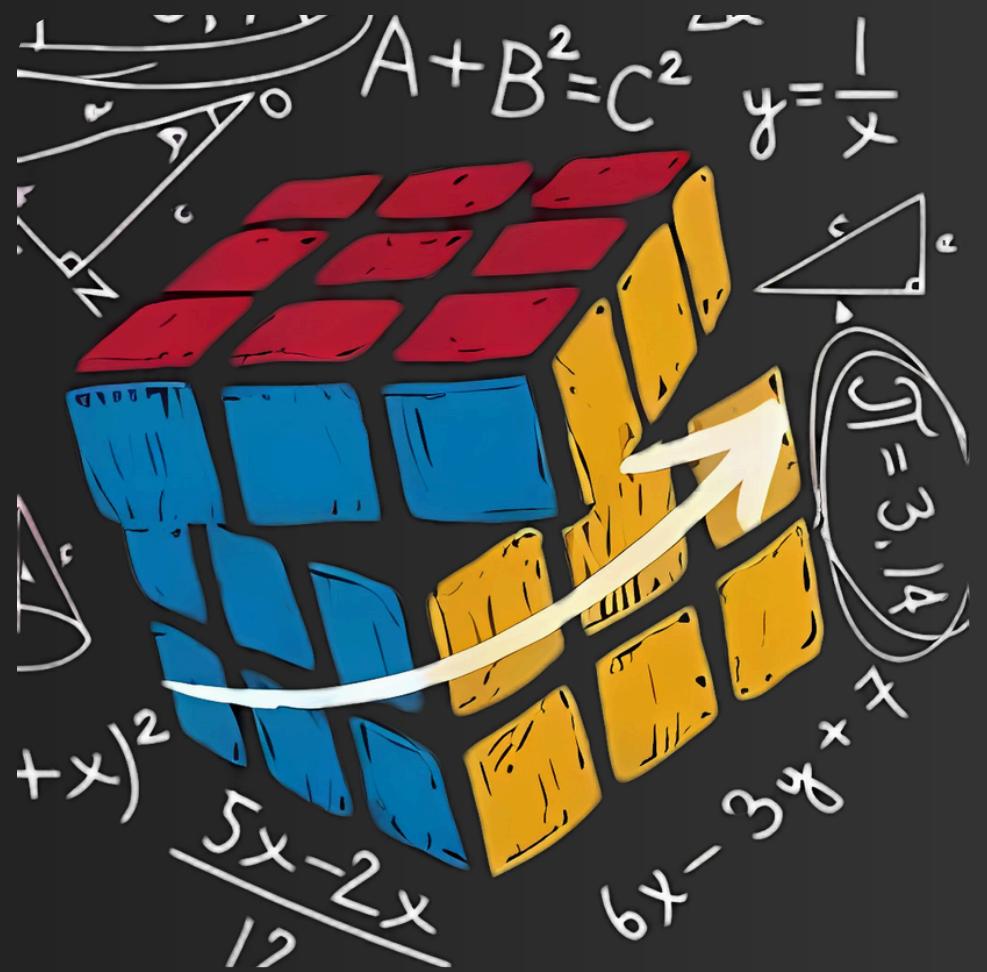


Do **caos** à **clareza**:  
preparando dados  
para o BI

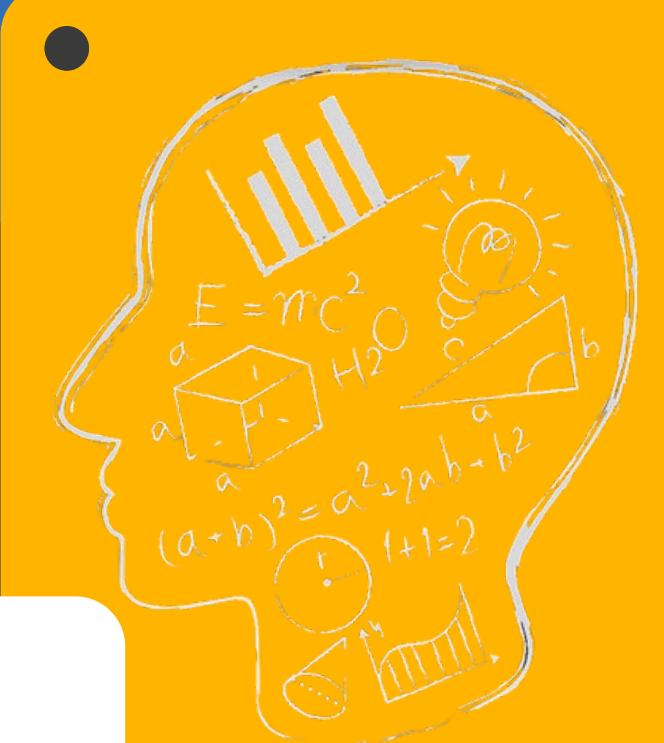
# TRATAMENTO DE DADOS



# 1. Parte Teórica



Antes de BI, é preciso entender os dados.



# Dado vs Informação

## O que é?

Dados e informações são frequentemente usados como sinônimos, mas representam conceitos distintos:

- **Dados:** Material bruto ou não processado.
- **Informação:** conhecimento, inteligência, dado com um significado ou uma função especial. Geralmente, é um resultado como combinar, comparar, analisar ou executar cálculos.

• Dados: são registros brutos, sem contexto ou processamento.

Ex: "35", "SP", "João", "01/02/2024".

• Informação: é o resultado do processamento dos dados, com significado e utilidade.

Ex: "João, de SP, comprou 35 unidades em 01/02/2024".



# Alguns conceitos importantes

- O que é um banco de dados?

- Um banco de dados é um conjunto estruturado de dados armazenados em sistemas computacionais, que permite adicionar, recuperar, modificar e excluir dados com eficiência.
- Esses dados podem ser convertidos em informações úteis, como estatísticas, médias ou relatórios.

- Sim! Praticamente todos os aplicativos e sites que utilizados hoje armazenam dados em bancos: redes sociais, mensagens, transporte, e-commerce, streaming — tudo depende de dados.
- E mais: na era da informação, dado virou ativo financeiro!

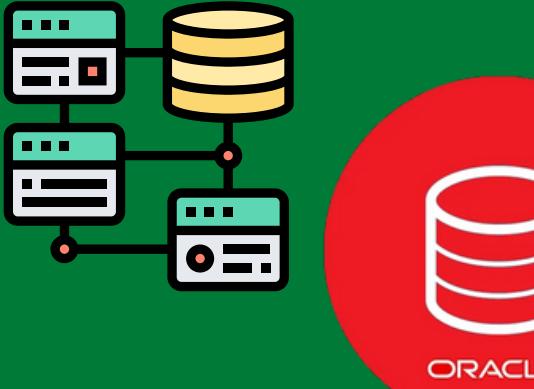
- Tem relação com o meu dia a dia?

## Modelagem de dados

- A modelagem de dados é o primeiro passo na construção de um banco de dados eficiente.
- Envolve as etapas mínimas de coleta, análise e diagramação de dados que resolvem um certo problema de uma empresa ou grupo de pessoas, ou melhor, atendam a um requisito de negócio (linguagem mais formal da área técnica).

# Tipos de Banco de Dados

## Relacional (SQL)



Estruturado em tabelas com **colunas** e **linhas**

Usa chaves **primárias** e **estrangeiras**

Linguagem: **SQL**

**Exemplos:** MySQL, PostgreSQL, SQL Server, Oracle

## Não Relacional (NoSQL)



**mongo DB**

Flexível, ideal para **dados não estruturados**

### Tipos:

- Documentos: MongoDB, CouchDB
- Chave-Valor: Redis, DynamoDB
- Colunar: Cassandra, HBase
- Grafos: Neo4j

## Temporal



**ORACLE**

Flashback Technologies

Armazena o **histórico das alterações dos dados**

Útil para **auditorias e versões**

**Exemplo:** Oracle Flashback, Bitemporal DBs

## Distribuído



**Google Bigtable**

Dados **armazenados em múltiplos servidores**

Alta **disponibilidade e escalabilidade**

**Exemplos:** Google Bigtable, CockroachDB, Cassandra

## Orientado a Objetos



**OP**

Dados **representados como objetos**, como em linguagens de programação

**Exemplo:** db4o, ObjectDB

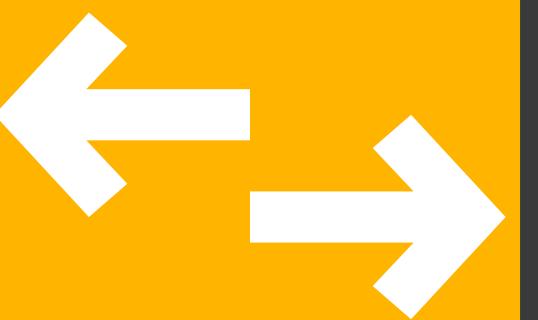
“ —

**Bancos relacionais** são os **mais usados** em sistemas corporativos tradicionais, enquanto os **NoSQL** ganham força em **Big Data** e **aplicações web escaláveis**.

