

Introdução a Banco de dados

O que é SQL?

Introdução a Banco de dados

SQL (Structured Query Language, ou Linguagem de Consulta Estruturada em português) é a linguagem padrão utilizada para interagir com bancos de dados relacionais.

Introdução a Banco de dados

História do SQL

Introdução a Banco de dados

A SQL foi criada pela IBM na década de 1970 como uma solução para simplificar a interação com sistemas de gerenciamento de bancos de dados relacionais (SGBDs). Seu desenvolvimento teve como base a necessidade de oferecer uma abordagem mais prática e intuitiva para acessar e manipular dados armazenados nesses sistemas. Surgiu no contexto do projeto System R, uma iniciativa da IBM para desenvolver um sistema de banco de dados relacional. O conceito de banco de dados relacional foi introduzido por Edgar F. Codd em 1970, propondo o uso de tabelas para armazenar dados e relacioná-los por meio de chaves primárias e estrangeiras. Diante dessa inovação, surgiu a necessidade de criar uma linguagem que possibilitasse consultas eficientes e amigáveis para os usuários.

A SQL foi idealizada para ser uma linguagem declarativa, permitindo que o usuário defuisse o que desejava realizar sem se preocupar com os detalhes técnicos de implementação

Introdução a Banco de dados

Após sua criação, a SQL foi gradualmente adotada por diversos sistemas de banco de dados, o que levou à criação de padrões para garantir sua consistência e interoperabilidade. Em 1986, a linguagem foi formalmente padronizada pelo ANSI (American National Standards Institute) e pela ISO (International Organization for Standardization). Desde então, a SQL passou por várias revisões e atualizações que incorporaram novas funcionalidades e ajustes na sintaxe.

Introdução a Banco de dados

SQL-86: Padrão Oficial: SELECT, INSERT, UPDATE, DELETE

SQL-92: Consolidação e Expansão

SQL-1999: Programação Orientada a Objetos

SQL-2003: Dados Não Estruturados e Funções Analíticas

SQL-2008: integração com dados externos e maior compatibilidade com SGBDs em nuvem

Introdução a Banco de dados

SQL-2011: Interoperabilidade com Tecnologias Emergentes

SQL-2016: Big Data e Nuvem

SQL-2017: Dados Temporais

SQL-2019: Dados JSON e Ferramentas Analíticas

SQL-2022:Novos Paradigmas Analíticos

Introdução a Banco de dados

O que é um Banco de dados?

Introdução a Banco de dados

Em um sentido mais amplo, qualquer ferramenta que colete e organize dados, como planilhas e arquivos de texto, pode ser considerada uma forma básica de armazenamento de dados. No entanto, quando falamos de bancos de dados no contexto da tecnologia da informação, estamos nos referindo a sistemas mais complexos e robustos, como os Sistemas de Gerenciamento de Banco de Dados Relacionais (SGBDRs ou RDBMSs em inglês). Esses sistemas são projetados especificamente para armazenar, organizar e gerenciar grandes volumes de dados de forma eficiente e segura, utilizando uma estrutura baseada em tabelas relacionadas.

Introdução a Banco de dados

A imagem abaixo exemplifica o funcionamento do relacionamento entre o cliente, a aplicação SGBD e o banco de dados:



Introdução a Banco de dados

Características Principais de um SGBD:

Independência de dados

Integridade dos dados

Concorrência

Segurança

Backup e recuperação

Introdução a Banco de dados

Tipos de Banco de Dados

Introdução a Banco de dados

Os sistemas de gerenciamento de bancos de dados (SGBDs) podem ser categorizados em relacionais (SQL) e não relacionais (NoSQL).

Introdução a Banco de dados

Bancos de Dados Relacionais:

Os bancos de dados relacionais organizam as informações em tabelas estruturadas por colunas e linhas, utilizando o modelo relacional para definir conexões entre os dados por meio de chaves primárias e estrangeiras. Esses bancos empregam a linguagem SQL (Structured Query Language) para realizar a gestão, manipulação e execução de consultas sobre os dados, oferecendo eficiência, integridade e consistência no armazenamento de informações.

Bancos de Dados Relacionais



Bancos de Dados Relacionais

MySQL:

Amplamente utilizado no desenvolvimento de aplicações web devido à sua robustez, escalabilidade e desempenho confiável.

