Introdução a Banco de Dados com MySQL

O que veremos por aqui:

1. Introdução ao Banco de Dados
2. Tipos de Banco de Dados
3. Introdução ao Banco de dados Relacional
4. Tipos de Dados
5. Chaves
6. MER - Modelo Entidade Relacionamento
7. DER - Diagrama Entidade Relacionamento
8. Entendendo: Relacionamento entre Tabelas

1. Introdução ao Banco de Dados

Segundo Korth, um banco de dados é *"uma coleção de dados inter-relacionado, representando informações sobre um domínio específico"*, ou seja, sempre que for possível agrupar informações que se relacionam e tratam de um mesmo assunto, posso dizer que tenho um banco de dados. Por exemplo, as informações armazenadas de uma lista telefônica, um registros de usuários em um sistema de e-commerce, um sistema de controle de RH de uma empresa.

Já um **Sistema de Gerenciamento de Banco de Dados (SGBD)** é um software que possui recursos capazes de manipular as informações do banco de dados e interagir com o usuário. Exemplos de SGBDs são: Oracle, SQL Server, PostgreSQL, MySQL, Mongo DB, Cassandra e etc.

O objetivo de um **Sistema de Banco de Dados** é isolar o usuário dos detalhes internos do banco de dados (promover a abstração de dados) e promover a independência dos dados em relação às aplicações, ou seja, tornar independente da aplicação, a estratégia de acesso e a forma de armazenamento.

2. Tipos de Banco de Dados

Há dois tipos Bancos de dados no mercado, que são classificados de acordo com a maneira como organizam as informações: **Relacional** e **Não Relacional**.

2.1. Relacionais

Esta forma de organização dos Bancos de dados se caracteriza por organizar as informações em tabelas e depende da integração entre colunas e linhas.

	id	nome	quantidade	preco
	1	Camiseta	20	10.00
	2	Bermuda	12	17.90
	3	Jaqueta de Couro	3	40.00
	4	Calça Jeans	40	60.00

Assim, o atributo fica na coluna, e o conteúdo do atributo fica nas linhas. É o modelo ideal quando precisa-se armazenar informações tabulares de pouca complexidade, e que precisam ser recuperadas agilmente.

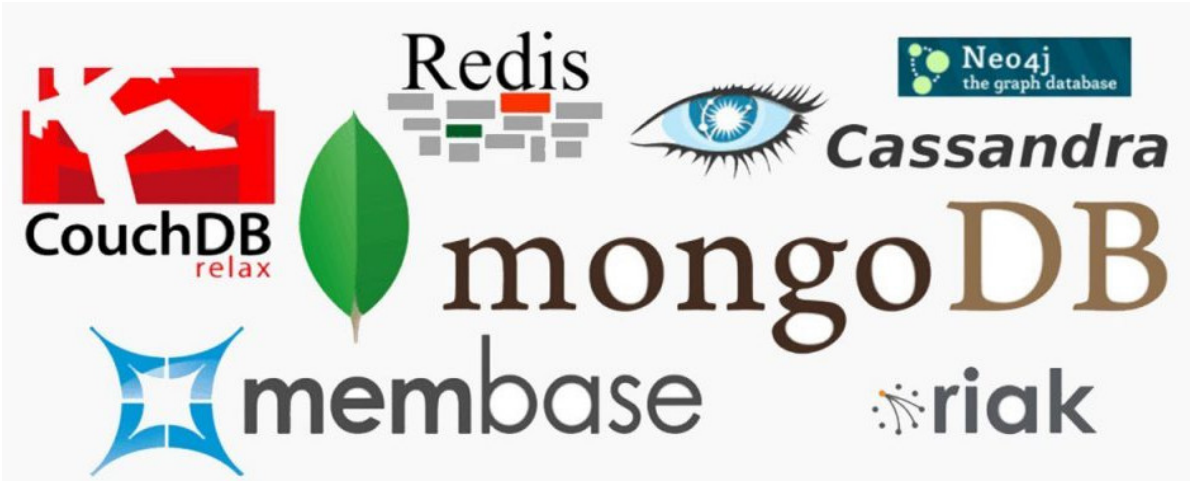
Essa é a forma de armazenamento mais utilizada, pois oferece alta confiabilidade. Porém, é necessário uma estrutura de relacionamento entre tabelas para que as informações possam ser recuperadas.

