CONTINUAÇÃO DE BANCO DE DADOS

Banco de dados relacional

Um banco de dados relacional é um tipo de sistema de gerenciamento de banco de dados (SGBD) que armazena informações em tabelas compostas por linhas e colunas. Cada tabela é composta por registros que contêm informações específicas e são identificados por chaves primárias.

As tabelas em um banco de dados relacional são conectadas por meio de relacionamentos que estabelecem links entre as informações contidas em diferentes tabelas. Esses relacionamentos podem ser de um para um, de um para muitos ou de muitos para muitos.

Os bancos de dados relacionais são amplamente utilizados em aplicativos comerciais e empresariais porque permitem a fácil organização e gerenciamento de grandes quantidades de dados, garantindo integridade, consistência e segurança dos dados. Eles também permitem que os dados sejam acessados e manipulados de maneira eficiente e flexível usando linguagens de consulta padrão, como SQL (Structured Query Language).

Em um banco de dados, o relacionamento é uma associação entre duas ou mais tabelas que compartilham informações relacionadas. Os relacionamentos são estabelecidos para permitir que as informações sejam organizadas e armazenadas de maneira mais eficiente e para garantir a integridade dos dados.

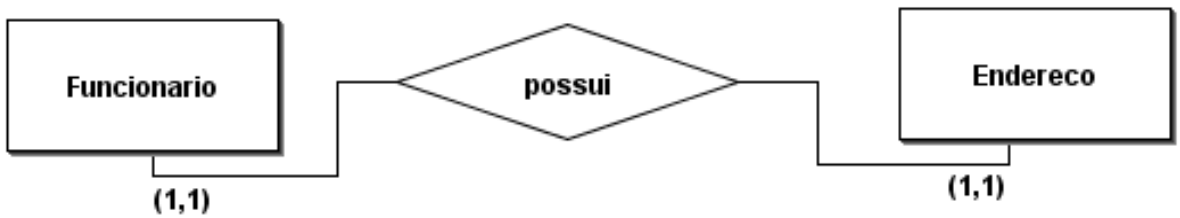
Entidade-Relacionamento (ER)

Entidade-Relacionamento (ER) é um modelo de dados amplamente utilizado na área de banco de dados. Foi proposto originalmente por Peter Chen em 1976, em seu artigo "The Entity-Relationship Model - Toward a Unified View of Data" (O Modelo Entidade-Relacionamento - Rumo a uma Visão Unificada de Dados, em tradução livre), publicado na revista ACM Transactions on Database Systems.

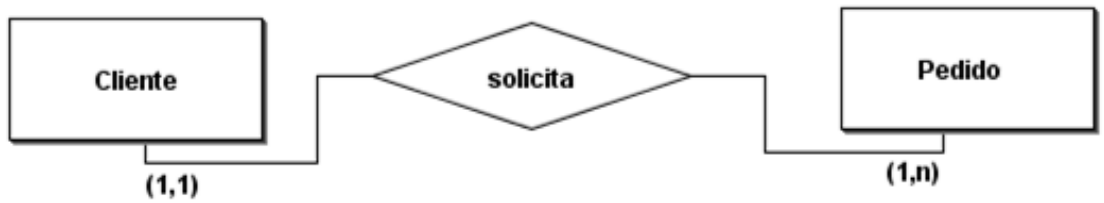
Desde então, o modelo ER tem sido amplamente utilizado em aplicações empresariais e sistemas de gerenciamento de informações. Uma das principais vantagens do modelo ER é sua capacidade de representar de forma clara e precisa as relações entre as entidades de um sistema.

