Professor: Ruan Chaves Machado

Data do Material Didático Criado: 19/05/2025

Data da Aula: 30/05/2025

Tópicos e Temas abordados nessa aula:

* Compreendendo Bancos de Dados: O Que São e Qual Sua Importância
* Sistemas de Gerenciamento de Banco de Dados (SGBD): A Ponte entre Aplicações e Dados
* Fundamentos da Modelagem de Dados: Como Organizar Informações de Forma Eficiente
* O que é o MySQL? Introdução ao Banco de Dados Relacional mais popular do mundo
* Tabela de Comandos de Consulta
* Tabela de Tipos do SQL

**Compreendendo Bancos de Dados: O Que São e Qual Sua Importância**

**O que é um banco de dados?**

Um **banco de dados** é uma estrutura organizada para **armazenar, gerenciar e recuperar informações** de forma eficiente. Ele permite que dados sejam mantidos de maneira persistente — ou seja, mesmo que o programa ou computador seja desligado, os dados continuam salvos.

Em essência, um banco de dados:

* **Armazena dados** de maneira estruturada (normalmente em tabelas, se for um banco relacional)
* **Permite acesso rápido e seguro** às informações
* **Garante integridade e consistência** dos dados
* Suporta operações de **inserção, leitura, atualização e exclusão** de dados (as famosas operações CRUD)

**Tipos de banco de dados**

Existem diferentes tipos de banco de dados, sendo os principais:

|  |  |
| --- | --- |
| Relacionais | Dados organizados em tabelas com linhas e colunas (ex: MySQL, PostgreSQL) |
| |  | | --- | | Não-relacionais |  |  | | --- | |  | | |  | | --- | |  |  |  | | --- | | Flexíveis, ideais para dados sem estrutura rígida (ex: MongoDB, Redis) | |
| |  | | --- | | Orientados a objetos |  |  | | --- | |  | | |  | | --- | |  |  |  | | --- | | Armazenam dados como objetos (menos comuns, ex: db4o) | |
| |  | | --- | | Em memória |  |  | | --- | |  | | |  | | --- | |  |  |  | | --- | | Armazenam dados na RAM para performance rápida (ex: Redis, SQLite em modo memória) | |

**Exemplo do mundo real: Sistema de Vendas**

Imagine uma empresa de e-commerce. Ela precisa armazenar:

* Produtos disponíveis
* Informações dos clientes
* Pedidos realizados
* Estoque de cada item

Essas informações podem ser representadas em tabelas como:

**Tabela: produtos**

|  |  |  |
| --- | --- | --- |
| ID | NOME | PRECO |
| 1 | Teclado Mecânico | 249.90 |
| 2 | Mouse Gamer | 129.90 |

Essas tabelas estão armazenadas em um banco de dados. O sistema pode buscar um cliente, mostrar os produtos, registrar uma nova venda e atualizar o estoque — tudo de forma rápida e estruturada.

**Para Memorizar:**

* Um **banco de dados** é uma ferramenta essencial para armazenar e gerenciar dados de forma persistente.
* Permite que as informações sejam organizadas, acessadas e modificadas com eficiência.
* Existem diferentes tipos de banco (relacional, NoSQL, em memória, etc.), cada um com finalidades específicas.
* Sem bancos de dados, aplicações modernas (sites, apps, sistemas financeiros) seriam quase impossíveis de manter com segurança e escalabilidade.

**Sistemas de Gerenciamento de Banco de Dados (SGBD): A Ponte entre Aplicações e Dados**

O que é um SGBD?

Um SGBD (Sistema de Gerenciamento de Banco de Dados) é um software responsável por controlar, organizar, manipular e proteger os dados armazenados em um banco de dados. Ele atua como uma camada intermediária entre a aplicação (como um sistema web, por exemplo) e os dados propriamente ditos.

Imagine que você tem milhares de registros de clientes, pedidos e produtos. Você precisa inserir, consultar, alterar e excluir essas informações — mas com controle de acesso, desempenho, consistência e segurança. É aí que entra o SGBD.

| **Função** | **Descrição** |
| --- | --- |
| **Armazenamento de Dados** | Gerencia como e onde os dados são fisicamente armazenados no disco. |
| **Acesso e Consulta** | Permite a busca eficiente dos dados através da linguagem SQL. |
| **Controle de Acesso** | Define permissões de leitura/escrita para usuários diferentes. |
| **Confiabilidade e Segurança** | Protege os dados contra falhas, corrupção ou acessos indevidos. |
| **Transações e Consistência** | Garante que operações ocorram de forma completa e correta (ACID). |
| **Backup e Recuperação** | Possibilita salvar cópias e restaurar dados em caso de falhas. |

**O que é ACID?**

Conjunto de propriedades que garantem a integridade das transações em bancos de dados:

* **Atomicidade**: tudo ou nada é executado.
* **Consistência**: garante que o banco vá de um estado válido a outro.
* **Isolamento**: transações simultâneas não se interferem.
* **Durabilidade**: uma vez concluída, a transação não é desfeita, mesmo com falhas.

