrutura principal para armazenar dados.
- **Chaves primárias:** Identificam unicamente cada registro em uma tabela.
- **Chaves estrangeiras:** Relacionam as tabelas entre si.
- **Normalização:** Processo de dividir as tabelas para evitar redundância e inconsistências.
- **Integridade:** Regras para garantir a qualidade dos dados.

Vantagens

- **Simplicidade:** Fácil de entender e usar.
- **Flexibilidade:** Permite modelar diversos tipos de dados.
- **Eficiência:** Desempenho otimizado para consultas e operações.
- **Integridade:** Mecanismos para garantir a qualidade dos dados.
- **Escalabilidade:** Suporta grandes volumes de dados.

Exemplos de SGBDRs

- **MySQL:** Gratuito, de código aberto e muito utilizado.
- **Microsoft SQL Server:** Robusto, escalável e com bom desempenho.
- **Oracle Database:** Alto desempenho, confiabilidade e segurança.

Para quem é recomendado?

- Desenvolvedores de software.
- Administradores de bancos de dados.
- Cientistas de dados.
- Analistas de negócios.
- Qualquer pessoa que precise trabalhar com dados.

Conceitos importantes

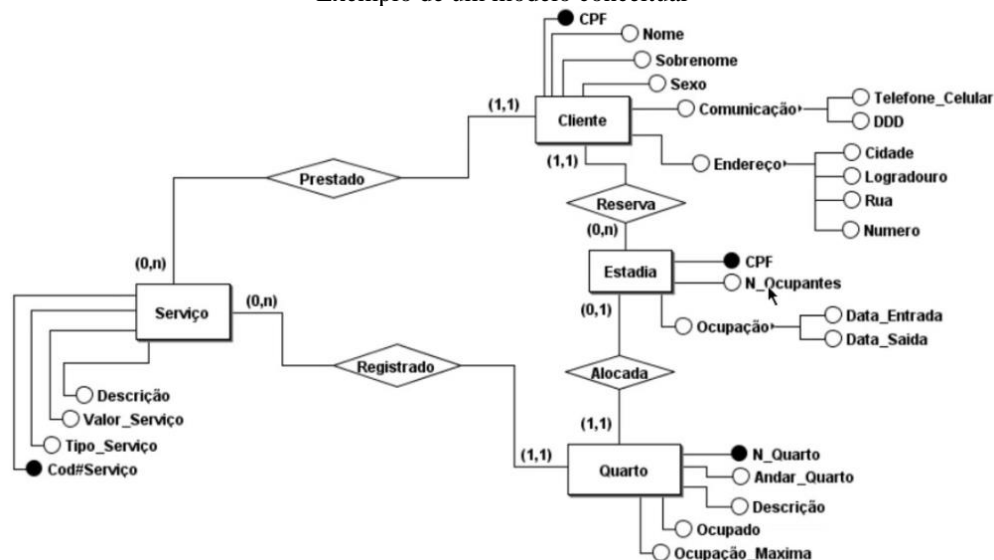
- **Normalização:** Processo de dividir as tabelas para evitar redundância e inconsistências.
- **Integridade:** Regras para garantir a qualidade dos dados.
- **Chaves primárias e estrangeiras:** Mecanismos para relacionar as tabelas entre si.
- **SQL:** Linguagem padrão para manipular dados em bancos de dados relacionais.

Os bancos de dados relacionais são uma ferramenta fundamental para armazenar e gerenciar dados de forma eficiente e organizada. Se você precisa trabalhar com dados, aprender sobre bancos de dados relacionais é um excelente investimento.

Modelo conceitual

Finalidade do modelo conceitual de dados é capturar os requisitos de informação e regras de negócio sob o ponto de vista do negócio. Para isto, torna-se necessário o entendimento e a correta aplicação dos mecanismos de abstração, utilizados na modelagem conceitual de dados.⁴

Exemplo de um modelo conceitual



Símbolos de entidade de diagramas ER

Entidades são objetos ou conceitos que representam dados importantes. Entidades são normalmente substantivos, como produto, cliente, localização ou promoção. Existem três tipos de entidades comumente utilizados em diagramas entidade-relacionamento.⁵

Símbolo de entidade	Nome	Descrição
	Entidade forte	Essas formas são independentes de outras entidades e são frequentemente chamadas de entidades primárias, pois normalmente têm entidades fracas que dependem delas. Elas também terão uma chave primária, destacando cada ocorrência da entidade.
	Entidade fraca	Entidades fracas dependem de algum outro tipo de entidade. Elas não possuem chaves primárias e não têm significância no diagrama sem sua entidade primária.
	Entidade associativa	Entidades associativas associam as instâncias de diversos tipos de entidades. Elas também contêm atributos voltados especificamente ao relacionamento entre tais instâncias de entidades.



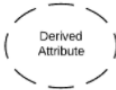
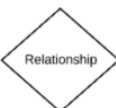
⁴<https://www.blrdata.com.br/single-post/2016/03/19/modelo-conceitual-de-dados-aprenda-a-utilizar-os-principais-mecanismos-de-abstra%C3%A7%C3%A3o>

https://i.ytimg.com/vi/_PVII4Y5GK4/maxresdefault.jpg

⁵ <https://www.lucidchart.com/pages/pt/simbolos-de-diagramas-entidade-relacionamento>

Símbolos de atributos de diagramas ER

Atributos de diagramas ER são características da entidade que ajudam usuários a entender melhor o banco de dados. Atributos são incluídos para incluir detalhes das várias entidades que são destacadas em um diagrama ER conceitual.

Símbolos de atributos	Nome	Descrição
	Atributo	Atributos são características de uma entidade, de um relacionamento de muitos para muitos ou de um relacionamento de um para um.
	Atributo multivalorado	Atributos multivalorados são aqueles capazes de assumir mais de um valor.
	Atributo derivado	Atributos derivados são atributos cujo valor pode ser calculado a partir de valores de atributos relacionados.
	Relacionamento	Relacionamentos são associações entre entidades.

Entidades, atributos e relacionamentos

Entidades, atributos e relacionamentos são os pilares dos bancos de dados relacionais.

- **Entidades:** São os objetos, pessoas, lugares ou eventos que armazenamos no banco de dados, representadas por tabelas.
- **Atributos:** São as características das entidades, representados por colunas nas tabelas, exemplos: nome, idade, endereço, data de nascimento.
- **Relacionamentos:** São as associações entre as entidades, representados por linhas que ligam as tabelas, exemplos: um cliente pode ter vários pedidos, um produto pode ter vários fornecedores.

Exemplos

- **Entidade:** Cliente
- **Atributos:** nome, idade, endereço, telefone
- **Relacionamento:** Pedido (um cliente pode ter vários pedidos)

- **Entidade:** Produto
- **Atributos:** nome, descrição, preço
- **Relacionamento:** Fornecedor (um produto pode ter vários fornecedores)

Tipos de Relacionamentos

- **Um para um:** Uma entidade em uma tabela se relaciona com no máximo uma entidade em outra tabela.
- **Um para muitos:** Uma entidade em uma tabela se relaciona com várias entidades em outra tabela.
- **Muitos para muitos:** Várias entidades em uma tabela se relacionam com várias entidades em outra tabela.

Chaves

- **Chave primária:** Identifica unicamente cada registro em uma tabela.
- **Chave estrangeira:** Estabelece o relacionamento entre as tabelas.

Entidade

Uma entidade (entity) é um objeto que existe e é distinguível dos outros objetos. Por exemplo, Paulo Silva com número de CPF 123.456.789-00 é uma entidade, visto que isso identifica unicamente uma pessoa particular do universo. Assim a conta número 40167-9 na agência Lapa é uma entidade que identifica unicamente uma conta corrente particular. Uma entidade pode ser concreta, como uma pessoa ou um livro, ou pode ser abstrata, como um feriado ou um conceito.

Um conjunto de entidades (entity set) é um conjunto de entidades do mesmo tipo. O conjunto de todas as pessoas com conta em um banco, por exemplo, pode ser definido como o conjunto de todas as entidades cliente. Similarmente, o conjunto de entidades conta pode representar o conjunto de todas as contas de um banco particular. É convenção adotar nomes de conjuntos de entidades no singular, mas não é obrigatório.

