

# Programação Orientada a Objetos: 101

Uma visão do tema para todos os  
mestres do SQL e do Excel

**APRENDA EM MENOS DE  
10 MINUTOS (SÉRIO)**

Você finalmente vai  
entender esse conceito



# A Pluralidade

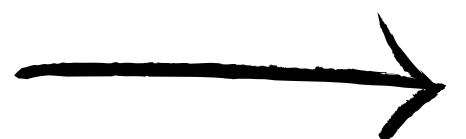
O universo de dados é diversificado. Em cada canto, encontramos especialistas com habilidades diferentes:

- Alguns são ninjas de **SQL**, comandando query complexas para gerar insights;
- Outros são magos do **Excel**, e fazem magia com procv e tabela dinâmica;
- E há aqueles que preferem **Python**, usando a programação para fazer tudo.

Mas com tanta diversidade e expertise, surge uma pergunta crucial...



# Como montar esse quebra cabeça?



# A "Cola" que une

Como garantir que uma consulta **SQL** possa se integrar harmoniosamente a um script no **Jupyter Notebook**?

Ou que um algoritmo **Python** possa se beneficiar das maravilhas de uma planilha **Excel**?

A resposta está na **Programação Orientada a Objetos (POO)**. A POO serve como a cola que une esses mundos.



# Difícil? Será?

Talvez você já tenha ouvido falar que Programação Orientada a Objetos é um bicho de sete cabeças, mas **não se preocupe!**

Dê uma chance para a gente te levar nessa jornada e, se você aprendeu PROCV meu amigo, **você aprende qualquer coisa.**



# POO

De cara, posso te falar que ela é usada para **abstrair e modelar dados**, garantindo que informações fluam suavemente entre bancos de dados, planilhas e scripts de programação, ou seja, **cria uma ponte** entre os mundos de dados!

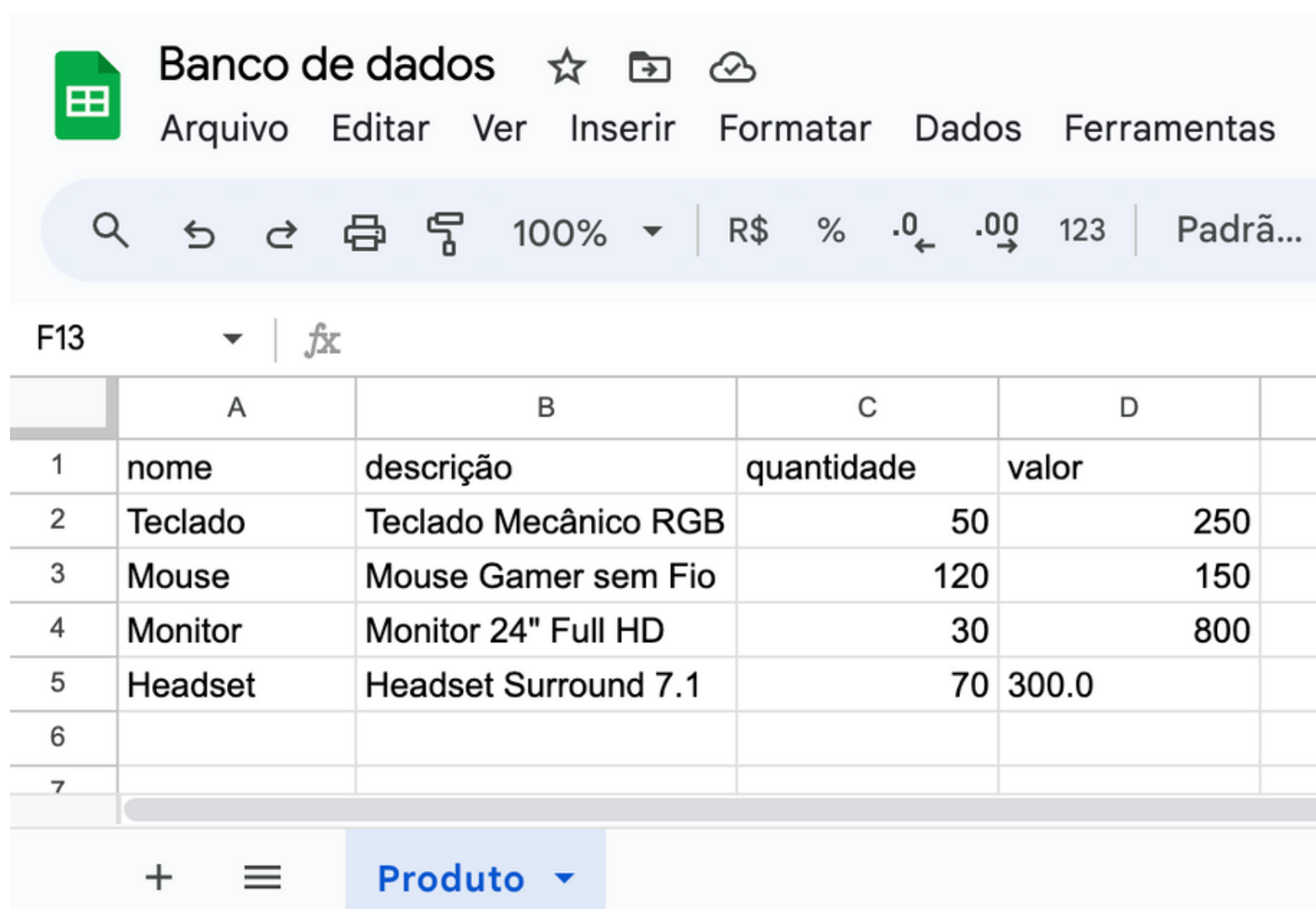
Beleza, não entendeu nada?