**Exemplos práticos**

**1. Exemplo simples: usando MySQL com Python**

Um sistema de pedidos faz a seguinte operação:

* O usuário faz uma compra
* O sistema:
  + Registra o pedido
  + Atualiza o estoque
  + Gera uma nota fiscal

Essas três ações devem ocorrer **juntas** — se uma falhar, nenhuma deve ser concluída. O SGBD (como o MySQL) garante isso através de **transações**.

**2. Interagindo com um SGBD via terminal**

-- Criar banco e tabela

**CREATE** **DATABASE** loja**;**

**USE** loja**;**

**CREATE** **TABLE** produtos **(**

id INT AUTO\_INCREMENT **PRIMARY** **KEY,**

nome VARCHAR**(**100**),**

preco DECIMAL**(**10**,**2**)**

**);**

-- Inserir dados

**INSERT** **INTO** produtos **(**nome**,** preco**)** **VALUES** **(**'Notebook'**,** 3200.00**);**

**SELECT** **\*** **FROM** produtos**;**

Esses comandos são processados por um SGBD como o **MySQL**, **PostgreSQL** ou **SQLite**.

**Exemplos de SGBDs Populares:**

|  |  |  |
| --- | --- | --- |
| Relacionais | MySQL, PostgreSQL, SQLite | Organizam dados em tabelas; usam SQL |
| NoSQL | MongoDB, Firebase | Estrutura flexível (documentos, chave-valor) |
| |  | | --- | | Em memória |  |  | | --- | |  | | Redis | Alta velocidade, ideal para cache |
| Corporativos | Oracle, Microsoft SQL Server | Robustos, com recursos avançados |

**Para memorizar:**

* O **SGBD** é o software responsável por **gerenciar os dados**, mantendo controle sobre acesso, transações e segurança.
* Ele **separa a aplicação dos dados**, oferecendo uma interface para consultas e modificações, normalmente através da linguagem SQL.
* SGBDs garantem integridade, desempenho e escalabilidade — requisitos essenciais em sistemas reais.
* Cada tipo de SGBD possui suas **características e aplicações ideais**.

**Fundamentos da Modelagem de Dados: Como Organizar Informações de Forma Eficiente**

## O que é modelagem de dados?

A **modelagem de dados** é o processo de **estruturar logicamente os dados que um sistema irá armazenar e como eles se relacionam entre si**. Ela é essencial antes mesmo de começar a criar tabelas no banco de dados.

**Pense nisso como o projeto arquitetônico de uma casa**: você não começa construindo sem saber onde ficarão os cômodos, portas e janelas. Na modelagem, definimos onde e como os dados vão "morar" dentro do sistema.

### Por que modelar dados?

* Para **evitar redundâncias**
* Para **aumentar a consistência e integridade** dos dados
* Para **facilitar consultas, relatórios e manutenções**
* Para garantir que os dados **representem a realidade do negócio**

## Elementos da modelagem

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Entidade | |  | | --- | |  |  |  | | --- | | Um "objeto" do mundo real que queremos representar | | Cliente, Produto, Pedido |
| Atributo | |  | | --- | |  |  |  | | --- | | Uma informação que descreve a entidade | | Nome, CPF, Preço |
| Relacionamento | |  | | --- | |  |  |  | | --- | | Como duas entidades estão conectadas | | Cliente faz Pedido |

### Modelos mais comuns

* **Modelo Conceitual (DER)**  
  É o primeiro desenho, mais abstrato. Usa **diagramas entidade-relacionamento**.
* **Modelo Lógico**  
  Já começa a considerar os tipos de dados e chaves primárias/estrangeiras.
* **Modelo Físico**  
  É a tradução final para o que será implementado no SGBD.