Conjuntos de entidades não precisam ser disjuntos. Por exemplo, é possível definir o conjunto de entidades de todos os funcionários de um banco (funcionários) e o conjunto de todos os clientes do banco (clientes). Uma entidade pessoa pode ser uma entidade funcionário, uma entidade cliente, ambas ou nenhuma delas.⁶

Atributos

São propriedades (características) que identificam as entidades. Uma entidade é representada por um conjunto de atributos.

Entidades.

Os atributos podem ser simples, composto, multivalorado ou determinante.

• Distinguir Atributos e Entidades;

Uma entidade é um objeto do mundo real que pode ser distinguido de outros objetos.
Toda entidade tem seus respectivos atributos.

• Analisar e modelar de Atributos;

Atributos são informações referentes a uma entidade que necessitam ser conhecidos, armazenados. Atributos descrevem uma Entidade pela qualificação, identificação, classificação, quantificação ou expressão de estado.

São propriedades (características) que identificam as entidades. Uma entidade é representada por um conjunto de atributos.

Relacionamentos:

✓ definição e classificações.

Os relacionamentos entre tabelas são implementados para evitar a repetição desnecessária dos dados, além desta situação também implementamos a integridade referencial, onde um dado não pode ser excluído ou cadastrado somente de acordo com algumas regras estabelecidas pelo banco de dados.

De acordo com a cardinalidade existem 3 tipos básicos de relacionamentos entre as entidades.

- Relacionamentos UM para UM
- Relacionamentos UM para MUITOS
- Relacionamentos MUITOS para MUITOS

➤ **Um para Um (1:1)**

São relacionamentos em que uma ocorrência de uma entidade em A está associada no máximo a uma ocorrência em uma entidade B e uma ocorrência na entidade B está associada no máximo a uma ocorrência na entidade A.

Neste relacionamento, escolhemos qual tabela irá receber a chave estrangeira, e para cada valor do campo na tabela A, há no máximo um valor na tabela B.

➤ **Um para Muitos (1:*)**

Este é o principal relacionamento utilizado em bancos de dados relacionais, neste caso um registro na tabela tb01 pode ter muitos registros coincidentes em tb02, porém um registro na tabela tb02 possui apenas um registro coincidente na tabela tb01.

Um relacionamento 1:m ocorre com frequência em situações de negócio.

Às vezes ocorre em forma de árvore ou em forma hierárquica.

➤ **Muitos para Muitos (*:*)**

Uma ocorrência de uma entidade em A está associada a qualquer número de ocorrências na entidade B, e cada ocorrência da entidade em B está associada a qualquer número de ocorrências na entidade A.

Exemplo:

Considere o caso em que itens são vendidos.

Podemos identificar imediatamente duas entidades: VENDA e ITEM. Uma venda pode consistir em muitos itens de mercadorias e um item de mercadoria pode aparecer em muitas vendas.

Não estamos dizendo que um mesmo item possa ser vendido muitas vezes, mas que o tipo específico de item (por exemplo, um livro) pode ser vendido muitas vezes; temos, portanto, um relacionamento de muitos-para-muitos (m:m) entre VENDA e ITEM.

Em um relacionamento m:m, criamos uma terceira entidade, chamada entidade associativa que é usada para associar as entidades por meio de dois relacionamentos 1:m.⁷

Grau de relacionamento (binário/ ternário)

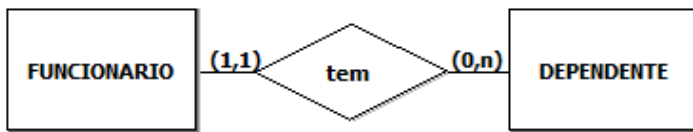
➤ **Comparação entre relacionamentos.**

Relacionamento binários se dão entre duas entidades.

- Conjunto de associações entre entidades sobre as quais deseja-se manter informações no BD.
- Descreve um conjunto de vínculos entre elementos de modelo.
- Relacionamento estrutural que especifica objetos de um item conectados a objetos de outro item:

⁷ http://www.cadcobol.com.br/db2_novo_tipos_relacionamentos.htm

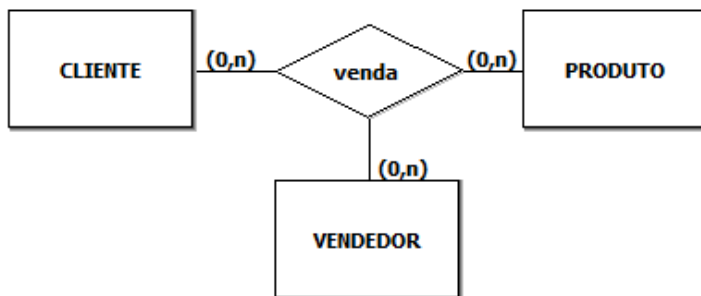
Exemplo:



Relacionamento ternário se dão entre três entidades.

- Denominamos ternários os relacionamentos entre três conjuntos de entidades.

Exemplo: Para que haja uma venda, tem que haver um cliente, um produto e um vendedor.



Grau de cardinalidade

- Definição e classificações.

Identifica quantas vezes cada instância de uma entidade pode participar do relacionamento. É diferente do grau de relacionamento – que mostra quantas entidades estão atreladas a um verbo / relacionamento. A cardinalidade mostra quantas vezes uma mesma entidade está atrelada ao mesmo verbo / relacionamento.

A cardinalidade indica a quantidade de ocorrências (tuplas) em uma tabela que estão associadas à quantidade de ocorrências na outra.

Cardinalidade Mínima é representada pelos símbolos 0 ou 1 – É o primeiro número.

Cardinalidade Máxima é representada pelos símbolos 1 ou N (ou M) – É o segundo número;

(Cardinalidade mínima, Cardinalidade máxima) = (X,Y).⁸

Auto relacionamento

Este tipo de relacionamento ocorre toda a vez que temos uma ocorrência de uma entidade que está associada a um ou mais ocorrências da mesma entidade.

Ou seja, temos uma entidade onde suas ocorrências possuem relacionamentos entre si.

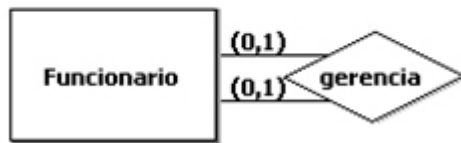
Os autorelacionamentos são na verdade uma forma de representarmos relações de hierarquia entre ocorrências de uma mesma entidade.

Por exemplo, vamos considerar uma entidade **EMPREGADO** sendo que no modelo conceitual devemos representar o conceito de que um empregado possui um gerente.

Ou seja, existe um relacionamento entre as ocorrências da entidade **EMPREGADO** que estabelece que um empregado é gerente de outro empregado.

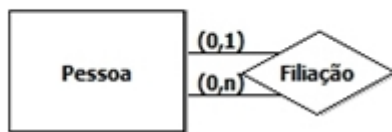
⁸ <https://cadernodeprova.com.br/relacionamentos-e-cardinalidades/>

Veja o exemplo abaixo de como deve ser representado este relacionamento:



Um outro exemplo que podemos representar seria o de uma entidade PESSOA que possui PAI.

Ora o pai de uma pessoa e a própria pessoa são ocorrências da mesma entidade PESSOA, portanto temos um relacionamento entre ocorrências da mesma entidade PESSOA que seriam as ocorrências da pessoa, do pai. representação desta situação seria feita da seguinte forma:



Observe que a cardinalidade do auto relacionamento indica opcionalidade visto que a PESSOA pode não ter pai conhecido ou ter somente um pai.

Os autorelacionamentos podem possuir qualquer tipo de cardinalidade.

No caso já vimos auto relacionamentos do tipo um para um. Mas também existem auto relacionamentos onde podemos ter cardinalidade muitos para muitos.

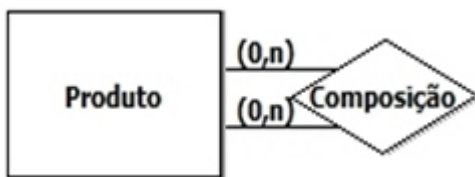
Considere um cenário de uma indústria onde produtos são compostos por componentes que são também produtos, de forma que um produto é composto por muitos componentes e pode compor muitos outros produtos.

Observe que estamos lidando com um único conceito que Produto, tanto componente como produto são produtos.

Neste caso, temos uma composição de produtos.