A linguagem usada para manipular essas informações é o **SQL (Structured Query Language)**. Na imagem abaixo temos alguns exemplos de Banco de dados Relacionais.

2.2. Não Relacionais

O formato Não Relacional atende uma parcela de dados que não podem ser inseridos e acessados através de tabela, como por exemplo, imagens. É muito valorizado pela sua alta performance e por manter todos os registros em um único lugar. Outro uso muito comum do Banco de dados Não Relacional são as áreas de conhecimento como **Data Science**, devido ao grande volume de dados.

Observe que nos Bancos de dados Não Relacionais, não é necessário criar um sistema de relacionamento entre informações, como nos Bancos de Dados relacionais. Como as informações não estão associadas entre si, é mais fácil fazer alterações e exclusões no conteúdo. A linguagem utilizada é o **NoSQL (Not Only SQL)**. Na imagem abaixo temos alguns exemplos de Banco de dados Não Relacionais:



SQL	NoSQL
Armazenamento de Dados Estruturados por Tabela	Armazenamento de Dados estruturados e não-estruturados por colunas, grafos, chave-valor e documentos.
Esquema estático	Esquema dinâmico
Maturidade de suporte maior (geralmente pago)	Suporte por comunidade independente (open source)
Escalabilidade vertical	Escalabilidade horizontal
Pago e Gratuito	Gratuito
O desempenho não é alto em todas as consultas. Não suporta pesquisas e cruzamentos muito complexos.	Alto desempenho em consultas
Necessidade de predefinição de um esquema de tabela antes da adição de qualquer dado	Altamente flexível (fácil adição de colunas e campos de dados não estruturados)

2.3. Quando usar um banco de dados relacional ou não relacional?

A melhor maneira de saber quando usar cada um deles é destacando os pontos fortes de cada uma dessas tecnologias.

O Banco de dados Relacional sempre irá fornecer dados íntegros e imutáveis, garantindo um controle transacional consistente. Além disso, seu esquema é rígido, sendo possível atribuir campos e estabelecer se o dado de uma coluna é nulo ou não nulo.

Já o banco de dados Não Relacional, que representa diversos tipos de Bancos de dados, não exige a rigidez de esquemas para armazenar os dados, ou seja, ele não limita os campos, diferente das colunas do SQL. Além disso, é possível adicionar novas propriedades, sem a preocupação com o impacto nas demais informações já armazenadas.

Caso sua empresa esteja aplicando metodologias ágeis modernas, um Banco de dados Relacional provavelmente não seria uma boa opção nesse contexto, pois ela requer um nível maior de preparação.

Não existe um modelo que seja melhor do que o outro, pois cada um tem seu ponto forte. Tudo dependerá do contexto e da necessidade da empresa.

IMPORTANTE: ***No Bootcamp da Generation Brasil utilizaremos o Banco de Dados Relacional MySQL.***

3. Introdução ao Modelo Relacional

Todos os Bancos de dados Relacionais são constituídos por 4 elementos básicos: campos, colunas, linhas e tabelas.