Então veja esse exemplo do Excel.



# Tipos básicos

Ao trabalhar com dados, rapidamente percebemos a **importância dos tipos de dados**. Veja no Excel, um erro de "tipagem" bastante comum



The screenshot shows the Microsoft Excel interface. The title bar reads "Banco de dados". The ribbon includes "Arquivo", "Editar", "Ver", "Inserir", "Formatar", "Dados", and "Ferramentas". The formula bar shows "F13" and a function icon. The worksheet contains a table with the following data:

	A	B	C	D
1	nome	descrição	quantidade	valor
2	Teclado	Teclado Mecânico RGB	50	250
3	Mouse	Mouse Gamer sem Fio	120	150
4	Monitor	Monitor 24" Full HD	30	800
5	Headset	Headset Surround 7.1	70	300.0
6				
7				

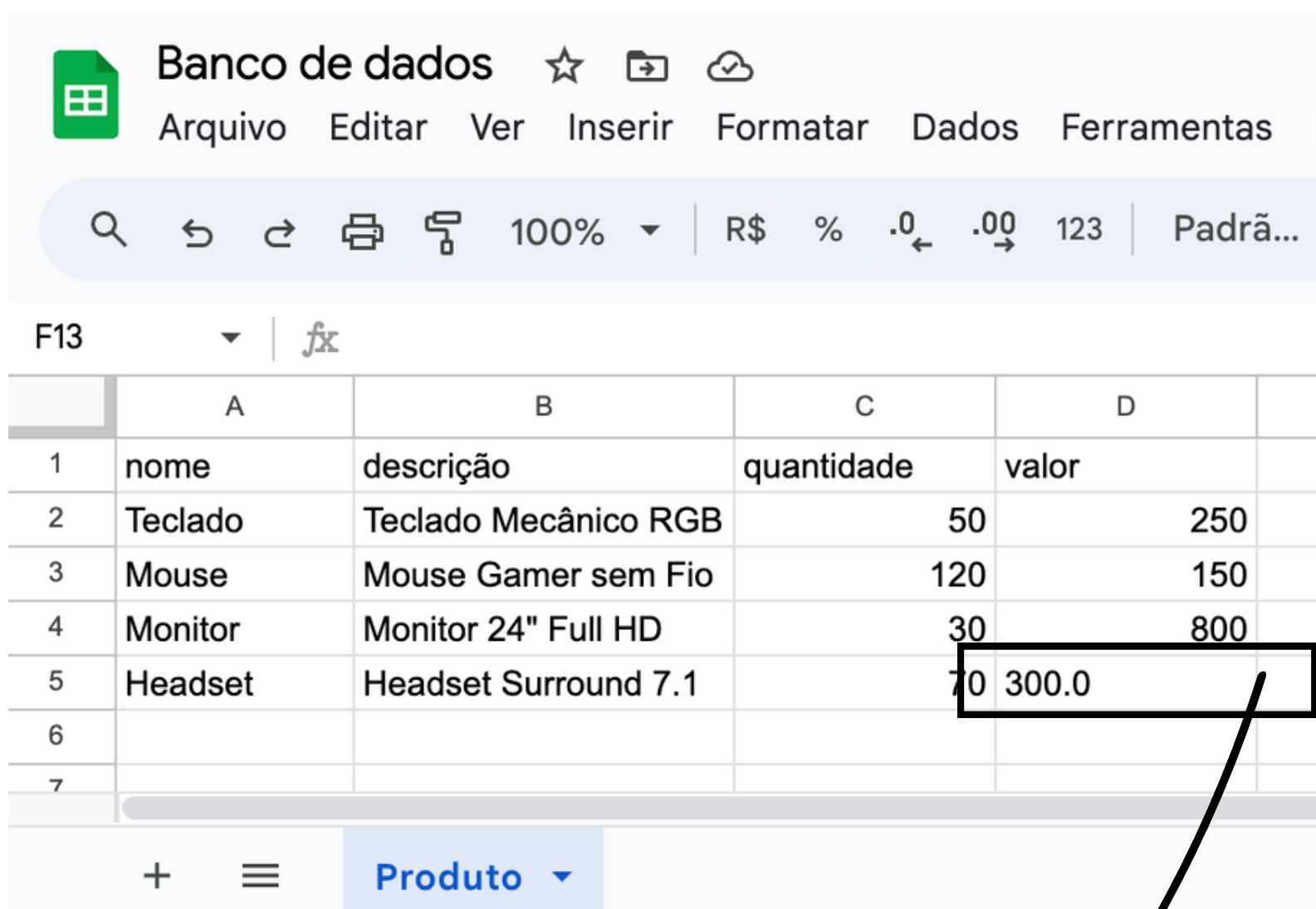
At the bottom of the interface, there is a tab labeled "Produto" with a dropdown arrow.

Você consegue achar o erro?



# O Erro

Na coluna 'valor' os 3 primeiros campos estão como **"inteiro"**



Banco de dados

Arquivo Editar Ver Inserir Formatar Dados Ferramentas

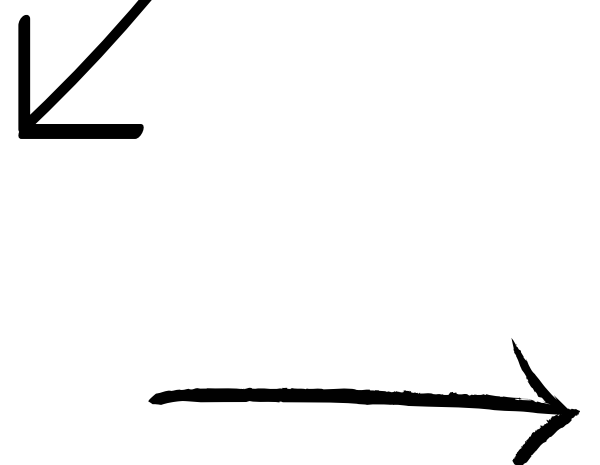
100% R\$ % .0 .00 123 Padrã...