# Diferença entre Banco

## Banco Transacional (OLTP)

- Usado para **registrar operações do dia a dia** do sistema.
- **Alta frequência de inserções, atualizações e exclusões.**
- Foco em **velocidade e integridade dos dados.**
- Estrutura **altamente normalizada** (para evitar redundância).
- **Exemplo de uso:**
  - Sistemas bancários, ERPs, CRMs, e-commerces.



## Banco Analítico (OLAP)

- Usado para **consulta e análise de grandes volumes de dados.**
- Foco em **leitura otimizada e desempenho em consultas complexas.**
- Dados históricos, agregados e **organizados por tempo**, local, categoria etc.
- **Estrutura desnormalizada** (com tabelas fato e dimensão).

### • Exemplo de uso:

- Consumo de dados em Dashboards



“Enquanto o banco **transacional** responde ‘O que está acontecendo agora?’, o banco **analítico** responde ‘O que aconteceu nos últimos meses ou anos e o que podemos aprender com isso?’”

# Níveis de modelagem

## Conceitual

**Representa o que será modelado**, focando nas entidades e seus relacionamentos, sem se preocupar com detalhes técnicos.

## Lógica

**Transforma o modelo conceitual em uma estrutura lógica**, com foco em atributos, chaves primárias e estrangeiras, tipos de dados e restrições.

## Física

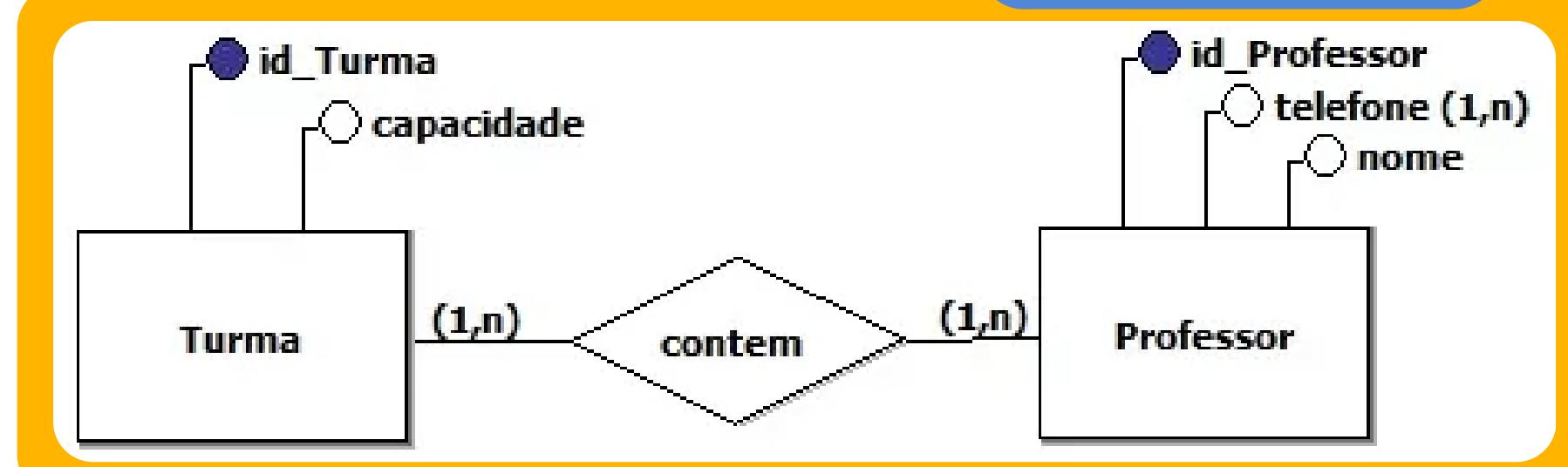
**É a implementação real do modelo em um banco de dados** específico, considerando otimizações, índices, performance e recursos do SGBD.

# Modelagem Conceitual: ER Diagram

## O que é?

- O **Diagrama Entidade-Relacionamento (ER)** é uma representação visual dos elementos principais de um sistema: entidades (como "Cliente", "Produto") e os relacionamentos entre elas.
- Ele ajuda a **entender a estrutura dos dados** antes de implementá-los em um banco.
- É o **primeiro passo na modelagem de dados** e não depende de tecnologia específica.

## Exemplo



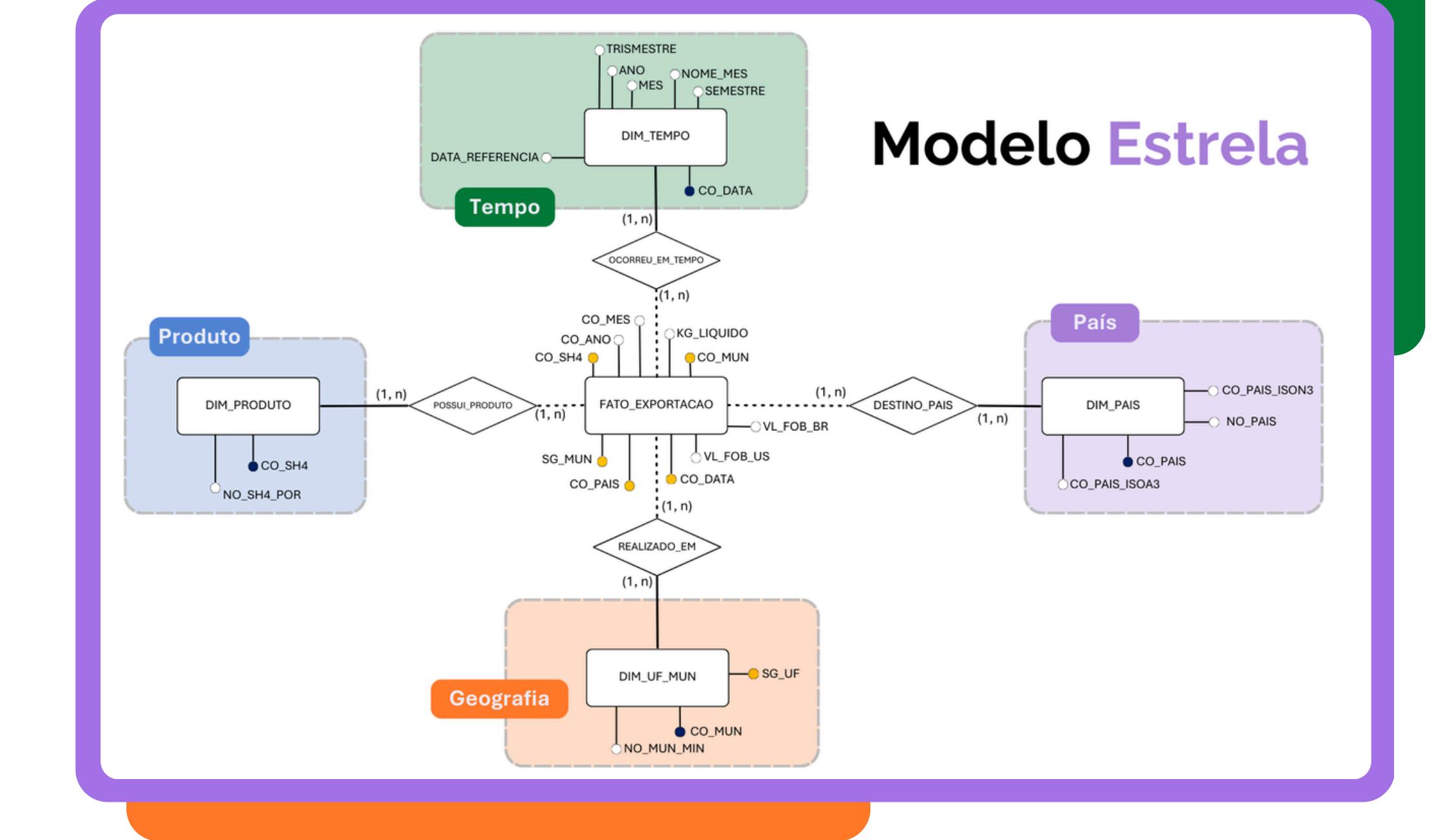
## Ferramentas disponíveis

- **dbdiagram.io** – simples, online e com exportação para SQL.
- **Diagrams.net (Draw.io)** – livre, flexível, permite criar DER manualmente.

