

# Aprendendo SQL

25/08/2025

Primeiras magias na linguagem dos dados

instituto  
AARCN  
SWARTZ

TÉO  
ME WHY?

Aprendendo SQL © 2025 by Téo Calvo is licensed under CC BY-NC-SA 4.0.

To view a copy of this license, visit: <https://creativecommons.org/licenses/by-nc-sa/4.0/>



# Quem

█ Teodoro “Téo” Calvo

█ Bacharel em Estatística - FCT UNESP

█ Especialista em Big Data & Data Science - UFPR

█ Streamer - Twitch

█ Membro do Conselho e Instrutor - Instituto Aaron Swartz





# Trajetória

2016 2017 2018 2019 2020 2021 2022 2023 2024 2025



Grupo  
Boticário



2019



2020



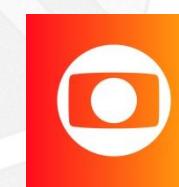
2021



2018



2023



2024



2025



# Combinados

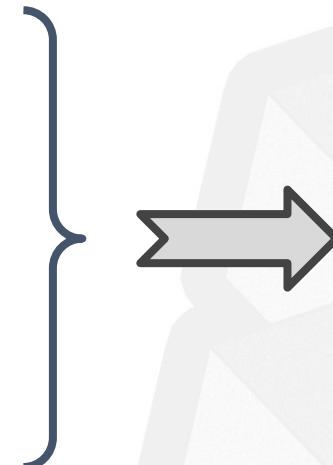
- Sempre das **9AM às ~11 horas** estaremos online
- Pausa às **10:00hrs** até as 10:15
- Notificações serão **desligadas**, mas sua contribuição é sempre bem-vinda
- Quando houver ADs, a explicação é **pausada**
- Evite **spammar** a mesma coisa no chat, moderadores estarão de olho
- Comentários discriminatórios e babacas serão **banidos** sem aviso prévio
- **!ppt** para acessar todo material o curso
- Sorteio no último dia!





# Avisos

- Participe de nossa comunidade: [comunidade.teomewhy.org](http://comunidade.teomewhy.org)
- Nosso foco é o básico bem feito. Estamos melhorando ano após ano!
- O sucesso do curso também depende de você!
- Me ajude com as perguntas repetidas;
- Tem algo que não abordei?
- Tem algo que você explicaria diferente?
- Tem algo mais complexo que deixei passar?



Faça um  
vídeo/material



# Nos Apoie

## Nossos cursos são 100% gratuitos

e você pode nos apoiar em diferentes plataformas:

- Apoia.se: [apoia.se/teomewhy](https://apoia.se/teomewhy)
- LivePix: [livepix.gg/teomewhy](https://livepix.gg/teomewhy)
- Assinante na Twitch: [twitch.tv/teomewhy](https://twitch.tv/teomewhy)
- Membros no YouTube: [youtube.com/@teomewhy](https://youtube.com/@teomewhy)
- GitHub Sponsors: [github.com/sponsors/TeoMeWhy](https://github.com/sponsors/TeoMeWhy)





# Para quem é este curso

- Pouco ou nenhum conhecimento em SQL;
- Migração de carreira;
- Acessar dados em bancos de dados;
- Analisar dados;
- Gerar relatórios;
- Tomar decisões suportadas em dados;





# Para quem NÃO é este curso

- Vasta experiência e usa SQL intensivamente no seu dia a dia;
- Deseja construir um banco de dados;
- Conhecer melhores práticas em modelagem de dados;



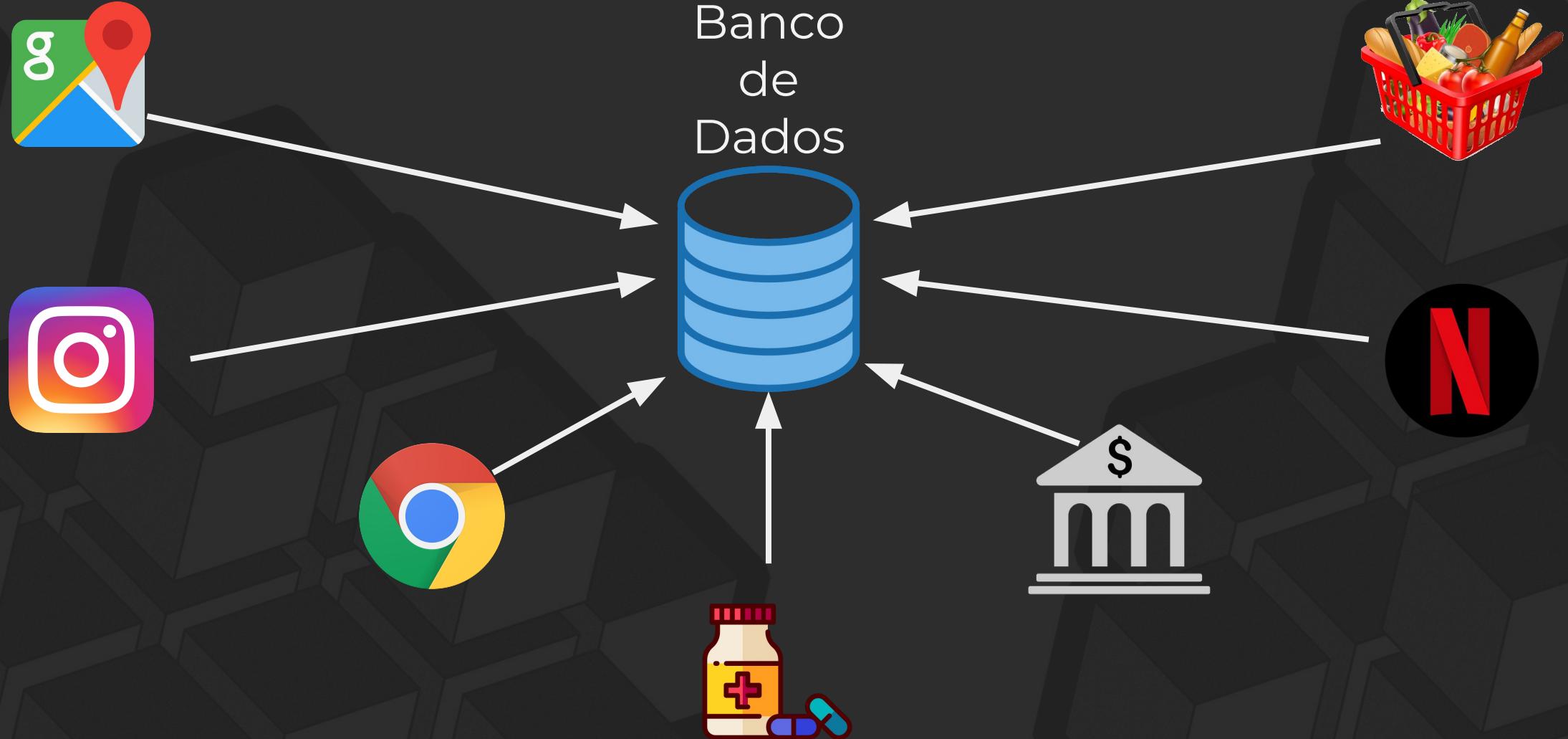
# Um pouco sobre a área de dados

**Quando se fala em dados, qual a primeira coisa que vem na sua cabeça?**

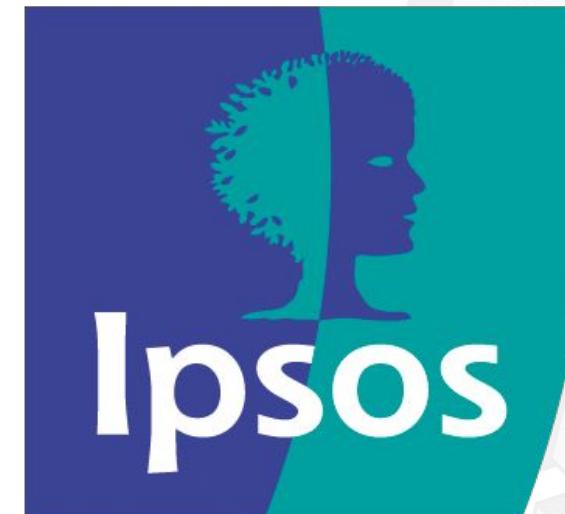
**Por que dados são importantes? Por que coletar dados?**