F13

	A	B	C	D	
1	nome	descrição	quantidade	valor	
2	Teclado	Teclado Mecânico RGB	50	250	
3	Mouse	Mouse Gamer sem Fio	120	150	
4	Monitor	Monitor 24" Full HD	30	800	
5	Headset	Headset Surround 7.1	70	300.0	
6					
7					

+ ≡ Produto

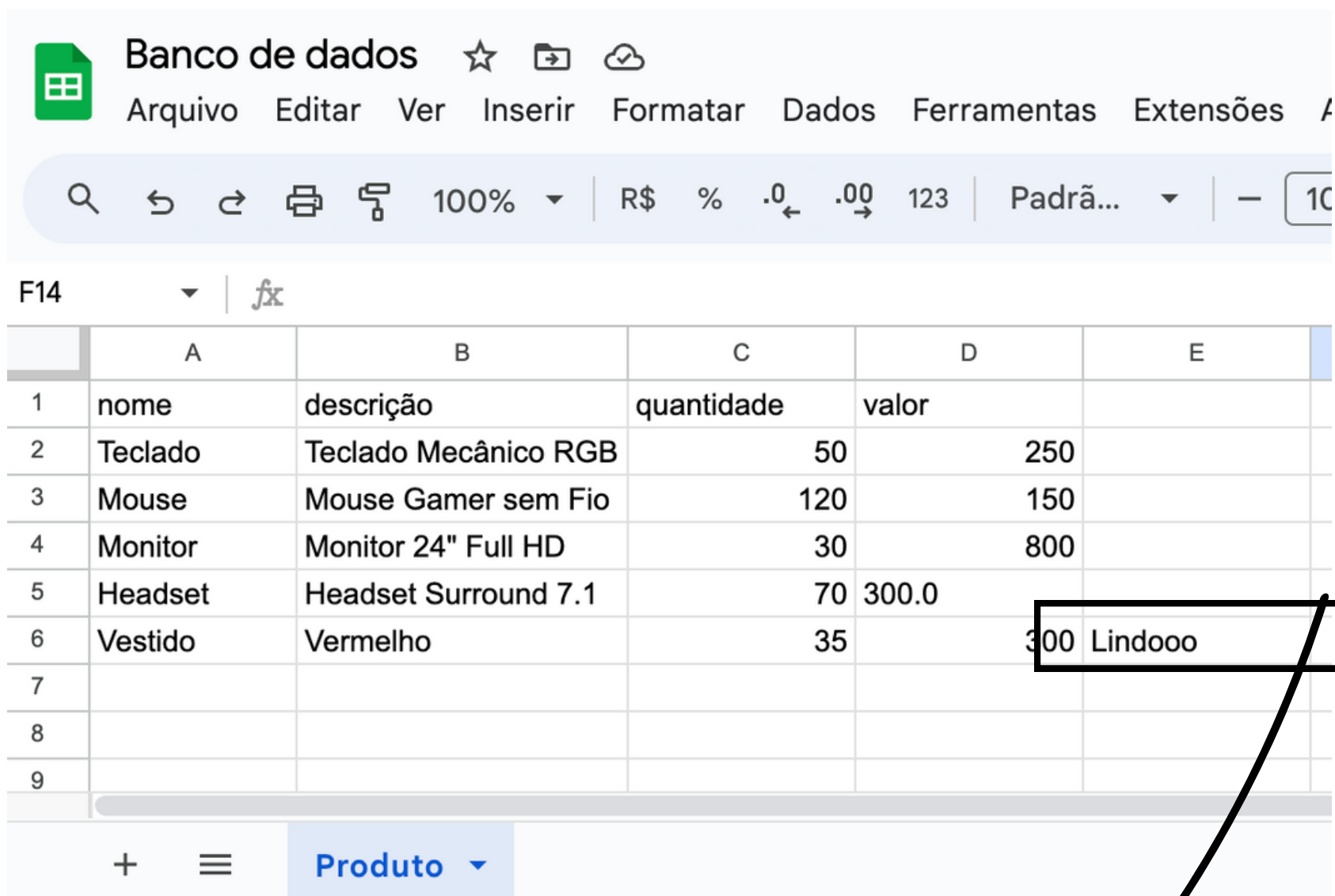
Já no campo 5D, por conta do ponto, acabou ficando como **"texto"**





# Erros de estrutura

Outro erro comum é você esperar 4 colunas, e do nada...



