

Banco de Dados (MSSQL)

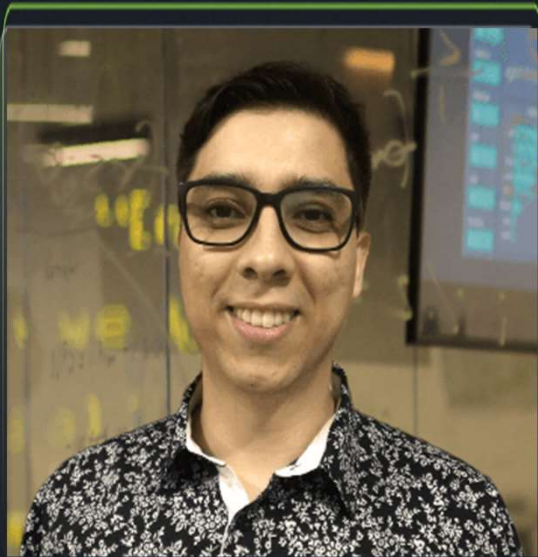
---

Singia - 955 | #BeTheNext - C#

## Apresentação

---

## Professores



Sobre o professor...

<https://www.linkedin.com/in/michael-tadeu>

## Estrutura do Módulo e Metodologia de Avaliação

---



## Estrutura do Módulo e Metodologia de Avaliação

### -> Módulo


- > Modelagem Entidade Relacionamento
- > Modelo Físico e Normalização
- > Queries Simples
- > Queries complexas
- > Otimização

### -> Avaliação

- > Projeto Final
- > Exercícios em Aula e no Class
- > Participação em Aula

Combinados

---



Metodologia de Ensino do Módulo

## Estrutura do Módulo e Metodologia de Avaliação

-> Aulas Expositivas

-> Live Coding

-> Exercícios/ Desafios



Ferramentas

---

## Ferramentas

-> Repositório

-> GitHub

-> SGBD

-> Microsoft SQL Management Studio

O que vamos aprender...

---

O que vamos aprender...

-> Introdução

-> Configuração de Ambiente

-> ACID



## Introdução

---

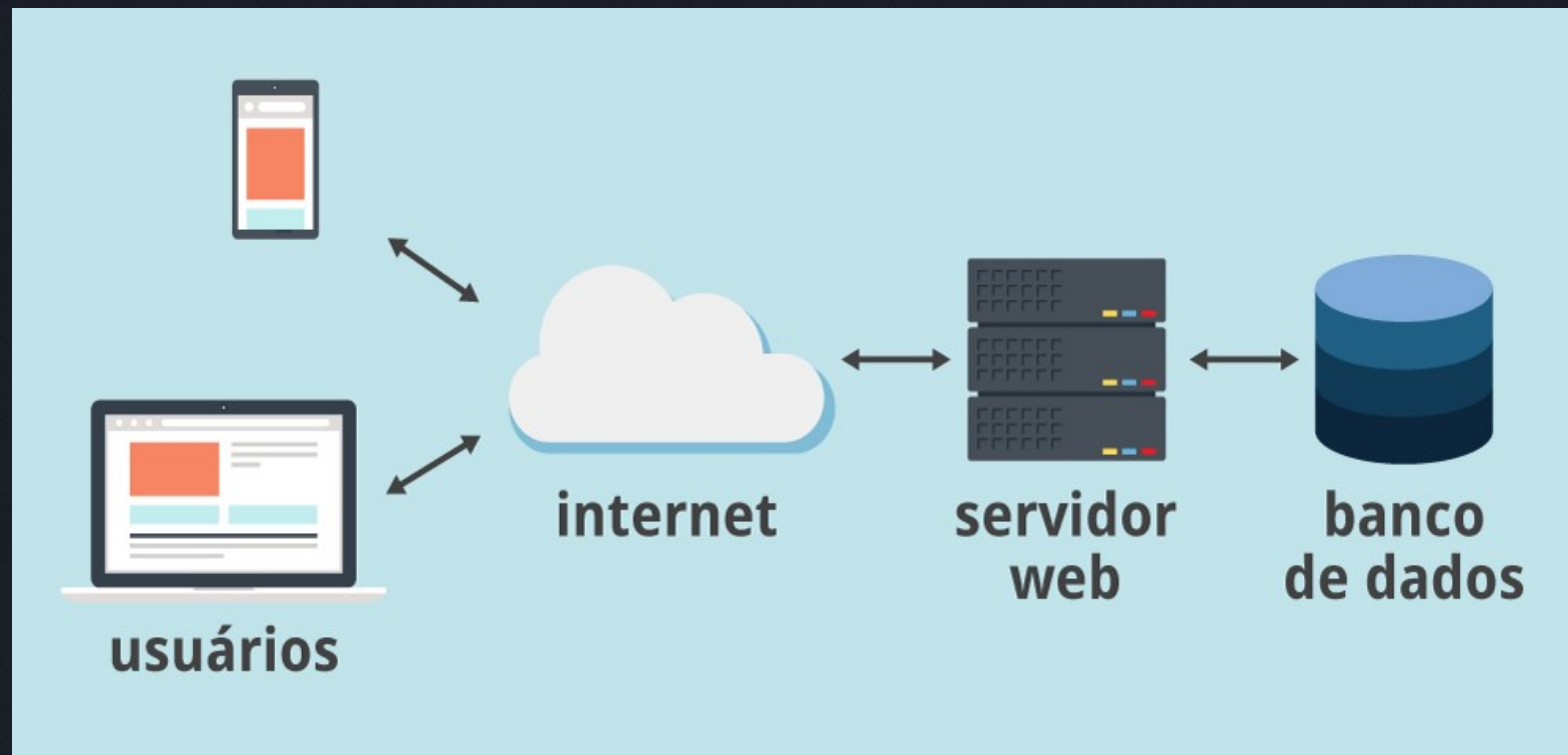
## Introdução

"O princípio básico de um Banco de Dados é armazenar informações de um sistema."

## Introdução

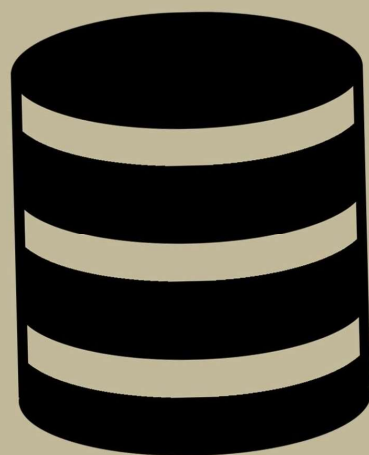
- > Dados: Unidade de informação a ser armazenada.
- > Banco de Dados: Coleção de dados relacionados logicamente.
- > Sistema Gerenciador de Banco de Dados – SGBDs: Coleção de programas que permite a criação e o gerenciamento de bancos de dados.

## Introdução

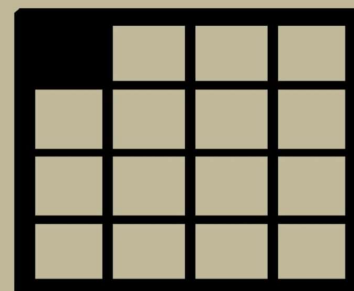




## Introdução



Vs.