**Que tipo de uso de dados você conhece?**

# Coleta de dados



# Coleta de dados (pesquisa e censo)



# Possibilidades em dados

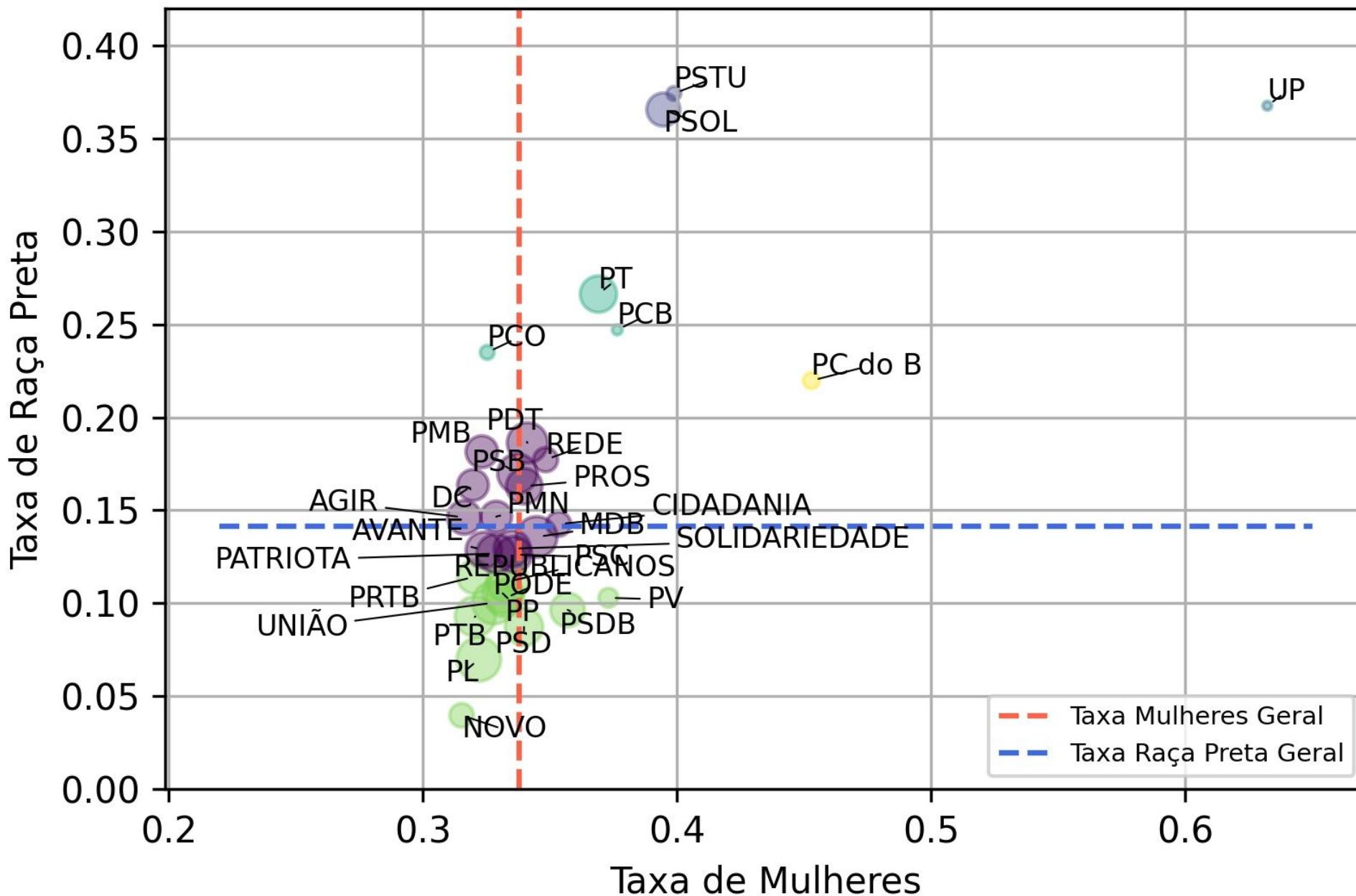
- Previsão de vendas
- Tempo de garantia
- Limite de crédito
- Manutenção de equipamento industrial
- Recomendação de produtos
- Definição de metas
- Processamento de Linguagem Natural
- Conhecer diferentes grupos de clientes
- Validar produtos com usuários
- Entender sazonalidade
- Visão computacional
- Performance de atletas
- Eficácia de vacinas e medicamentos