- Campos são os espaços reservados para inserção de um determinado dado (em vermelho na figura);
- Colunas são os registros de um determinado campo (em verde na figura);
- Linhas (tuplas) são os registros de um conjunto de campos (em azul na figura);
- Tabelas são os conjuntos de linhas, campos e colunas.

id	nome	quantidade	preco	categoria_id
1	tomate	100	8.00	NULL
3	laranja	50	10.00	NULL
4	banana	200	12.00	NULL
5	uva	1200	30.00	NULL
6	pêra	500	3.00	NULL
NULL	NULL	NULL	NULL	NULL

3.1. Linguagem SQL

SQL significa **Structured Query Language** e é a linguagem padrão utilizada pelos banco de dados relacionais. Ela está dividida em 2 categorias principais:

- **DML - Linguagem de Manipulação de Dados:** instruções DML indicam uma ação para o SGBD executar. Utilizados para recuperar, inserir e modificar um registro no banco de dados. Seus principais comandos são: **INSERT, DELETE, UPDATE e SELECT;**
- **DDL - Linguagem de Definição de Dados:** instruções DDL são responsáveis pela criação, alteração e exclusão dos objetos no banco de dados, ou seja, pela estrutura do Banco e suas Tabelas. Seus principais comandos são: **CREATE DATABASE, CREATE TABLE, ALTER TABLE e DROP TABLE;**

4. Tipos de Dados

Agora que entendemos o conceito de Banco de Dados, também, precisamos entender que assim como tudo que manipula dados precisamos definir a ***"Tipagem"*** dos dados, vamos ver algumas:

Dado	Tipo
Texto	VARCHAR(n)
Caractere	CHAR(n)
Texto muito grande	TEXT
Data	DATE
Data e Hora	DATETIME
Hora	TIME
Data e Hora do sistema	TIMESTAMP
Inteiro	INT
Inteiro grande	BIGINT
Ponto Flutuante	FLOAT
Ponto Flutuante mais casas decimais	DOUBLE
Numero decimal com ponto fixo (10,0)	DECIMAL
Booleano (Verdadeiro ou Falso)	BOOLEAN
Valores pequenos podendo ser (0 e 1)	Bit
Guardar Bytes (arquivos e imagens)	BLOB


Documentação: Tipos de Dados - W3Schools

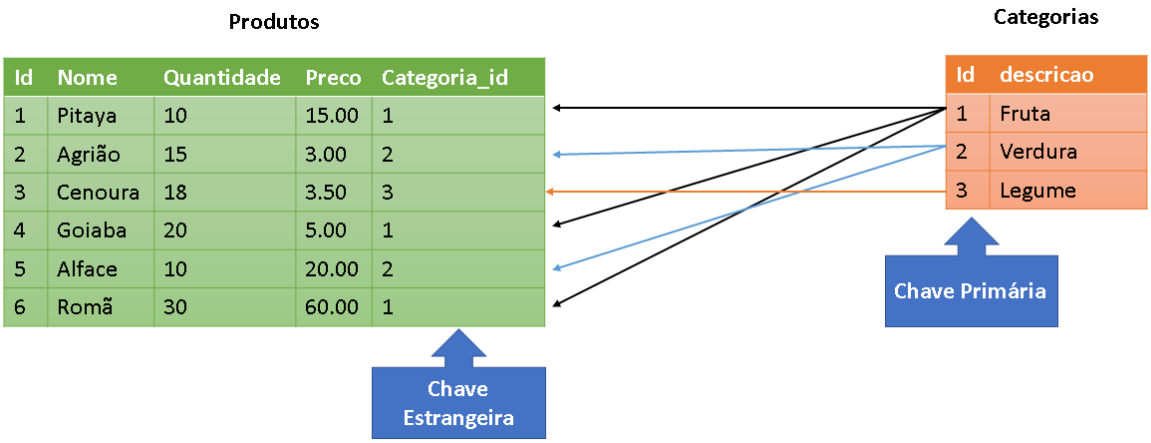
5. Chaves

5.1. Chave Primária (PK)

A **chave primária** é o que torna a linha ou o registro de uma tabela únicos. Geralmente, é utilizada uma sequência numérica automática para a geração dessa chave para que ela não venha a se repetir. Nenhuma linha possuirá o mesmo valor na coluna que é chave primária, ou seja, um identificador único do registro.