Existem três tipos principais de relacionamentos em um banco de dados relacional:

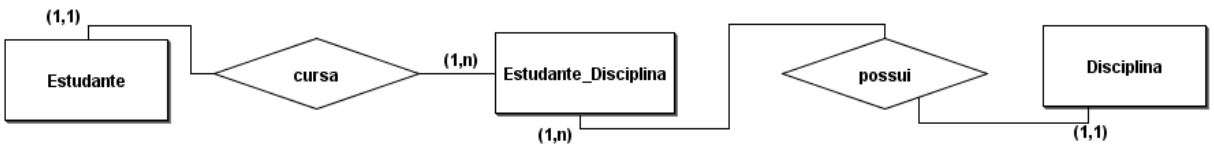
1. Relacionamento um para um (1:1): ocorre quando uma linha em uma tabela está associada a uma única linha em outra tabela e vice-versa. Por exemplo, uma tabela de funcionários pode estar relacionada a uma tabela de endereços, onde cada funcionário tem um endereço único.



2. Relacionamento um para muitos (1:N): ocorre quando uma linha em uma tabela está associada a várias linhas em outra tabela, mas as linhas na segunda tabela estão associadas a apenas uma linha na primeira tabela. Por exemplo, uma tabela de clientes pode estar relacionada a uma tabela de pedidos, onde cada cliente pode fazer vários pedidos, mas cada pedido está associado a um único cliente.



3. Relacionamento muitos para muitos (N:N): ocorre quando várias linhas em uma tabela estão associadas a várias linhas em outra tabela. Esse tipo de relacionamento requer uma terceira tabela intermediária, chamada tabela de junção, para registrar as associações entre as duas tabelas. Por exemplo, uma tabela de estudantes pode estar relacionada a uma tabela de disciplinas, onde cada estudante pode ter muitas disciplinas e cada disciplina pode ter muitos estudantes.



Ao estabelecer relacionamentos entre tabelas, é possível criar consultas complexas que recuperam informações de várias tabelas ao mesmo tempo, garantindo a integridade dos dados e evitando redundância e inconsistência.

Banco de dados não relacional: nessa estrutura de banco de dados, todos os dados armazenados nesse banco de dados estão em uma única tabela e guardados em um único computador, ou seja, simplesmente arquivos isolados.

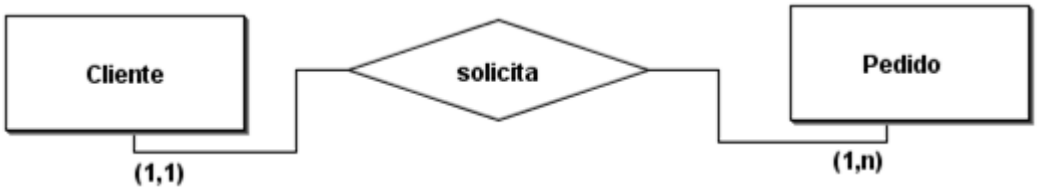
Modelos de Diagrama Entidade Relacionamento (E.R.)

Existem três modelos de diagrama Entidade-Relacionamento (E.R.) que são comumente utilizados:

- Conceitual
- Lógico
- Físico

Modelo conceitual: também chamado de modelos abstrato, descreve as estruturas do banco graficamente, a partir de uma análise de uma realidade que necessita de um banco de dados. Para representar esse modelo, é utilizada a abordagem entidade-relacionamento, representado pelo DER. Essa abordagem faz uma descrição gráfica de elementos do banco de dados e como as tabelas relacionam-se. Conforme Oliveira (2000, p.10), “baseia-se na percepção de mundo real, que consiste em uma coleção de objetos básicos, chamados entidades, e em relacionamentos entre esses objetos.”