# Nosso Modelo

Modelagem  
Conceitual:  
ER Diagram

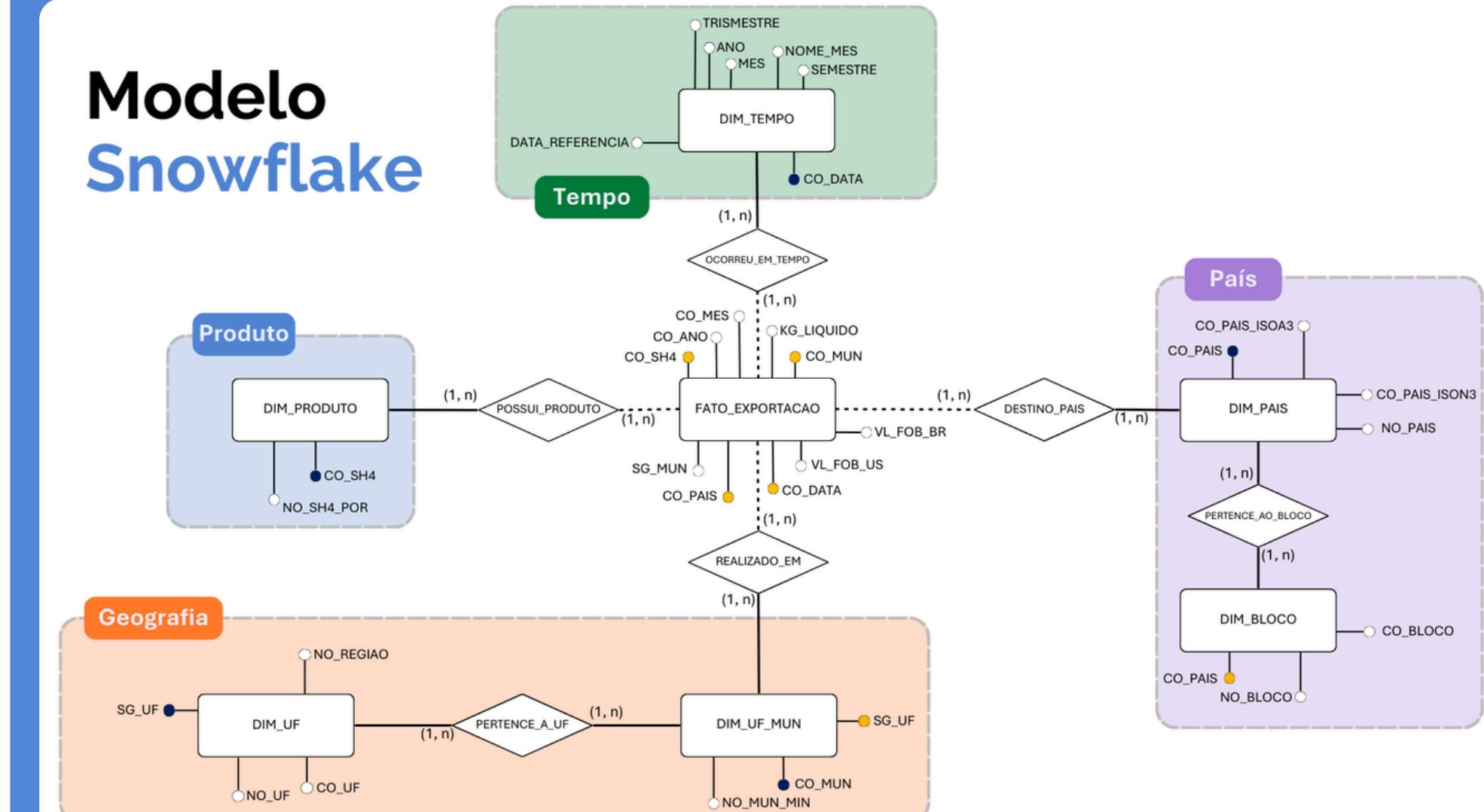
## Modelo Estrela



# Nosso Modelo

# Modelagem Conceitual: ER Diagram

## Modelo Snowflake



# Modelagem Lógica

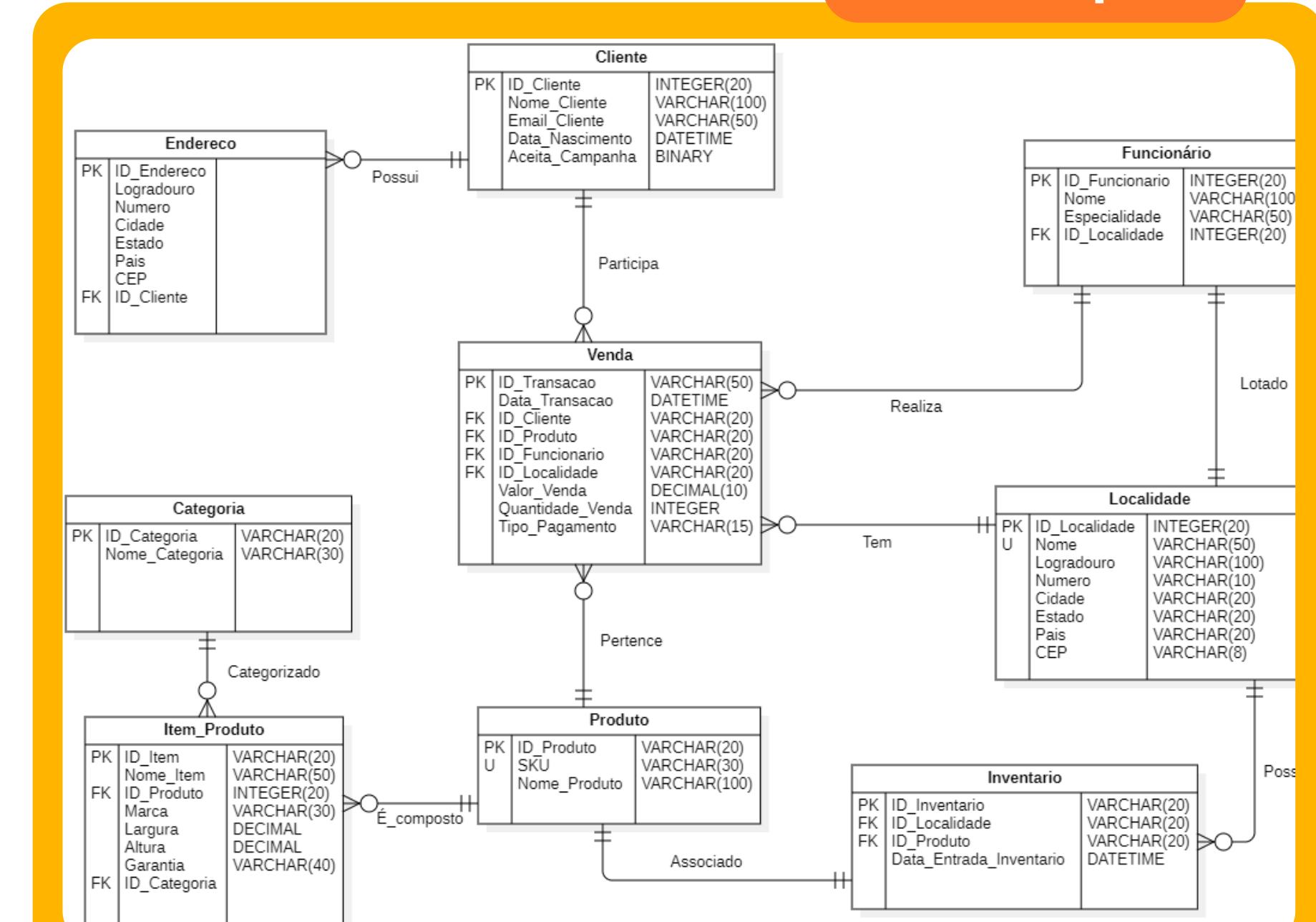
## Exemplo

### O que é?

- A **modelagem lógica define como será a relação dos dados no banco**, incluindo atributos (colunas), tipos de dados, chaves primárias e estrangeiras.
- Ela não depende de um **SGBD específico**, mas já respeita as regras de banco de dados relacionais.
- É a **ponte entre a ideia** (modelo conceitual) e a **implementação** (modelo físico).