## Introdução



## Introdução

ORACLE®



```
SELECT * FROM usuario WHERE  
estado = "São Paulo"
```



## Introdução



Cassandra

mongoDB

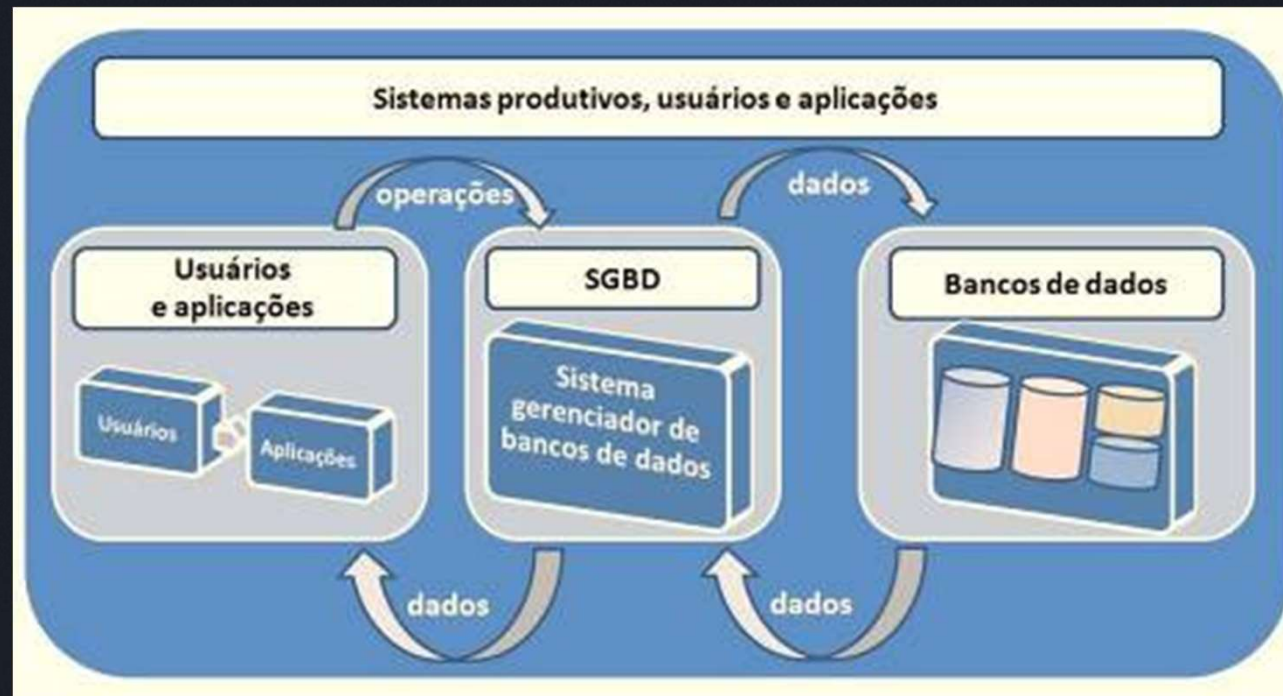


membase



```
db.usuario.find(  
  { estado: { $eq: "São Paulo" } }  
)
```

## Introdução



## Introdução

-> Como Acessar um SGBD?

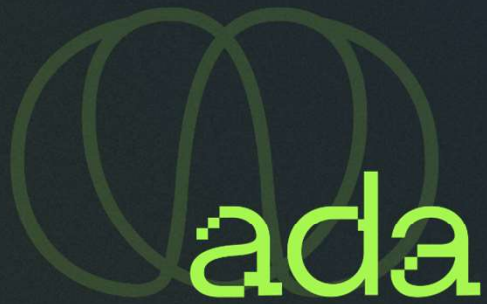
-> Dados: Através de SQL.

-> Componentes: Através de interfaces fornecidas pelos  
fabricantes do SGBD.



## Configuração de Ambiente





Banco de Dados (MSSQL)

---

Singia - 955 | #BeTheNext - C#

Resolvendo Problemas



Microsoft®  
SQL Server®



O que vamos aprender...

---



## O que vamos aprender...

-> Projeto Final

-> ACID

-> Tipos de Dados

-> SQL

Projeto Final

---

## Projeto Final

### -> Descrição

Desenvolva um projeto individual (tema livre) que aplique todo o conteúdo deste módulo.

Banco de Dados (defina um modelo de dados consistente com sua aplicação)

Scripts de inicialização dos dados: defina um arquivo data.sql para realizar os inserts iniciais da aplicação.

## Projeto Final

Crie um arquivo README.md explicando o escopo do seu projeto e adicione os diagramas que vier a criar.

Sugestão: Aplicação spring boot usando hibernate para comunicar com o banco de dados.

Seu banco de dados deve definir mais de uma tabela, relacionamentos e constraints (FKs).

Defina pelo menos um CRUD (Create, Read, Update and Delete).





## Projeto Final

### -> Avaliação

O grupo será avaliado pela coerência do projeto e aplicação dos seus conceitos vistos em aula.

Cada membro do grupo será avaliado individualmente por seus commits no repositório.

### -> Prazo

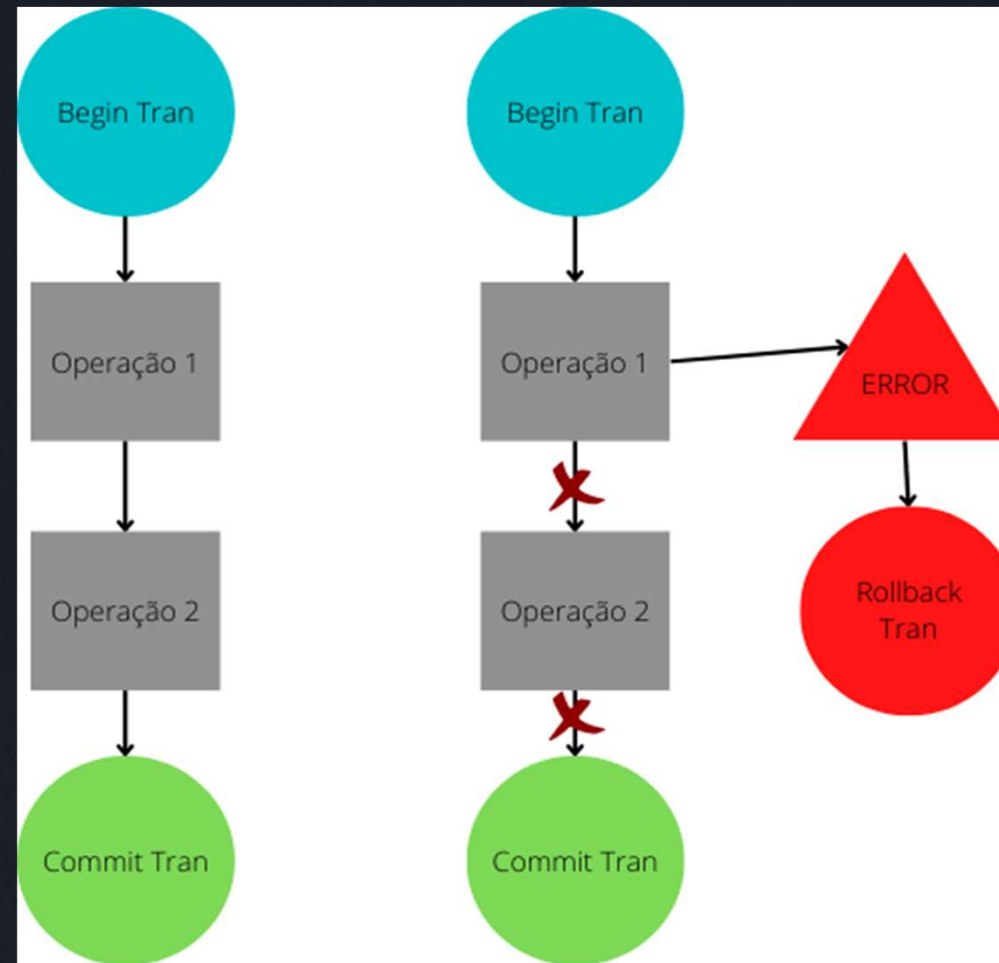
Até o dia 15/02/2023 antes da aula para apresentar.

Entrega final para o dia 16/02/2023.

ACID

---

## ACID



## ACID

<b>A</b>		<b>Atomicidade</b> Indivisível
<b>C</b>		<b>Consistência</b> Coerência
<b>I</b>		<b>Isolamento</b> Segregado
<b>D</b>		<b>Durabilidade</b> Persistência



## Tipos de Datos

---

## Tipos de Dados - Numéricos

bit;

tinyint;

smallint;

int;

bigint;

decimal(p,s);

numeric(p,s);

smallmoney;

money;

float(n);

real.