Exemplos de uso: Plataformas como Facebook, YouTube, Twitter e o sistema de gerenciamento de conteúdo WordPress utilizam o MySQL como seu banco de dados principal.

Bancos de Dados Relacionais

PostgreSQL:

Reconhecido por sua robustez, escalabilidade e recursos avançados, é ideal para sistemas corporativos e aplicações financeiras.

Exemplos: Plataformas de e-commerce, sistemas de gestão empresarial, aplicações financeiras e soluções de análise de dados em larga escala.

Bancos de Dados Relacionais

Oracle Database:

Amplamente adotado por grandes corporações devido ao seu alto desempenho, segurança avançada e recursos robustos de gerenciamento de dados.

Exemplos: Sistemas de planejamento de recursos empresariais (ERP), aplicações financeiras, soluções de BI (Business Intelligence) e plataformas de gestão de grandes volumes de dados críticos.

Bancos de Dados Relacionais

Microsoft SQL Server:

Destaca-se por sua integração nativa com o ecossistema da Microsoft, oferecendo alto desempenho em sistemas corporativos e aplicações empresariais.

Exemplos: Aplicações que utilizam o Microsoft Azure, soluções de ERP, ferramentas de BI (Business Intelligence) e plataformas corporativas com integração Microsoft 365.

Bancos de Dados Relacionais

SQLite:

Um banco de dados leve, simples e de fácil implementação, ideal para aplicações que exigem baixo consumo de recursos.

Comumente utilizado em aplicativos móveis, dispositivos embarcados e cenários onde a portabilidade e a eficiência são prioridades.

Exemplos: Aplicativos móveis (iOS e Android), dispositivos IoT e pequenos sistemas que demandam armazenamento local sem servidor dedicado.

Introdução a Banco de dados

Bancos de Dados Não Relacionais (NoSQL):

Os bancos de dados não relacionais, conhecidos como NoSQL (Not Only SQL), não utilizam tabelas com estruturas rígidas e relacionamentos entre elas. Eles são mais flexíveis e são frequentemente usados para armazenar grandes volumes de dados variados, dados não estruturados ou semi-estruturados.

Os bancos não relacionais se destacam por sua escalabilidade, alta disponibilidade e por suportarem dados em formatos variados, como JSON ou documentos.

Bancos de Dados Não Relacionais



Graph Database 4.1



Amazon
DynamoDB

Bancos de Dados Não Relacionais

MongoDB (orientado a documentos):

Armazena dados no formato JSON: O MongoDB utiliza documentos no formato JSON para armazenar informações. Esses documentos são estruturas flexíveis, semelhantes a objetos JavaScript, que permitem armazenar dados com variações nas estruturas, sem a necessidade de um esquema fixo.

Exemplo prático: Aplicações que precisam armazenar dados com estruturas variáveis, como catálogos de produtos (onde cada produto pode ter atributos diferentes, como tamanho, cor ou categoria).

Bancos de Dados Não Relacionais

Cassandra (orientado a colunas):

Armazena dados em formato de colunas em vez de tabelas tradicionais: O Cassandra organiza as informações em estruturas de linhas e colunas, mas cada linha pode ter um conjunto diferente de colunas, o que proporciona flexibilidade para armazenar grandes volumes de dados distribuídos de maneira eficiente.

Exemplo prático: Aplicações que exigem alta escalabilidade e desempenho em operações de leitura e gravação, como sistemas de análise de dados em tempo real ou registros de logs em larga escala.

Bancos de Dados Não Relacionais

Redis (base de dados em memória):

Armazena dados na memória RAM para acesso rápido: O Redis é uma base de dados que mantém as informações em memória, proporcionando tempos de acesso extremamente rápidos em comparação com bancos de dados tradicionais baseados em disco.

Exemplo prático: Aplicações que exigem cache de dados para melhorar o desempenho, sessões de usuário em aplicações web ou contadores em tempo real, como análises de métricas de acesso.

Bancos de Dados Não Relacionais

Neo4j (Banco de Dados Orientado a Grafos) :

Armazena dados na forma de grafos com nós, arestas e propriedades: O Neo4j é projetado para modelar e armazenar relações complexas entre entidades de forma eficiente, utilizando grafos como estrutura principal. Isso facilita consultas e análises relacionadas a conexões e redes.