**Tomar melhores decisões**

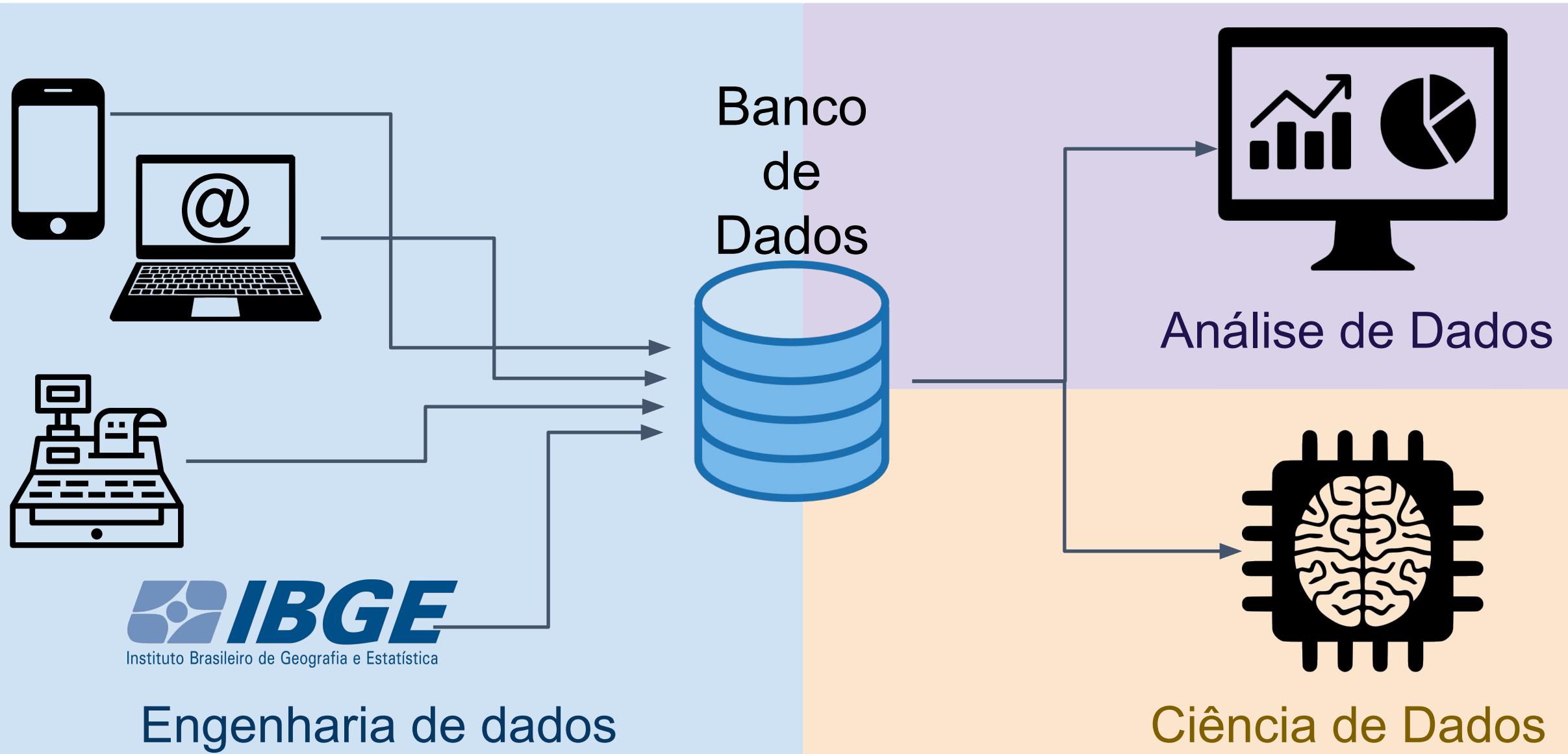


# Grupos de Partidos - Candidatos 2022

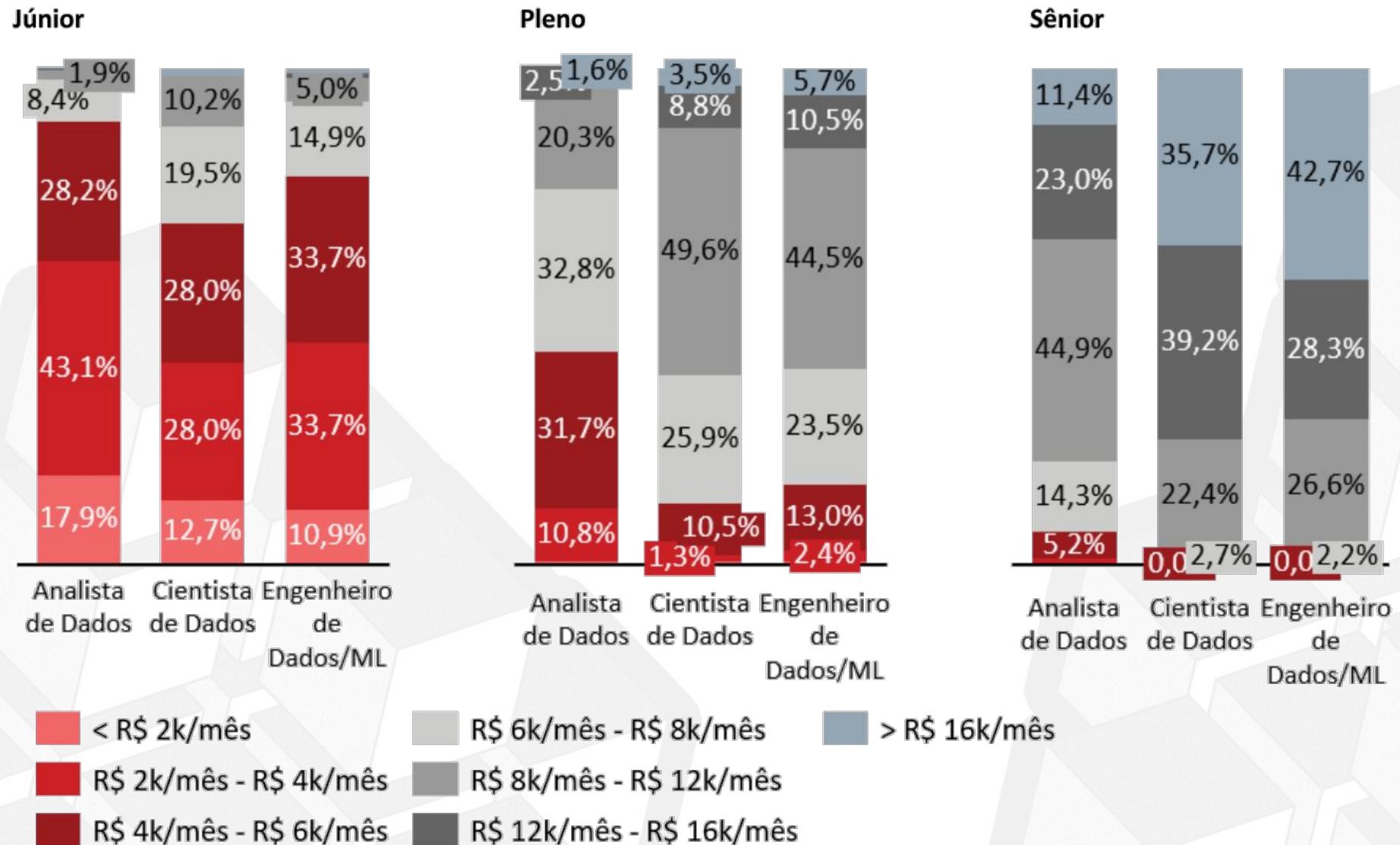
Quanto maior a bolha, mais candidatos



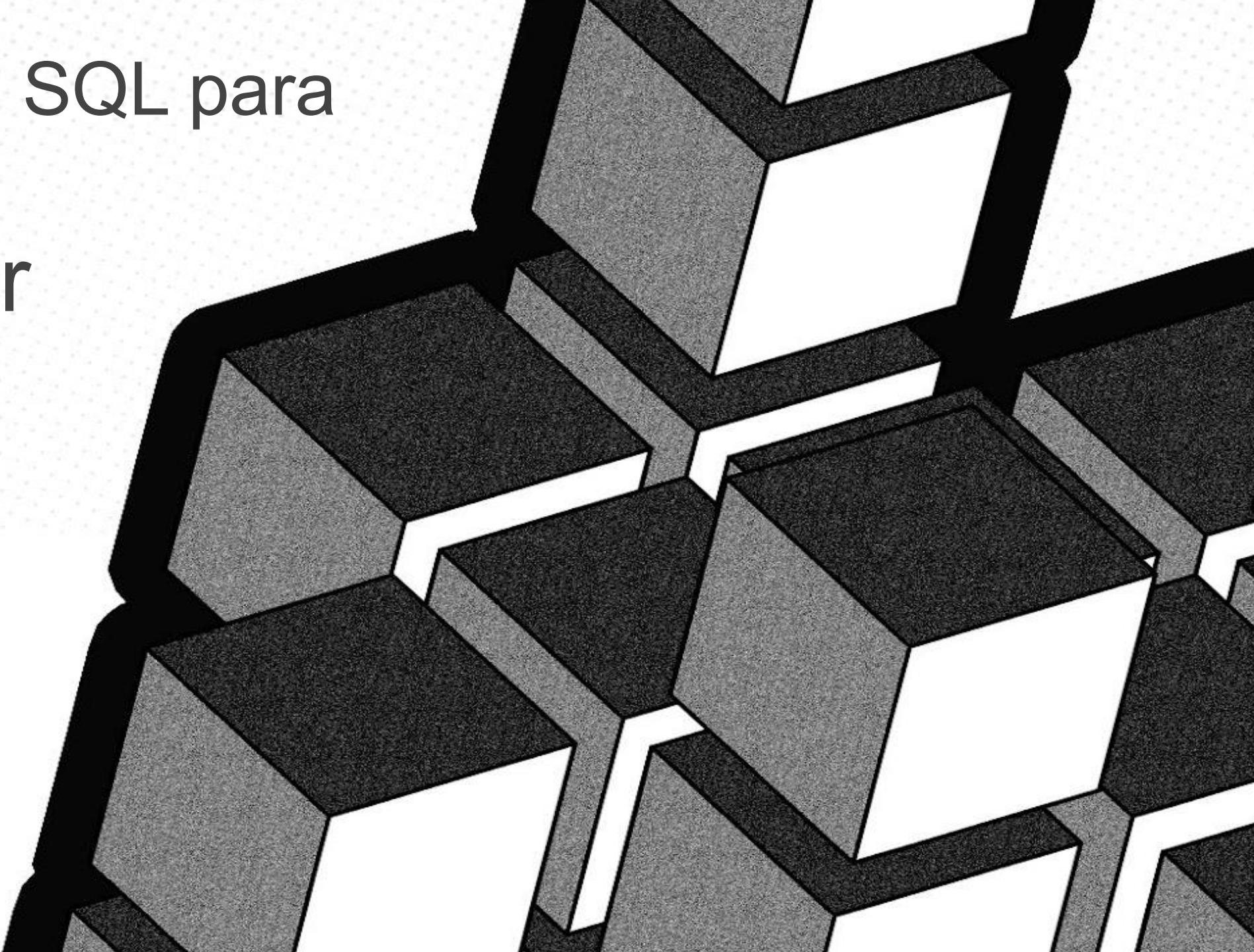
# Profissionais de Dados



# Cargos e salários



**Todos usam SQL para**  
armazenar  
organizar  
consultar  
**dados!**





# O que vamos abordar

## 1. Introdução + Setup

- Acesso aos dados
- Visual Studio Code
- Extensões para SQLite

## 4. JOINs:

- LEFT
- RIGHT
- INNER

## 7. Window Functions:

- OVER
- PARTITION BY
- LAG
- LEAD
- ROW\_NUMBER

## 2. Primeiras consultas (DQL)

- SELECT
- FROM
- WHERE
- CASE WHEN

## 5. Juntando dados com UNIONs

## 8. DDL (Data Definition Language)

- CREATE
- DROP
- TRUNCATE

## 3. Agregações

- Funções de agregações
- GROUP BY
- ORDER BY
- HAVING

## 6. Subqueries

- CTE

## 9. DML (Data Manipulation Language)

- INSERT
- DELETE
- UPDATE



# Banco de dados

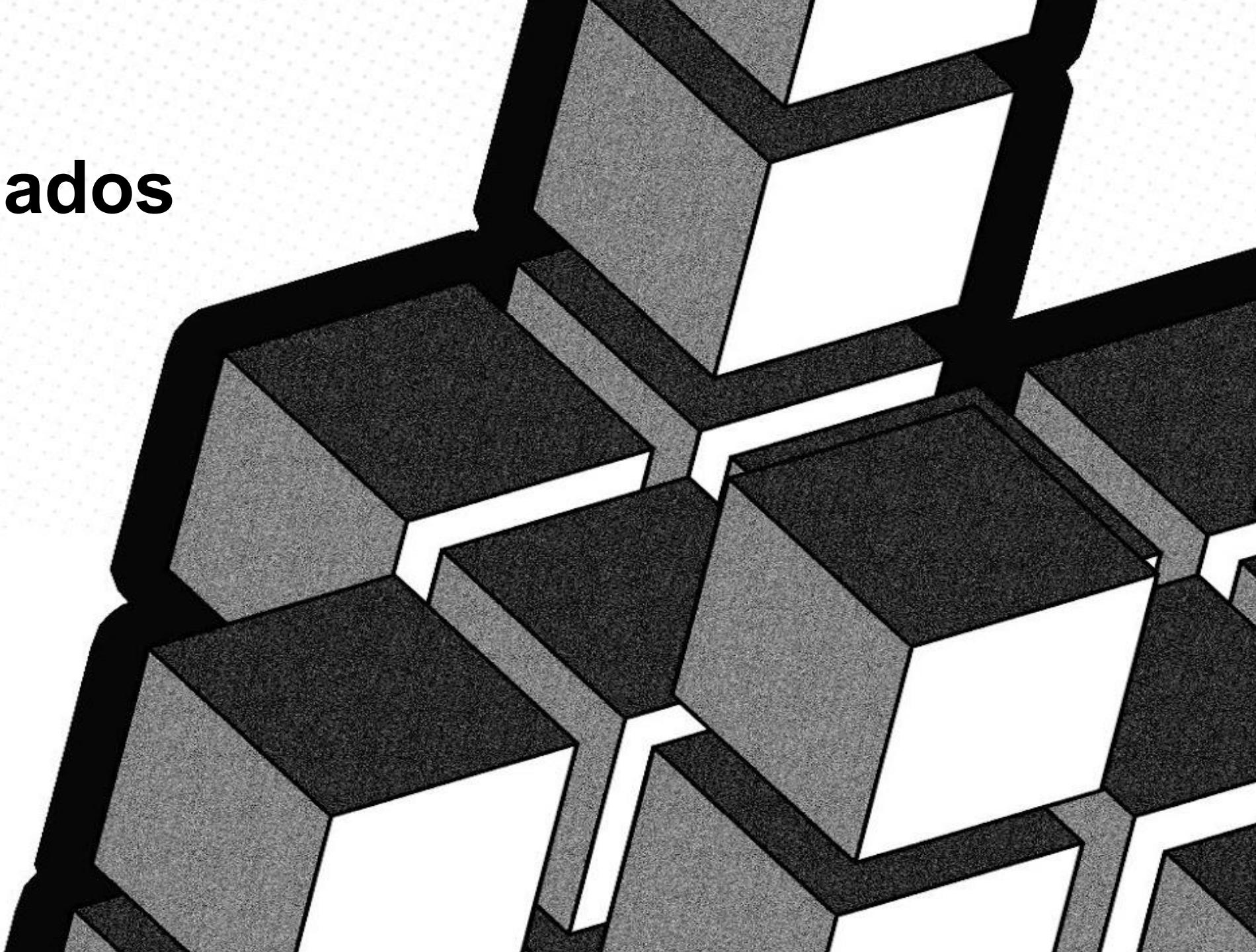


Tabela de Cliente

ID Cliente	Nome	email	Data Cadastro
1	Téo Calvo	teozinhoBoladao@gmail.com	2024-06-13
2	Nah Ataide	nahnah@gmail.com	2021-05-04
3	Téo Cabeludo		2025-03-10