## Tipos de Dados - Textos

`char(n);`

`varchar(n);`

`varchar(max);`

`text;`

`nchar;`

`nvarchar;`

`nvarchar(max);`

`ntext;`

`binary(n);`

`varbinary;`

`varbinary(max);`

`image.`

## Tipos de Dados - Data

`datetime;`

`time;`

`datetime2;`

`datetimeoffset.`

`smalldatetime;`

`date;`



## Tipos de Datos - Data

sql\_variant;

table;

uniqueidentifier;

tipos de geometria espacial;

xml;

tipos de geografia espacial.

cursor;

## Tipos de Dados – Criando Tabelas

CLIENTE (cod\_cli, nome\_cli, endereco, cidade, cep, uf)

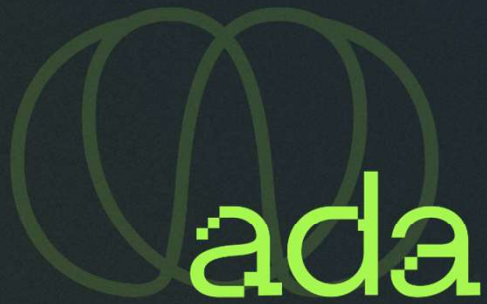
VENDEDOR (cod\_vend, nome\_vend, sal\_fixo, faixa\_comiss)

PRODUTO (cod\_prod, unid\_prod, desc\_prod, val\_unit)

PEDIDO ( cod\_ped, prazo\_entr, cod\_cli, cod\_vend)

ITEM\_PEDIDO (cod\_item\_ped, qtd\_ped, cod\_ped, cod\_prod)

Obrig.ada



Banco de Dados (MSSQL)

---

Singia - 955 | #BeTheNext - C#



O que vamos aprender...

---

O que vamos aprender...

-> SQL

-> Modelagem de Dados

-> Tipos de Dados

-> SQL

# SQL

DDL (Data Definition Language);

DML (Data Manipulation Language);

DCL (Data Control Language);

TCL (Transactional Control Language).

## Modelagem de Dados

- > Modelagem conceitual;
- > Modelagem logica;
- > Modelagem física (interna).

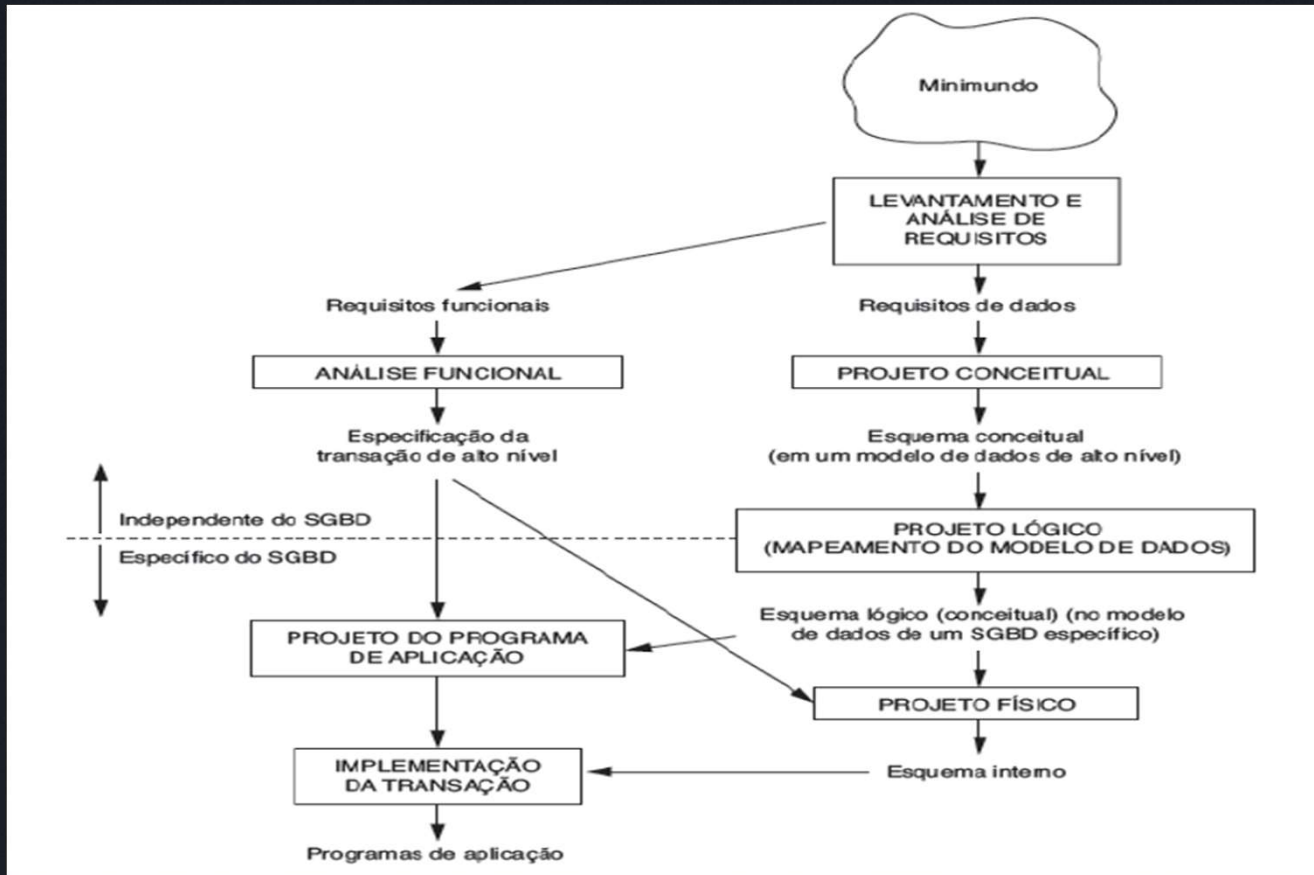


## Modelagem de Dados

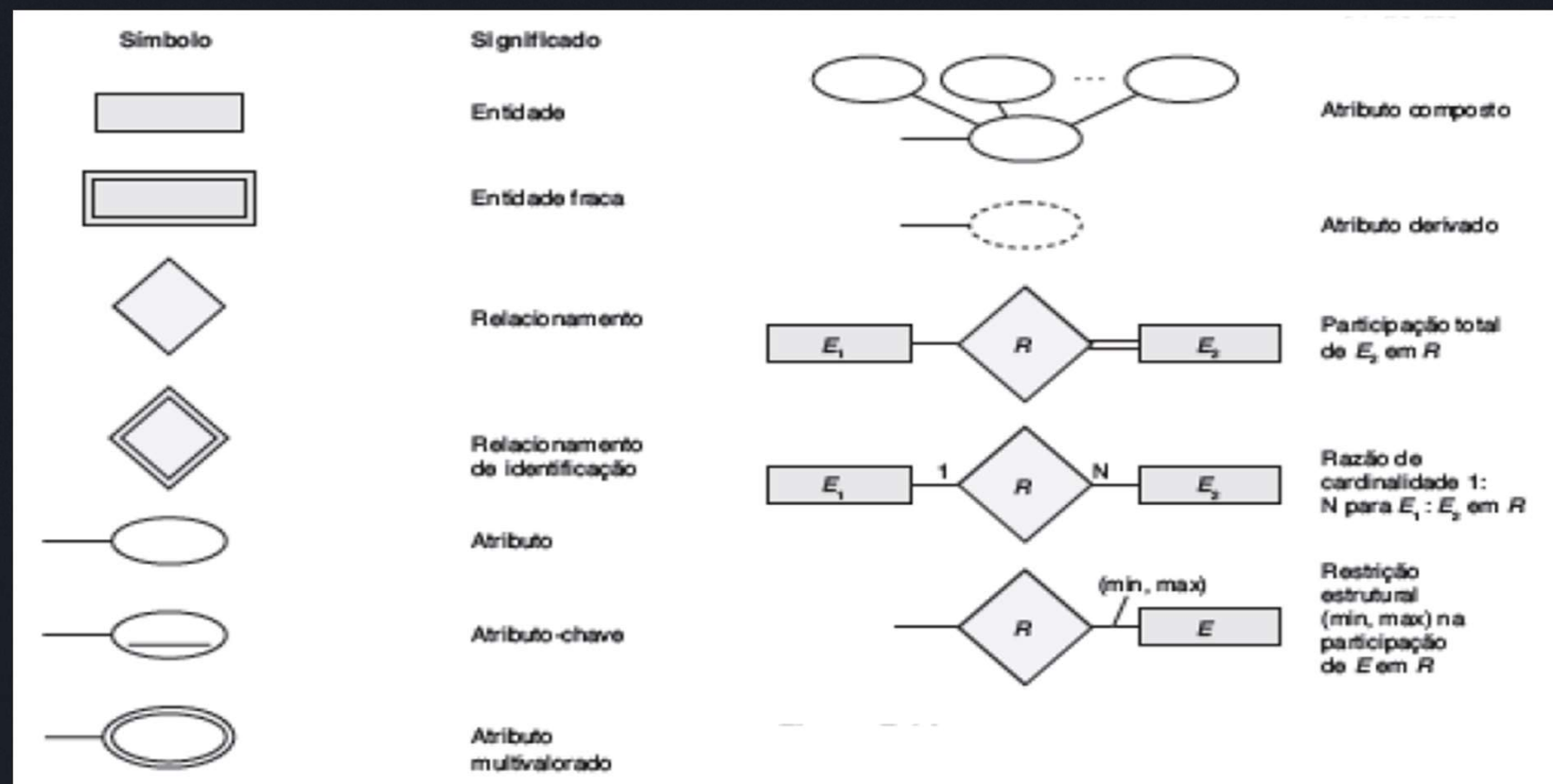
-> Abordagem Top-Down;