Exemplo prático: Aplicações como redes sociais (para mapear conexões entre usuários), sistemas de recomendação (como sugerir produtos com base no comportamento do usuário) e análise de fraudes em transações financeiras, onde identificar padrões de conexões é fundamental

Bancos de Dados Não Relacionais

Amazon DynamoDB (Banco de Dados NoSQL Gerenciado):

É um banco de dados NoSQL totalmente gerenciado pela AWS, com alta escalabilidade e baixa latência: O DynamoDB armazena dados em um modelo chave-valor ou documento, oferecendo desempenho consistente sem a necessidade de administração de infraestrutura.

Exemplo prático: Aplicações que exigem escalabilidade automática e alta disponibilidade, como sistemas de e-commerce, jogos online com armazenamento de perfis de usuários e catálogos de produtos em tempo real.

Introdução a Banco de dados

Após explorarmos os conceitos fundamentais sobre SQL e compreendermos a diferença entre bancos de dados relacionais e não relacionais, agora avançaremos para um tema essencial no processo de desenvolvimento de sistemas de banco de dados: A modelagem.

Modelagem de Banco de Dados.

A modelagem de bancos de dados é um processo fundamental para a criação de sistemas de informação eficientes e robustos. Existem três níveis principais de abstração na modelagem de dados: conceitual, lógico e físico.

Modelagem de Banco de Dados.

A modelagem conceitual: Representa a visão de alto nível do sistema, focando nos conceitos do negócio e nas relações entre eles. É como um esboço inicial, onde identificamos as entidades (objetos do mundo real, como clientes, produtos, pedidos) e os relacionamentos entre essas entidades (por exemplo, um cliente pode fazer muitos pedidos). O objetivo principal é capturar os requisitos de negócio e estabelecer uma base sólida para as etapas seguintes da modelagem.

Modelagem de Banco de Dados.

A modelagem lógica detalha a estrutura dos dados de forma mais precisa, traduzindo o modelo conceitual para uma representação mais próxima do sistema de gerenciamento de banco de dados (SGBD). Nesta etapa, as entidades são mapeadas para tabelas, os atributos para colunas e os relacionamentos para chaves estrangeiras. A modelagem lógica define as regras de integridade e as restrições dos dados.

Modelagem de Banco de Dados.

A modelagem física é a representação mais detalhada do banco de dados, considerando os aspectos físicos de implementação. Define índices, partições, alocação de espaço em disco e outros detalhes específicos do SGBD. O objetivo é otimizar o desempenho do banco de dados, considerando fatores como volume de dados, frequência de acesso e tipo de operações.

Modelagem lógica.

O **Modelo Entidade-Relacionamento (MER)** é uma técnica de modelagem de dados que representa graficamente as entidades (objetos), seus atributos (características) e os relacionamentos (vínculos) entre elas. Essa representação facilita a compreensão da estrutura lógica dos dados e a comunicação entre analistas, desenvolvedores e usuários. O MER permite identificar redundâncias, inconsistências e otimizar a organização dos dados, resultando em um banco de dados mais eficiente e flexível.

Modelagem Lógica.

O Modelo Entidade-Relacionamento (ER) é composto por três elementos principais: Entidades, Atributos e Relacionamentos. Esses elementos representam, de forma estruturada, as informações e suas conexões dentro de um sistema, servindo como base para o projeto de banco de dados.

Modelagem Lógica.

Entidade: Representam os objetos ou conceitos do mundo real que possuem relevância para o sistema, como clientes, produtos ou pedidos. Por convenção ao descrever uma Entidade:
Nome único, no singular e em caixa alta.

Modelagem Lógica.

Atributos: Características ou propriedades associadas a cada entidade, como o nome do cliente, o código do produto ou a data do pedido. Por convenção ao descrever um atributo:

Nome no singular, caixa baixa identificador único '#' e atributos obrigatórios marcados com ''**

Modelagem Lógica.

Relacionamentos: Representam as associações entre as entidades, definindo como elas interagem e se conectam no contexto do sistema. Por convenção ao descrever um relacionamento: nome com verbo.

Modelagem Lógica.

Cardinalidade: Define a quantidade de instâncias que podem estar associadas entre duas entidades em um relacionamento. Temos três tipos de cardinalidade sendo elas: 1:1, 1:N e N:N

Modelagem Lógica.

1:1 (Um para Um): Uma instância de uma entidade está associada a apenas uma instância de outra entidade, e vice-versa.

Exemplo:

Cada aluno possui apenas uma matricula.

Cada matricula pertence apenas a um aluno.

