ENIAC
Centro Universitário de Excelência

Material de Estudos

Banco de Dados - MySQL

Conceito de Banco de Dados

Um banco de dados é de demasiada importância, pois serve especificamente para armazenar dados úteis, armazenando esses dados de forma inteligente.

Neste mercado temos dois tipos de profissionais: o profissional que gera ou cria o banco de dados (Administrador de Dados), e o outro que realiza a manutenção e administra dados já existentes (Database Administrator - DBA).

A linguagem a ser utilizada para estruturar um banco de dados, é a **SQL**, tendo a função de armazenar arquivos em diferentes lugares, além de gerar as informações úteis, de forma que possamos interagir.

A transação engloba todas as atividades que um SGBD (Sistema de Gerenciamento de Banco de Dados) realiza após a interação que o usuário tem com o próprio sistema.

Para criar um banco de dados, temos **4 regras** que são fundamentais:

1. **ATOMICIDADE**

Deve ser realizado uma transação em dois estados, **commit** (sucesso) ou **rollback** (falha), sendo que se ocorrer um problema nesta transação, retornará ao estado original.

2. **CONSISTÊNCIA**

As regras em sua totalidade sempre devem ser respeitadas, garantindo que todo tipo de dados estejam corretos, tendo consistência. Como atributos de: VARCHAR (Conjunto de caracteres), INT (Número inteiro sem casa decimal “8”),FLOAT (Número inteiro com casa decimal “8.5”), DATE, etc.

Nota: Devemos indicar o tipo de atributo, pois o computador o identifica e determina um valor específico em kb para utilização, variando de acordo com estes valores atribuídos.

3. **ISOLAMENTO**

Cada transação é única e independente, sendo assim, duas transações que alterem o mesmo valor de uma tabela não entram em conflito. Toda operação é uma engrenagem na SGBD.

4. **DURABILIDADE**

Quando uma transação é finalizada, seus dados estão salvos de qualquer modificação. Somente outra transação pode modificá-los. Os dados, portanto, ficam protegidos.

Sendo assim, para que um banco de dados funcione, deve-se respeitar essas quatro regras, que irão garantir o bom funcionamento do mesmo.

Evolução dos Bancos de Dados

Nasceu na década de **60**, sendo armazenado de forma rudimentar, até que a empresa **IBM** investiu em pesquisas para relacionamento e atividades de armazenamento de dados do formato virtual.
Através do pesquisador **Edgar Frank Codd** que tem a autoria de um modelo relacional de banco de dados.

Até que na década de **70**, surgiu o **sistema R.** e a **Linguagem SQL** (*Structured Query Language*).

Nos anos **80** a **Oracle**, outra empresa gigante entrou na área dos Bancos de Dados, lançou o **Oracle 2** e então **IBM** para não ficar para trás, lançou o **SQL DS** (**DB2** dos tempos atuais), surgimento também do **Microsoft SQL Server** uma SQL que era livre, mas acabou sendo comprada pela Oracle. E seus idealizadores, que realizaram um Spin Off para o MariaDB. que muitos nem mesmo consideram como banco de dados.

Nos dias atuais temos como disponibilidade para nosso uso, os Bancos de Dados free e os pagos, como por exemplo temos:

- MySQL
- MariaDB
- Microsoft SQL Server
- DB2
- PostgreSQL
- Oracle
- MongoDB

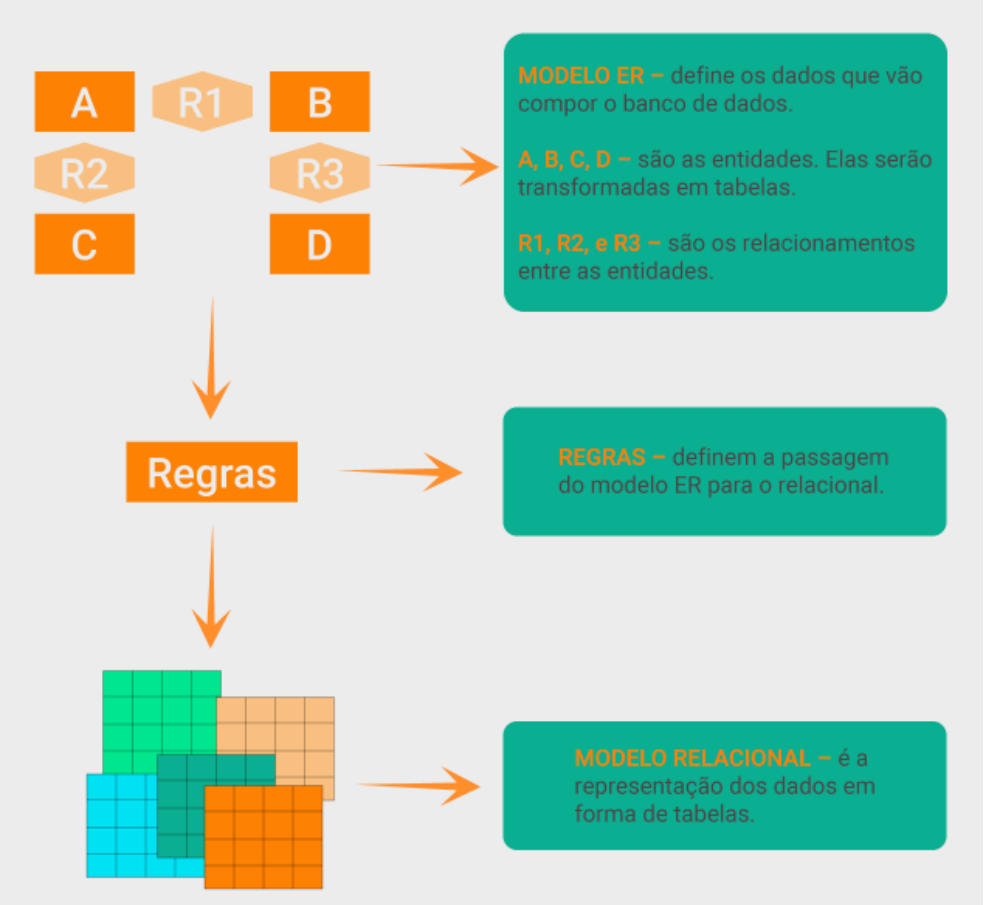
Com base em todo o decorrer desta história, concluímos que os Sistemas de bancos de dados atuais são seguros e extremamente confiáveis, proporcionando também imensas possibilidades aos desenvolvedores por meio de seu uso.

Modelo Relacional

O modelo relacional é um modelo físico, que representa o banco de dados como uma coleção de tabelas (relações), enquanto o modelo Entidade-Relacionamento, é usado para descrever as entidades e as suas características (atributos), ilustrando como esses elementos se relacionam entre si.

O modelo conceitual é utilizado para descrever os objetos (Entidades), suas características (atributos) e os relacionamentos entre si,representando de forma abstrata um banco de dados, fornecendo informações sobre os aspectos relacionados ao seu domínio.

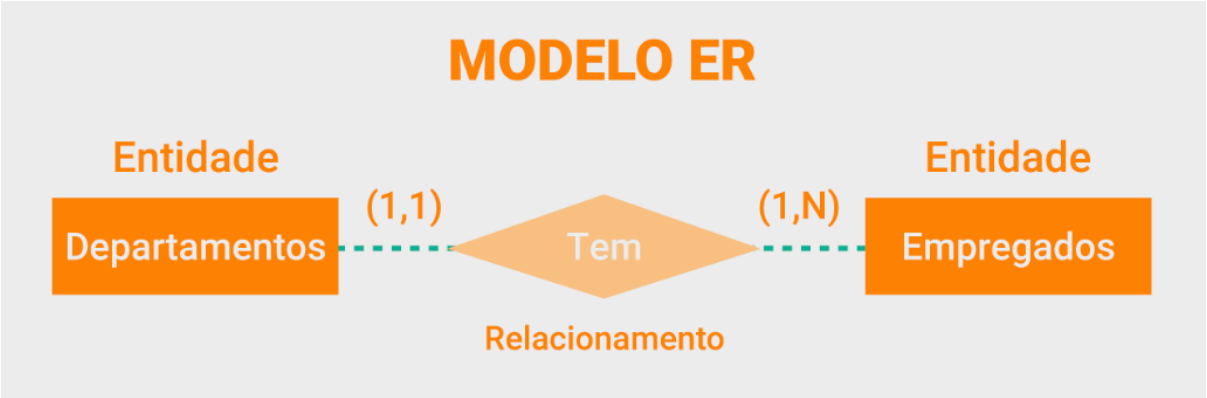
O modelo relacional é o passo seguinte da modelagem de banco de dados, mostrando com clareza o “mapeamento” entre as diversas tabelas que compõem o banco de dados, sendo um modelo físico representando os dados de um banco de dados, por meio de uma coleção de tabelas, também chamadas de relações ou instâncias.