-> Abordagem Bottom-up.

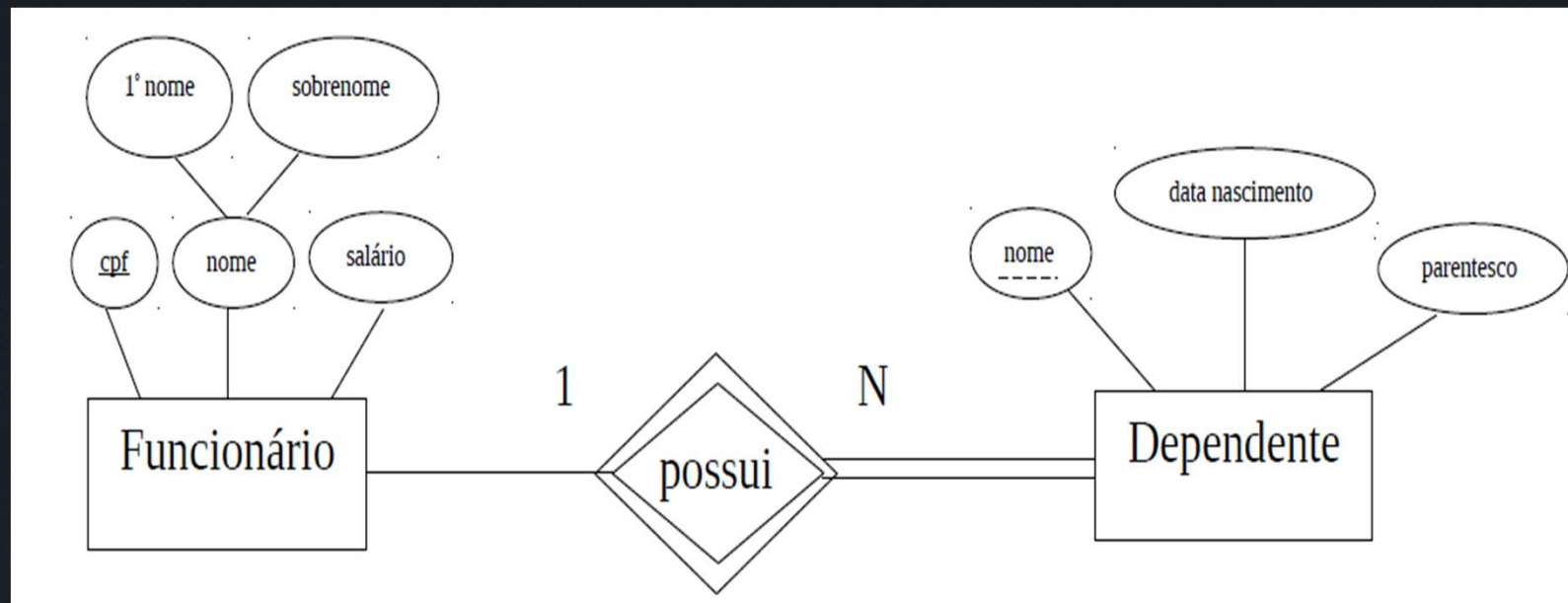
## Modelagem de Dados



## Modelagem de Dados - MER



## Modelagem de Dados - MER





## Modelagem de Dados - Cardinalidade

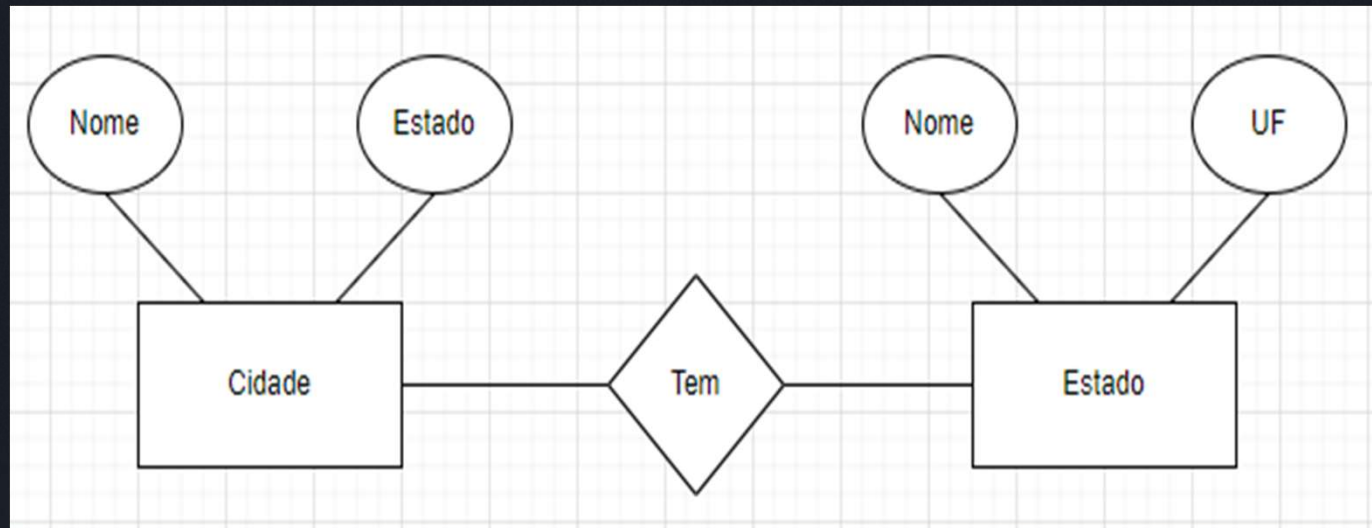
-> Cardinalidade;

-> Duas cardinalidade a considerar:

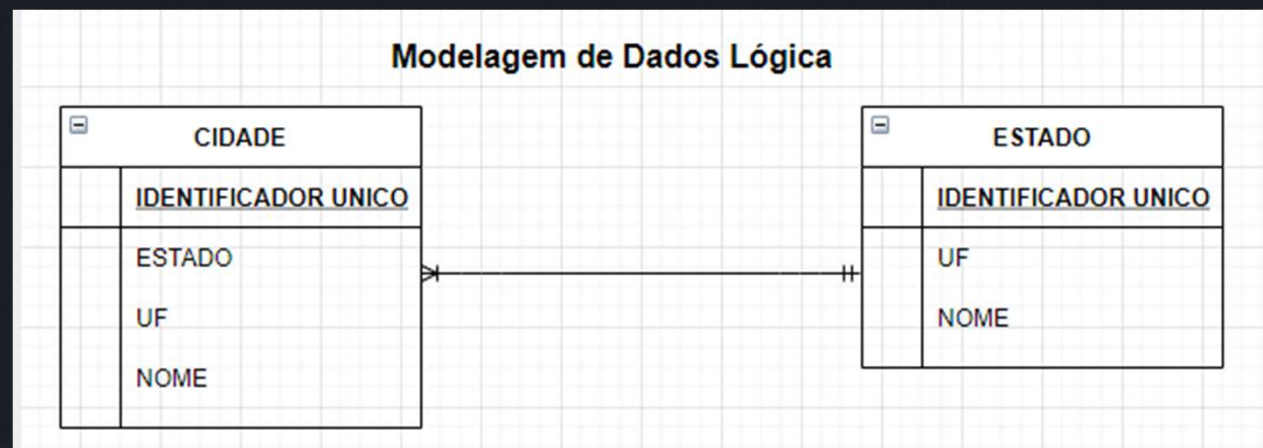
Cardinalidade mínima

Cardinalidade máxima.