Banco de dados

Arquivo Editar Ver Inserir Formatar Dados Ferramentas Extensões

100% R\$ % .0 .00 123 Padrã... 10

F14

	A	B	C	D	E
1	nome	descrição	quantidade	valor	
2	Teclado	Teclado Mecânico RGB	50	250	
3	Mouse	Mouse Gamer sem Fio	120	150	
4	Monitor	Monitor 24" Full HD	30	800	
5	Headset	Headset Surround 7.1	70	300.0	
6	Vestido	Vermelho	35	300	Lindooo
7					
8					
9					

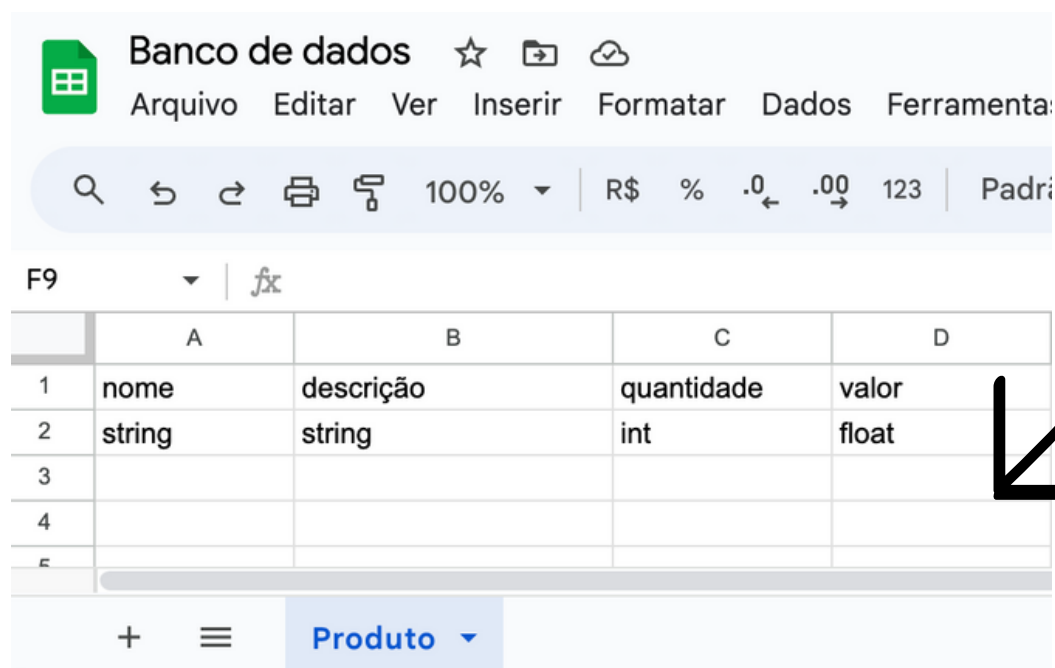
+ ≡ Produto

...alguém "*para te ajudar*"  
inclui outra coluna sem te  
avisar, você acaba tendo  
um **erro de estrutura**

# E como resolver isso?

Uma das formas de resolver, é você fixando essa estrutura através do **CREATE TABLE do SQL**

```
CREATE TABLE Produtos (  
  Nome VARCHAR(255) NOT NULL,  
  Descrição VARCHAR(255),  
  Quantidade INT NOT NULL DEFAULT 0,  
  Valor DECIMAL(10, 2) NOT NULL  
);
```



The screenshot shows the Microsoft Excel interface with a table named 'Produtos' in the 'Banco de dados' workbook. The table has four columns: 'nome', 'descrição', 'quantidade', and 'valor'. The first two rows are filled with data: 'string' and 'string' for 'nome' and 'descrição', and 'int' and 'float' for 'quantidade' and 'valor'. The rest of the rows are empty.