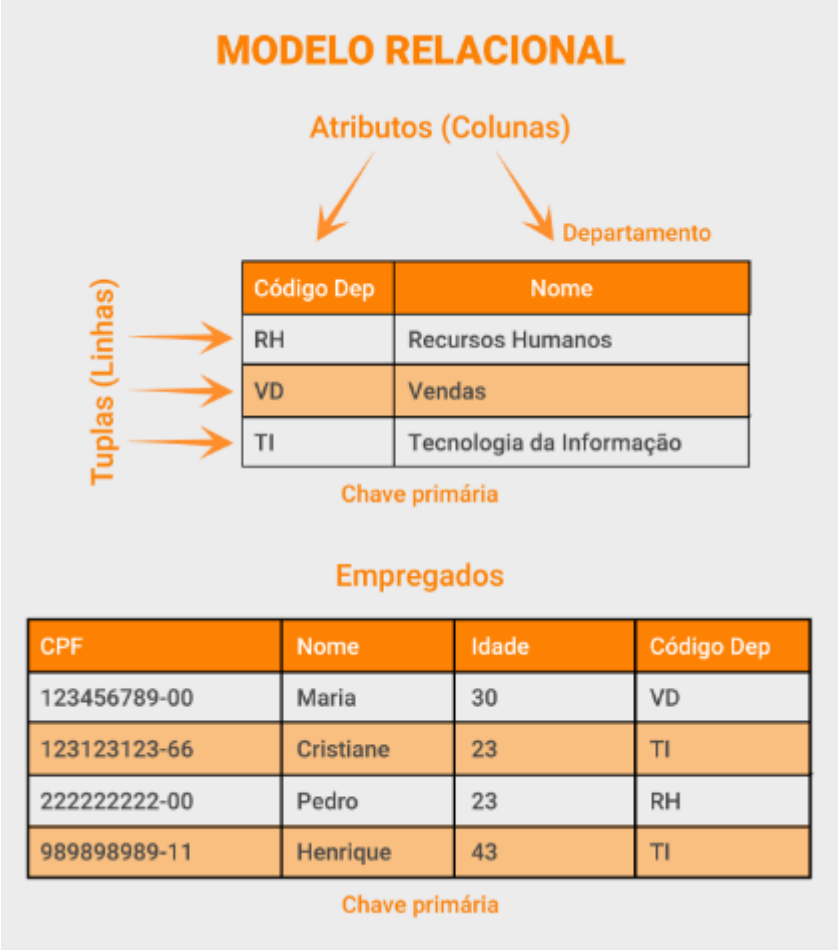


- Cada **Entidade** torna-se uma **Tabela**;
- Cada **atributo** torna-se um **campo** da tabela criada;
- Os **atributos identificadores** formam a **chave primária**.

Atributos são as informações que referenciam a entidade.

Atributos identificadores são únicos para cada entidade, como CPF.





Tipos de Dados

- **binary** = Número binário (0 ou 1);
- **blob** = Um objeto do tipo arquivo, com dados brutos imutáveis;
- **boolean** = Verdadeiro ou falso (0 e 1);
- **date** = Data;
- **datetime** = Data e hora;
- **decimal** = Maior número de dígitos significativos para um número;
- **float** = Número com vírgula;
- **integer** = Número inteiro;
- **string** = Poucas letras;
- **text** = Texto;
- **time** = Hora/tempo;
- **timestamp** = Basicamente representa um instante único, um ponto específico na linha do tempo;
- **varchar** = Caracteres (letras e números)[Sem limite];
- **char** = Caracteres [com limite].

Normalização de Dados

As formas normais correspondem a um conjunto de regras e de simplificação de tabelas.

Principais especificações de cada formal normal	
Forma normal	Característica principal.
1FN	Apenas atributos atômicos.
2FN	Estar na 1FN e sem dependência funcional parcial.
3FN	Estar na 2FN e sem dependência funcional transitiva.
Boyce-Codd	Estar na 3FN e não pode existir um atributo A determinando outro B, sendo que A não é super-chave.
4FN	Estar na 3FN e não pode existir dependência funcional multivalorada.
5FN	Estar na 4FN e não pode existir dependência funcional de junção.

Sendo que apenas as três primeiras são essenciais, **1FN**, **2FN** e **3FN.** Pois as duas últimas são utilizadas apenas quando o banco de dados já está construído sendo aplicados depois de anos da existência das demais.