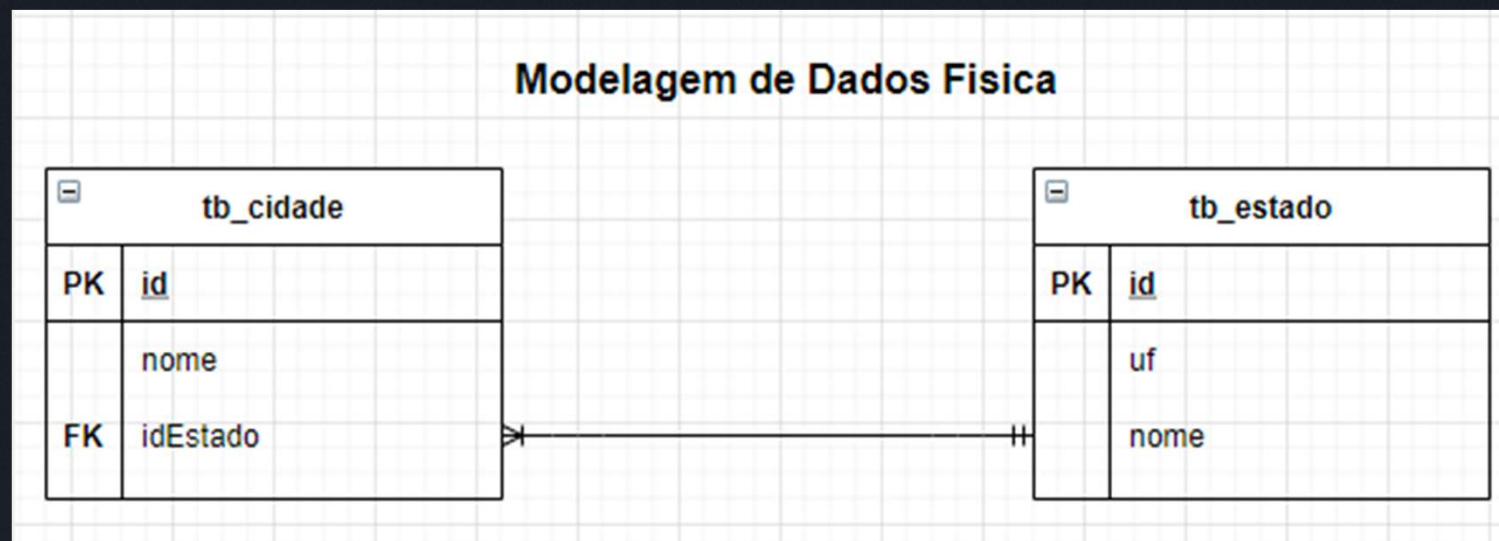
## Modelagem de Dados - Conceitual



## Modelagem de Dados - Lógico



## Modelagem de Dados - Físico





## Modelagem de Dados

Característica	Conceitual	Logica	Física
Nome de Entidades	X	X	
Relacionamentos de Entidades	X	X	
Atributos	X	X	
Chave Primária (PK)		X	X
Chave Estrangeira (FK)		X	X
Nome das Tabelas			X
Nome das Colunas			X
Tipo das Colunas			X

## Modelagem de Dados

*“... o acervo de uma biblioteca é composto por exemplares de livros. Cada livro é caracterizado por um ou mais autores, um título, uma editora, local de edição, um código ISBN (único), um tipo (didático/não) e um conjunto de palavras-chave... Cada autor tem um nome e um e-mail. Cada editora tem um nome e um endereço.”*

## Modelagem de Dados

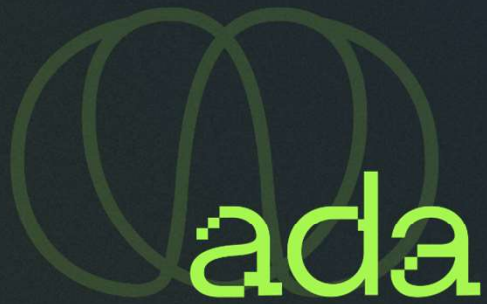
*“... o acervo de uma **biblioteca** é composto por **exemplares de livros**. Cada **livro** é caracterizado por um ou mais **autores**, um **título**, uma **editora**, **local de edição**, um **código ISBN** (único), um **tipo** (didático/não) e um conjunto de **palavras-chave**... Cada **autor** tem um **nome** e um **e-mail**. Cada **editora** tem um **nome** e um **endereço**.”*

## Modelagem de Dados

“... o acervo de uma **biblioteca** é composto por **exemplares de livros**. Cada **livro** é caracterizado por um ou mais **autores**, um **título**, uma **editora**, **local de edição**, um **código ISBN** (único), um **tipo** (didático/não) e um conjunto de **palavras-chave**... Cada **autor** tem um **nome** e um **e-mail**. Cada **editora** tem um **nome** e um **endereço**.”



Obrig.ada



Banco de Dados (MSSQL)

---

Singia - 955 | #BeTheNext - C#

O que vamos aprender...

---

O que vamos aprender...

-> Normalização de Dados e Formas Normais

-> Constraints



## Normalização de Dados e Formas Normais

-> Normalização de tabelas;

-> Ideia;

-> Objetivo principal;

-> Uso da Normalização;

-> Engenharia Reversa;

## Normalização de Dados e Formas Normais

- > 1º Forma Normal.
- > 2º Forma Normal.
- > 3º Forma Normal.
- > FNBC (Forma normal de Boyce e Codd).
- > 4º Forma Normal.
- > 5º Forma Normal;

**Diminuir a Redundância de Dados**

**Aumenta o Desempenho das Consultas**

## Normalização de Dados e Formas Normais

-> 1º Forma Normal

- > Possui chave primária;
- > Não possui grupos repetitivos;
- > Todos os seus atributos são atômicos, ou seja, não precisa ser decomposto.

## Normalização de Dados e Formas Normais

-> 2º Forma Normal

- > Deve estar na primeira forma normal;
- > Não deve haver dependência funcional parcial;
- > Dependência funcional: atributo(s) não-chave depende(m) da chave primária.



## Normalização de Dados e Formas Normais

-> 3º Forma Normal

- > Deve estar na segunda forma normal;
- > Se nenhum dos campos foram determinados transitivamente pela chave primária;
- > Dependência funcional transitiva: atributo(s) não-chave depende(m) de outro(s) atributo(s) não-chave.

## Normalização de Dados e Formas Normais

Formal Normal	Teste	Solução (normalização)
1FN	A relação não deve ter qualquer atributo não-atômico nem relações agrupadas.	Forme novas relações para cada atributo não-atômico ou relação aninhada.
2FN	Para relações nas quais a chave primária contém múltiplos atributos, nenhum atributo não-chave deve ser funcionalmente dependente de uma parte da chave primária.	Decomponha e monte uma relação para cada chave parcial com seu(s) atributo(s) dependente(s). Certifique-se de manter uma relação com a chave primária original e quaisquer atributos que sejam completamente dependentes dela em termos funcionais.
3FN	A relação não deve ter um atributo não-chave funcionalmente determinado por um outro atributo não -chave (ou por um conjunto de atributos não-chave). Ou seja, não deve haver dependência transitiva de um atributo não chave na chave primária.	Decomponha e monte uma relação que inclua o(s) atributo(s) não-chave que funcionalmente determine(m) outros atributos não-chave.