É importante observar que nesse caso a forma de representação é a mesma pois se trata da entidade PRODUTO que mantém um relacionamento COMPOSIÇÃO com ela mesma, sendo que a única alteração é com respeito à cardinalidade deste relacionamento que é muitos para muitos.

Segue abaixo exemplo de como seria representado este auto relacionamento.



Os autorelacionamentos são muito usados para representar hierarquias e composições de elementos do mundo real que são ocorrências em uma mesma entidade.

A forma de representação é simples sendo que devemos ter especial atenção a cardinalidade do auto relacionamento.

Especialização e generalização.

Especialização e generalização são técnicas de modelagem de dados que permitem organizar e estruturar os dados de forma hierárquica, representando entidades com diferentes níveis de granularidade.

O que é especialização?

A especialização é o processo de dividir uma entidade genérica em entidades mais específicas, cada uma com suas próprias características e propriedades. As entidades especializadas herdam as características da entidade genérica, mas também possuem características próprias que as diferenciam.

Exemplo

- **Entidade genérica:** (Veículo)
- **Entidades especializadas:** (Carro, Motocicleta, Caminhão...)

Cada entidade especializada possui características próprias que a diferenciam das demais. Por exemplo, um carro possui portas e assentos, enquanto uma motocicleta possui guidão e baú.

O que é generalização?

A generalização é o processo de agrupar entidades com características semelhantes em uma única entidade genérica. A entidade genérica representa as características comuns às entidades especializadas.

Exemplo

- **Entidade genérica:** (Animal)
- **Entidades especializadas:** (Cachorro, Gato, Cavalo, ...)

Todas as entidades especializadas são animais, portanto, herdam as características comuns aos animais, como ter pelos, respirar e se alimentar.

Tipos de especialização

Existem dois tipos de especialização:

- **Especialização total:** Cada ocorrência da entidade genérica deve pertencer a uma única entidade especializada.
- **Especialização parcial:** Uma ocorrência da entidade genérica pode pertencer a nenhuma, uma ou mais entidades especializadas.

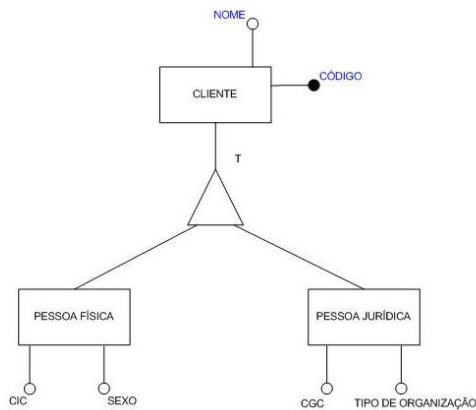
Exemplos

Imagine uma empresa de seguros que vende seguros para seus clientes que podem ser tanto cidadãos como empresas.

Neste caso teríamos a situação abaixo:

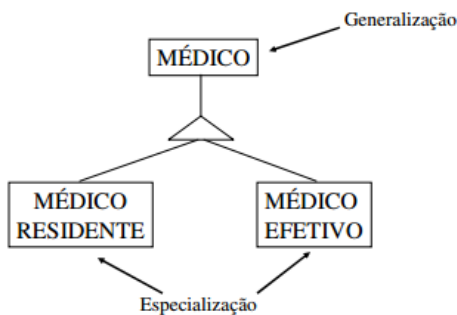
CLIENTE
PESSOA JURÍDICA
PESSOA FÍSICA

No Diagrama Entidade e Relacionamentos este conceito pode ser representado assim:



Em uma empresa de Planos de Saúde poderíamos ter a seguinte situação:

- PACIENTE
- MÉDICO
- MÉDICO RESIDENTE
- MÉDICO EFETIVO



Cada uma dessas categorias, além de características comuns, possui atributos distintos.

Generalização/Especialização

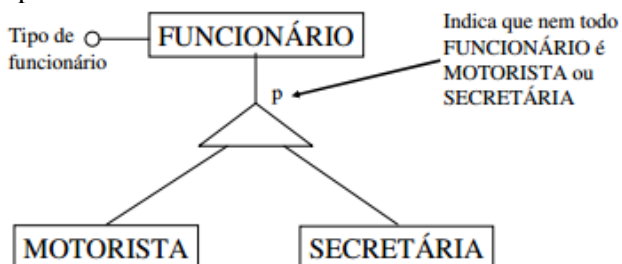
Através deste conceito é possível atribuir propriedades particulares a um subconjunto das ocorrências (especializadas) de uma entidade genérica ou entidade Pai.

Herança de propriedades: cada ocorrência da entidade especializada possui, além de seus próprios atributos e relacionamentos, todos os atributos da entidade generalizada.

Tipos de Generalização/Especialização

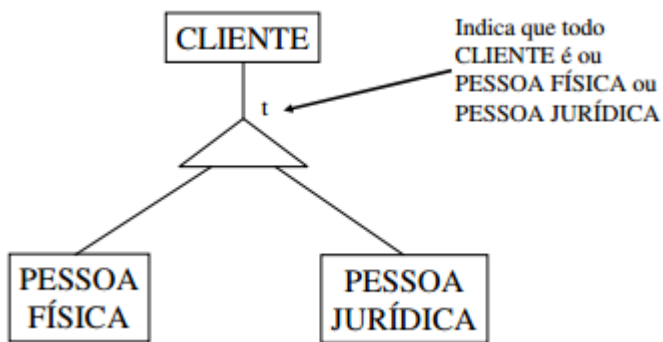
A Generalização/Especialização pode ser classificada em dois tipos:

Parcial: nem toda ocorrência da entidade genérica possui uma ocorrência correspondente em uma entidade especializada.



No exemplo acima, nem todo funcionário é motorista e nem todo funcionário é secretária. Pode haver funcionários que não sejam nem motorista e nem secretária.

Total: para toda ocorrência da entidade genérica existe sempre uma ocorrência em uma das entidades especializadas.



No exemplo acima, todo cliente ou é uma pessoa física ou uma pessoa jurídica. Não existe a possibilidade de haver um cliente que não seja pessoa física OU pessoa jurídica.

Vantagens da especialização e generalização

- **Melhor organização dos dados:** As entidades são organizadas de forma hierárquica, facilitando a compreensão e o gerenciamento dos dados.
- **Redução da redundância:** As características comuns às entidades especializadas são armazenadas apenas na entidade genérica, evitando redundância no banco de dados.
- **Maior flexibilidade:** O modelo de dados pode ser facilmente adaptado para novas necessidades.

Desvantagens da especialização e generalização

- **Aumento da complexidade do modelo:** O modelo de dados pode se tornar mais complexo, dificultando a compreensão e a manutenção.
- **Dificuldade de implementação:** A implementação da especialização e generalização pode ser complexa em alguns sistemas de gerenciamento de banco de dados.

Quando usar especialização e generalização?

A especialização e generalização devem ser utilizadas quando:

- Existem entidades com características comuns que podem ser agrupadas em uma única entidade genérica.
- Existem entidades que podem ser divididas em entidades mais específicas.

É importante avaliar as vantagens e desvantagens da especialização e generalização antes de utilizá-las em um modelo de dados.

Dicionário de dados em banco de dados

Um dicionário de dados é um repositório central de metadados que descreve os objetos de um banco de dados, como tabelas, colunas, chaves e índices.

Ele fornece informações sobre:

- **Nome do objeto:** Nome da tabela, coluna, chave ou índice.
- **Descrição:** Uma breve descrição do que o objeto representa.
- **Tipo de dado:** O tipo de dado armazenado na coluna.
- **Tamanho:** O tamanho do campo em bytes.
- **Restrições:** Restrições de integridade, como NOT NULL, UNIQUE e FOREIGN KEY.
- **Relacionamentos:** Relacionamentos entre as tabelas.

- **Proprietário:** O usuário responsável pelo objeto.
- **Data de criação:** A data em que o objeto foi criado.
- **Data de modificação:** A data em que o objeto foi modificado pela última vez.