Linha  
ou  
Registro

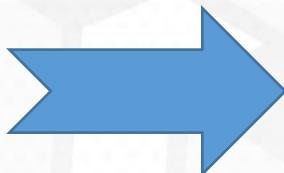
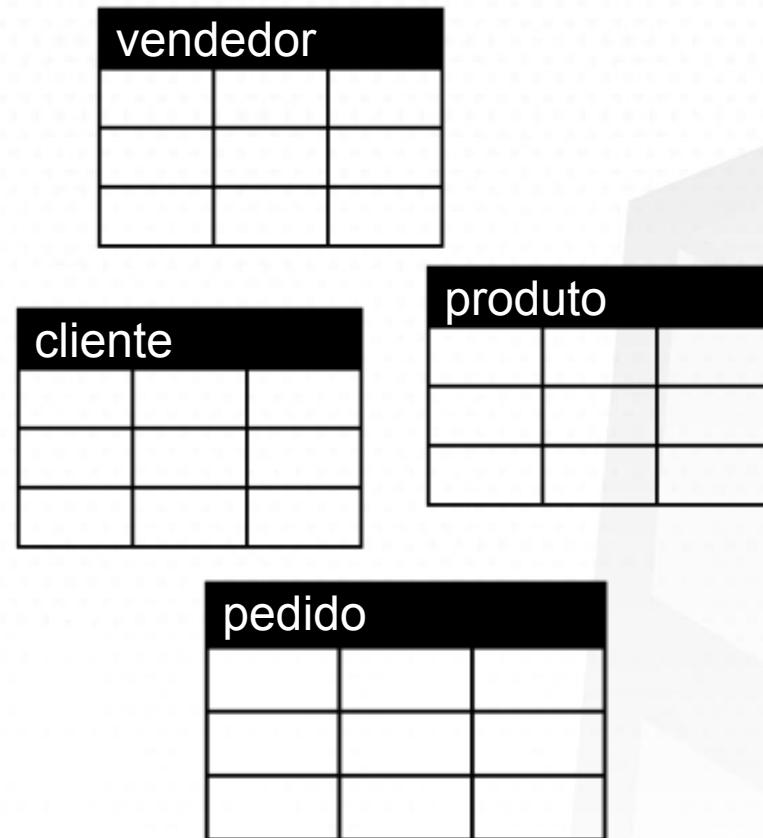
Tabela de Vendas

ID Venda	ID Cliente	ID Vendedor	Data Venda	Valor Venda
1	1	1	2024-07-15	R\$ 120,00
2	2	1	2022-01-11	R\$ 63,90
3	2	4	2024-06-09	R\$ 25,89

Coluna ou Campo



Data Venda



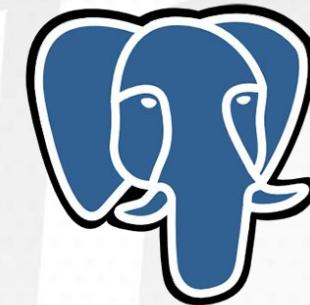
## Banco de dados



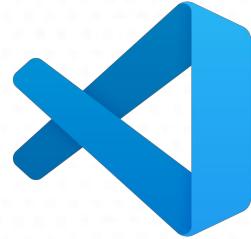
**Sistema de Gerenciamento  
de Banco de Dados  
(SGBD)**



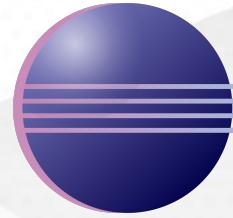
# Alguns SGBD Relacionais (sabores de engine)



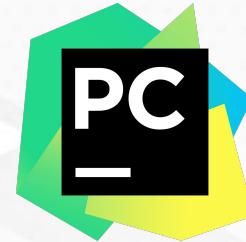
# Algumas IDEs (Ambiente de Desenvolvimento Integrado)



Visual Studio Code



Eclipse



PyCharm



Sublime Text



MySQL Workbench



Metabase



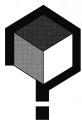
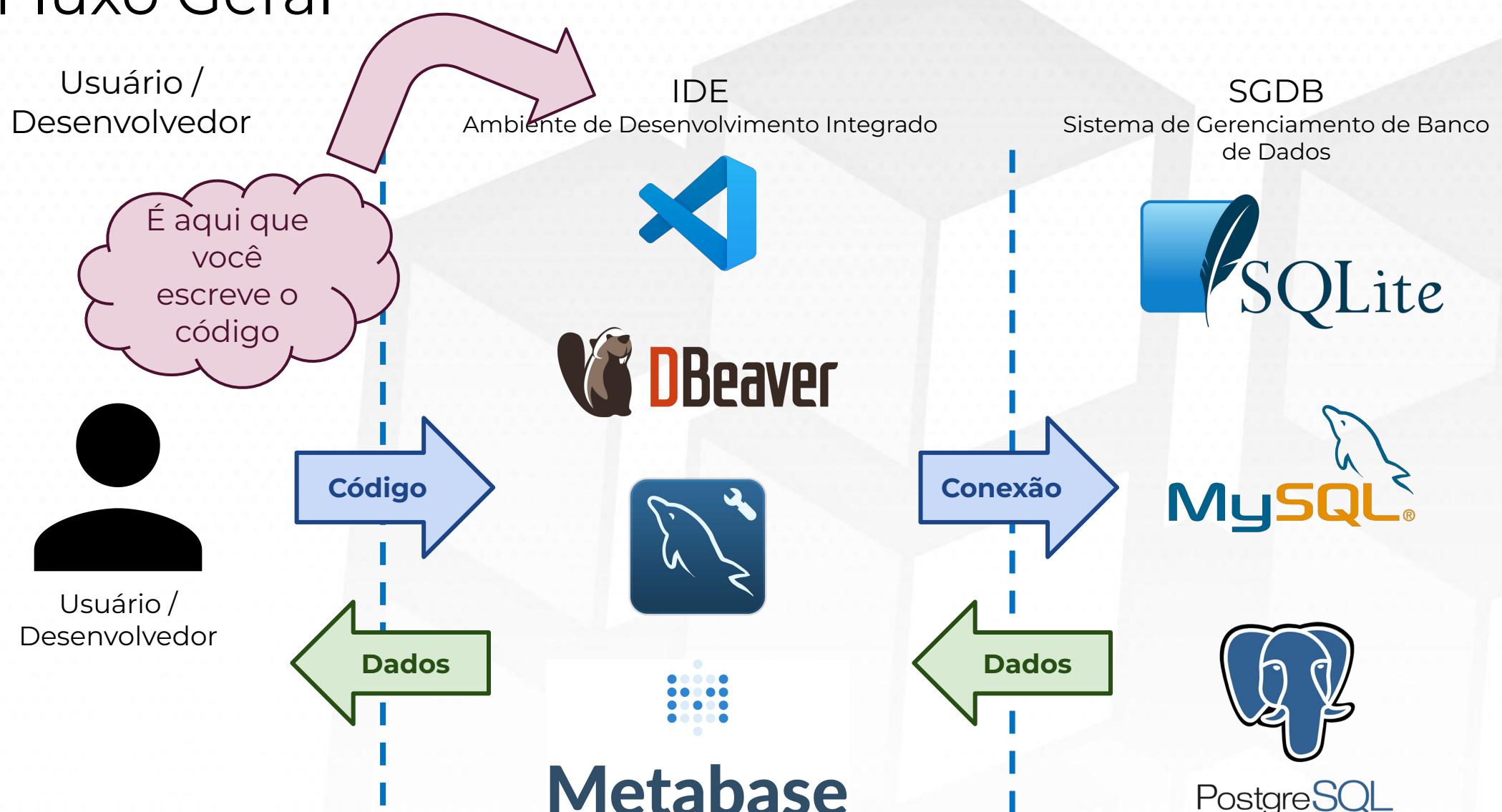
databricks