### Analogia prática: Cadastro de uma escola

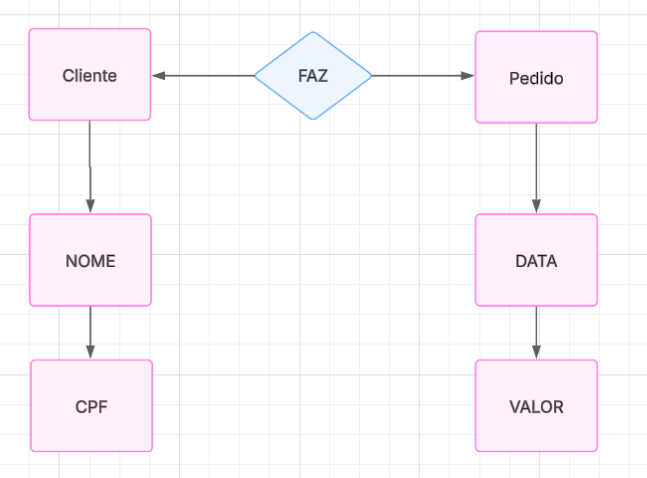
Imagine que você está organizando fichas em uma escola usando pastas e etiquetas.

* Cada **pasta** é uma **entidade**: Aluno, Professor, Turma.
* Cada **informação dentro da pasta** é um **atributo**: Nome, CPF, Disciplina.
* As **ligações entre pastas** são os **relacionamentos**:  
  Exemplo: Um Professor pode dar aula em várias Turmas.

Se você tentar guardar essas informações aleatoriamente em folhas soltas, sem estrutura, rapidamente se perderá. O banco de dados é esse **arquivo organizado com estrutura**, e a modelagem é o **planejamento de como ele será montado**.

### Exemplos práticos

### Exemplo de DER simples (Diagrama Entidade Relacionamento)



### Como isso vira um modelo relacional (SQL):

-- Inserir dados

**INSERT** **INTO** produtos **(**nome**,** preco**)** **VALUES** **(**'Notebook'**,** 3200.00**);**

**SELECT** **\*** **FROM** produtos**;**

**CREATE** **TABLE** cliente **(**

id INT **PRIMARY** **KEY** AUTO\_INCREMENT**,**

nome VARCHAR**(**100**),**

cpf VARCHAR**(**14**)**

**);**

**CREATE** **TABLE** pedido **(**

id INT **PRIMARY** **KEY** AUTO\_INCREMENT**,**

**data** DATE**,**

valor DECIMAL**(**10**,**2**),**

cliente\_id INT**,**

**FOREIGN** **KEY** **(**cliente\_id**)** **REFERENCES** cliente**(**id**)**

**);**

### Para memorizar:

* Modelagem de dados é o **planejamento lógico de como os dados serão organizados e inter-relacionados** no sistema.
* Trabalhamos com conceitos como **entidades**, **atributos** e **relacionamentos**.
* Um bom modelo evita erros como **dados duplicados**, facilita a manutenção e garante integridade.
* A modelagem é uma etapa **fundamental antes de criar um banco de dados real**.

**O que é o MySQL? Introdução ao Banco de Dados Relacional mais popular do mundo**

O **MySQL** é um **SGBD relacional** (Sistema de Gerenciamento de Banco de Dados Relacional) de **código aberto**, usado para criar, armazenar, gerenciar e consultar bancos de dados. Ele armazena os dados em **tabelas**, assim como o Excel — mas com muito mais poder, segurança e flexibilidade.

O nome "MySQL" vem da junção de “**My**” (o nome da filha do cofundador Michael Widenius) e “**SQL**” (Structured Query Language, a linguagem usada para interagir com os dados).

Ele é utilizado por grandes empresas como:

* Facebook
* Twitter
* GitHub
* YouTube

É um dos bancos de dados mais usados **em aplicações web**, devido à sua estabilidade, velocidade e compatibilidade com diversas linguagens, como **Python, PHP, Java, C#**, entre outras.

**Conceito de Banco de Dados Relacional**

MySQL organiza os dados em **tabelas** que se relacionam entre si por meio de **chaves primárias e estrangeiras**, seguindo o modelo relacional proposto por Edgar Codd na década de 70.

**Um pouco da história**

| **Ano** | **Evento** |
| --- | --- |
| 1995 | Primeira versão pública lançada por **David Axmark**, **Allan Larsson** e **Michael Widenius** |
| 2008 | A Sun Microsystems comprou o MySQL |
| 2010 | A Oracle comprou a Sun e passou a controlar o MySQL |
| Atualmente | O MySQL possui **versões gratuitas** (Community) e **pagas** (Enterprise), sendo amplamente usado no mundo todo |

**Principais características**

| **Característica** | **Descrição** |
| --- | --- |
| Código aberto | Gratuito para uso geral |
| Multiplataforma | Funciona no Windows, Linux, macOS |
| Suporte a transações | ACID-compliant (com InnoDB) |
| Performance e escalabilidade | Usado em sistemas com milhões de registros |
| Conectividade | Fácil integração com Python, PHP, Java e mais |