Modelagem Lógica.

1:N (Um para Muitos): Uma instância de uma entidade está associada a várias instâncias de outra entidade, mas cada instância da outra entidade está associada a apenas uma instância da primeira.

Exemplo:

Um usuário pode realizar vários empréstimos, mas cada empréstimo pertence a apenas um usuário.

Modelagem Lógica.

N:N (Muitos para Muitos): Várias instâncias de uma entidade podem estar associadas a várias instâncias de outra entidade.

Exemplo:

Um aluno pode se matricular em vários cursos.

Um curso pode ter vários alunos matriculados.

Modelagem Lógica.

Identificador Único (UID): Ele representa um atributo especial que garante a unicidade de cada instância de uma entidade. Em outras palavras, o UID é como um “CPF” para cada registro em um banco de dados, assegurando que não haja duplicações.

Modelagem Lógica.

Exemplo de (MER):

Vamos considerar um sistema de biblioteca com as entidades Usuário, Livro, e Empréstimo.

Modelo Entidade Relacionamento.

Entidade:

Usuário

Atributos:

ID, Nome, Endereço

Relacionamentos:

Usuário (1:N) → Empréstimo

Livro (1:N) → Empréstimo

Entidade:

Livro

Atributos:

ISBN, Título, Autor

Entidade:

Empréstimo

Atributos:

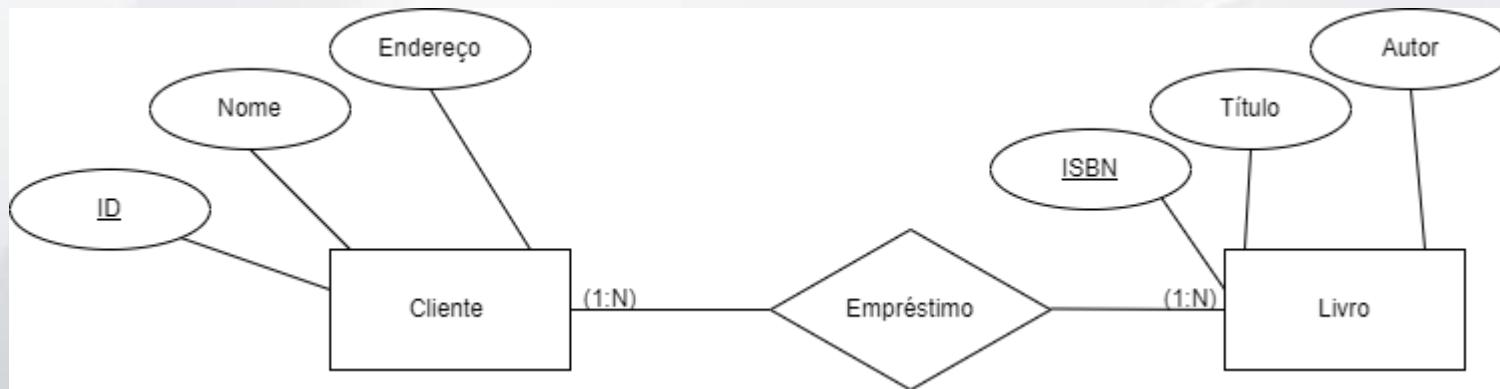
**Data_Empréstimo,
Data_Devolução**

Modelagem Física.

Ao passarmos para a modelagem lógica, daremos vida ao nosso modelo conceitual através de diagramas. Nesses diagramas, as entidades serão representadas por retângulos, os atributos por elipses e os relacionamentos por linhas que conectam as entidades. A cardinalidade, que indica o número de ocorrências permitidas em cada relacionamento, será representada por números ou símbolos específicos ao lado das linhas.

Modelagem Física.

Exemplo Vamos considerar um sistema de biblioteca com as entidades Usuário, Livro, e Empréstimo (DER):



Referências Bibliográficas

Elmasri, R.; Navathe, S. B. Sistemas de Banco de Dados. 7. ed.
São Paulo: Pearson, 2019.

NIELD, Thomas. Introdução à Linguagem SQL: Abordagem Prática Para Iniciantes. 1. ed. São Paulo: Novatec Editora, 26 abr. 2016.

KLINE, Kevin E.; KLINE, Daniel. SQL: O Guia Essencial - Manual de Referência Profissional. 3. ed. Rio de Janeiro: Alta Books, 15 set. 2010.