Benefícios de um dicionário de dados

- Melhora a documentação do banco de dados: Facilita a compreensão da estrutura e do funcionamento do banco de dados.
- Facilita o gerenciamento do banco de dados: Permite que os administradores e desenvolvedores gerenciem os objetos de forma mais eficiente.
- Melhora a qualidade dos dados: Ajuda a garantir a consistência e a integridade dos dados.
- Reduz a redundância de dados: Evita que as mesmas informações sejam armazenadas em vários lugares.
- Melhora a comunicação entre os usuários do banco de dados: Facilita a comunicação entre os usuários do banco de dados, pois fornece um vocabulário comum para descrever os objetos.

Ferramentas para criar um dicionário de dados

Ferramentas de modelagem de dados: A maioria das ferramentas de modelagem de dados possui recursos para criar dicionários de dados.

SGBDs: Alguns SGBDs, como o Oracle e o SQL Server, possuem ferramentas integradas para criar dicionários de dados.

Ferramentas de terceiros: Existem diversas ferramentas de terceiros para criar dicionários de dados.

Recomendações

Mantenha o dicionário de dados atualizado.

Defina um padrão para a documentação dos objetos.

Utilize ferramentas para facilitar a criação e a manutenção do dicionário de dados.

Treine os usuários do banco de dados sobre como usar o dicionário de dados.

Junto com o modelo de entidade e relacionamento, é necessário que se mantenha um documento com a explicação de todos os objetos nele criados. Este documento, que pode ser chamado de dicionário de dados, permite que os analistas obtenham informações sobre todos os objetos do modelo de forma textual, contendo explicações por vezes difíceis de incluir no diagrama. É válido lembrar que o objetivo do documento é ser claro e consistente.¹⁰

Exemplo de dicionário de dados:¹¹

Entidade: Cliente				
Atributo	Classe	Domínio	Tamanho	Descrição
Codigo_cliente	Determinante	Númerico		
Nome	Simples	Texto	50	
Telefone	Multivalorado	Texto	50	Valores sem as máscaras de entrada
Cidade	Simples	Texto	50	
data_nascimento	Simples	Data		Formato dd/mm/aaaa

¹¹

<https://www.google.com/imgres?imgurl=https%3A%2F%2Fwww.luis.blog.br%2Fuserfiles%2Fimage%2Fdicionario%2520de%2520dados.gif&imgrefurl=https%3A%2F%2Fwww.luis.blog.br%2Fdicionario-de-dados.html&tbnid=FYV9srGEeKCgEM&vet=12ahUKEwiVwJWaseH1AhWwOLkGHTIDD1oQMygBegUIARC4AQ..i&docid=Qe-i3kKbaq3KBM&w=484&h=137&itg=1&q=dicion%C3%A1rio%20de%20dados%20o%20que%20%C3%A9&ved=2ahUKEwiVwJWaseH1AhWwOLkGHTIDD1oQMygBegUIARC4AQ>

Normalização

- Processo de dividir as tabelas para evitar redundância e inconsistências.
- Garante a qualidade dos dados no banco de dados.

Entidades, atributos e relacionamentos são os conceitos básicos dos bancos de dados relacionais. Compreendê-los é fundamental para modelar e gerenciar dados de forma eficiente.

Integridade referencial.

A integridade referencial é uma regra em bancos de dados relacionais que garante a consistência dos dados. Ela garante que cada valor em uma coluna de chave estrangeira corresponda a um valor existente em uma coluna de chave primária em outra tabela.

Importância

- Evita inconsistências: Impede a criação de dados "órfãos" sem referencia em outras tabelas.
- Melhora a confiabilidade: Aumenta a confiança na qualidade dos dados.
- Facilita a recuperação de dados: Permite recuperar dados de forma mais precisa e eficiente.

Chave Primária: Cada tabela possui uma coluna que identifica unicamente cada registro.

Chave Estrangeira: Uma coluna em uma tabela que referencia uma coluna de chave primária em outra tabela.

Regras

- Cada valor na chave estrangeira deve existir na chave primária referenciada.
- Um valor na chave primária não pode ser excluído se estiver referenciado por uma chave estrangeira.

Tipos de Integridade Referencial

- **Integridade referencial restrita:** Não permite a exclusão de um valor na chave primária se estiver referenciado por uma chave estrangeira.
- **Integridade referencial em cascata:** Exclui automaticamente os registros na tabela referenciada quando um valor na chave primária é excluído.
- **Integridade referencial nula:** Permite que a chave estrangeira tenha valores nulos.

Exemplos

- **Tabela Clientes:** Coluna id_cliente (chave primária)
- **Tabela Pedidos:** Coluna id_cliente (chave estrangeira) que referencia id_cliente em Clientes

Violações da Integridade Referencial

Ocorre quando um valor na chave estrangeira não existe na chave primária referenciada.

Pode ser causada por exclusão de um valor na chave primária ou inserção de um valor inválido na chave estrangeira.

Soluções para Violações

- **Rejeitar a operação:** Impedir a operação que causa a violação.
- **Cascadeamento:** Excluir automaticamente os registros na tabela referenciada.
- **Atualização em cascata:** Atualizar automaticamente os valores na chave estrangeira para NULL.

A integridade referencial é uma ferramenta fundamental para garantir a consistência e confiabilidade dos dados em bancos de dados relacionais. É importante compreender como ela funciona e como configurá-la de acordo com as necessidades do seu sistema.

Modelagem E-R e Normalização

Modelagem E-R

A modelagem entidade-relacionamento (E-R) é uma técnica para projetar bancos de dados relacionais. Ela utiliza diagramas para representar as entidades (objetos, pessoas, lugares ou eventos) que serão armazenados no banco de dados, seus atributos (características) e os relacionamentos entre elas.

Componentes da Modelagem E-R

- **Entidades:** Representadas por retângulos.
- **Atributos:** Representados por ovais dentro das entidades.
- **Relacionamentos:** Representados por linhas que ligam as entidades.
- **Cardinalidade:** Indica o número de entidades que podem participar de um relacionamento.
- **Modalidade:** Indica o tipo de relacionamento (um para um, um para muitos, muitos para muitos).

Tipos de Diagramas E-R

- Diagrama E-R conceitual: Representa a visão geral do banco de dados.
- Diagrama E-R lógico: Representa a estrutura detalhada do banco de dados.

Normalização

O conceito de normalização foi introduzido por E. F. Codd em 1970 (primeira forma normal). Esta técnica é um processo matemático formal, que tem seus fundamentos na teoria dos conjuntos.

A normalização é um processo de dividir as tabelas em um banco de dados relacional para evitar redundância e inconsistências nos dados. Ela visa garantir a integridade e a eficiência do banco de dados.

Através deste processo pode-se, gradativamente, substituir um conjunto de entidades e relacionamentos por um outro. O qual se apresenta "purificado" em relação às anomalias de (atualização, inclusão, alteração e exclusão) as quais podem causar certos problemas.

Tais como:

- Grupos repetitivos (atributos multivalorados) de dados;
- Dependências parciais em relação a uma chave concatenada;
- Redundâncias de dados desnecessárias;
- Perdas acidentais de informação;
- Dificuldade na representação de fatos da realidade observada;
- Dependências transitivas entre atributos.

Benefícios da Normalização

- **Redução da redundância:** Evita que os mesmos dados sejam armazenados em vários lugares.
- **Eliminação de inconsistências:** Melhora a qualidade dos dados.
- **Melhoria do desempenho:** Aumenta a velocidade e a eficiência das consultas.

Formas normais (1FN, 2FN e 3FN);

Primeira forma normal (1FN): Cada célula da tabela deve conter apenas um valor único. Os atributos da tabela não contêm grupos de repetição (tabelas aninhadas).

Segunda forma normal (2FN): Todos os atributos não-chave devem ser dependentes da chave primária completa.

Terceira forma normal (3FN): Todos os atributos não-chave devem ser dependentes transitivamente da chave primária. Cada coluna não PK depende DIRETAMENTE da PK.

O processo de normalização leva ao refinamento das entidades, retirando delas grande parte das redundâncias e inconsistências.

Naturalmente, para que haja uma associação entre entidades é preciso que ocorram redundâncias mínimas de atributos que evidenciam estes relacionamentos.

Sem estas redundâncias não haveria relacionamento entre entidades.

A modelagem E-R e a normalização são técnicas importantes para projetar bancos de dados relacionais eficientes e confiáveis. Compreendê-las é fundamental para qualquer profissional que trabalha com bancos de dados.