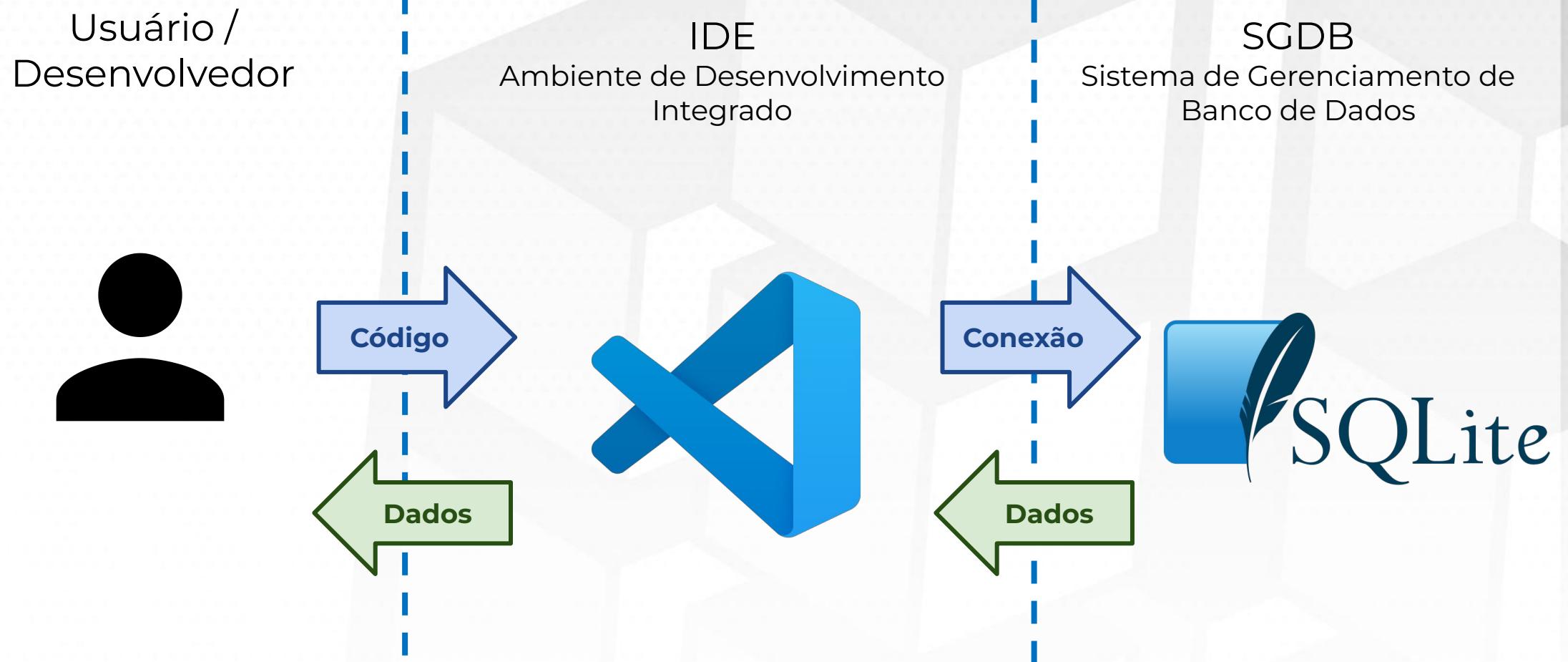
DBeaver



# O Fluxo Geral



# O Fluxo em nosso curso

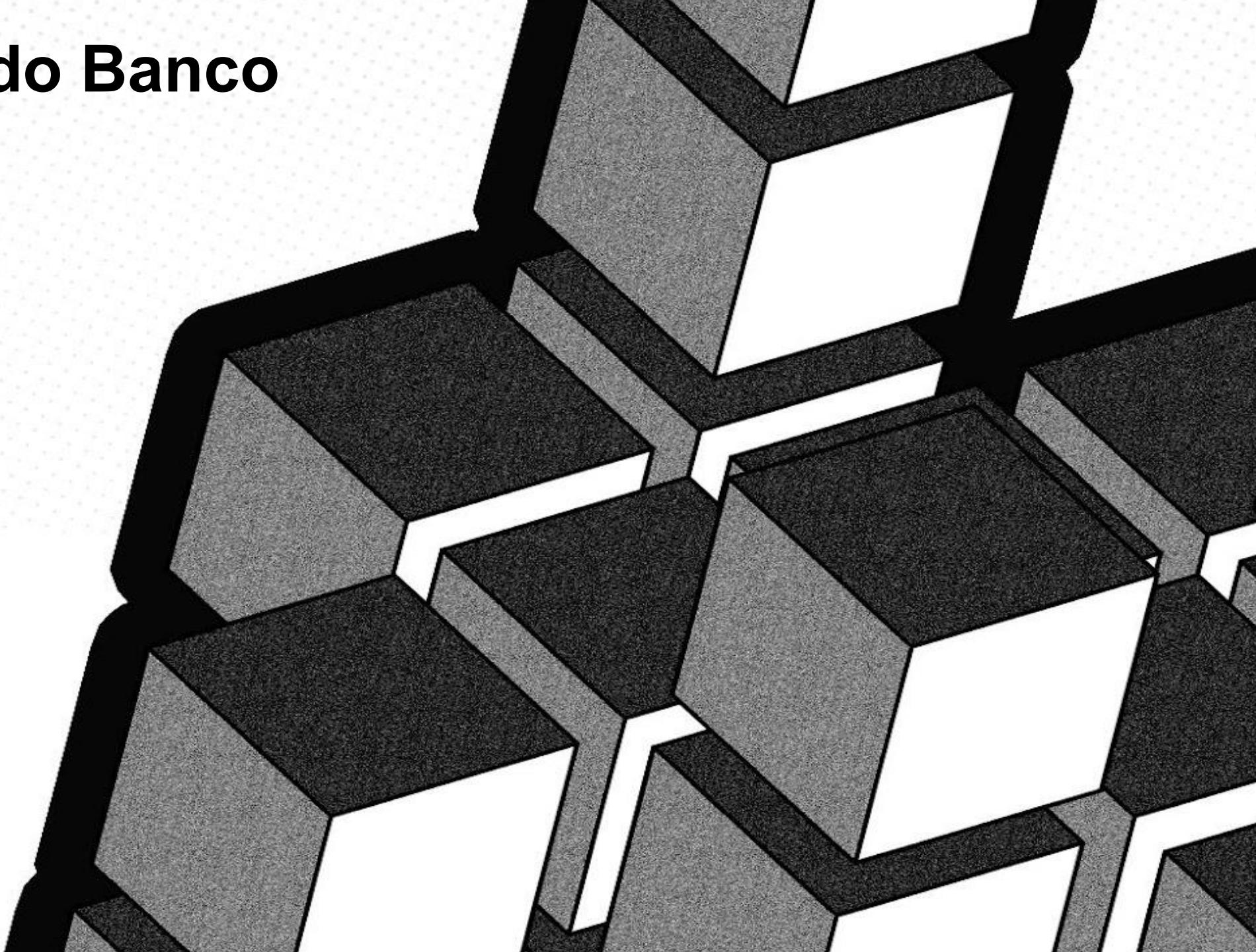


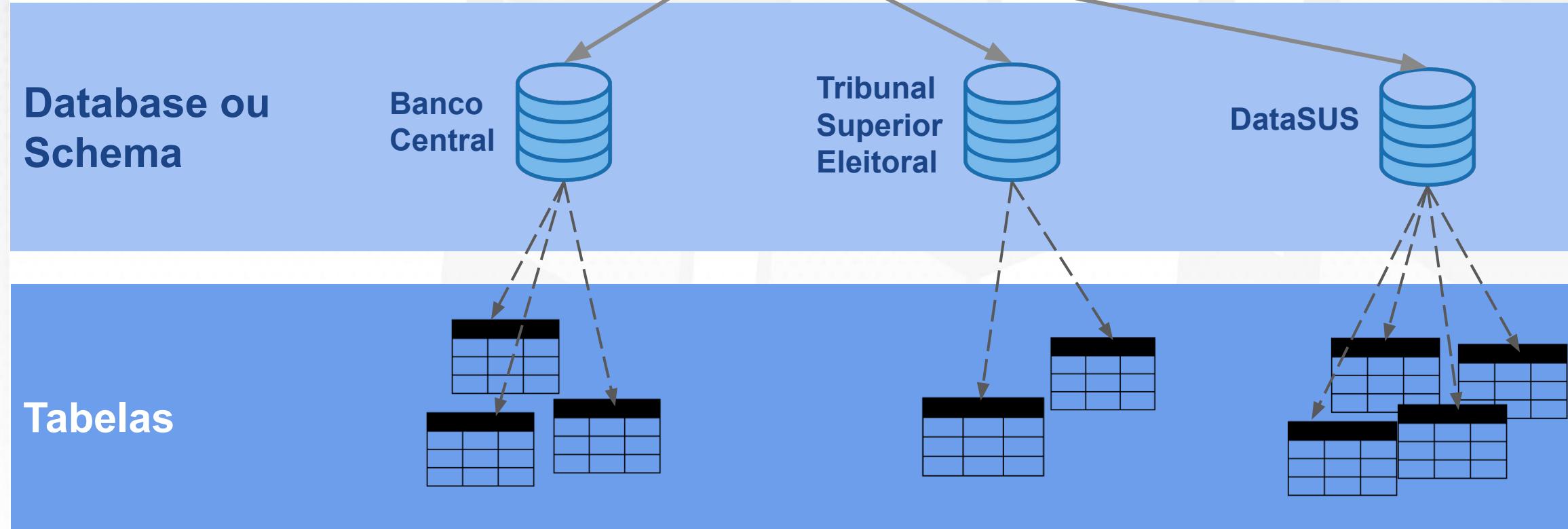
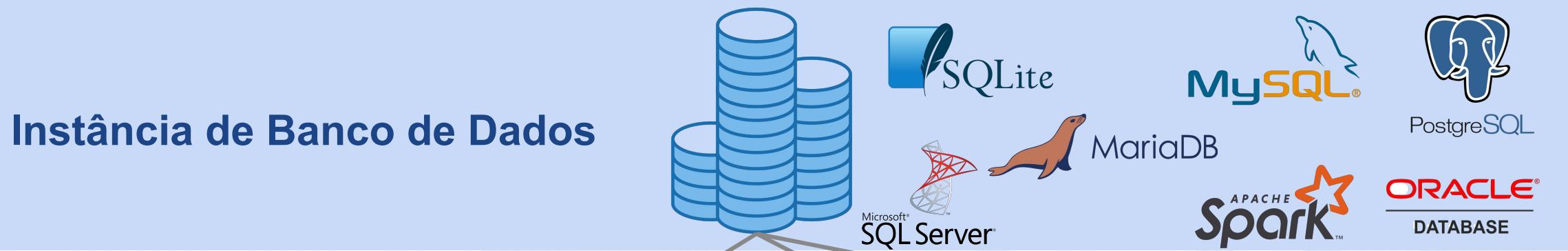
-> Instância do Banco

