



Fundamentos de Banco de Dados

Consultas e informações

Nome	CPF	Sexo	Ocupação	Matrícula	Data de Nascimento	Data de Nascimento	Idade
10.040	0		10	9		32	
18.000	0		26	6		30	
36.046	0		19	8		19	
36.046	0		23	13		32	
30.040	0		15	4		25	
36.046	0		26	11		36	
36.046	0		10	8		20	
18.007	0		34	16		22	
36.046	0		10	10		32	
36.000	0		16	0		18	
30.000	0		13	17		22	
36.086	0		12	17		12	
40.086	0		24	13		10	
36.000	0		12	10		22	
36.000	0		12	10		20	

O que é uma Tabela?



Pense em uma tabela como uma planilha do Excel!

Estrutura de uma Tabela

 Exemplo:

- Colunas: Nome, CPF, Data de Nascimento
- Linhas: Cada pessoa (registro) da sua turma

Componentes

-  Cada coluna = um atributo
-  Cada linha = um registro

Relacionamentos entre Tabelas

Tabelas podem se conectar!



Tabela de Clientes

Armazena informações de cada cliente



Relacionamento

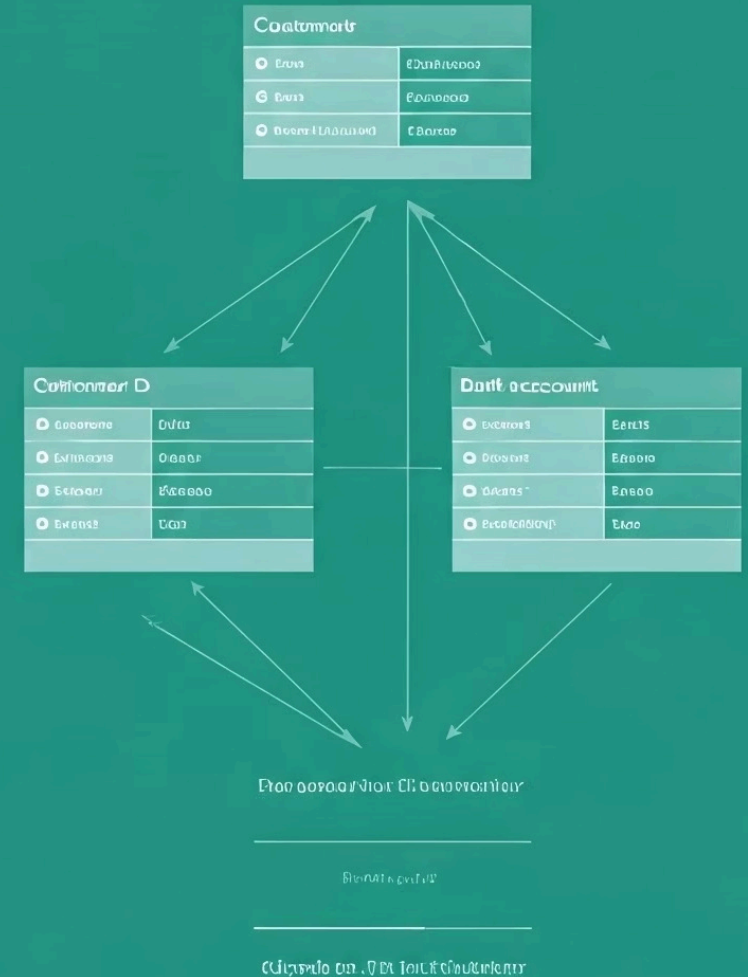
Conexão entre as tabelas



Tabela de Contas Bancárias

Armazena dados das contas

👉 Elas se relacionam para mostrar qual cliente tem qual conta



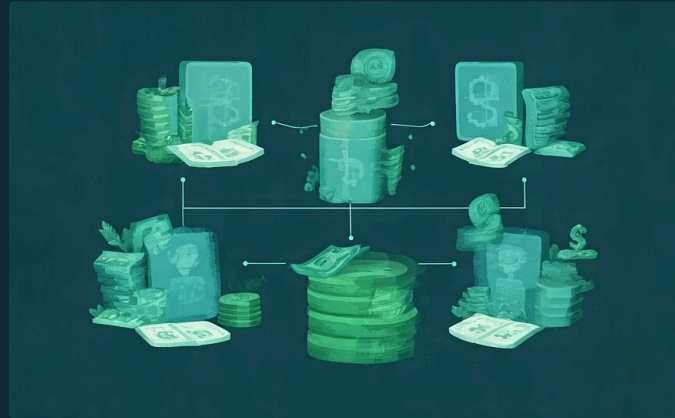
Tipos de Relacionamentos



Um para Um (One-to-One)

Exemplo do mundo real: 🧑 Pessoa ↔ CPF

Cada pessoa possui **um único CPF**, e cada CPF corresponde a **uma única pessoa**.



Um para Muitos (One-to-Many)

Exemplo do mundo real: 🧑 Cliente → 🏠 Contas Bancárias

Um cliente pode ter **várias contas**, mas cada conta está associada a **um único cliente**.