## Constraints – NOT NULL

```
CREATE TABLE cliente (  
    cod_cli INTEGER NOT NULL,  
    cpf CHAR(11) UNIQUE,  
    nome_cli VARCHAR(40) NOT NULL,  
    idade INTEGER CHECK (idade = 18)  
    endereco VARCHAR(40) null,  
    cidade VARCHAR(20) null,  
    cep CHAR(8) null,  
    uf CHAR(2) null,  
    ativo BIT null  
);
```

## Constraints – UNIQUE

```
CREATE TABLE cliente (  
    cod_cli INTEGER NOT NULL,  
    cpf CHAR(11) UNIQUE,  
    nome_cli VARCHAR(40) NOT NULL,  
    idade INTEGER CHECK (idade = 18)  
    endereco VARCHAR(40) null,  
    cidade VARCHAR(20) null,  
    cep CHAR(8) null,  
    uf CHAR(2) null,  
    ativo BIT null  
);
```



## Constraints – DEFAULT

```
CREATE TABLE conta (  
    cod_conta BIGINT NOT NULL,  
    saldo DOUBLE DEFAULT 0.0  
);
```

## Constraints – CHECK

```
CREATE TABLE cliente (  
    cod_cli INTEGER NOT NULL,  
    cpf CHAR(11) UNIQUE,  
    nome_cli VARCHAR(40) NOT NULL,  
    idade INTEGER CHECK (idade = 18)  
    endereco VARCHAR(40) null,  
    cidade VARCHAR(20) null,  
    cep CHAR(8) null,  
    uf CHAR(2) null,  
    ativo BIT null  
);
```

## Constraints – CHAVE PRIMÁRIA

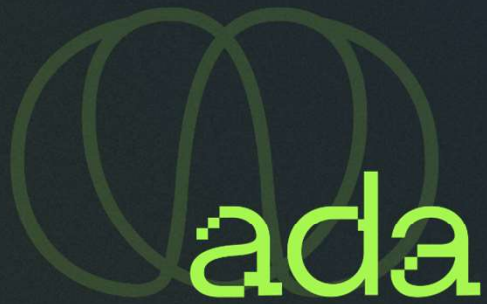
```
CREATE TABLE letscode_genero (  
    id_genero INTEGER NOT NULL,  
    nome VARCHAR(40) NOT NULL,  
    PRIMARY KEY (id_genero)  
);
```

## Constraints – CHAVE ESTRANGEIRA

```
CREATE TABLE letscode_participacao (  
    id_participacao INTEGER NOT NULL,  
    id_filme INTEGER NOT NULL,  
    id_ator INTEGER NOT NULL,  
    FOREIGN KEY (id_filme)  
        REFERENCES letscode_filme (id_filme),  
    FOREIGN KEY (id_ator)  
        REFERENCES letscode_ator (id_ator),  
    PRIMARY KEY (id_participacao)  
);
```



Obrig.ada



Banco de Dados (MSSQL)

---

Singia - 955 | #BeTheNext - C#

O que vamos aprender...

---

O que vamos aprender...

-> Funções Agregadas



## Funções Agregadas – MIN() e MAX()

```
SELECT MIN(valor_unitario) FROM tb_entrada_produto;
```

```
SELECT MAX(valor_unitario) FROM tb_entrada_produto;
```

## Funções Agregadas – COUNT()

```
SELECT COUNT(*) FROM tb_saida_produto;
```

## Funções Agregadas – SUM()

```
SELECT qtde, SUM(valor_unitario) FROM tb_saida_produto  
GROUP BY 1;
```

## Funções Agregadas – AVG()

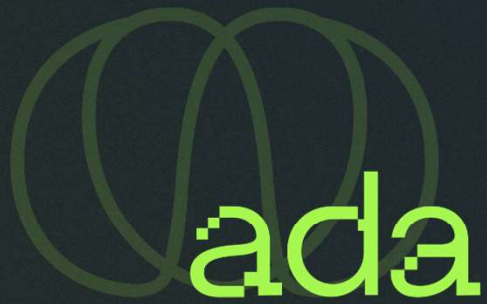
```
SELECT valor_unitario, AVG(qtde) FROM tb_entrada_produto  
GROUP BY 1;
```



## Funções Agregadas – HAVING

```
SELECT valor_unitario, COUNT(*) FROM tb_estoque  
GROUP BY 1  
HAVING COUNT(*) > 2;
```

Obrig.ada



Banco de Dados (MSSQL)

---

Singia - 955 | #BeTheNext - C#

O que vamos aprender...

---



O que vamos aprender...

-> Uso da Expressão WHERE

-> JOIN

-> UNION

## Uso da Expressão WHERE

-> Operadores de comparação:

=, <>, <, <=, >, >=

<coluna> BETWEEN 'valorInicial' AND 'valorFinal'

<coluna> IN ('valor 1', ..., 'valor n')

<coluna> LIKE '%banco de %'

<coluna> LIKE '19\_ \_'