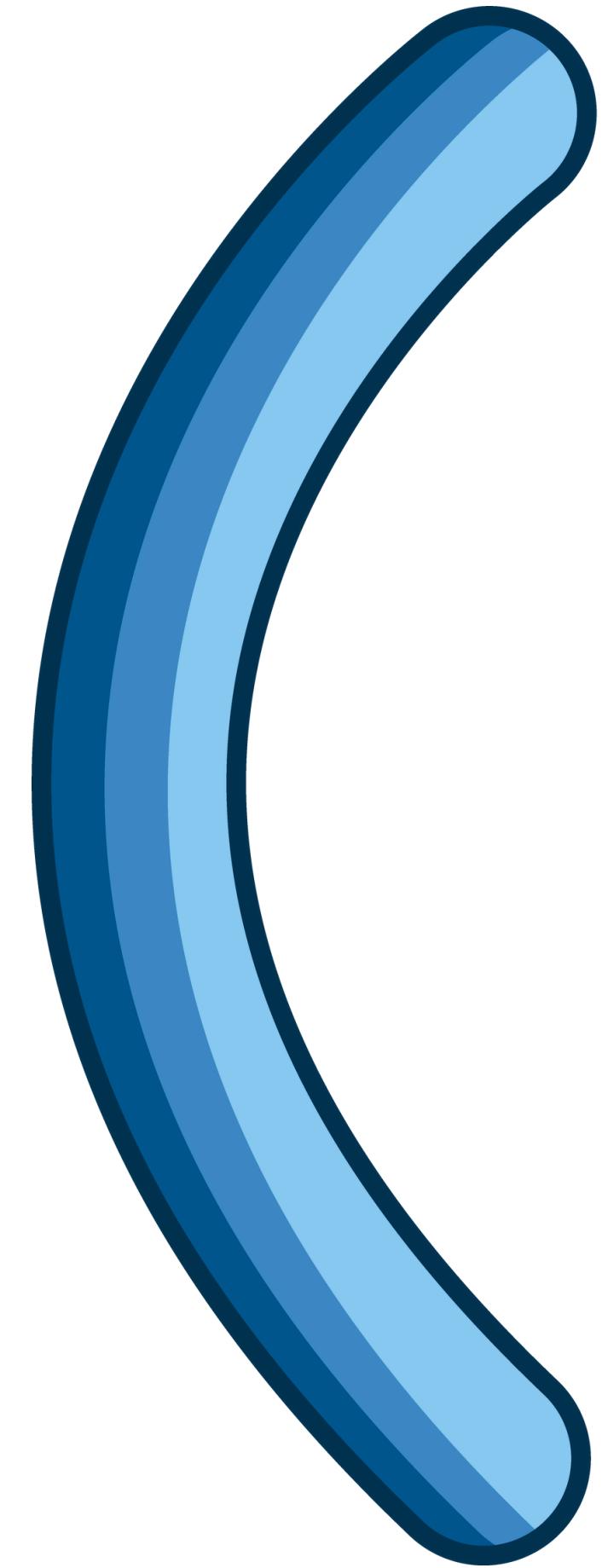
### Ferramentas disponíveis

- [dbdiagram.io](#) – gera modelos lógicos com sintaxe simplificada.
- [DBeaver](#) – permite visualizar e criar estrutura lógica com suporte a vários bancos.



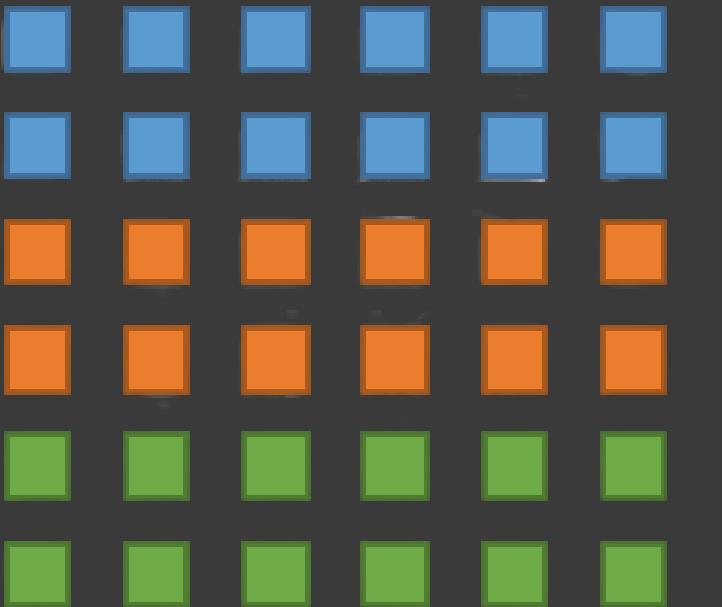
**SGBD:** Sistema  
Gestor de  
Banco de  
Dados

**Abrindo  
parênteses....**



# Dados Estruturados

## Estruturado

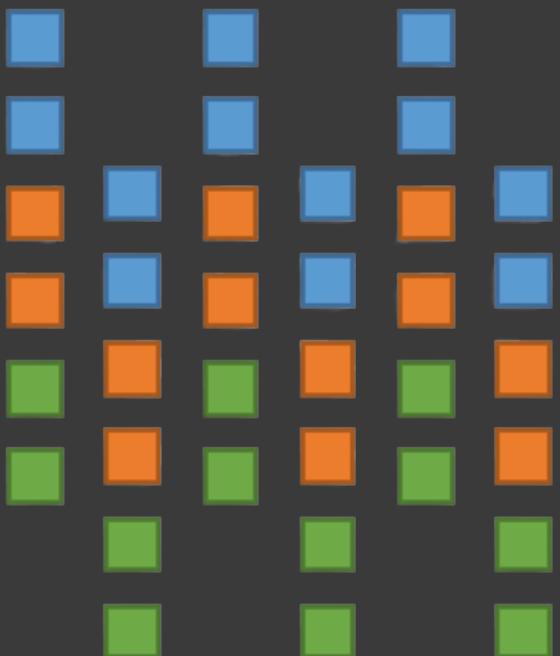


“—

À medida que os dados ficam menos estruturados, a complexidade para tratá-los aumenta — mas o valor potencial também!

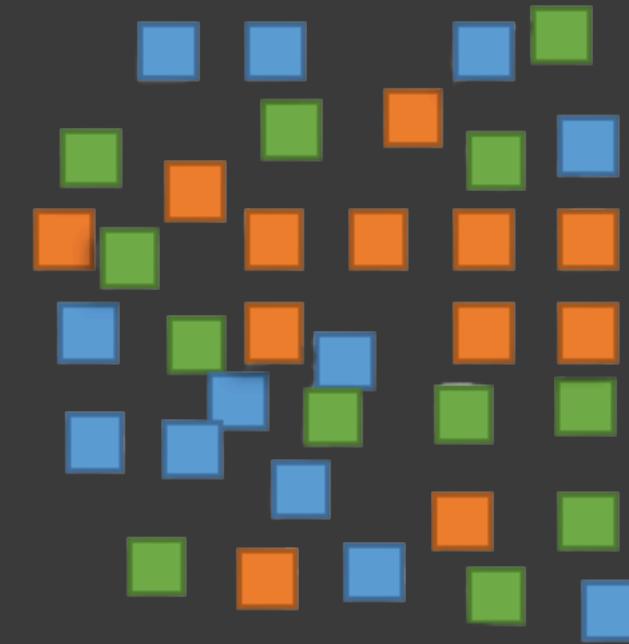
- Fácil de armazenar, consultar e analisar.
  - Ex: bancos de dados relacionais (MySQL, PostgreSQL, Oracle).