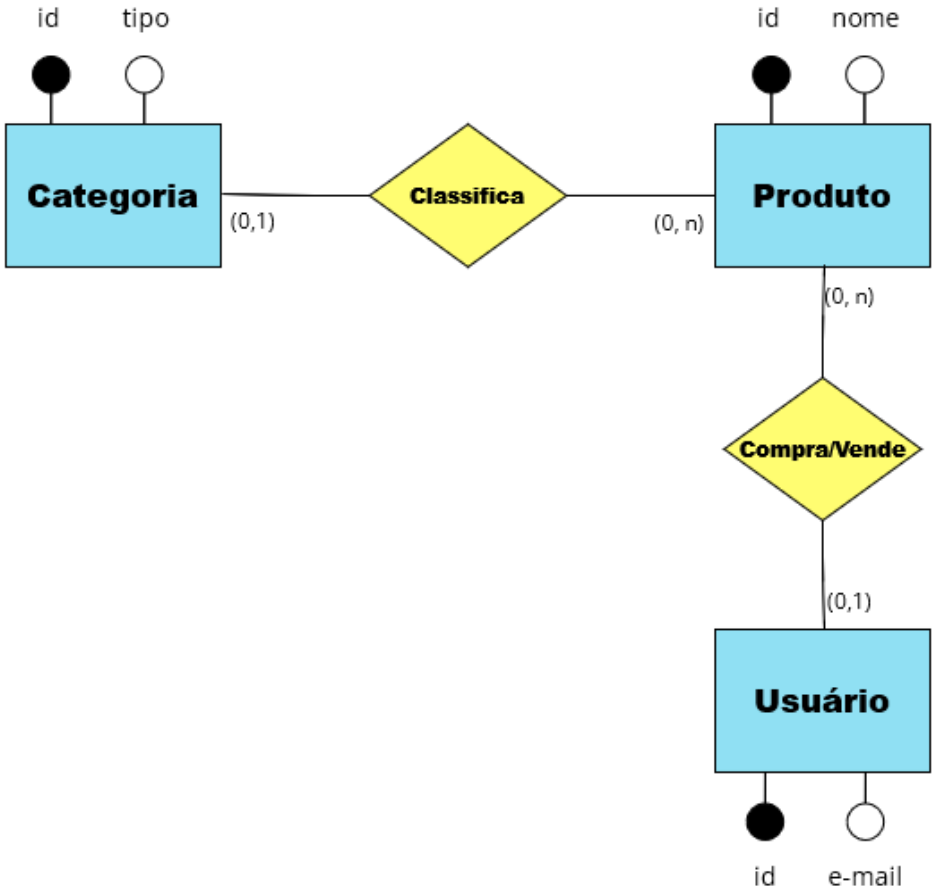
5.2 Chave Estrangeira (FK)

A chave estrangeira define um relacionamento entre tabelas, comumente chamado de integridade referencial. Esta regra baseia-se no fato de que uma chave estrangeira em uma tabela é a chave primária em outra. Na imagem abaixo, a tabela produtos tem o campo **categoria_id (Chave Estrangeira)**, isto é, ela pode se repetir na tabela de produtos. No entanto, deve ser único na tabela de categorias **(Chave Primária)**, pois assim terá uma referência exclusiva.



6. MER - Modelo Entidade Relacionamento

Assim como todo os projetos, devemos iniciar por um "esboço" do que de fato será nossa base de dados, quando desenhamos esta estrutura, á chamamos de **MER - Modelo Entidade Relacionamento** onde com apenas um papel e uma caneta podemos desenhar o modelo do nosso banco de dados expondo todos os relacionamento entre as tabelas.