Ferramenta CASE para criação de Diagramas E-R;

O método do caso é uma metodologia de ensino que usa casos reais de organizações ou situações de negócio para colocar os alunos no papel dos decisores que foram confrontados com decisões difíceis em algum momento. Em contraste com outros métodos de ensino, o método do caso exige que os professores evitem dar as suas próprias opiniões sobre as decisões em causa. Em vez disso, a principal tarefa dos professores que usam o método do caso é pedir aos alunos para elaborarem e defenderem soluções e planos de ação para os problemas centrais de cada caso. O método do caso propõe o desenvolvimento de habilidades gerenciais à medida que os participantes buscam solucionar um problema. O método do caso está fundamentado na aprendizagem baseada em problemas

Definição de ferramentas CASE (Computer-Aided Software Engineering)

Ferramentas CASE (do inglês Computer-Aided Software Engineering) é uma classificação que abrange todas as ferramentas baseadas em computadores que auxiliam atividades de engenharia de software, desde análise de requisitos e modelagem até programação e testes. Podem ser consideradas como ferramentas automatizadas que tem como objetivo auxiliar o desenvolvedor de sistemas em uma ou várias etapas do ciclo de vida do desenvolvimento de um software.¹²

Elas fornecem recursos para criar diagramas E-R: Ferramentas de desenho para representar entidades, atributos, relacionamentos e cardinalidades.

- **Validar o modelo:** Verificação de erros e inconsistências no modelo.
- **Gerar código SQL:** Criação automática do código SQL para criar o banco de dados.
- **Documentar o modelo:** Geração de documentação detalhada do modelo E-R.

Ferramentas CASE Populares

Visio: Ferramenta popular da Microsoft com recursos para modelagem E-R, mas exige pagamento de licença.

- **PowerDesigner:** Ferramenta robusta com suporte para diversos tipos de modelagem, incluindo E-R, UML e BPMN, mas também exige pagamento de licença.
- **Altova DatabaseSpy:** Ferramenta completa para design, desenvolvimento e gerenciamento de bancos de dados, com recursos para modelagem E-R, mas com custo elevado.
- **Rational Rose:** Ferramenta profissional para modelagem UML, com suporte a modelagem E-R, mas com custo elevado.

Ferramentas CASE Gratuitas

- **MySQL Workbench:** Ferramenta completa para modelagem E-R, com suporte a diversos bancos de dados, e de código aberto.
- **Dia:** Software de código aberto com recursos básicos para modelagem E-R.

¹² https://pt.wikipedia.org/wiki/Ferramenta_CASE

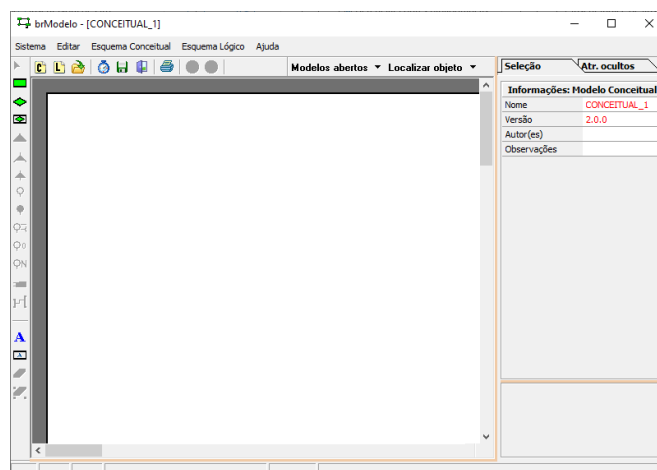
- **yEd Graph Editor:** Ferramenta online gratuita para criação de diversos tipos de diagramas, incluindo diagramas E-R.
- **Graphviz:** Ferramenta de código aberto para criação de gráficos e diagramas, incluindo diagramas E-R.

Outras Ferramentas CASE

- **Lucidchart:** Ferramenta online com interface amigável e diversos modelos pré-definidos, mas exige pagamento de assinatura.
- **ERDPlus:** Software leve e fácil de usar, ideal para projetos simples, mas com recursos limitados.
- **brModelo:** é uma ferramenta de modelagem de dados originalmente desenvolvida no Brasil. Ela pode ser usada para criar e manipular Diagramas Entidade-Relacionamento (DER), que é uma maneira de representar visualmente as relações entre diferentes entidades em um banco de dados.

brModelo

Em 2005 foi desenvolvida uma ferramenta de código aberto e totalmente gratuita voltada para ensino de modelagem de banco de dados relacionais com base na metodologia defendida por Carlos A. Heuser no livro “Projeto de Banco de Dados”. A ferramenta foi concebida pelo autor como trabalho de conclusão do curso de especialização em banco de dados pelas universidades UFSC (SC) e UNIVAG (MT), orientado pelo Professor Dr. Ronaldo dos Santos Mello, após se constatar a inexistência de uma ferramenta nacional que pudesse ser utilizada para essa finalidade.¹³



O brModelo vem em duas versões:

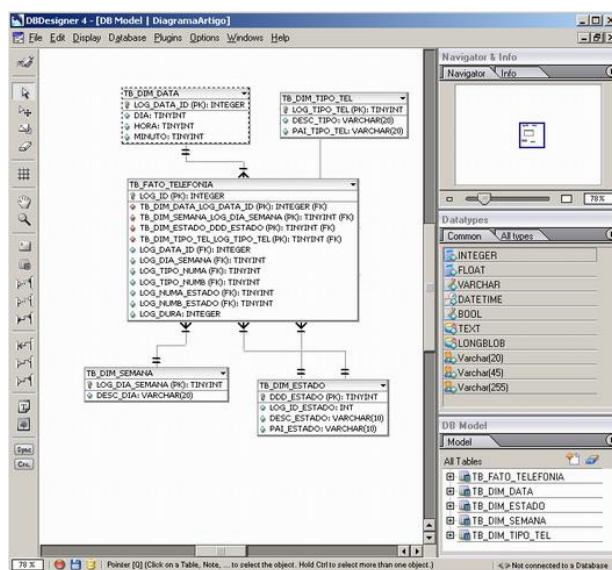
- **brModelo Desktop:** É um aplicativo gratuito para download que você pode instalar no seu computador. Existem versões disponíveis para Windows e possivelmente outros sistemas operacionais [buscar download brModelo].
- **BRModelo Web:** É uma versão web do brModelo que você pode acessar de qualquer navegador. Há um plano gratuito e planos pagos com recursos adicionais [buscar BRModeloWeb].

O brModelo é uma ferramenta popular no Brasil, mas também é usada por profissionais de banco de dados em todo o mundo.

Link para utilizar web: <https://www.brmodeloweb.com/lang/pt-br/index.html>

¹³ <http://www.sis4.com/brModelo/>

O DBDesigner 4 é uma ferramenta CASE (Computer-Aided Software Engineering) desenvolvida pela empresa Fabulous Force Database Tools. Esta ferramenta é livre e utilizada para a modelagem de dados visual que está disponível sob a licença GNU General Public License (GLP).¹⁴



- DBDesigner: é uma ferramenta de design de banco de dados visual que integra todos os níveis de design, modificação, criação e manutenção na estrutura de um banco de dados. Ele combina recursos profissionais e uma interface de usuário clara e simples para oferecer a maneira mais eficiente de lidar com seus bancos de dados.

O DBDesigner pode ser usado para:

- Criar ERDs (Diagramas Entidade-Relacionamento)
- Criar modelos físicos e lógicos de banco de dados
- Gerar SQL a partir de modelos de banco de dados
- Sincronizar modelos de banco de dados com bancos de dados existentes
- Colaborar em projetos de banco de dados com outros

O DBDesigner está disponível para Windows, macOS e Linux. Existe uma versão gratuita e paga do DBDesigner. A versão gratuita é limitada em termos de recursos, mas é suficiente para a maioria dos usuários. A versão paga inclui recursos adicionais, como suporte para vários bancos de dados, geração de código avançada e ferramentas de colaboração.

- **Suporte para vários bancos de dados:** O DBDesigner pode ser usado com uma variedade de bancos de dados, incluindo MySQL, PostgreSQL, Microsoft SQL Server, Oracle e DB2.
- **Diagramação ERD fácil:** O DBDesigner fornece uma interface fácil de usar para criar ERDs.
- **Geração de SQL:** O DBDesigner pode gerar SQL a partir de seus modelos de banco de dados. Isso pode economizar muito tempo e esforço.
- **Sincronização de modelo:** O DBDesigner pode sincronizar seus modelos de banco de dados com bancos de dados existentes. Isso garante que seus modelos estejam sempre atualizados.
- **Colaboração:** O DBDesigner permite que você colabore em projetos de banco de dados com outros.