## Semi Estruturado



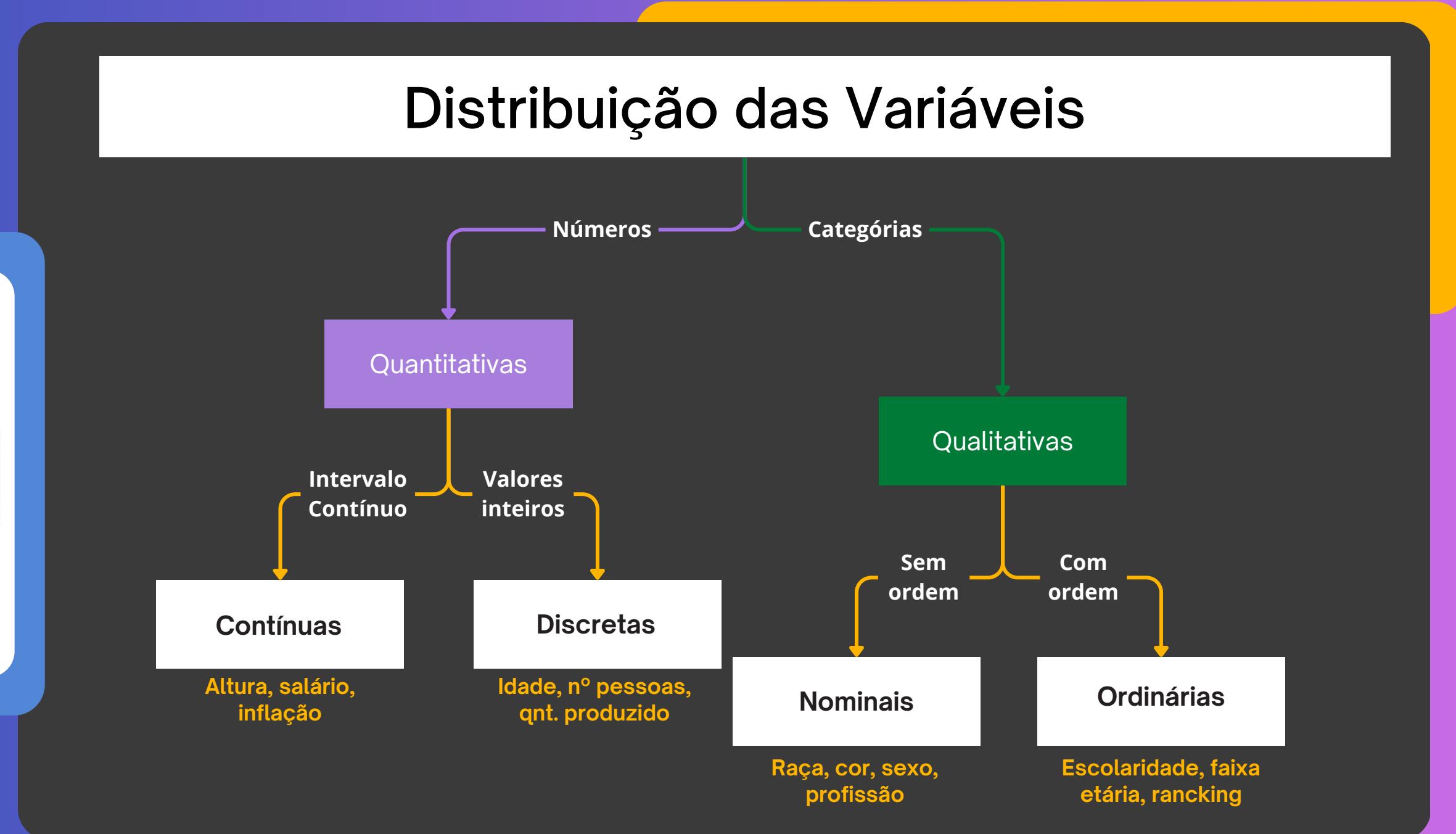
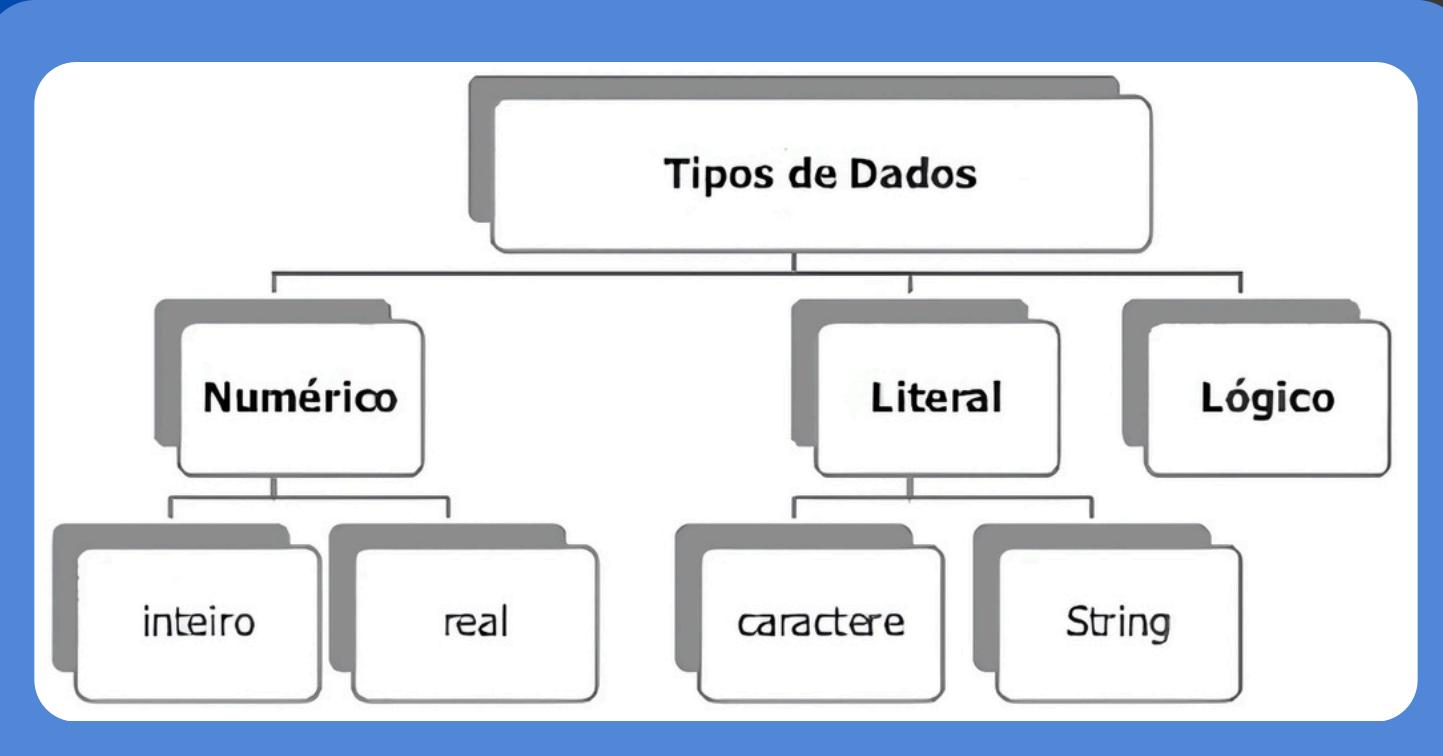
- Mais flexível que o modelo relacional.
  - Ex: arquivos JSON, XML, planilhas com abas variadas,

## Não Estruturado



- Difíceis de processar diretamente.
  - Ex: imagens, vídeos, áudios, PDFs, posts em redes sociais.

# Do Conceito ao Banco: Variáveis e Seus Tipos de Dados



# Entendendo: Tabelas Fato e Dimensão

O que é?