**Principais Comandos de Consulta com o SQL:**

| **Comando SQL** | **Descrição** | **Exemplo de uso** |
| --- | --- | --- |
| CREATE DATABASE | Cria um novo banco de dados | CREATE DATABASE escola; |
| DROP DATABASE | Remove um banco de dados | DROP DATABASE escola; |
| USE | Define qual banco de dados será usado | USE escola; |
| CREATE TABLE | Cria uma nova tabela | CREATE TABLE alunos (...); |
| DROP TABLE | Exclui uma tabela | DROP TABLE alunos; |
| ALTER TABLE | Modifica a estrutura de uma tabela | ALTER TABLE alunos ADD COLUMN email VARCHAR(100); |
| RENAME TABLE | Renomeia uma tabela | RENAME TABLE alunos TO estudantes; |
| INSERT INTO | Insere novos dados em uma tabela | INSERT INTO alunos (nome, idade) VALUES ('Maria', 22); |
| SELECT | Recupera dados da tabela | SELECT \* FROM alunos; |
| WHERE | Filtra resultados de uma consulta | SELECT \* FROM alunos WHERE idade > 18; |
| UPDATE | Atualiza dados existentes | UPDATE alunos SET idade = 23 WHERE nome = 'Maria'; |
| DELETE | Remove dados de uma tabela | DELETE FROM alunos WHERE nome = 'Maria'; |
| ORDER BY | Ordena os resultados | SELECT \* FROM alunos ORDER BY idade DESC; |
| GROUP BY | Agrupa resultados iguais | SELECT idade, COUNT(\*) FROM alunos GROUP BY idade; |
| HAVING | Filtra grupos agregados (usado com GROUP BY) | SELECT idade, COUNT(\*) FROM alunos GROUP BY idade HAVING COUNT(\*) > 1; |
| JOIN | Junta dados de duas tabelas | SELECT \* FROM alunos JOIN cursos ON alunos.curso\_id = cursos.id; |
| INNER JOIN | Retorna apenas os registros que têm correspondência em ambas as tabelas | SELECT \* FROM alunos INNER JOIN cursos ON alunos.curso\_id = cursos.id; |
| LEFT JOIN | Retorna todos os registros da tabela da esquerda, e correspondentes da direita | SELECT \* FROM alunos LEFT JOIN cursos ON alunos.curso\_id = cursos.id; |
| RIGHT JOIN | Retorna todos os registros da tabela da direita, e correspondentes da esquerda | SELECT \* FROM alunos RIGHT JOIN cursos ON alunos.curso\_id = cursos.id; |
| UNION | Combina resultados de múltiplos SELECTs | SELECT nome FROM alunos UNION SELECT nome FROM professores; |
| LIMIT | Limita o número de resultados | SELECT \* FROM alunos LIMIT 10; |
| DISTINCT | Elimina duplicatas nos resultados | SELECT DISTINCT idade FROM alunos; |

**Tipos de Dados em SQL:**

| **Tipo SQL** | **Descrição** | **Exemplo de uso** |
| --- | --- | --- |
| INT | Número inteiro | idade INT |
| BIGINT | Número inteiro maior (até 64 bits) | populacao BIGINT |
| DECIMAL(p,s) | Número com precisão exata (p = total dígitos, s = casas decimais) | salario DECIMAL(10,2) |
| FLOAT | Número com ponto flutuante | altura FLOAT |
| DOUBLE | Número com ponto flutuante com mais precisão | media DOUBLE |
| CHAR(n) | Texto fixo de n caracteres | sigla CHAR(2) |
| VARCHAR(n) | Texto com até n caracteres (mais eficiente que CHAR) | nome VARCHAR(100) |
| TEXT | Texto longo (até 65.535 caracteres) | biografia TEXT |
| DATE | Data no formato YYYY-MM-DD | data\_nascimento DATE |
| DATETIME | Data e hora combinadas | criado\_em DATETIME |
| TIMESTAMP | Data e hora com fuso horário UTC (usado para logs, auditorias) | log\_criado TIMESTAMP |
| TIME | Hora no formato HH:MM:SS | hora\_entrada TIME |
| BOOLEAN / TINYINT(1) | Representa verdadeiro/falso (em MySQL, geralmente 0 = False, 1 = True) | ativo BOOLEAN |
| ENUM('a', 'b') | Aceita apenas valores definidos | sexo ENUM('M', 'F', 'Outro') |
| BLOB | Armazena dados binários (imagens, arquivos, etc.) | foto BLOB |
| JSON | Armazena dados no formato JSON (a partir do MySQL 5.7) | config JSON |