	A	B	C	D
1	nome	descrição	quantidade	valor
2	string	string	int	float
3				
4				
5				

Você declara  
as colunas, os  
tipos etc

Você perde a flexibilidade do Excel, mas ganha uma **estrutura padrão**.



# Você entendeu o que fez?

Você saiu de uma estrutura simples

Para **criar a sua própria estrutura**

Uma estrutura chamada **Produtos**.

Você entendeu mesmo?

Então parabéns, você **entendeu OOP**.

Vamos pro Python...



# Tipos Primitivos

Também temos nossos tipos em Python, podemos declarar nossas variáveis de forma isoladas.



```
nome = "Teclado" # tipo string  
descrição = "Tipo mecânico" # tipo string  
quantidade = 50 # tipo int  
valor = 250.00 # tipo float
```

Ou empacotar elas em listas, tuplas e dicionários....



# Tipos empacotados

Mas mesmo trabalhando com dicionários, no final, são **estruturas simples** e que geram **os mesmos problemas do excel**.

```
produto_1 = {
    "nome": "Teclado",
    "descrição": "Teclado Mecânico RGB",
    "quantidade": 50,
    "valor": 250
}

produto_2 = {
    "nome": "Teclado",
    "descrição": "Teclado Mecânico RGB",
    "quantidade": 50,
    "valor": "300.0"
}

produto_3 = {
    "nome": "Teclado",
    "descrição": "Teclado Mecânico RGB",
    "quantidade": 50,
    "valor": "300.0",
    "opinioao": "lindoooo"
}
```

Como resolver? Criando nossas **Classes**



# Classe e tabelas


Assim como as tabelas em SQL têm **colunas** para definir dados, as classes em OOP têm **atributos**. Pense em classes como 'tabelas moldáveis'."

python

 Copy code

```
class Cliente:
    def __init__(self, nome, email):
        self.nome = nome
        self.email = email
```

sql

 Copy code

```
CREATE TABLE clientes (
    nome VARCHAR(50),
    email VARCHAR(50)
);
```



# Classe Pessoas

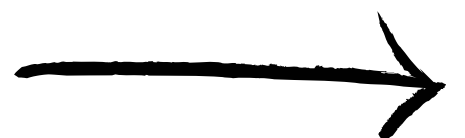
A orientação a objetos nos permite **definir classes com propriedades e métodos específicos.**

```
class Produto:
    def __init__(self, nome: str,
                  descricao: str,
                  quantidade: int,
                  valor: float):

        self.nome = nome
        self.descricao = descricao
        self.quantidade = quantidade
        self.valor = valor

teclado = Produto("Teclado", "Teclado Mecânico RGB", 50, 300)
```

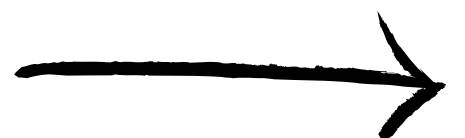
Se você entende como definir tipos em uma tabela, você está no caminho certo para compreender POO.



# Conclusão

A Programação Orientada a Objetos oferece uma abordagem para **modelar e estruturar software**, aproximando-o de entidades e **conceitos do mundo real**.

Além disso, a relação entre classes e objetos em OOP tem uma semelhança clara com tabelas e registros em bancos de dados, proporcionando **uma ponte conceitual** entre programação e gerenciamento de dados.





# Difícil?

Te falei que **não é era**.



Não deixe ninguém te falar que programação é **não é para você**, ok?

Apesar de ter muitos termos **específicos**, no final é criar uma espécie de "blueprint" / "tabela"!



# Obrigado!

Se essa postagem te ajudou, curta e comente!

**Sua interação pode ajudar outras pessoas** a se beneficiarem desse conhecimento.

Siga-me para receber dicas, insights e tutoriais de problemas que já passei muita noite virado para resolver!

(e espero que você não passe!)