-> Database

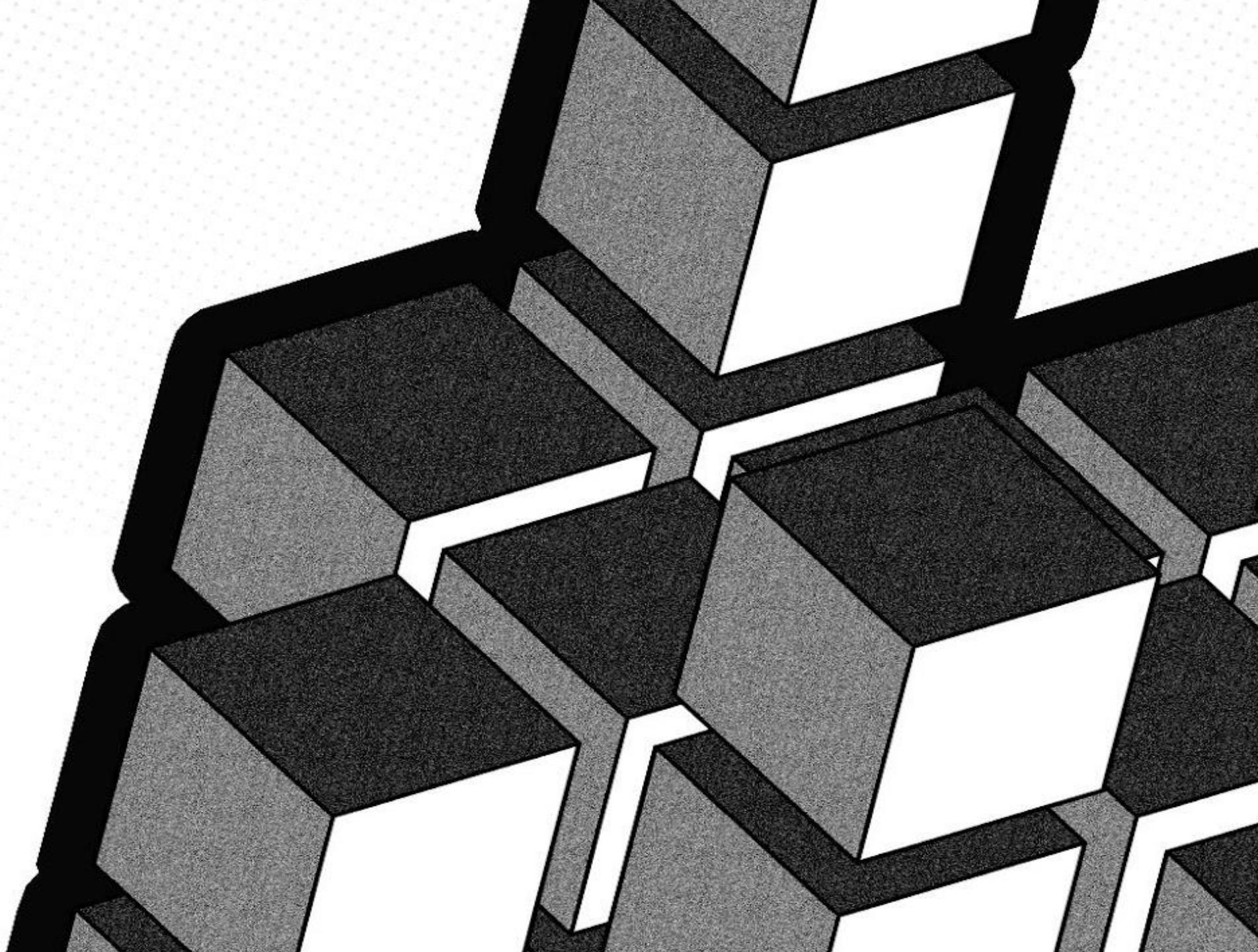
-> Tabela

-> Registros





# SQL



# Mas o que é SQL?

*“Structured Query Language, ou Linguagem de Consulta Estruturada ou SQL, é a linguagem de pesquisa declarativa padrão para banco de dados relacional (base de dados relacional).” - Wikipedia*



# Mas o que é SQL?

Alguns de seus principais elementos:

- **Comandos** – executados resultam em efeito persistente sobre dados e estruturas, ou controlam transações, conexões, sessões, etc.
  - EX: SELECT, INSERT, UPDATE, DELETE, CREATE TABLE, GRANT, etc.
- **Cláusulas** - componentes dos comandos. Em muitos casos, alguns são opcionais.
  - Ex: FROM, WHERE, GROUP BY
- **Expressões** - os quais produzem tanto valores escalares, como tabelas consistindo de colunas e linhas
  - Ex: A+B, A\*ABS(B-20)
- **Predicados** - especificam condições que podem ser avaliadas pela lógica, retornando verdadeiro, falso ou desconhecido
  - Ex: A>B, C BETWEEN 20 AND 200
- **Queries** - que retornam dados baseados em critérios específicos
  - Ex: SELECT COLUNA FROM TABELA



# Tipos de dados

Linguagem Natural	Spark / SQL
Número (inteiro)	Int
Número (com casa decimal)	Float
Texto	String
Sim/Não	Boolean
Data	Date
Data - Hora	Timestamp
Vazio / Nulo	null



# Outras Estruturas Importantes

- **Chaves primárias (Primary Key) e referenciais (Foreign Key)**
  - PK, FK - garantem integridade e relacionamento consistente entre tabelas
  - Servem como uma documentação também
  - Exemplos: CPF, CNPJ, ID de cliente – PKs. Colocar um ID de cliente válido num pedido - FK
- **Constraints**
  - Assim como PKs e FKs, servem para garantir integridade
  - Não são mais muito utilizadas pois acabam onerando o SGBD
  - A mais comum é a NOT NULL
- **Sequências**
  - Servem para garantir que informações únicas sejam criadas, com continuidade e unicidade
  - Normalmente são identificadores internos das aplicações





# Setup

## Download dos dados

- Kaggle: [kaggle.com/datasets/teocalvo/teomewhy-loyalty-system](https://kaggle.com/datasets/teocalvo/teomewhy-loyalty-system)

## • Visual Studio Code

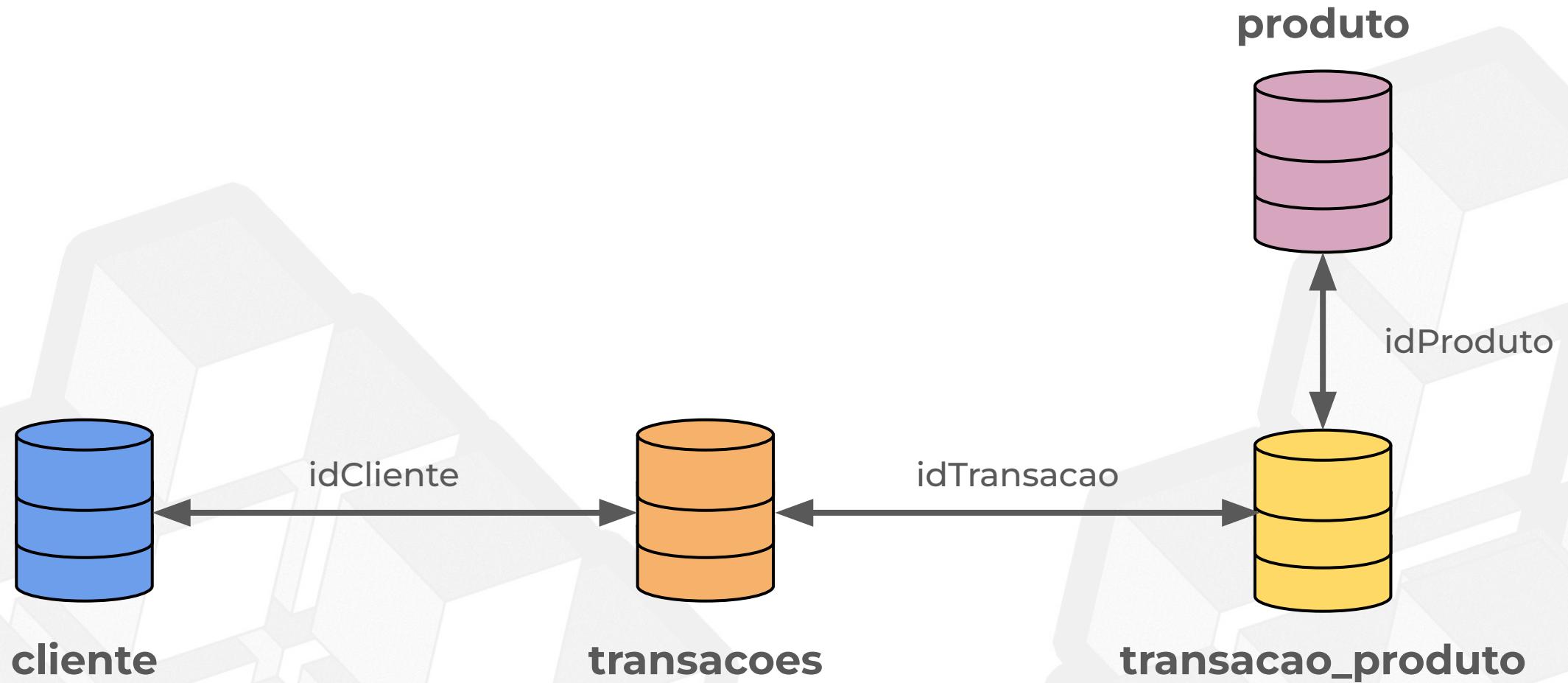
- Download: [code.visualstudio.com/](https://code.visualstudio.com/)
- Extensão: [marketplace.visualstudio.com/items?itemName=alexcvzz.vscode-sqlite](https://marketplace.visualstudio.com/items?itemName=alexcvzz.vscode-sqlite)

## Repositório

- <https://github.com/TeoMeWhy/sql-2025>



# Database: sistema de pontos Téo Me Why



# ***SELECT***

Toda consulta de tabelas precisa ao menos de duas instruções em sua sintaxe. A primeira é a *SELECT*, onde é passado uma lista de **colunas** que se deseja obter, bem como realizar a criação de novas colunas e cálculos.