Muitos para Muitos (Many-to-Many)

Exemplo do mundo real: 🧑 Clientes ↔ 🏠 Projetos

Vários clientes podem estar envolvidos em **múltiplos projetos**, e cada projeto pode contar com **diversos clientes**.

O que é ORM?

ORM = Object-Relational Mapping



Código Python

🧠 Em vez de escrever SQL, usamos Python!



ORM (Django)

Traduz Python para SQL



Banco de Dados

Armazena os dados

🔧 O Django cuida de:

Criar as tabelas

Fazer buscas no banco

Salvar e deletar dados

O que é um Modelo no Django?


Modelo = Estrutura da Tabela

No Django:

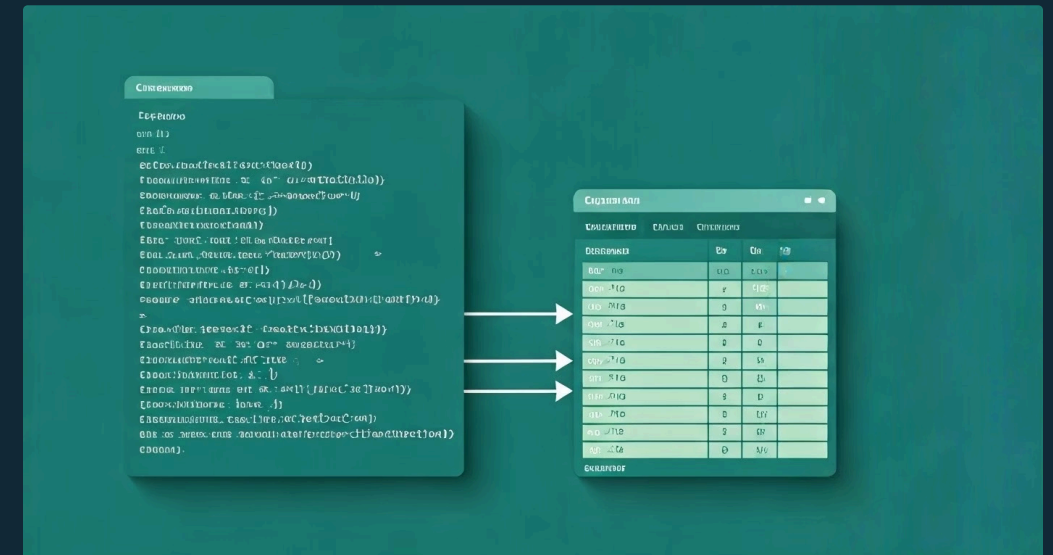
```
from django.db import models
```

```
class Cliente(models.Model):  
    nome = models.CharField(max_length=100)  
    email = models.EmailField()  
    data_nascimento = models.DateField()
```

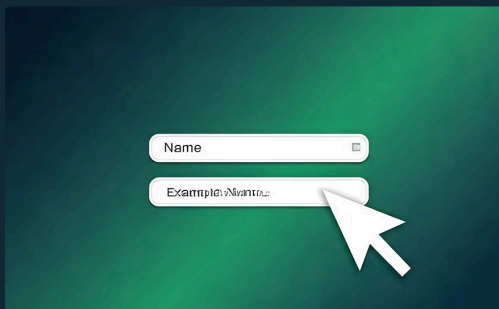
Equivalência

 Cada classe = uma tabela

 Cada atributo = uma coluna



Tipos de Campos de Modelo



CharField

Texto curto (ex: nome, CPF)



TextField

Texto longo (ex: descrição)



IntegerField

Números inteiros (ex: idade)



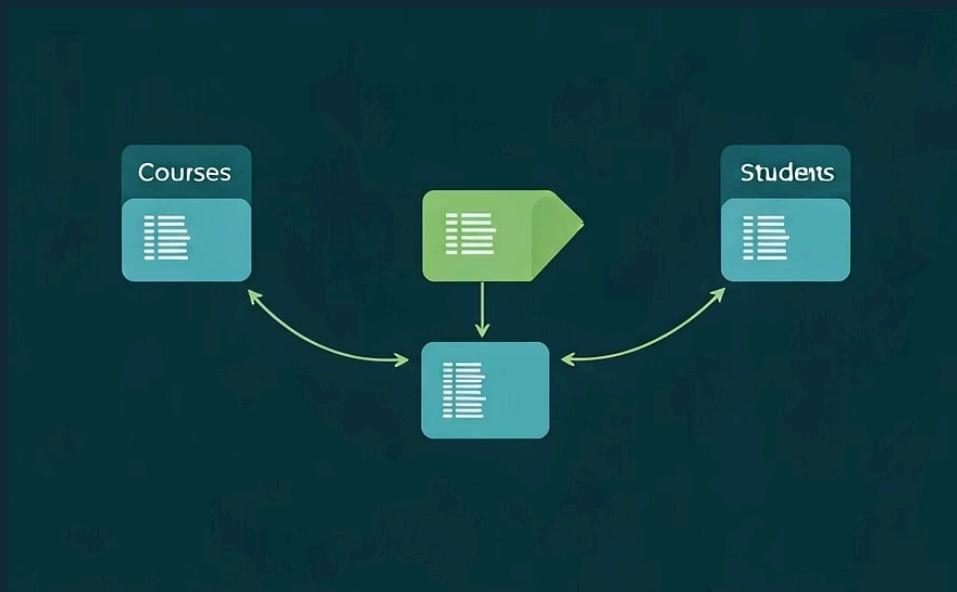
BooleanField

Verdadeiro ou falso (ex: ativo?)