Dependências funcionais

Uma dependência funcional ocorre quando um valor A depende de um valor B. As dependências funcionais se dividem em três:

- **Dependência funcional total** - Quando houver uma chave primária concatenada, isto é, duas ou mais colunas são a chave primária

de uma tabela, as demais colunas dependerão exclusivamente dessa ligação para que possam ser inseridas corretamente.

Exemplo: „
Tabela ItemVenda
■ CodVenda PRIMARY KEY
■ CodProduto PRIMARY KEY
■ Qtd

Repare que a coluna Qtd irá depender totalmente da chave primária concatenada entre CodVenda e CodProduto.

- **Dependência funcional parcial** - Ocorre quando um item da tabela depende de uma parte da chave primária concatenada da tabela, e não da chave toda.

  Exemplo: „
  Tabela ItemVenda
  ■ CodVenda PRIMARY KEY
  ■ CodProduto PRIMARY KEY
  ■ Qtd
  ■ PrecoProduto

A coluna PrecoProduto depende do valor da coluna CodProduto, que faz parte da chave primária da tabela ItemVenda. Portanto, PrecoProduto depende parcialmente da chave primária concatenada da tabela.

- **Dependência funcional transitiva** - Acontece quando uma coluna da tabela depende de outra coluna da tabela que não é chave dessa tabela.

  Exemplo: „
  Tabela ItemVenda
  ■ CodVenda PRIMARY KEY
  ■ CodProduto PRIMARY KEY
  ■ Qtd
  ■ PrecoProduto
  ■ TotalParcial

A coluna TotalParcial depende do resultado da multiplicação das colunas Qtd por PrecoProduto, e essas colunas não são chave da tabela.

Sendo assim, para estruturar uma tabela de banco de dados bem otimizada, deve-se seguir estas três regras.

Linguagem de Definição de Dados

A Linguagem SQL é uma linguagem de consulta estruturada, transformando os dados (nível físico) em informação (nível lógico).

- **DDL (Linguagem de Definição de Dados)**

Comandos (CREATE, ALTER E DROP) - Usados na estrutura da data base.

CREATE = Comando utilizado para criação de tabelas;
- CREATE DATABASE/TABLE nome

ALTER = Comando utilizado para alteração de tabelas;
- ALTER TABLE nome ADD nome2
- ALTER TABLE nome DROP COLUMN nome
- ALTER TABLE nome MODIFY COLUMN nome
- ALTER TABLE nome CHANGE `nome` `nome2` VARCHAR(-)

DROP = Comando utilizado para exclusão de tabelas.
- DROP DATABASE/TABLE nome

Linguagem de Manipulação de Dados

- **DML (Linguagem de Manipulação de Dados)**

Comandos (INSERT, UPDATE, SELECT E DELETE) - Usando para adicionar dados na data base.

INSERT *= Comando utilizado para inserção de dados;*
- INSERT INTO (atributo1, atributo2,…) VALUE (valor-atributo1 valor-atributo2, …)
- INSERT INTO alunos VALUE (valor-atributo1, valor-atributo2, …)

UPDATE *= Comando utilizado para atualização de dados;*

- UPDATE tabela SET atributo = '' WHERE PK = ;
- UPDATE tabela SET atributo = CONCAT('value', atributo);
***SELECT** = Comando utilizado para pesquisa de dados;*

***DELETE** = Comando utilizado para exclusão de dados.*

Criando uma DataBase

Passo a passo

1. Baixar o **XAMPP** : https://www.apachefriends.org/xampp-files/7.3.29/xampp-windows-x64-7.3.29-1-VC15-installer.exe
2. Com o mesmo já aberto, instale os serviços necessários, neste caso (Apache e MySQL)
3. Clique **Start** para Apache e para MySQL
4. Após iniciado, clique em **Admin** em MySQL
5. Quando aberta a LocalHost clique em **SQL** e no campo de texto digite: `CREATE DATABASE nome_database ;`

Referências

- Conceitos de Banco de Dados (grupoa.education)
- Evolução de Banco de Dados (grupoa.education)
- Modelo Relacional (grupoa.education)
- Diagrama de Entidade e Relacionamento (DER) (grupoa.education)
- Normalização de Dados (grupoa.education)
- Linguagem de definição de dados (DDL) – Data Definition Language (grupoa.education)
- Linguagem de manipulação de dados (DML) – Data Manipulation Language (grupoa.education)