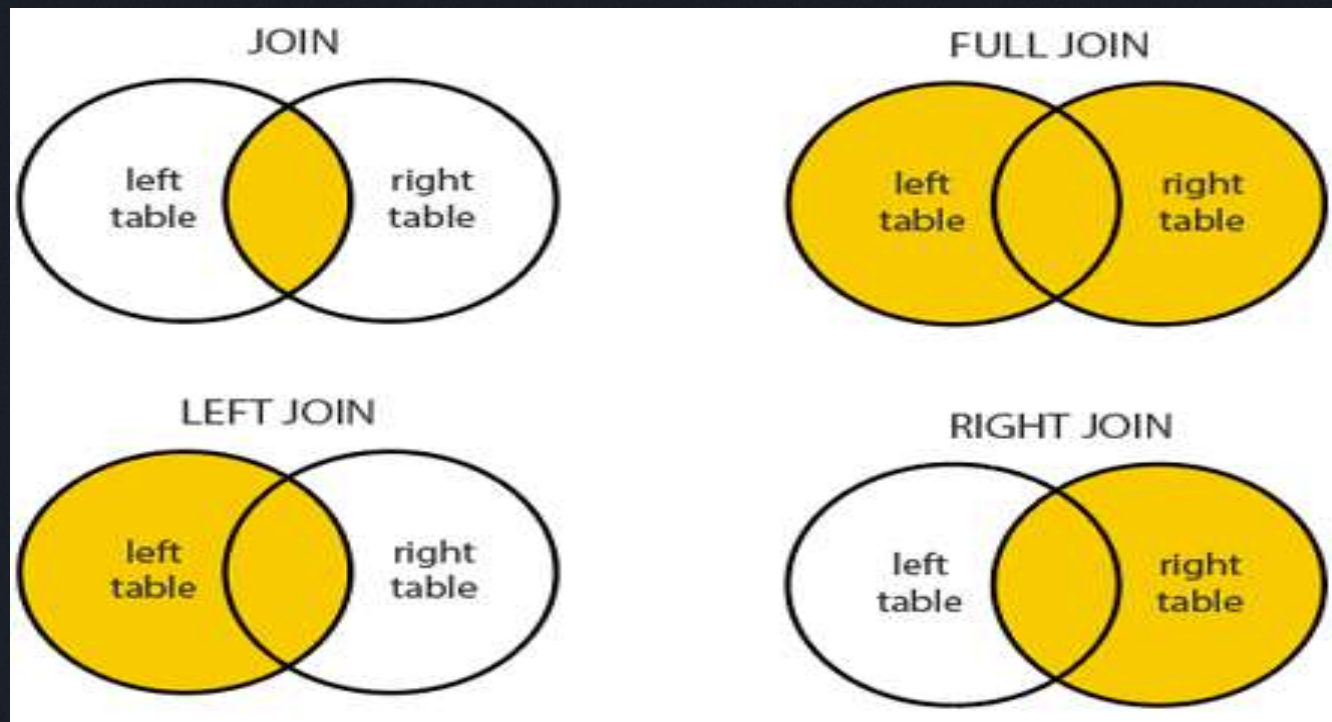
<coluna> IS [NOT] NULL

"\_" = um caracter

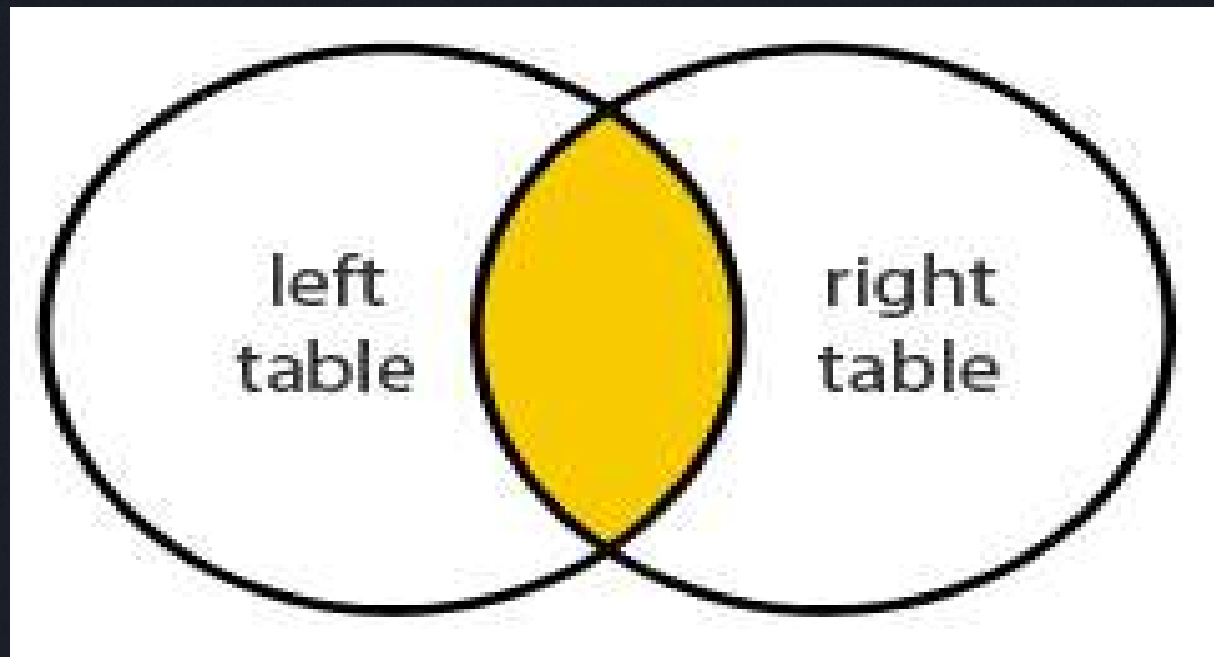
"%" = sequência de caracteres

Operadores lógicos: AND e OR

## Introdução a 'Otimização de Consultas no Banco de Dados'

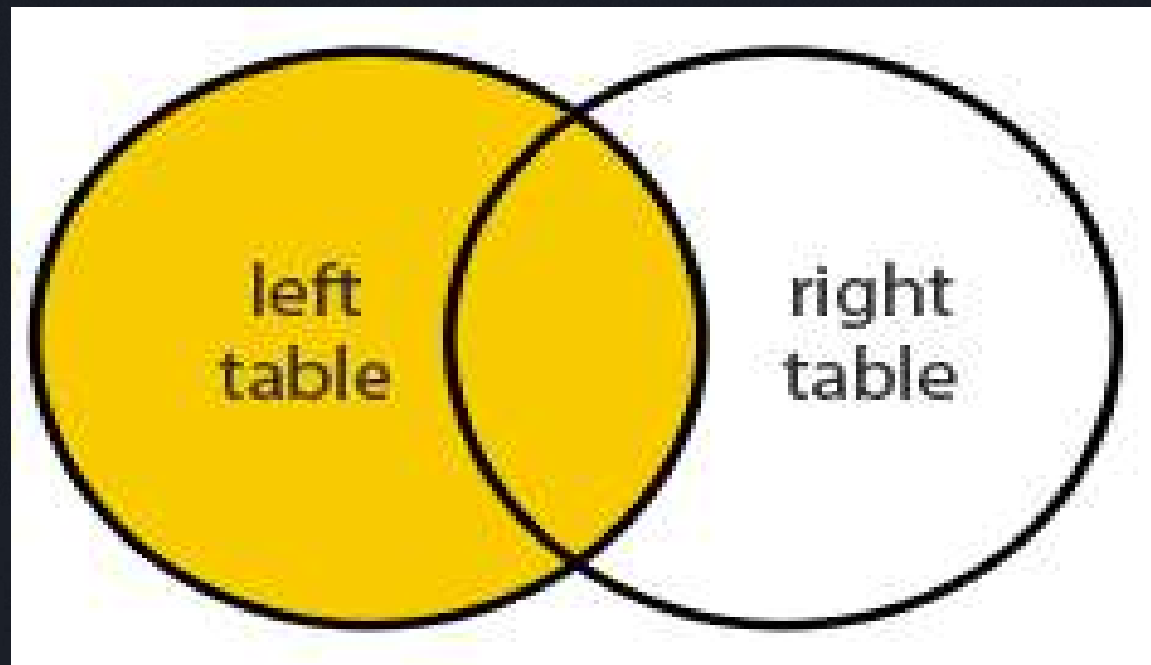


## Introdução a 'Otimização de Consultas no Banco de Dados' – INNER JOIN

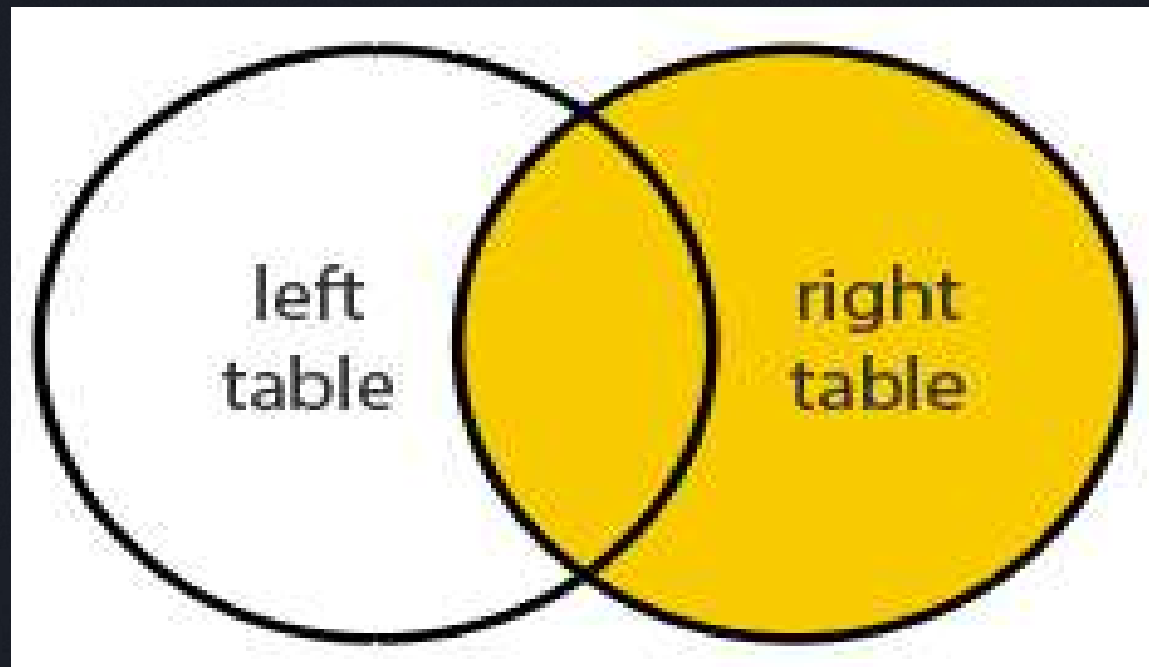




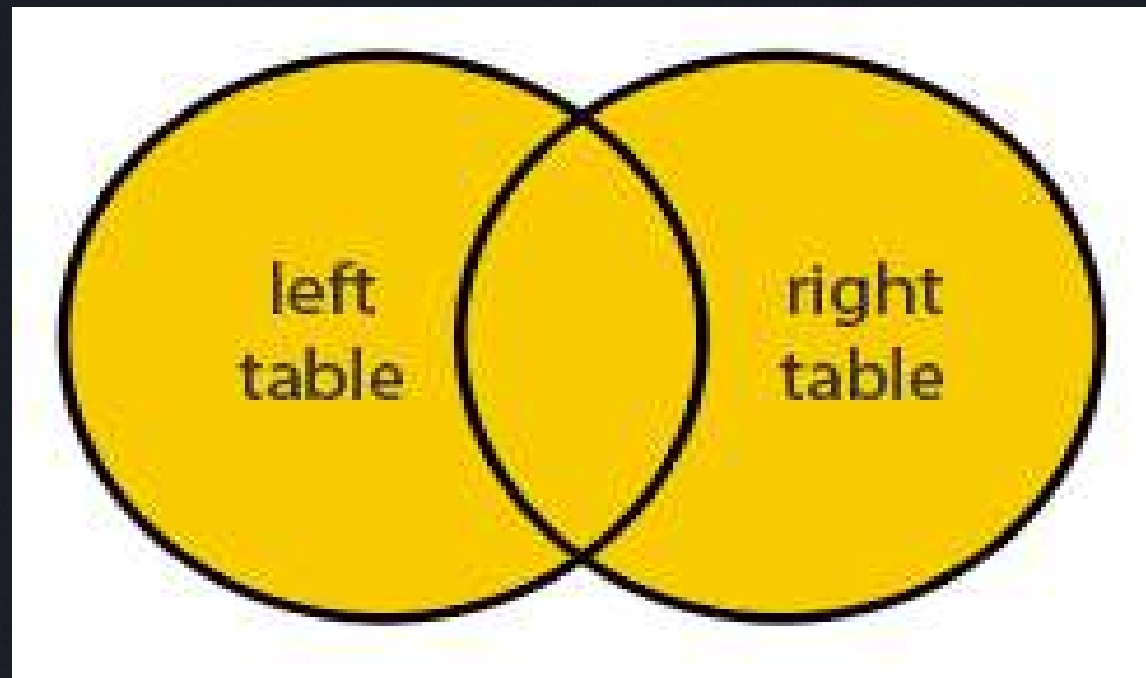