**SELECT** 'Olá mundo!'

**SELECT** 2 \* 136



# **FROM**

Esta é a segunda instrução necessária para realizar uma consulta simples. Devemos informar a partir de qual tabela desejamos obter os dados.

Assim, para selecionar **todas** colunas de uma tabela, temos:

**SELECT \***

**FROM tabela**

Note que o **\*** (asterisco) representa todas as colunas.



# Nossa primeira Query (consulta)

```
SELECT *  
FROM cliente
```

Podemos ler esse código em linguagem natural da seguinte forma:  
“selecione **todas** colunas da **tabela cliente**”



# **WHERE**

Agora que temos uma query simples, podemos adicionar algumas instruções adicionais. Talvez seja necessário realizar algum tipo de filtro, para isso utilizamos o WHERE com uma condição lógica.

**SELECT \***

**FROM cliente**

**WHERE flEmail = 1**

Podemos ler esse código em linguagem natural da seguinte forma:  
“selecione **todas** colunas da **tabela cliente onde** os valores da coluna **flEmail** sejam iguais a **1**”



# CASE

Como podemos identificar se um pedido atrasou ou não? Podemos realizar comparações lógicas ao criar novas variáveis?

```
SELECT *,  
       CASE WHEN QtdePontos > 500 THEN 'alto' END AS NivelPontos  
FROM cliente
```



# Bora praticar?

- Selecione todos os clientes com email cadastrado
- Selecione todas transações de 50 pontos (exatos)
- Selecione todos clientes com mais de 500 pontos
- Selecione produtos que contêm ‘churn’ no nome

# Tempo de foco

1. Lista de transações com apenas 1 ponto;
2. Lista de pedidos realizados no fim de semana;
3. Lista de clientes com 0 (zero) pontos;
4. Lista de clientes com 100 a 200 pontos (inclusive ambos);
5. Lista de produtos com nome que começa com “Venda de”;
6. Lista de produtos com nome que termina com “Lover”;
7. Lista de produtos que são “chapéu”;
8. Lista de transações com o produto “Resgatar Ponei”;
9. Listar todas as transações adicionando uma coluna nova sinalizando “alto”, “médio” e “baixo” para o valor dos pontos [ $<10$  ;  $<500$ ;  $\geq 500$ ]

# Agregações

Em muitos momentos desejamos alguns dados summarizados, como por exemplo: contagem, média, variância, mínimo, máximo, etc.

Contagem de linhas: `count(*)`

Contagem de linhas distintas: `count(distinct column)`

Soma: `sum(column)`

Média: `avg(column)`

Máximo: `max(column)`



## *GROUP BY*

Será que temos uma forma melhor para summarizar dados em diferentes níveis?

Para isso, podemos AGREGAR os dados utilizando a instrução GROUP BY

```
SELECT idCliente,  
       count( idTransacao )  
FROM transacoes  
GROUP BY idCliente
```

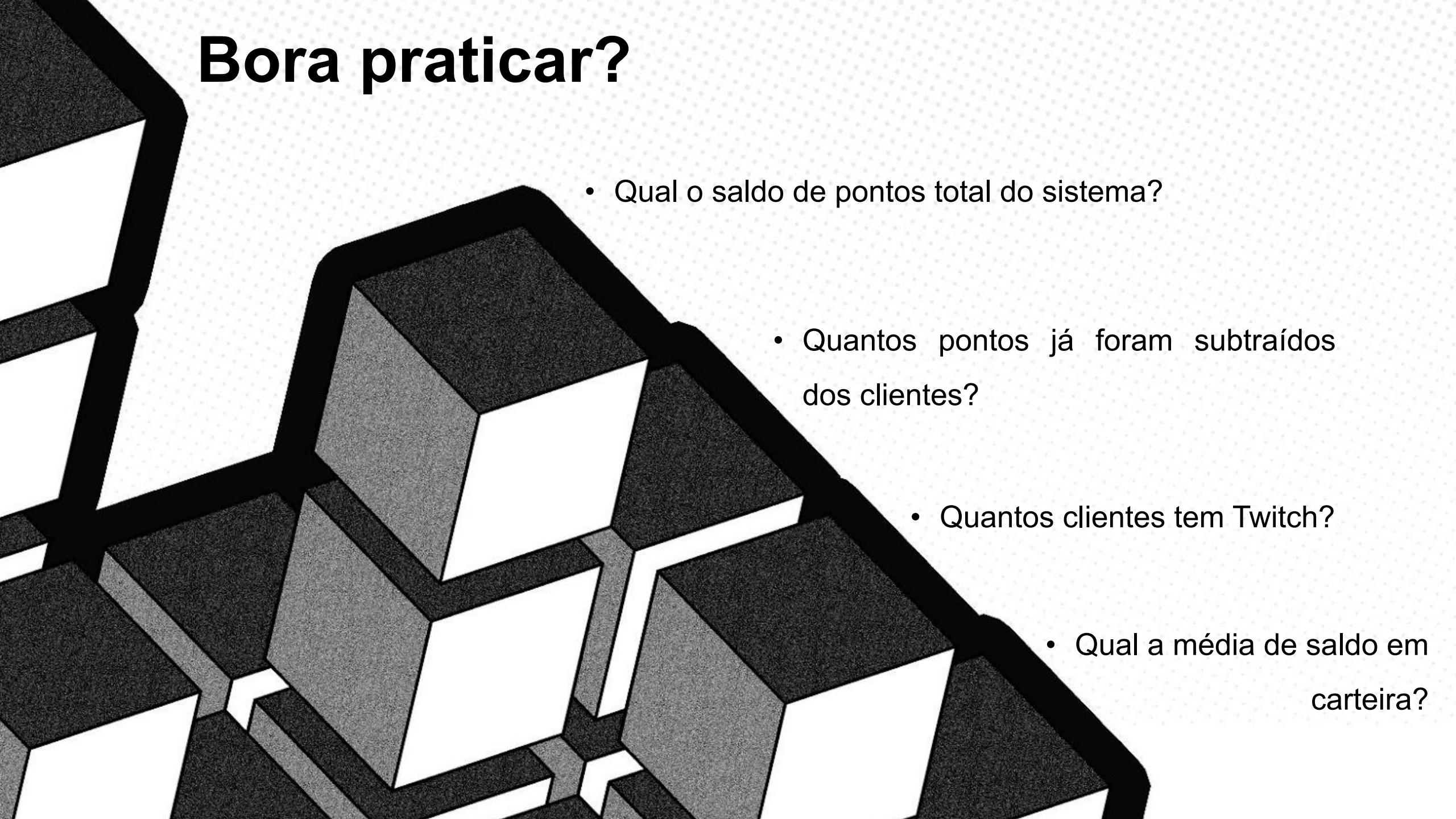


# *GROUP BY*

Qual a diferença da consulta anterior para essa?

```
SELECT idCliente,  
       count(distinct idTransacao)  
FROM transacoes  
GROUP BY idCliente
```





# Bora praticar?

- Qual o saldo de pontos total do sistema?
- Quantos pontos já foram subtraídos dos clientes?
- Quantos clientes tem Twitch?
- Qual a média de saldo em carteira?

# Tempo de foco (Group By)

1. Quantos clientes tem email cadastrado?
2. Qual cliente juntou mais pontos positivos em 2025-05?
3. Qual cliente fez mais transações no ano de 2024?
4. Quantos produtos são de rpg?
5. Qual o valor médio de pontos positivos por dia?
6. Qual dia da semana quem mais pedidos em 2025?
7. Qual o produto mais transacionado?
8. Qual o produto com mais pontos transacionados?



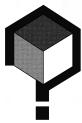
# **JOIN**

Como podemos cruzar dados de diferentes tabelas para obter informações mais interessantes a respeito de nossas vendas?

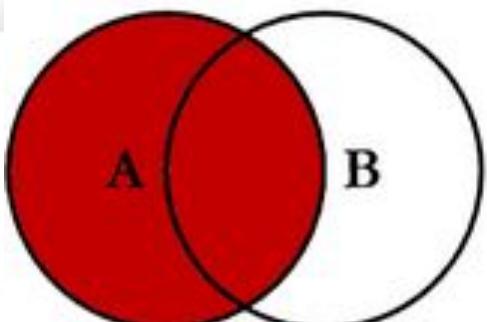
Por exemplo, quantidade de vendas realizadas por vendedores do Ceará?