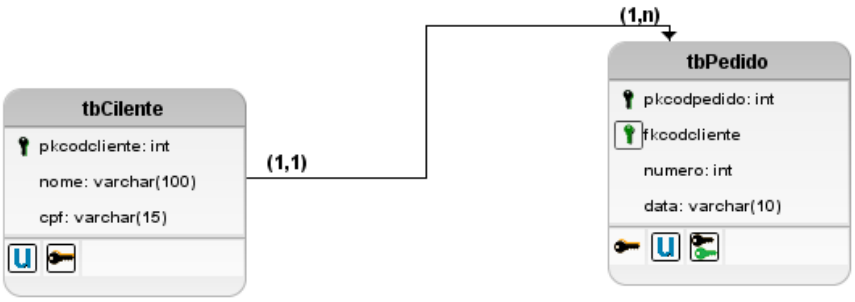
Exemplo de diagrama ER modelo conceitual



Modelo lógico são construídos baseados no modelo conceitual. Nesse

modelo são levadas em conta regras de derivação e restrições. As regras de derivações atendem aos processos de normalização, agregação, generalização/especialização, entre outros processos. Já as regras de restrições, atendem aos processos restrições de domínio, implementação e integridade. Conforme Date (2004), as restrições ligadas à integridade foram as que mais evoluíram nos últimos anos. Inicialmente estavam ligadas somente as chaves. Com a necessidade de novas formas de relacionamentos entre as tabelas, essas restrições atendem a outras estruturas. Para Costa (2006), além dos modelos de banco de dados conceitual e lógico, propostos por Heuser (2004), em sua descrição adiciona o modelo físico.

Exemplo de diagrama ER modelo lógico



Modelo físico: o desenvolvimento desse modelo segue a partir das regras estabelecidas no modelo lógico.Para Martins (2007, p.425),

A partir do modelo lógico é desenvolvido o modelo físico, que define os tipos físicos dos atributos de cada tabela e considera os requisitos de performance, através do uso de índices e views, por exemplo. Ambos os modelos devem ser coesos entre si.

Exemplo de diagrama ER modelo físico

```
CREATE TABLE tbCilente (
   pkcodcliente INT PRIMARY KEY NOT NULL AUTOINCREMENT,
   nome VARCHAR (100),
   cpf VARCHAR (15)
);

CREATE TABLE tbPedido (
   pkcodpedido INT PRIMARY KEY NOT NULL AUTOINCREMENT,
   fkcodcliente INT NOT NULL ,
   numero INT,
   data VARCHAR(10),
   valor DECIMAL
   FOREIGN KEY (fkcodcliente) REFERENCES tbCilente (pkcodcliente)
);
```

Esse modelo trabalha com as restrições imposta pelo SGBD, escolhido para o desenvolvimento do banco de dados e também é desenvolvido utilizando uma linguagem de banco de dados relacionais chamada de SQL (Structured Query Language, ou Linguagem de Consulta Estruturada).

Sistema Gerenciadores de Banco de Dados

O SGBD, sistema gerenciadores de banco de dados, é um conjunto de sistemas responsáveis pelo gerenciamento de um banco de dados. Esse sistema de gerenciamento não é algo novo, conforme Rodriguez e Ferrante (2000). Essa ideia foi introduzida nos anos de 1960. No início os dados eram organizados através de índices, para facilitar a sua localização. Entre as décadas de 1980 e 1990, os SGBDS tiveram uma otimização nas suas funções, tornando-

os mais seguros e confiáveis. Rodriguez e Ferrante (2000) descrevem os principais modelos de SGBD como:

Sistemas gerenciadores de arquivos, sistema de gerência hierárquica de dados, sistema de banco de dados em rede, sistema de banco relacionais e sistema de banco de dados orientado a objetos.