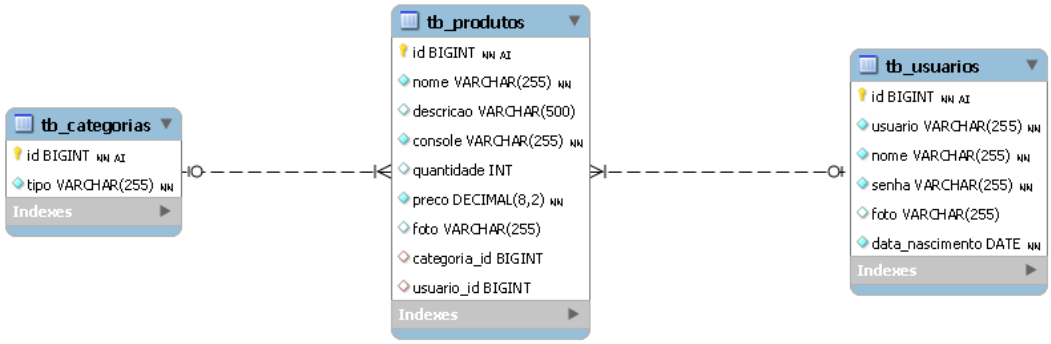
Um Modelo Entidade Relacionamento é uma maneira sistemática de descrever e definir um processo de negócio. O processo é modelado como componentes (Entidades) que são ligadas umas as outras por Relacionamentos que expressam as dependências e exigências entre elas.

No exemplo acima, uma Categoria pode possuir zero ou mais Produtos, mas um Produto só pode ser classificado em apenas uma Categoria.

Entidades podem ter várias propriedades (Atributos) que os caracterizam. Uma Entidade e seus Atributos se transformam em uma ou mais Tabelas no Banco de dados, dependendo do tipo Relacionamento.

7. DER - Diagrama Entidade Relacionamento

O **DER - Diagrama Entidade Relacional** também pode ser visto como um "esboço" de uma forma mais detalhada em comparação ao **MER**, porém ele é feito através de uma ferramenta especifica, onde podemos compartilhar de uma forma segura com os demais membros de nossa equipe, e também extrair do nosso **DER** o código SQL usado para criar o Banco de dados.

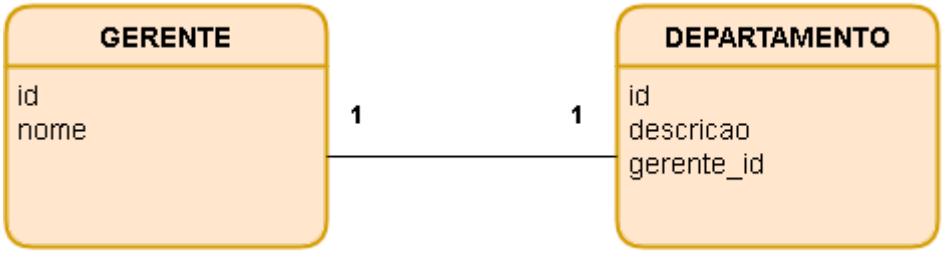


8. Entendendo: Relacionamento entre Tabelas

Uma vez que terminamos o DER, precisamos definir o relacionamento entre as tabelas, De acordo com a quantidade de objetos envolvidos em cada lado do relacionamento, classificando-os de três formas:

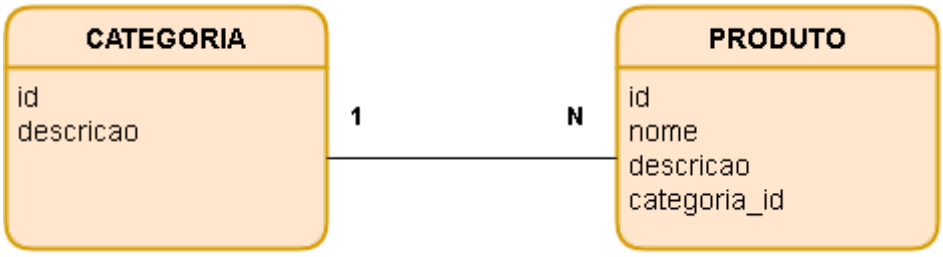
- **Relacionamento 1...1 (um para um):** cada uma das duas entidades envolvidas referenciam obrigatoriamente apenas uma unidade da outra. Por exemplo, em um banco de dados de currículos, cada usuário cadastrado pode possuir apenas um currículo na base e cada currículo só pode pertencer a um único usuário cadastrado.