O DBDesigner é uma ferramenta poderosa e versátil que pode ser usada por uma variedade de usuários, desde desenvolvedores individuais até grandes empresas. Se você está procurando uma ferramenta de design de banco de dados que seja fácil de usar e poderosa, o DBDesigner é uma ótima opção.

¹⁴ https://wiki.ifsc.edu.br/mediawiki/index.php/PI_S5_DSW_I_DouglasARS

Fatores a Considerar na Escolha da Ferramenta CASE

- **Complexidade do projeto:** Ferramentas mais simples são adequadas para projetos menores, enquanto ferramentas mais robustas são necessárias para projetos complexos.
- **Recursos disponíveis:** Avalie o custo, as funcionalidades e a integração com outras ferramentas.
- **Experiência do usuário:** Escolha uma ferramenta com interface amigável e fácil de usar.
- **Suporte técnico:** Verifique se a ferramenta oferece suporte técnico de qualidade.

Dicas para Escolher a Ferramenta CASE Ideal

- Teste algumas ferramentas antes de tomar uma decisão.
- Leia avaliações de outros usuários.
- Verifique se a ferramenta oferece tutoriais e documentação.
- Considere suas necessidades específicas e escolha a ferramenta que melhor se adapta a elas.

Representação gráfica de entidades, atributos e relacionamentos utilizando a Ferramenta Case brModelo

The screenshot shows the brModelo CASE tool interface. The main window displays a conceptual model diagram with four entities: MEDICO, FORMACAO, PACIENTE, and QUARTO. The interface includes a menu bar (Sistema, Editar, Esquema Conceitual, Esquema Lógico, Ajuda) and a toolbar. A red circle highlights the 'Sistema' icon in the toolbar. On the right, a panel titled 'Informações: Modelo Conceitual' displays the following information:

Informações: Modelo Conceitual	
Nome	CONCEITUAL_1
Versão	2.0.0
Autor(es)	
Observações	

Below the screenshot, a legend identifies the symbols used in the diagram:

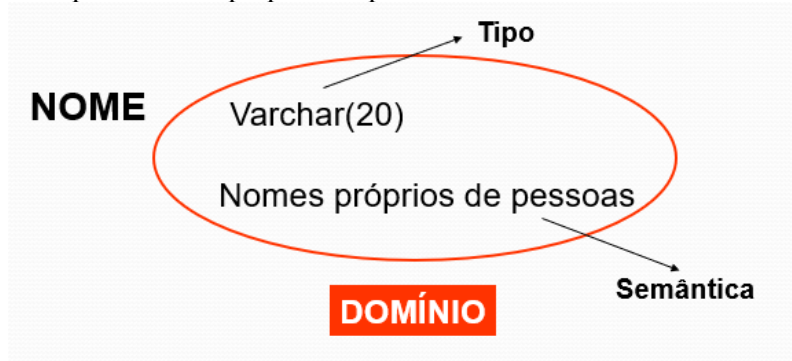
Ícone	Descrição
	Atributo obrigatório
	Atributo identificador
	Atributo composto
	Atributo opcional
	Atributo multivalorado
	Entidade
	Relação
	Ligação

Domínio

- Conjunto de valores permitidos para um dado
- Possui uma descrição física e outra semântica.
- A descrição física identifica o tipo e o formato dos valores que compõem o domínio a descrição semântica ajuda na interpretação de seus valores

exemplo: “Números de telefone válidos no Brasil (99)9999-9999”

exemplo: “Nomes próprios de pessoas até 20 caracteres”



Conceito de tabelas

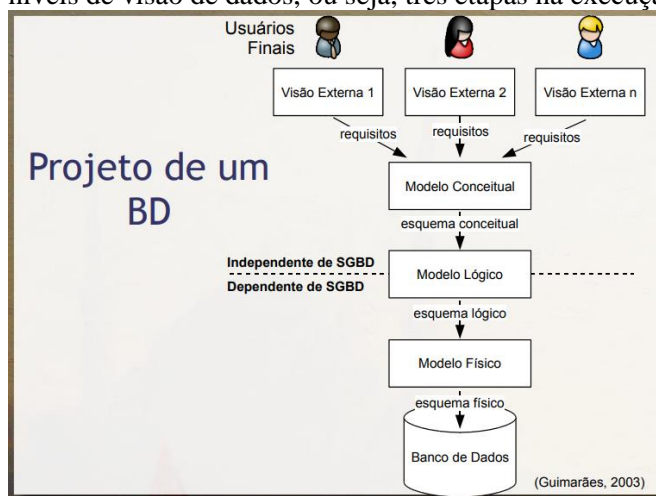
Uma tabela pode ser composta de uma lista completa de indivíduos em uma lista de discussão, incluindo seus endereços e números de telefone. As tabelas são frequentemente classificadas por finalidade ou aplicação a que se destinam. Alguns exemplos comuns incluem listas de discussão, tabelas de controle de qualidade, tabelas de estoque, ou tabelas de documentos. Tabelas também podem ser classificadas pelo grau de permanência que elas têm. Tabelas de transação são apenas temporárias, enquanto tabelas mestres são utilizadas por mais tempo.

Uma tabela é a estrutura dentro do seu banco de dados que contém os dados organizados em campos (colunas) e registros (linhas).¹⁵

Construção de projeto lógico de banco de dados

Todo projeto de um sistema de aplicação para banco de dados necessita de um coração, um centro nervoso. A modelagem de um sistema pela abordagem Entidade-Relacionamento representa esse ponto central no Projeto Conceitual de um sistema.

A utilização de uma abordagem correta de metodologia orientada a banco de dados envolve a estruturação nos três níveis de visão de dados, ou seja, três etapas na execução de um projeto: conceitual, lógico e físico.



¹⁵ <http://ehgomes.com.br/disciplinas/bdd/introducao.php>

Um banco de dados bem-estruturado:

- Economiza espaço em disco ao eliminar dados redundantes.
- Mantém a exatidão e a integridade dos dados.
- Oferece acesso aos dados de maneiras úteis.

Criar um banco de dados eficiente e útil é uma questão de seguir o processo adequado, incluindo as fases a seguir:

- Análise de requisitos, ou identificação do objetivo do banco de dados
- Organizando dados em tabelas
- Especificando chaves primárias e analisando relações
- Normalizando para padronizar as tabelas

Tipos de dados e NULL;

Tipos de Dados SQL Padrão ANSI

O SQL Padrão ANSI define um conjunto de tipos de dados básicos para garantir a interoperabilidade entre diferentes sistemas de gerenciamento de banco de dados (SGBDs).

Estes tipos de dados podem ser categorizados em:

1. Tipos Numéricos:

Inteiros:

- SMALLINT: Números inteiros com 2 bytes (de -32768 a 32767).
- INTEGER: Números inteiros com 4 bytes (de -2147483648 a 2147483647).
- BIGINT: Números inteiros com 8 bytes (de -9223372036854775808 a 9223372036854775807).

Reais:

- FLOAT: Números de ponto flutuante com precisão simples (7 dígitos).
- DOUBLE: Números de ponto flutuante com precisão dupla (15 dígitos).
- DECIMAL: Números de ponto flutuante com precisão e escala especificadas.

2. Tipos de Caractere:

- CHAR(n): String fixa de caracteres com tamanho n.
- VARCHAR(n): String variável de caracteres com tamanho máximo n.
- TEXT: String longa de caracteres (sem limite de tamanho).
- NCHAR(n): String fixa de caracteres Unicode com tamanho n.
- NVARCHAR(n): String variável de caracteres Unicode com tamanho máximo n.

3. Tipos de Data e Hora:

- DATE: Data no formato AAAA-MM-DD.
- TIME: Hora no formato HH:MM:SS.
- DATETIME: Data e hora no formato AAAA-MM-DD HH:MM:SS.
- TIMESTAMP: Data e hora com precisão de frações de segundo.