Os SGBDS são sistemas responsáveis pelo gerenciamento de dados, possibilitando a realização de rotinas como consultas, inserção de dados no banco, atualização desses dados, entre outras rotinas. O SGBD é definido por Heuser (2004, p.15) como um “software que incorpora as funções de definição, recuperação e alteração de dados em um banco de dados”. O SGBD traz uma interface amigável 0 ao usuário, permitindo que a informação do banco de dados seja acessada por qualquer pessoa, não exigindo nenhum conhecimento técnico. Para Ferrari (2007), o SGBD é um ambiente para utilização de comandos de uma linguagem de desenvolvimento de banco de dados, que auxilia na manipulação de dados. Quando falamos de gerenciamento de banco de dados encontramos sistemas de gerenciamentos com características especificas para auxiliar em um determinado tipo de banco de dados. Os sistemas de gerenciamento de banco de dados distribuídos, SGBDD, é responsável pelo gerenciamento de banco de dados distribuídos. Para Alves (2004, p.145), “uma das funções principais é permitir que diversos bancos de dados distribuídos pela rede sejam manipulados de forma transparente, ou seja, não cabe ao usuário conhecer os detalhes da distribuição dos dados nem suas localizações. “Podem existir distinções entre os SGBDD, essa distinção define-os como homogêneos ou heterogêneos. A homogeneidade acontece quando os clientes e o servidor utilizam softwares idênticos e heterogêneos quando existe

diferença de softwares utilizadas pelas duas partes. Para que o sistema funcione mesmo com a heterogeneidade, é necessária a comunicação entre aplicativos de diferentes fabricantes. Conforme Alves (2004, p.152) “deve haver uma convivência pacífica entre equipamentos de diferentes marcas e modelos, mesmo nos casos em que até os sistemas operacionais são distintos”. Para Alves (2004), os SGBDD possuem três autonomias: de comunicação, comunica-se com outro SGBD; de execução, executa operação sem sofrer interferências; de associação, decide as funcionalidades de serão compartilhadas.

Os sistemas de gerenciamento de banco de dados orientados a objetos, SGBDOO, é um sistema de gerenciamento de segue a estrutura do paradigma de orientação a objetos. Segundo Alves (2004), uma das principais características dos sistemas orientados a objetos é que permite a especificação de estruturas e também de objetos que desempenham operações chamados de métodos. Ferrari (2007) afirma que no mercado existem diversos tipos de SGBD e os desenvolvedores seguem discutindo para definir qual é o melhor. Atualmente existe uma variedade de SGBD's disponíveis no mercado, como por exemplo o Oracle, PostgreSQL, MySQL, entre outros.

A Linguagem SQL (Structured Query Language, ou Linguagem de Consulta Estruturada)

As linguagens de programação descrevem rotinas que são interpretadas pelo computador, que resultam em softwares ou simplesmente em rotinas. A linguagem SQL é uma linguagem que representa um conjunto de comandos utilizados somente para o desenvolvimento de estruturas de um de banco de

dados, como exemplo, a criação de tabelas e atualização de dados. Segundo Ferrari (2007,p.11),

(...) o SQL não serve para criar rotinas de procedimentos para serem executadas pelo computador, e sim, para informar quais dados (ou conjunto de dados) queremos manipular. De fato, a finalidade do SQL, é acessar dados, independente do tipo de hardware ou software que estamos usando.

A linguagem SQL é utilizada para toda a administração de um banco de dados, desde a sua criação até a sua manutenção. Essa linguagem deriva inicialmente da linguagem SEQUEL, já em desuso. O SEQUEL tinha como principal objetivo tornar-se uma linguagem universal em função de seus comandos que basicamente passavam instruções para o computador em inglês.

A primeira versão do SQL foi distribuída na década de 1980 pela IBM, com o nome de SQL/DS. Alguns anos mais tarde, a linguagem deixa de ser propriedade da IBM e passa a ser uma linguagem livre. Com a popularidade da linguagem, existiu a necessidade do desenvolvimento de padrões para a utilização da linguagem. Em 1986, o Instituto Americano Nacional de Padrões (ANSI) inicia a padronização da linguagem. No ano seguinte a Organização Internacional de Padronização (ISO) adota essas regras. Outros padrões foram definidos até chegar ao SQL:2003. Segundo Costa (2006), a linguagem em 1989 passa por um aperfeiçoamento. Em 1992 é lançado o SQL-92, também conhecido como SQL-2. Durante os anos de 1990 o mercado de SGBD teve grande crescimento. No final dos anos de 1990, é lançado um novo padrão reconhecido como SQL-99 ou SQL-3. Nesse padrão foram definidos novos comandos, estruturas e permite o desenvolvimento de banco de dados com orientação a objetos. Alguns anos depois surge um novo padrão o SQL-2003,

em que foram revisadas as regras do SQL-3 e incluídas as regras de XML. Mesmo com essa definição de padrões, existem variações na linguagem, trazendo comportamentos diferentes de um SGDB para outro. Essas variações podem comprometer a estrutura da linguagem.