Para isso, teremos que aprender os diferentes tipos de JOINS:

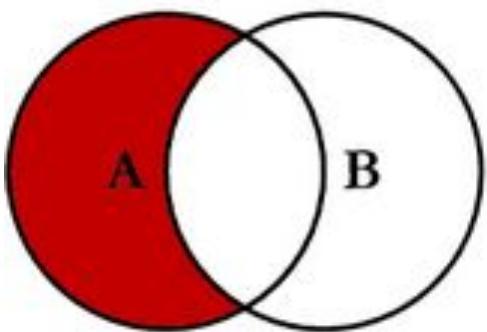
LEFT, RIGHT, INNER...



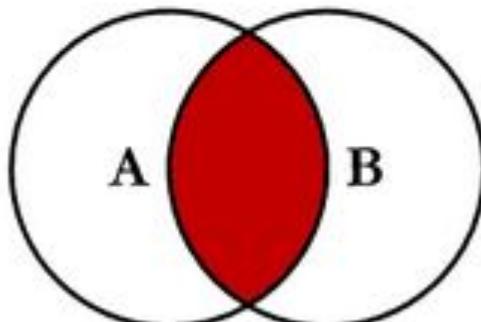
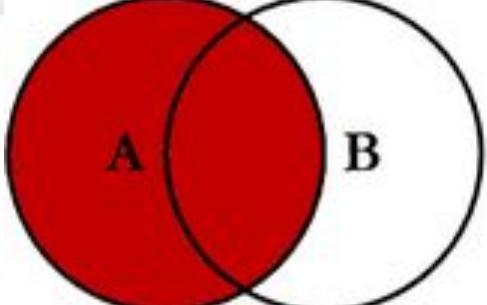
# SQL JOINS



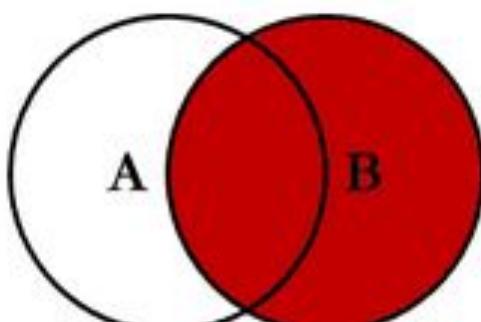
```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```



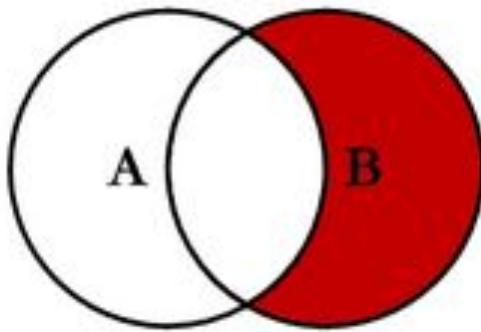
```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```



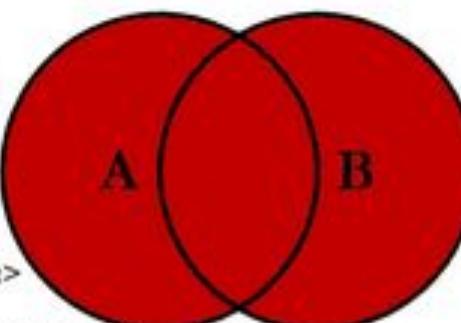
```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```



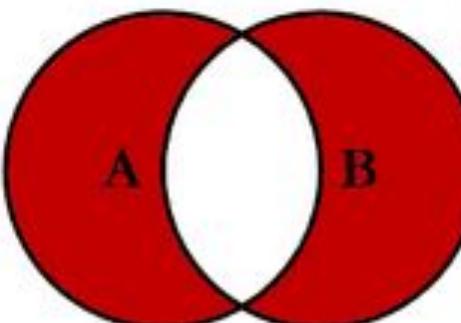
```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```



```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
```

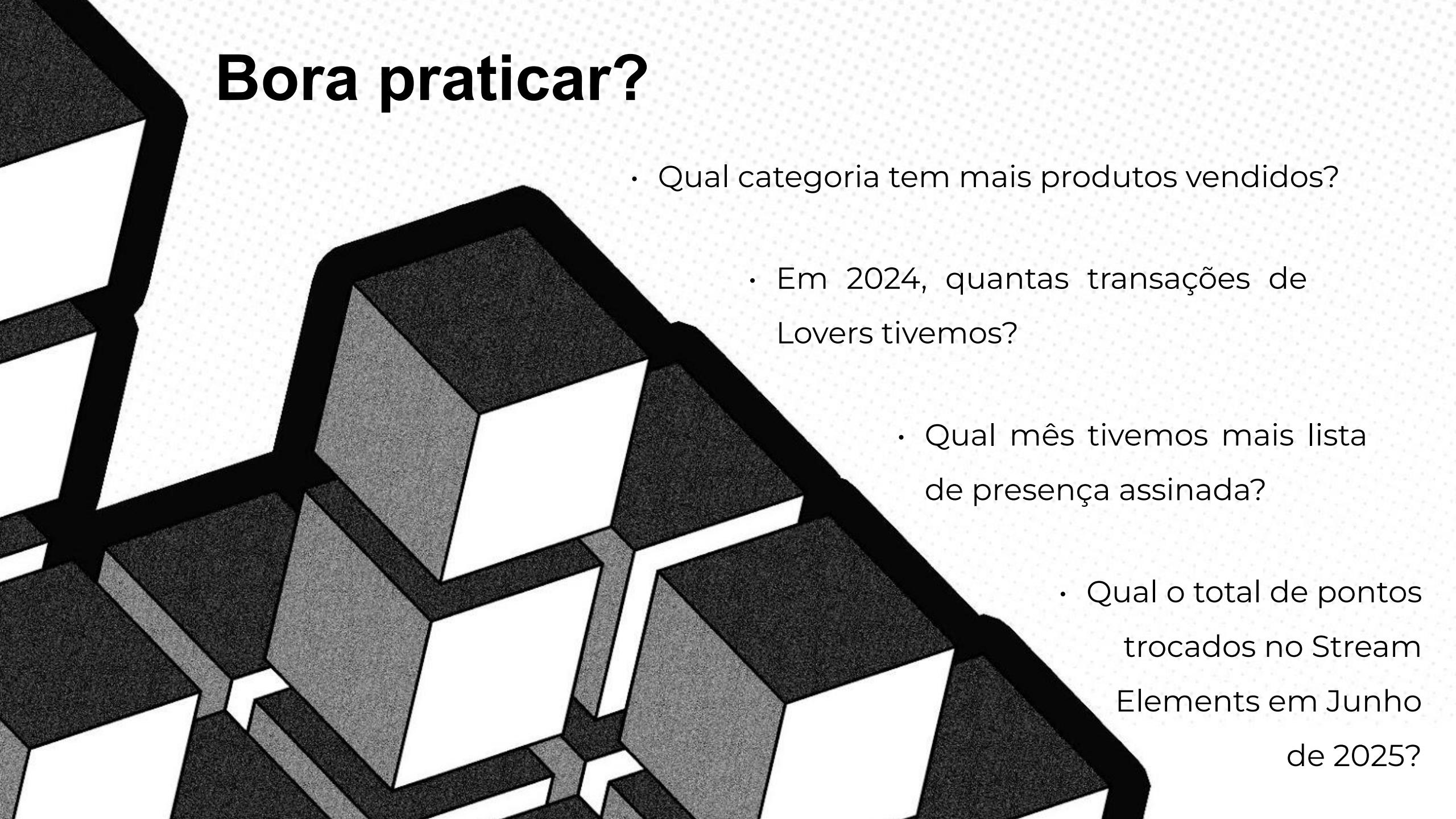


```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
```



```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
```





# Bora praticar?

- Qual categoria tem mais produtos vendidos?
- Em 2024, quantas transações de Lovers tivemos?
- Qual mês tivemos mais lista de presença assinada?
- Qual o total de pontos trocados no Stream Elements em Junho de 2025?

# Tempo de foco

1. Quais clientes mais perderam pontos por Lover?
  2. Quais clientes assinaram a lista de presença no dia 2025/08/25?
  3. Do início ao fim do nosso curso (2025/08/25 a 2025/08/29), quantos clientes assinaram a lista de presença?
  4. Clientes mais antigos, tem mais frequência de transação?
  5. Quantidade de transações Acumuladas ao longo do tempo?
  6. Quantidade de usuários que fizeram compras ao longo do tempo?
  7. Qual o dia da semana com maior número de compras?
  8. Saldo de pontos acumulado ao longo do tempo?
- 



# Tempo de foco

9. Dos clientes que começaram SQL no primeiro dia, quantos chegaram ao 5º dia?
10. Como foi a curva de Churn do Curso de SQL?
11. Quem iniciou o curso no primeiro dia, em média assistiu quantas aulas?
12. Dentre os clientes de janeiro/2025, quantos assistiram o curso de SQL?
13. Qual o dia com maior engajamento de cada aluno que iniciou o curso no dia 01?



# Projeto Final

Vamos construir uma tabela com o perfil comportamental dos nossos usuários.

1. Quantidade de transações históricas (vida, D7, D14, D28, D56);
2. Dia desde a última transação
3. Idade na base
4. Produto mais usado (vida, D7, D14, D28, D56);
5. Saldo de pontos atual;
6. Pontos acumulados positivos (vida, D7, D14, D28, D56);
7. Pontos acumulados negativos (vida, D7, D14, D28, D56);
8. Dias da semana mais ativos (D28)
9. Período do dia mais ativo (D28)
10. Engajamento em D28 versus Vida



# Um ótimo curso!

Téo Calvo

[teo.bcalvo@gmail.com](mailto:teo.bcalvo@gmail.com)

 /in/teocalvo

 /teomewhy