## Introdução a 'Otimização de Consultas no Banco de Dados' – LEFT JOIN



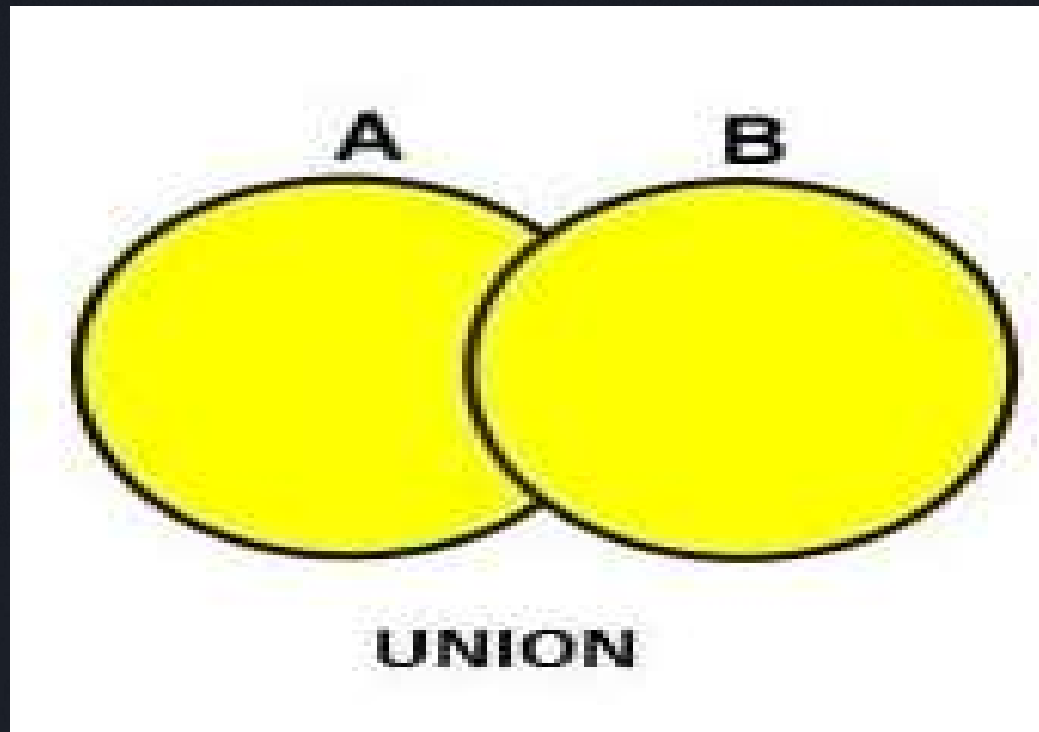
## Introdução a 'Otimização de Consultas no Banco de Dados' – RIGHT JOIN



## Introdução a 'Otimização de Consultas no Banco de Dados' – FULL JOIN

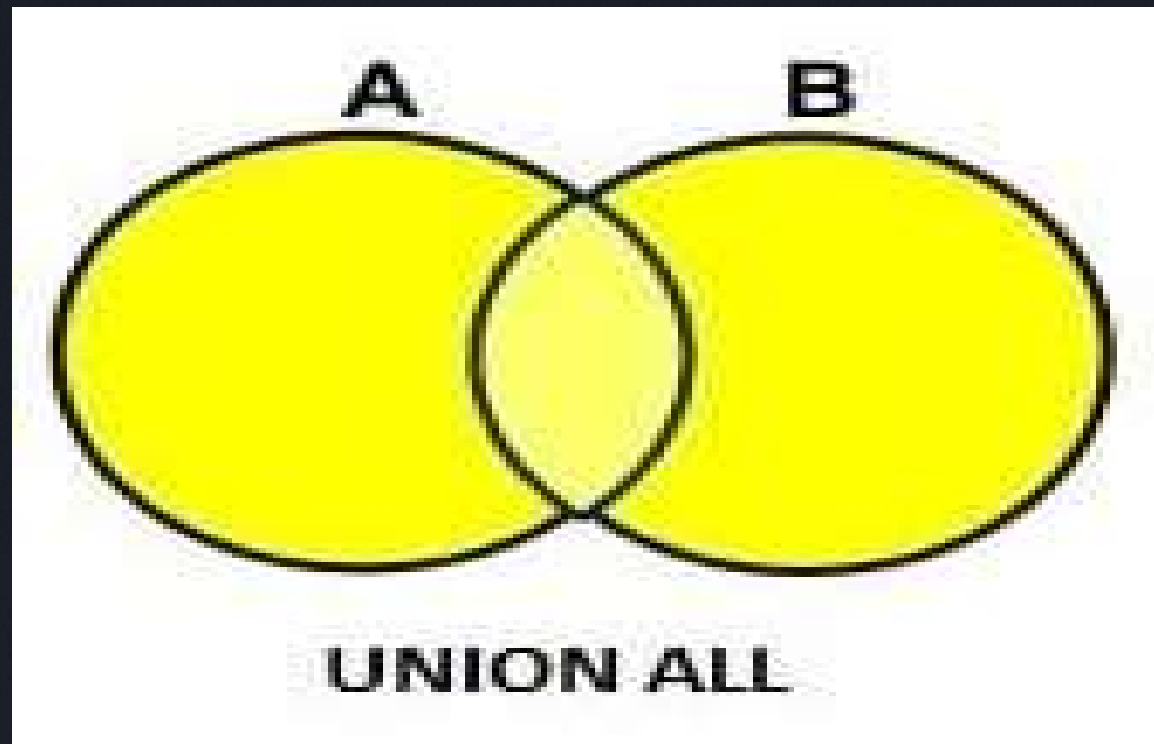


## Introdução a 'Otimização de Consultas no Banco de Dados' – UNIONS





## Introdução a 'Otimização de Consultas no Banco de Dados' – UNIONs



Obrig.ada