Conforme Ferrari (2007), a linguagem SQL é uma linguagem de acesso a banco de dados articulada e funcional que pode ser empregada em computadores de diferentes arquiteturas. Além do desenvolvimento de banco de dados, tabelas e consulta, entre outras estruturas, a linguagem nos possibilita a manipulação de dados, inserindo, alterando e até mesmo realizando cálculos. Segundo Costa 2006), os comandos da linguagem SQL subdividem-se nas seguintes linguagens, que são definidas em grupos de acordo com as suas funcionalidades. São elas:

- DDL (Data Definition Language): grupos de comandos utilizados para definição de dados. Esses comandos tem como objetivo a criação de estruturas como desenvolvimento do banco de dados, tabelas e outras estruturas. Os comandos alter, drop e create, fazem parte dessa linguagem.

Exemplo de DDL:

```
CREATE TABLE clientes (
    id INT PRIMARY KEY,
    nome VARCHAR(50),
    email VARCHAR(50),
    idade INT,
    cidade VARCHAR(50)
);
```

- DML (Data Manipulation Language): grupos de comandos utilizados para manipulação de dados. Esses comandos tem como objetivo

inserção, atualização de dados, consultas e exclusão de dados. Os comandos insert, delete, update e selec, fazem parte dessa linguagem.

Exemplo de DML:

```
//Cadastrando
INSERT INTO clientes (id, nome, email, idade, cidade)
VALUES (1, 'frodo', 'frodo@gmail.com', 25, 'Gravataí');

//Consultando
SELECT nome, idade, cidade
FROM clientes
WHERE idade >= 18;

//Alterando
UPDATE clientes
SET cidade = 'Canoas'
WHERE id = 1;

//Excluindo
DELETE FROM clientes
WHERE id = 1;
```

- DCL (Data Control Language): grupo de comandos utilizados para controle de dados. Esses comandos tem como objetivo definir privilégios de acesso aos usuários. Os comandos grant e revoke, fazem parte dessa linguagem.

Exemplo de DCL:

```
GRANT SELECT, INSERT, UPDATE, DELETE ON clientes TO usuario1;

REVOKE SELECT ON clientes FROM usuario1;
```

- DTL (Data Transaction Language): grupo de comando para realização de transações no banco de dados. Esses comandos tem como objetivo confirmar ou excluir, ações executadas pelos comandos da linguagem DML. Os comandos commit e rollback, fazem parte dessa linguagem.

Exemplo de DTL:

```
BEGIN TRANSACTION;
UPDATE contas SET saldo = saldo - 100 WHERE id = 1;
UPDATE contas SET saldo = saldo + 100 WHERE id = 2;
COMMIT;

BEGIN TRANSACTION;
UPDATE contas SET saldo = saldo - 100 WHERE id = 1;
UPDATE contas SET saldo = saldo + 100 WHERE id = 2;
IF saldo_da_conta_1 < 0 THEN
  ROLLBACK;
END IF;
COMMIT;
```

- DQL (Data Query Language): grupo de cláusulas e opções que são vinculadas, ao comando select, pois essa linguagem tem como objetivo realização de consultas. As cláusulas dessa linguagem são: where, from, distinct, having, order by e group by. As opções estão relacionadas a operadores lógicos, operadores matemáticos entre outras.

Exemplo de DQL:

```
SELECT * FROM clientes WHERE cidade = 'Porto Alegre';
```