Tipos de Campos Especiais

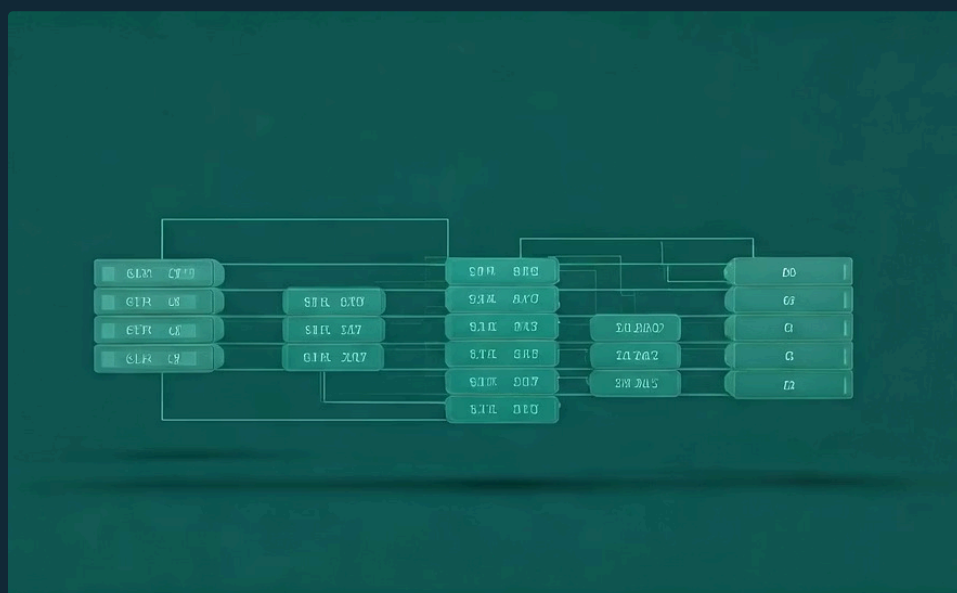


Campos de Relacionamento



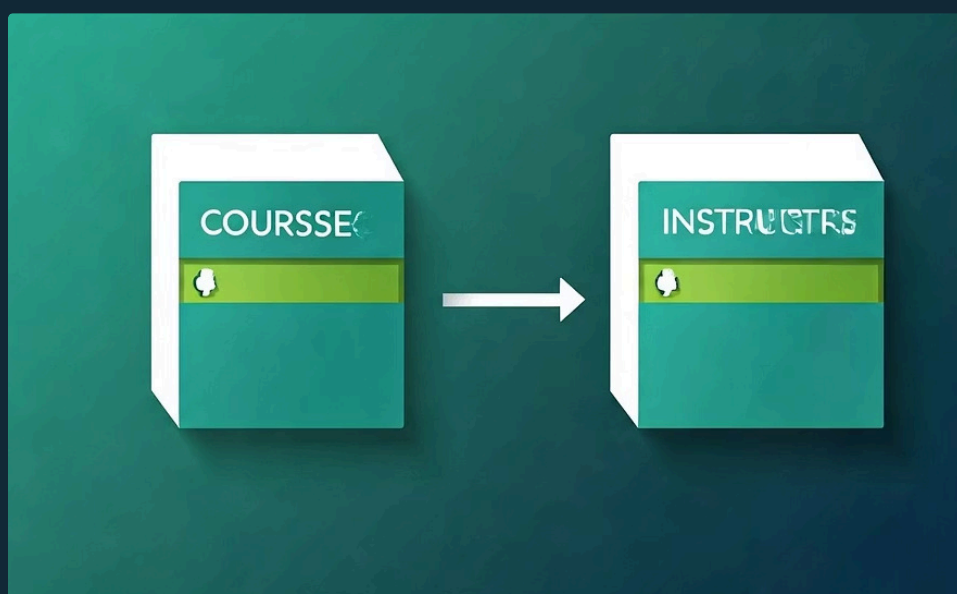
ForeignKey

Um para Muitos (ex: Conta → Cliente)



ManyToManyField

Muitos para Muitos (ex: Projeto ↔ Cliente)



OneToOneField

Um para Um (ex: Pessoa ↔ CPF)

Resumo para Levar pra Casa

Tabelas = Planilhas com dados

Organizam informações em linhas e colunas

Relacionamentos = Conexões entre tabelas

Um-para-Um, Um-para-Muitos, Muitos-para-Muitos

Django ORM = Escrevemos código Python no lugar de SQL

Simplifica a interação com o banco de dados

Models = Como dizemos ao Django o que deve existir no banco

Classes Python que definem a estrutura do banco