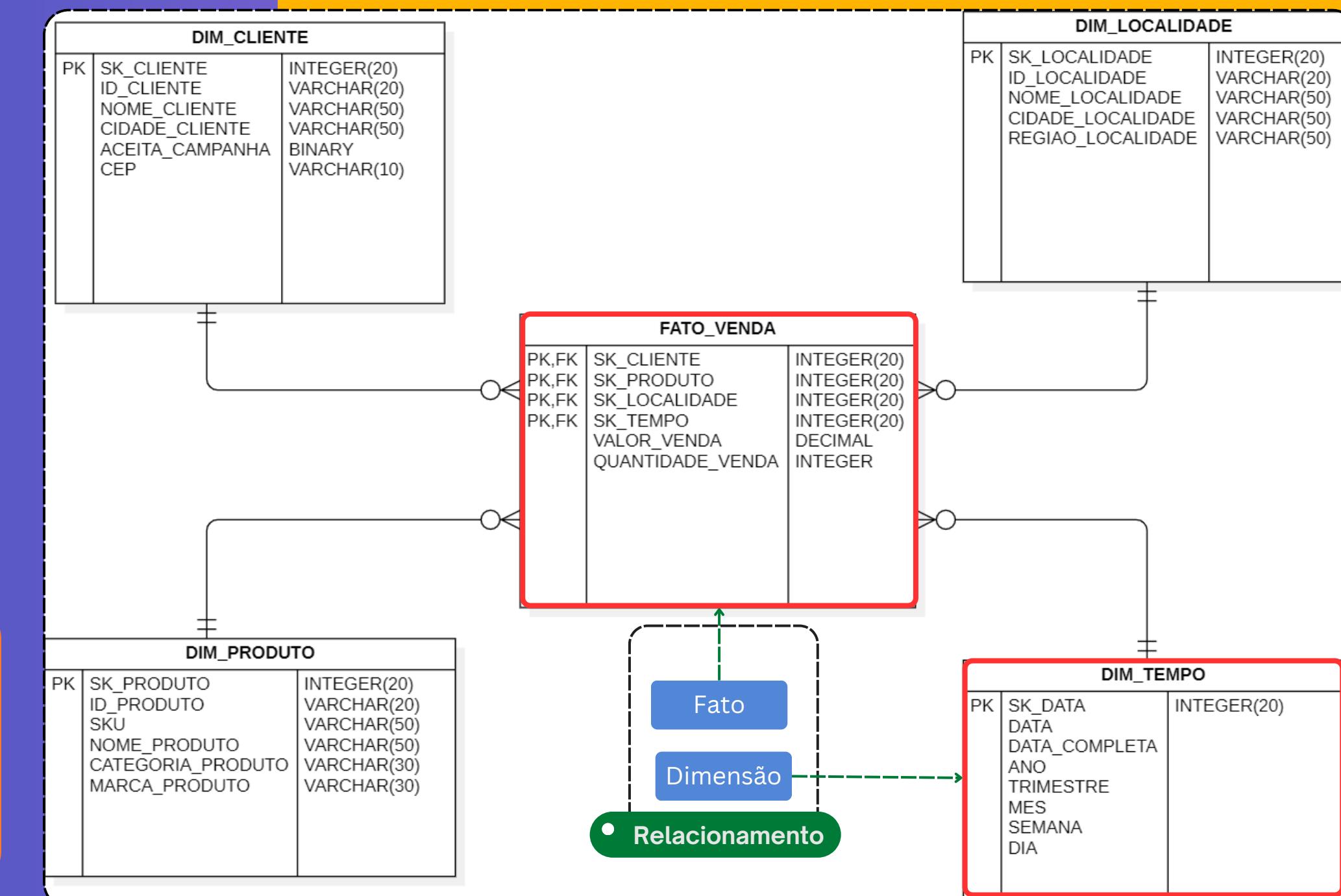
- Armazena informações descritivas relacionadas aos fatos, como "nome do cliente", "cidade", "data", "produto".
- São usadas para filtrar, agrupar ou categorizar as análises no BI.

Componentes da tabela

	FATO_VENDA		
Restrições	PK,FK	SK_CLIENTE	INTEGER(20)
	PK,FK	SK_PRODUTO	INTEGER(20)
	PK,FK	SK_LOCALIDADE	INTEGER(20)
	PK,FK	SK_TEMPO	INTEGER(20)
Atributo		VALOR_VENDA	DECIMAL
		QUANTIDADE_VENDA	INTEGER

O que é?

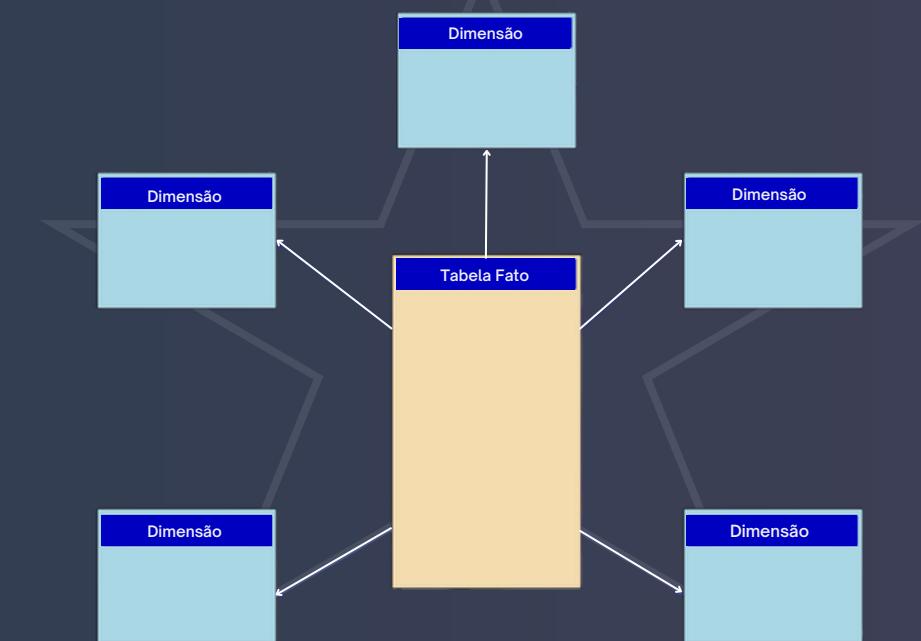
- Armazena os eventos ou transações do negócio (ex: uma venda).
- Contém medidas numéricas que serão analisadas (como quantidade ou valor), e chaves estrangeiras que se conectam às dimensões.



# Modelos estrela e snowflake

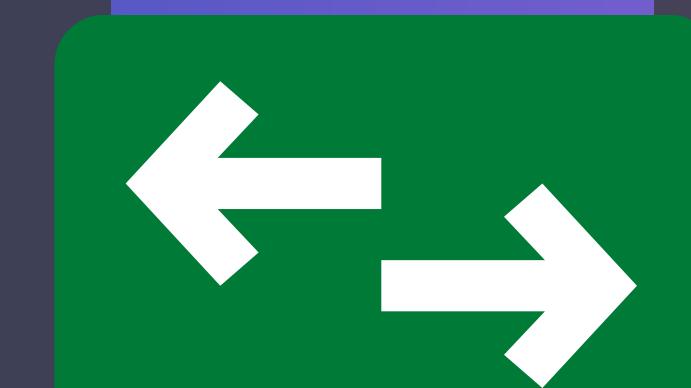
## Estrela

- O modelo estrela é o mais comum em ambientes de BI.
- Nele, uma **tabela fato** fica no centro, conectada diretamente a **várias tabelas dimensão**.
- Ele é **mais simples**, fácil de entender e rápido para consultas, mas **pode ter redundância de dados** nas dimensões.

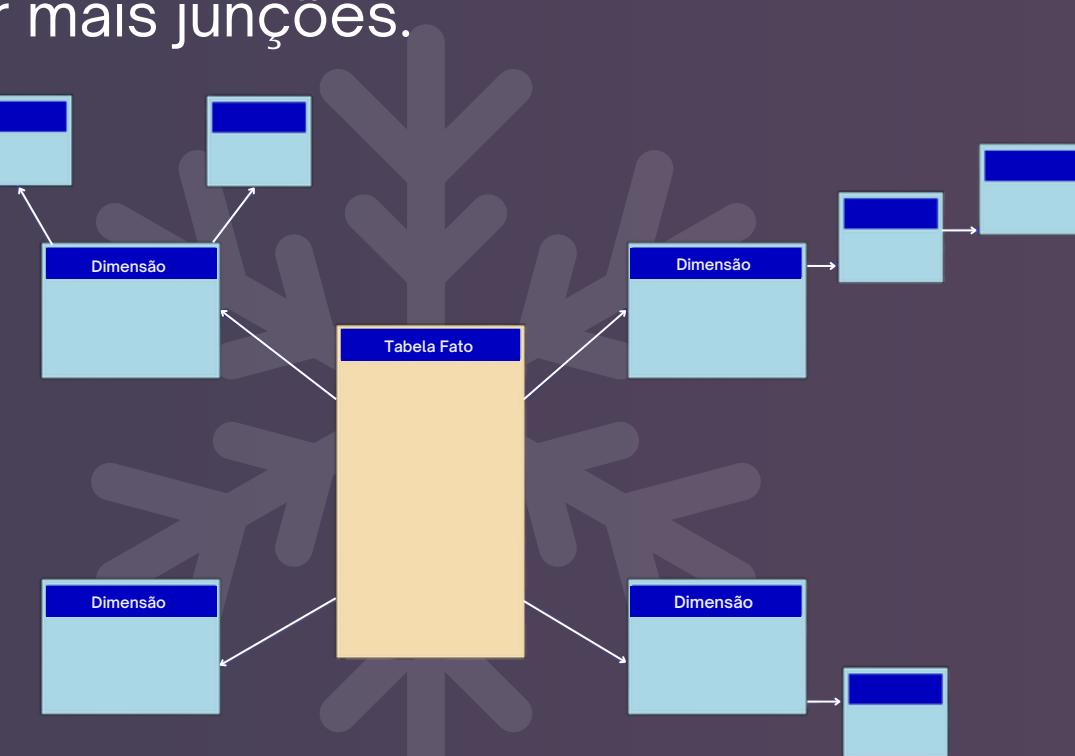


## Snowflake

- O modelo snowflake (foco de neve) é uma **variação mais normalizada** do modelo estrela.
- As **dimensões são divididas em subdimensões**, reduzindo redundância.
- Ele é **mais otimizado em espaço**, mas pode ser **mais lento em consultas** por exigir mais junções.