4. Tipos Especiais:

- BIT: Valor booleano (TRUE/FALSE).
- BINARY(n): Array de bytes com tamanho n.
- VARBINARY(n): Array variável de bytes com tamanho máximo n.

5. Tipos de Dados Nulos:

- NULL: Indica um valor ausente ou desconhecido.

Observações

A maioria dos SGBDs suportam tipos de dados adicionais além dos tipos padrão ANSI.

A sintaxe específica para declarar e usar tipos de dados pode variar entre diferentes SGBDs.

Consulte a documentação do seu SGBD para obter mais detalhes sobre os tipos de dados disponíveis.¹⁶

Modelo lógico

O Modelo Lógico tem por objetivo representar as estruturas que irão armazenar os dados dentro de um Banco de Dados, a partir deste momento é que são definidas com maior propriedade as entidades e os seus atributos.

• Regras de derivação

Regras de derivação descrevem atributos cujo valor depende de outro atributo ou de um cálculo. No diagrama, o símbolo RD é utilizado para identificar esse tipo de regra. A descrição completa da fórmula utilizada para compor o valor do atributo deve ser documentada no dicionário de dados.

Por exemplo, para modelar o valor total de uma compra, vamos acrescentar dois atributos no relacionamento COMPRA: quantidade comprada e total (o atributo preço já está modelado na entidade Livro). Esse último contém uma regra de derivação: seu valor é calculado através da soma da quantidade de livros comprados multiplicado pelo preço de venda e subtraído pelo desconto de promoção, caso o cliente tenha direito.¹⁷

• Regras de Restrição

O modelo relacional, ao definir conceitos como Tabela, Coluna, Nulo, Domínio, Chave Primária e Chave Estrangeira, deixa implícitas algumas regras fundamentais para a manutenção da consistência da base de dados. Elas são chamadas de Regras de Integridade e tratam dos cuidados que analistas, projetistas e programadores devem observar ao implementar as rotinas de Inclusão, Alteração e Exclusão de dados nos objetos.¹⁸

Desnormalização

A desnormalização é uma técnica usada em bancos de dados relacionais para aumentar o desempenho de consultas, principalmente consultas que envolvem várias tabelas relacionadas. Ela funciona através da introdução de redundância de dados.

- **Performance de consultas:** É a rapidez com que um banco de dados consegue responder a uma consulta. Consultas que envolvem várias tabelas (através de junções) geralmente são mais demoradas do que consultas em tabelas únicas.
- **Redundância de dados:** É quando o mesmo dado é armazenado em mais de um local no banco de dados. Normalmente, bancos de dados relacionais são projetados para minimizar a redundância para evitar problemas de consistência de dados. Porém, em alguns casos, a introdução de uma certa redundância estratégica pode melhorar significativamente a performance.

¹⁶ https://www.w3schools.com/sql/sql_datatypes.asp

<https://support.microsoft.com/pt-br/topic/tipos-de-dados-equivalentes-do-ansi-sql-7a0a6bef-ef25-45f9-8a9a-3c5f21b5c65d>

¹⁷ <https://www.devmedia.com.br/artigo-sql-magazine-04-projeto-de-banco-de-dados-modelo-logico/7647#:~:text=Regras%20de%20deriva%C3%A7%C3%A3o%20descrevem%20atributos,documentada%20no%20dicion%C3%A1rio%20de%20dados.>

¹⁸ <https://sites.google.com/site/fkbancodedados1/modelologico>

Aqui estão alguns cenários onde a desnormalização pode ser útil:

- **Consultas frequentes que envolvem junções:** Se você tem uma consulta que é executada com muita frequência e envolve a junção de várias tabelas, desnormalizar os dados necessários para essa consulta em uma tabela única pode melhorar drasticamente o desempenho.
- **Tabelas de dimensão em data warehouses:** Data warehouses são usados para armazenar dados históricos para análise. Tabelas de dimensão em data warehouses costumam ser desnormalizadas para melhorar o desempenho de consultas analíticas complexas.

É importante ressaltar que a desnormalização deve ser feita com cautela. Alguns pontos negativos a se considerar são:

- **Aumento do tamanho do banco de dados:** Devido à redundância, o banco de dados vai ocupar mais espaço de armazenamento.
- **Manutenção mais complexa:** Como o mesmo dado pode estar armazenado em vários lugares, a atualização desse dado precisa ser feita em todos os locais para manter a consistência, o que pode aumentar a complexidade da manutenção do banco de dados.

Resumindo, a desnormalização é uma técnica que pode ser usada para otimizar o desempenho de consultas em bancos de dados relacionais, mas deve ser usada com cuidado para evitar problemas com o tamanho e a manutenção do banco de dados.

Linguagem SQL Padrão ANSI

Existem inúmeras linguagens no mercado de TI, linguagens de programação orientadas a objeto, estruturadas, de marcação de texto entre outras. A linguagem voltada para banco de dados é a SQL (Linguagem de consulta estruturada, em português) é uma linguagem que todo programador, técnico ou administrador de banco de dados deve conhecer. Sua aplicação é extremamente ampla no mercado de banco de dados e programação.

A linguagem SQL (Structure query Language - Linguagem de Consulta Estruturada) é a linguagem padrão ANSI (American National Standards Institute - Instituto Nacional de Padronização Americano) para a operação em bancos de dados relacionais. A linguagem SQL foi criada para atender a todos os bancos de dados relacionais e permitir que usuários possam acessar qualquer banco usando a mesma base de conhecimento.

O padrão ANSI define os seguintes elementos da linguagem SQL:

- **Comandos básicos:** SELECT, INSERT, UPDATE, DELETE etc.
- **Funções:** Funções matemáticas, de agregação, de manipulação de strings etc.
- **Tipos de dados:** Inteiros, strings, datas etc.
- **Operadores:** Operadores aritméticos, lógicos, comparativos etc.
- **Restrições:** NOT NULL, UNIQUE, PRIMARY KEY etc.

Sublinguagens SQL

A linguagem SQL é composta por diversas sublinguagens que permitem realizar tarefas específicas:

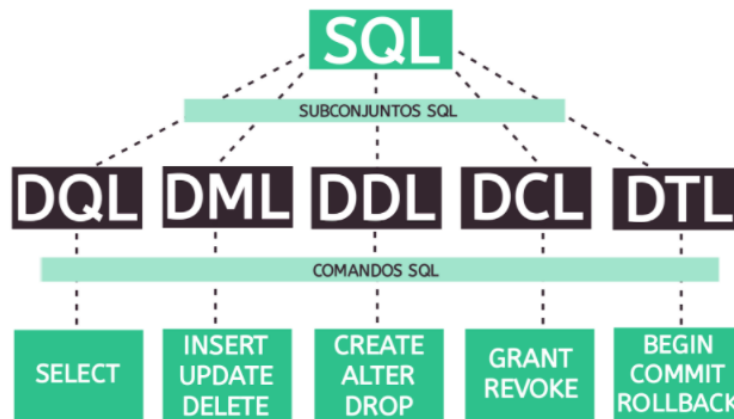
Data Definition Language (DDL): Define a estrutura do banco de dados, como tabelas, colunas e chaves.

Data Manipulation Language (DML): Permite manipular dados, como inserir, atualizar e excluir.

Data Query Language (DQL): Permite consultar dados, como selecionar e ordenar.

Data Control Language (DCL): Controla o acesso aos dados, como permissões e usuários.

Transaction Control Language (TCL): Controla transações, como commit e rollback.



O SQL Padrão ANSI define um conjunto básico de comandos e funcionalidades que todos os SGBDs devem suportar. As sublinguagens SQL estendem o padrão ANSI com funcionalidades específicas de cada SGBD.

Aprender o SQL Padrão ANSI é importante para:

- Escrever código SQL mais portátil: O código escrito em SQL Padrão ANSI pode ser executado em diferentes SGBDs sem necessidade de modificações significativas.
- Facilitar o aprendizado de sublinguagens: As sublinguagens SQL são extensões do padrão ANSI, portanto, aprender o padrão ANSI facilita o aprendizado das sublinguagens.
- Melhorar a compreensão da linguagem SQL: O padrão ANSI define a base da linguagem SQL, portanto, compreendê-lo ajuda a entender melhor a linguagem como um todo.

O SQL Padrão ANSI é a base da linguagem SQL e é importante para escrever código SQL mais portátil, facilitar o aprendizado de sublinguagens e melhorar a compreensão da linguagem como um todo.