No exemplo abaixo temos uma Relação **1:1**.



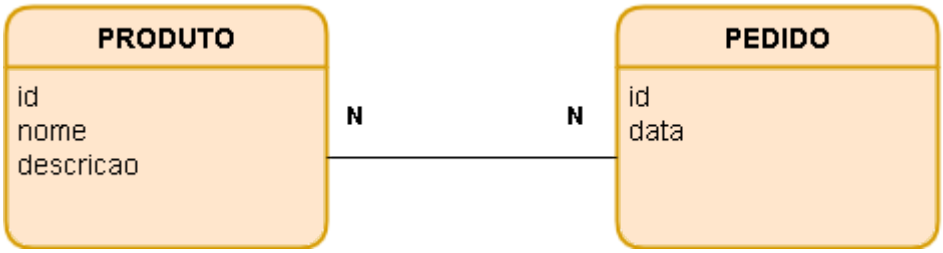
- **Relacionamento 1...n (um para muitos):** uma das entidades envolvidas pode referenciar várias unidades da outra, porém, do outro lado cada uma das várias unidades referenciadas só pode estar ligada a uma unidade da outra entidade. Por exemplo, em um sistema de plano de saúde, um usuário pode ter vários dependentes, mas cada dependente só pode estar ligado a um usuário principal.

No exemplo abaixo temos uma Relação **1:N**.

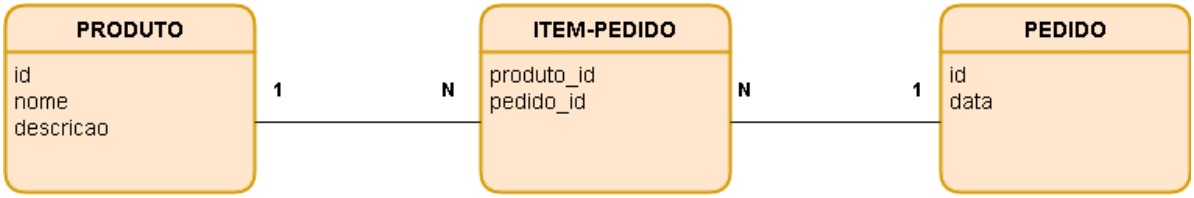


- **Relacionamento n...n ou ... (muitos para muitos):** neste tipo de relacionamento cada entidade, de ambos os lados, podem referenciar múltiplas unidades da outra. Por exemplo, em um sistema de biblioteca, um título pode ser escrito por vários autores, ao mesmo tempo em que um autor pode escrever vários títulos. Assim, um objeto do tipo autor pode referenciar múltiplos objetos do tipo título, e vice versa.

No exemplo abaixo temos uma relação **N:N**.



Observe que da forma como a Relação foi representada, a Entidade **PRODUTO** teria que possuir uma Chave Estrangeira relacionada com **PEDIDO** e vice versa. Como não é possível armazenar Chaves Estrangeiras nas duas entidades relacionadas simultaneamente, será necessário criar uma terceira Entidade(**ITEM-PEDIDO**), para armazenar as 2 Chaves Estrangeiras, como mostra o Diagrama abaixo.



Observe que tanto PRODUTO quanto PEDIDO possuem uma relação 1:N com ITEM_PEDIDO.

IMPORTANTE: ***Uma boa prática é nomearmos os relacionamentos com verbos ou expressões que representa a forma como as entidades (tabelas) interagem, ou a ação que uma exerce sobre a outra, este nome pode mudar de acordo com com a direção que se lê o relacionamento.***