**“** —  
O modelo estrela prioriza desempenho. O snowflake prioriza organização e normalização.





**Fechando  
parênteses....**

# Nosso Modelo

Modelagem  
Lógica

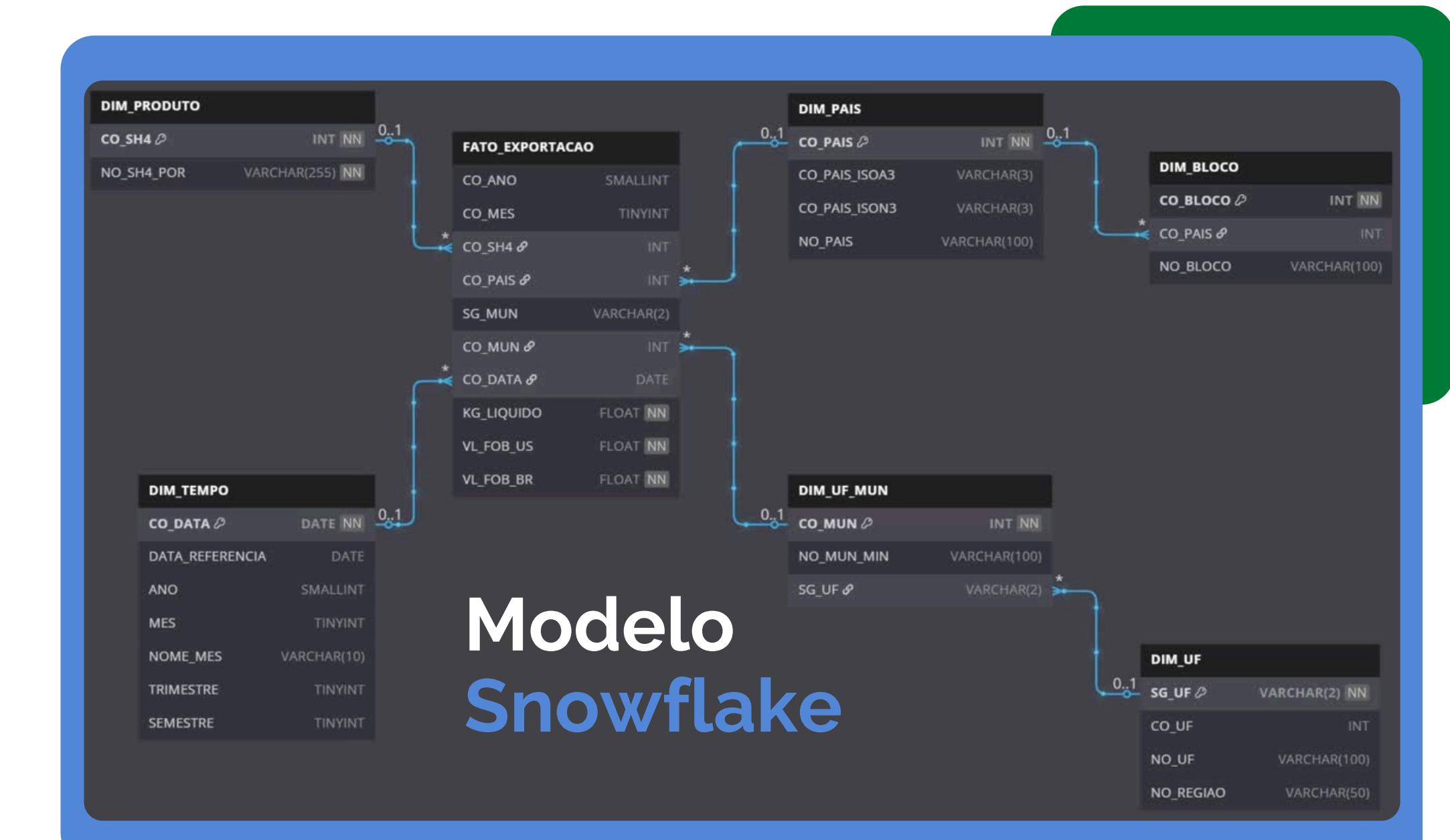


Modelo Estrela

[Acessando o modelo](#)

# Nosso Modelo

Modelagem  
Lógica



[Acessando o modelo](#)

# Modelagem Física

## O que é?

- A modelagem física é a **etapa em que o modelo lógico é implementado em um banco de dados real**, como PostgreSQL, Oracle ou MySQL.
- Envolve a **criação das tabelas, definição de índices, tipos de dados específicos, partições** e outras configurações que afetam a performance, integridade e armazenamento.
- Aqui, **decisões como usar VARCHAR(50) ou TEXT, criar índices, e definir estratégias de particionamento são tomadas com base nas necessidades do sistema**.

## Exemplo

```
CREATE TABLE FATO_VENDA (
    ID_VENDA INT PRIMARY KEY,
    SK_CLIENTE INT,
    SK_PRODUTO INT,
    VALOR_VENDA DECIMAL(10,2),
    DATA_VENDA DATE
);
```

## Ferramentas disponíveis

- **DBeaver** (open-source, com suporte a múltiplos bancos)
- **pgAdmin** (para PostgreSQL, muito usado em produção)
- **MySQL Workbench** (modelagem e execução de scripts no MySQL)
- **Oracle SQL Developer** (para bancos Oracle)
- **HeidiSQL** (leve, bom para MariaDB e MySQL)



```
CREATE TABLE `FATO_EXPORTACAO` (
  `CO_ANO` SMALLINT,
  `CO_MES` TINYINT,
  `CO_SH4` INT,
  `CO_PAIS` INT,
  `SG_MUN` VARCHAR(2),
  `CO_MUN` INT,
  `CO_DATA` DATE,
  `KG_LIQUIDO` FLOAT NOT NULL,
  `VL_FOB_US` FLOAT NOT NULL,
  `VL_FOB_BR` FLOAT NOT NULL
);

CREATE TABLE `DIM_TEMPO` (
  `CO_DATA` DATE PRIMARY KEY NOT NULL,
  `DATA_REFERENCIA` DATE,
  `ANO` SMALLINT,
  `MES` TINYINT,
  `NOME_MES` VARCHAR(10),
  `TRIMESTRE` TINYINT,
  `SEMESTRE` TINYINT
);

CREATE TABLE `DIM_PRODUTO` (
  `CO_SH4` INT PRIMARY KEY NOT NULL,
  `NO_SH4_POR` VARCHAR(255) NOT NULL
);

CREATE TABLE `DIM_PAIS` (
  `CO_PAIS` INT PRIMARY KEY NOT NULL,
  `CO_PAIS_ISO3` VARCHAR(3),
  `CO_PAIS_ISON3` VARCHAR(3),
  `NO_PAIS` VARCHAR(100)
);

CREATE TABLE `DIM_UF_MUN` (
  `CO_MUN` INT PRIMARY KEY NOT NULL,
  `NO_MUN_MIN` VARCHAR(100),
  `SG_UF` VARCHAR(2)
);

ALTER TABLE `FATO_EXPORTACAO` ADD FOREIGN KEY (`CO_DATA`) REFERENCES `DIM_TEMPO`(`CO_DATA`);

ALTER TABLE `FATO_EXPORTACAO` ADD FOREIGN KEY (`CO_SH4`) REFERENCES `DIM_PRODUTO`(`CO_SH4`);

ALTER TABLE `FATO_EXPORTACAO` ADD FOREIGN KEY (`CO_PAIS`) REFERENCES `DIM_PAIS`(`CO_PAIS`);

ALTER TABLE `FATO_EXPORTACAO` ADD FOREIGN KEY (`CO_MUN`) REFERENCES `DIM_UF_MUN`(`CO_MUN`);
```