Linguagem de definição de dados (DDL) com SGBDR

DDL significa Data Definition Language (Linguagem de Definição de Dados). É uma sublinguagem dentro da ampla linguagem SQL (Structured Query Language) usada para gerenciar a estrutura de um banco de dados relacional.

O propósito das instruções DDL é usadas para criar, modificar e remover objetos dentro do esquema de um banco de dados. Esses objetos incluem:

- **Tabelas:** A unidade fundamental de armazenamento de dados em um banco de dados relacional, consistindo em linhas e colunas.
- **Colunas:** Definem os atributos ou características dos dados armazenados em uma tabela, cada um com um tipo de dado específico (por exemplo, inteiro, string, data).
- **Restrições:** Regras que impõem integridade e consistência de dados dentro de uma tabela. Exemplos incluem chaves primárias (identificadores exclusivos para cada linha), chaves estrangeiras (relacionamentos entre tabelas) e restrições NOT NULL (garantindo que as colunas tenham valores).
- **Índices:** Estruturas de dados especializadas que melhoram o desempenho das operações de recuperação de dados (consultas SELECT).

Principais comandos DDL

- **CREATE TABLE:** Define a estrutura de uma nova tabela, especificando suas colunas, tipos de dados e restrições.
- **ALTER TABLE:** Modifica a estrutura de uma tabela existente, permitindo adicionar, remover ou modificar colunas, restrições e índices.
- **DROP TABLE:** Remove permanentemente uma tabela do banco de dados.

Criação de banco de dados

Sintaxe para criar uma base de dados:

CREATE DATABASE <NomeDataBase> [;]

- **CREATE:** é um comando SQL para criar objetos.
- **DATABASE:** é o objeto do tipo base de dados.
- <NomeDataBase>: é o nome que damos a base de dados.
- [;]: ponto e vírgula indica o término do comando.

Exemplo: **CREATE DATABASE** dbExemplo;

Símbolos em sintaxe

Para conseguirmos ler e compreender as sintaxes com mais facilidade precisamos entender o que significa alguns símbolos:

- < > substituir por nome
- [] opcional
- | separador de opções

Criação de objetos (tabelas, colunas, chaves e índices)

Criando tabelas

Comando **CREATE TABLE**: Define o nome da tabela, suas colunas e seus tipos de dados.

Exemplo:

```
CREATE TABLE clientes (  
  id INT PRIMARY KEY,  
  nome VARCHAR(255) NOT NULL,  
  email VARCHAR(255) UNIQUE  
);
```

- **id:** Inteiro, chave primária (identifica cada registro de forma única).
- **nome:** String com até 255 caracteres, não nulo (obrigatório).
- **email:** String com até 255 caracteres, único (cada email só pode estar em um registro).

Restrições

- **PRIMARY KEY:** Define a coluna como chave primária.
- **NOT NULL:** Indica que a coluna não pode ter valor nulo.
- **UNIQUE:** Indica que cada valor na coluna deve ser único.

Adicionando colunas

Comando **ALTER TABLE ADD COLUMN**: Adiciona uma nova coluna à tabela.

Exemplo:

```
ALTER TABLE clientes ADD COLUMN telefone VARCHAR(255);
```


Criando chaves

- **Chave primária:** Identifica cada registro de forma única.
- **Chave estrangeira:** Relaciona uma tabela a outra.

Exemplo:

```
CREATE TABLE pedidos (  
  id INT PRIMARY KEY,  
  cliente_id INT,  
  FOREIGN KEY (cliente_id) REFERENCES clientes(id)  
);
```

A tabela pedidos possui uma coluna cliente_id que referencia a coluna id da tabela clientes.

Criando índices

Melhoram o desempenho de consultas que filtram ou ordenam dados.

Exemplo:

```
CREATE INDEX idx_cliente_nome ON clientes (nome);
```

Cria um índice na coluna nome da tabela clientes.

Considerações

- Defina os nomes dos objetos de forma clara e concisa.
- Escolha os tipos de dados adequados para cada coluna.
- Utilize restrições para garantir a qualidade dos dados.
- Crie índices para otimizar o desempenho das consultas.

Lembre-se:

A sintaxe SQL pode variar de acordo com o SGBD que você está usando.
É importante consultar a documentação do seu SGBD para obter mais informações.

Alteração e exclusão de objetos

Drop - Excluindo uma base de dados

Sintaxe para excluir uma base de dados:

```
DROP DATABASE <Nome da Base> [;]
```

- **DROP:** é um comando SQL para excluir objetos.

Exemplo: **DROP DATABASE** dbExemplo;

O comando drop pode ser entendido como excluir, apagar, eliminar, e em banco de dados a palavra dropar é a mais utilizada. Todas querem dizer a mesma coisa, retirar o objeto em questão do SGBD.

Introdução ao SGBD SQL (Server)

Todo negócio retém informações relacionadas às suas atividades, que ajudam a entender mais sobre os clientes, sobre o mercado e sobre a própria empresa. O SQL Server da Microsoft é um sistema especializado em gerenciar esses registros, funcionando como uma plataforma completa de soluções.

Ele permite controlar a qualidade dos dados, desenvolver relatórios avançados e ainda é capaz de se integrar com diversas fontes. Tudo isso por meio de um sistema voltado a vários níveis de usuários.

Antes de investir nesse software é preciso entender melhor o funcionamento, sua proposta e os benefícios que traz às empresas. Veja a seguir!

Afinal, o que é a plataforma SQL?

Quando surgiu no mercado, o SQL Server da Microsoft era um simples Sistema de Gerenciamento de Banco de Dados (SGBD). Entretanto, seu antigo foco é somente uma das aplicações atuais.

Links para ajudar nos estudos

Link para estudos com Sql Server

https://www.youtube.com/watch?v=1YQIRdWkMvs&list=PLucm8g_ezqNqI5cW3alteV5olcMCcHYRK&index=1

O que é um servidor

<https://www.youtube.com/watch?v=4ilLf8VqLa4>

O que é um SGBD

<https://www.youtube.com/watch?v=NnG7f60XPkQ>

Como funciona um DATA CENTER

<https://www.youtube.com/watch?v=jQx6wItPuSo>

Modelagem de Dados - Modelos Conceitual, Lógico e Físico

<https://www.youtube.com/watch?v=ZX7EuRWRdZg&t=531s>

Modelo Entidade-Relacionamento (MER)

<https://www.youtube.com/watch?v=oPlXecD-gZM>

Referências para estudos de construção de banco de dados relacional

Livros:

Fundamentos de Banco de Dados, Elmasri & Navathe (7ª edição)
Projeto de Banco de Dados, Date (10ª edição)
SQL em 10 Minutos, Ben Forta (6ª edição)
Banco de Dados Relacionais: Um Enfoque Prático, Codd (2ª edição)
Modelagem de Dados, Teorey (3ª edição)

Tutorial online:

W3Schools: <https://www.w3schools.com/sql/default.asp>

Ferramenta:

SQL Server 2019 Developer Edition: <https://www.microsoft.com/en-us/sql-server/sql-server-downloads>

Comunidades online para SQL Server:

Stack Overflow: <https://stackoverflow.com/questions/tagged/sql>

Uma das maiores comunidades online para programadores e profissionais de TI.

Possui uma seção dedicada a perguntas e respostas sobre SQL Server, onde você pode encontrar soluções para problemas específicos e aprender com outras pessoas.

É importante seguir as diretrizes da comunidade para formular perguntas claras e objetivas.

Microsoft Docs Q&A: <https://learn.microsoft.com/en-us/sql/sql-server/?view=sql-server-ver16>

Documentação oficial do Microsoft SQL Server com uma seção de perguntas e respostas.

Local ideal para encontrar respostas a perguntas básicas e relacionadas à documentação oficial.

Você pode pesquisar perguntas existentes ou fazer a sua própria.

TechNet Forums (arquivado, mas ainda acessível): https://techcommunity.microsoft.com/t5/sql-server/bd-p/SQL_Server

Fóruns oficiais da Microsoft para o SQL Server (atualmente arquivado, mas ainda acessível para consulta).

Embora não haja novas postagens, esta é uma grande coleção de tópicos resolvidos que podem ser úteis para encontrar soluções para problemas conhecidos.