## Modelo Estrela

Introdução rápida à modelagem de dados

Nosso Modelo

Modelagem  
Física

```
CREATE TABLE `FATO_EXPORTACAO` (
  `CO_ANO` SMALLINT,
  `CO MES` TINYINT,
  `CO_SH4` INT,
  `CO_PAIS` INT,
  `SG_MUN` VARCHAR(2),
  `CO_MUN` INT,
  `CO_DATA` DATE,
  `KG_LIQUIDO` FLOAT NOT NULL,
  `VL_FOB_US` FLOAT NOT NULL,
  `VL_FOB_BR` FLOAT NOT NULL
);

CREATE TABLE `DIM_TEMPO` (
  `CO_DATA` DATE PRIMARY KEY NOT NULL,
  `DATA_REFERENCIA` DATE,
  `ANO` SMALLINT,
  `MES` TINYINT,
  `NOME_MES` VARCHAR(10),
  `TRIMESTRE` TINYINT,
  `SEMESTRE` TINYINT
);

CREATE TABLE `DIM_PRODUTO` (
  `CO_SH4` INT PRIMARY KEY NOT NULL,
  `NO_SH4_POR` VARCHAR(255) NOT NULL
);

CREATE TABLE `DIM_PAIS` (
  `CO_PAIS` INT PRIMARY KEY NOT NULL,
  `CO_PAIS_ISO3` VARCHAR(3),
  `CO_PAIS_ISON3` VARCHAR(3),
  `NO_PAIS` VARCHAR(100)
);

CREATE TABLE `DIM_BLOCO` (
  `CO_BLOCO` INT PRIMARY KEY NOT NULL,
  `CO_PAIS` INT,
  `NO_BLOCO` VARCHAR(100)
);

CREATE TABLE `DIM_UF_MUN` (
  `CO_MUN` INT PRIMARY KEY NOT NULL,
  `NO_MUN_MIN` VARCHAR(100),
  `SG_UF` VARCHAR(2)
);

CREATE TABLE `DIM_UF` (
  `SG_UF` VARCHAR(2) PRIMARY KEY NOT NULL,
  `CO_UF` INT,
  `NO_UF` VARCHAR(100),
  `NO_REGIAO` VARCHAR(50)
);

ALTER TABLE `FATO_EXPORTACAO` ADD FOREIGN KEY (`CO_DATA`) REFERENCES `DIM_TEMPO` (`CO_DATA`);

ALTER TABLE `FATO_EXPORTACAO` ADD FOREIGN KEY (`CO_SH4`) REFERENCES `DIM_PRODUTO` (`CO_SH4`);

ALTER TABLE `FATO_EXPORTACAO` ADD FOREIGN KEY (`CO_PAIS`) REFERENCES `DIM_PAIS` (`CO_PAIS`);

ALTER TABLE `FATO_EXPORTACAO` ADD FOREIGN KEY (`CO_MUN`) REFERENCES `DIM_UF_MUN` (`CO_MUN`);

ALTER TABLE `DIM_UF_MUN` ADD FOREIGN KEY (`SG_UF`) REFERENCES `DIM_UF` (`SG_UF`);

ALTER TABLE `DIM_BLOCO` ADD FOREIGN KEY (`CO_PAIS`) REFERENCES `DIM_PAIS` (`CO_PAIS`);
```

## Modelo Snowflake

Introdução rápida à modelagem de dados

# Nosso Modelo

# Modelagem Física

## 2. Parte Prática



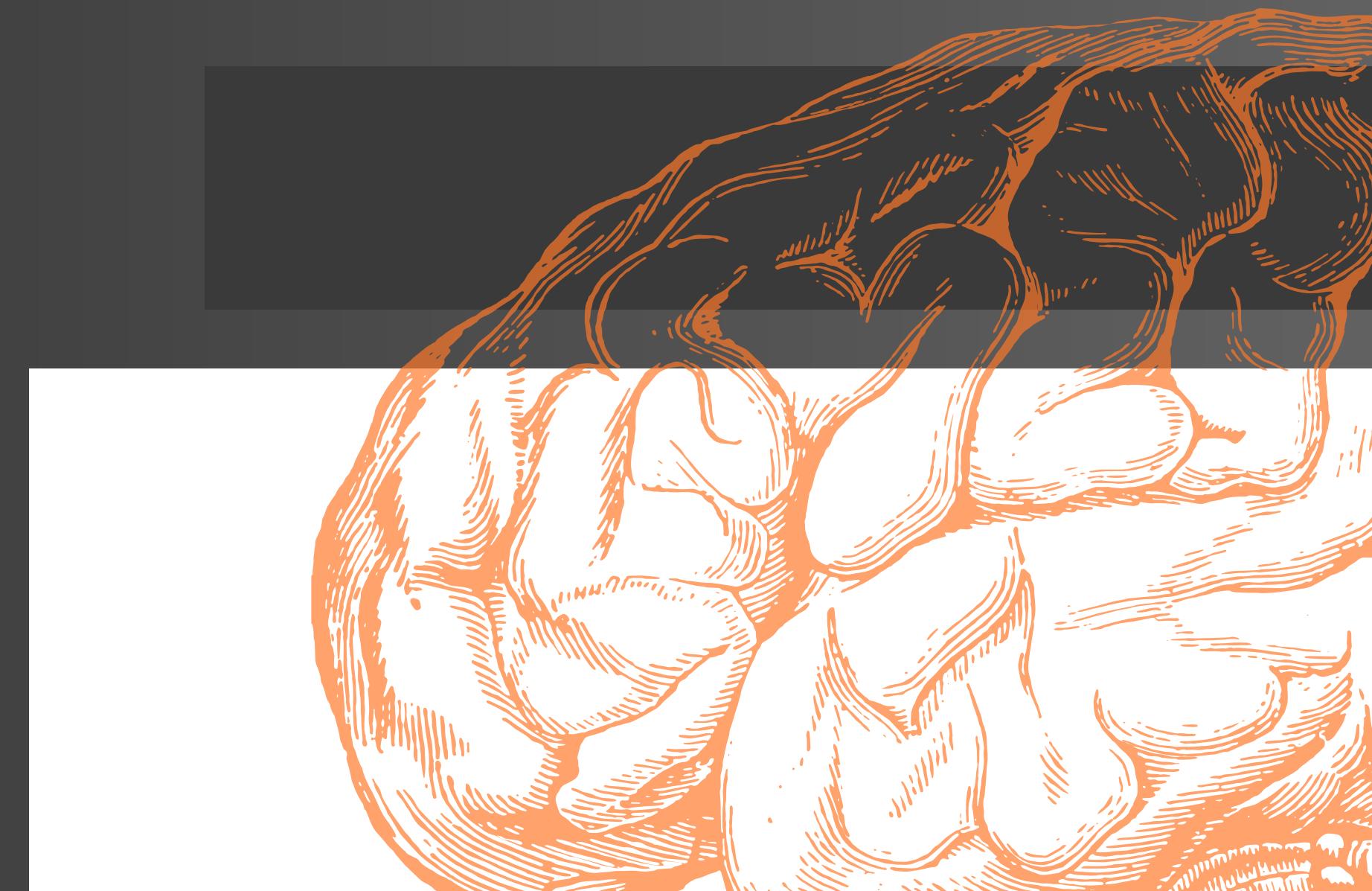
Hands on...

Practice  
MAKES  
Progress



# Sumário

- 00 PRÁTICA DE MODELAGEM E ANÁLISE DE DADOS DE EXPORTAÇÃO
- 01 SELEÇÃO E CARREGAMENTO DA BASE DE DADOS
- 02 IMPORTANDO BIBLIOTECAS NECESSÁRIAS
- 03 LEITURA DOS DADOS A PARTIR DE ARQUIVOS LOCAIS NO COLAB
- 04 ANÁLISE EXPLORATÓRIA E VALIDAÇÃO DOS DADOS
- 05 LIMPEZA E AJUSTES DAS TABELAS
- 06 SALVANDO O ARQUIVO FINAL
- 07 VISUALIZANDO O RESULTADO FINAL





IGOR MOREIRA



# OBRIGADO!



Site



Instagram



LinkedIn



Git-Hub



**FAT** Faculdade e  
Tecnologia  
UERJ - Resende

