En Django, les **modèles** (`models.Model`) représentent les structures de données de ton application. Ils définissent **les champs** (colonnes de la base de données) et **les types de données** autorisées.

Voici les **types de champs principaux** qu'on peut utiliser dans un modèle Django :

🧮 1. Champs de base

Type de champ	Description
`CharField(max_length=...)`	Texte court (ex : noms, titres)
`TextField()`	Texte long (ex : description)
`IntegerField()`	Entier
`FloatField()`	Nombre décimal flottant
`BooleanField()`	Vrai / Faux
`DateField()`	Date uniquement
`DateTimeField()`	Date + heure
`TimeField()`	Heure uniquement
`EmailField()`	Adresse email (validée)
`URLField()`	URL (validée)
`SlugField()`	Texte court compatible URL
`UUIDField()`	Identifiant unique UUID
`BinaryField()`	Données binaires (ex : fichiers bruts)

🔗 2. Champs de relation (entre modèles)

Type de champ	Description
`ForeignKey(Model, on_delete=...)`	Relation 1 à N (ex : un article a un auteur)
`OneToOneField(Model, on_delete=...)`	Relation 1 à 1 (ex : profil utilisateur)
`ManyToManyField(Model)`	Relation N à N (ex : tags d'un article)

🧰 3. Champs spécialisés

Type de champ	Description
`ImageField(upload_to=...)`	Image téléchargée
`FileField(upload_to=...)`	Fichier quelconque

Type de champ	Description
`DecimalField(max_digits=..., decimal_places=...)`	Nombre décimal précis (ex : montants)
`DurationField()`	Durée (ex : intervalle de temps)
`JSONField()`	Données au format JSON (Django 3.1+)

📦 Exemple pratique :

```
from django.db import models
from django.contrib.auth import get_user_model

class Recipe(models.Model):
    user = models.ForeignKey(get_user_model(), on_delete=models.CASCADE)
    title = models.CharField(max_length=255)
    time_minutes = models.IntegerField()
    price = models.DecimalField(max_digits=5, decimal_places=2)
    description = models.TextField(blank=True)
    link = models.URLField(blank=True)

    def __str__(self):
        return self.title
```

Bien sûr ! Voici une **explication détaillée** de chaque type de champ Django, avec ses usages et particularités :

---

1. Champs de base

`CharField(max_length=...)`

- **Description :** Champ texte court, obligatoire de spécifier la longueur maximale (ex : `max_length=255`).
- **Usage courant :** noms, titres, petites descriptions.
- **Exemple :**

```
name = models.CharField(max_length=100)
```

- **Note :** La longueur maximale est appliquée côté base de données.

`TextField()`

- **Description :** Texte long illimité, pas besoin de `max_length`.
- **Usage courant :** descriptions, commentaires, contenus longs.
- **Exemple :**

```
description = models.TextField()
```

- **Note :** Plus flexible que `CharField` mais peut être moins performant pour certaines requêtes.

`IntegerField()`

- **Description :** Nombre entier (positif ou négatif).
- **Usage courant :** quantités, âges, identifiants numériques.
- **Exemple :**

```
age = models.IntegerField()
```

`FloatField()`

- **Description :** Nombre à virgule flottante (décimal approximatif).
- **Usage courant :** valeurs numériques avec décimales, mais où la précision absolue n'est pas critique.
- **Exemple :**

```
rating = models.FloatField()
```

`DecimalField(max_digits=..., decimal_places=...)`

- **Description :** Nombre décimal avec précision définie, idéal pour les montants financiers.
- **Paramètres :**
  - `max_digits` : nombre total de chiffres
  - `decimal_places` : nombre de chiffres après la virgule
- **Usage courant :** prix, monnaies.
- **Exemple :**

```
price = models.DecimalField(max_digits=6, decimal_places=2)
```

- **Note :** Plus précis que `FloatField` pour les calculs monétaires.

`BooleanField()`

- **Description :** Valeur vraie ou fausse.
- **Usage courant :** champs « oui/non », activé/désactivé.
- **Exemple :**

```
is_active = models.BooleanField(default=True)
```

`DateField()`

- **Description :** Stocke une date (année, mois, jour).
- **Usage courant :** dates d'anniversaire, événements, date de création.
- **Exemple :**

```
birth_date = models.DateField()
```

`DateTimeField()`

- **Description :** Stocke date + heure.
- **Usage courant :** timestamps, horodatages.
- **Exemple :**

```
created_at = models.DateTimeField(auto_now_add=True)
```

`TimeField()`

- **Description :** Stocke uniquement l'heure (ex : `15:30:00`).
- **Usage courant :** horaires d'ouverture, durées.
- **Exemple :**

```
start_time = models.TimeField()
```

EmailField()

- **Description :** Champ texte avec validation automatique d'un email.
- **Usage courant :** adresse email.
- **Exemple :**

```
email = models.EmailField()
```

URLField()

- **Description :** Champ texte validant un URL.
- **Usage courant :** liens web.
- **Exemple :**

```
website = models.URLField(blank=True)
```

SlugField()

- **Description :** Champ texte court utilisé pour les URL (ex : « mon-article-2025 »).
- **Usage courant :** parties d'URL lisibles.
- **Exemple :**

```
slug = models.SlugField(max_length=50)
```

UUIDField()

- **Description :** Champ stockant un identifiant unique universel.
- **Usage courant :** clés primaires ou identifiants uniques non séquentiels.
- **Exemple :**

```
id = models.UUIDField(primary_key=True, default=uuid.uuid4,
editable=False)
```

`BinaryField()`

- **Description :** Stocke des données binaires (non lisibles).
- **Usage courant :** fichiers bruts, données cryptées.
- **Exemple :**

```
data = models.BinaryField()
```

---

2. Champs de relation

`ForeignKey(Model, on_delete=...)`

- **Description :** Relation « plusieurs à un ».
- **Usage courant :** une recette appartient à un utilisateur.
- **Paramètre important :** `on_delete` définit ce qui se passe si l'objet lié est supprimé (`CASCADE`, `SET_NULL`, etc).
- **Exemple :**

```
user = models.ForeignKey(User, on_delete=models.CASCADE)
```

`OneToOneField(Model, on_delete=...)`

- **Description :** Relation « un à un ».
- **Usage courant :** profil utilisateur lié à un seul utilisateur.
- **Exemple :**

```
profile = models.OneToOneField(User, on_delete=models.CASCADE)
```

`ManyToManyField(Model)`

- **Description :** Relation « plusieurs à plusieurs ».
- **Usage courant :** un article peut avoir plusieurs tags, un tag plusieurs articles.
- **Exemple :**

```
tags = models.ManyToManyField(Tag)
```

---

3. Champs spécialisés

`ImageField(upload_to=...)`

- **Description :** Champ pour stocker une image.
- **Usage courant :** photos de profil, images de produit.
- **Exemple :**

```
photo = models.ImageField(upload_to='photos/%Y/%m/%d/')
```

`FileField(upload_to=...)`

- **Description :** Champ pour stocker un fichier.
- **Usage courant :** documents, fichiers PDF.
- **Exemple :**

```
document = models.FileField(upload_to='documents/')
```

`DurationField()`

- **Description :** Durée de temps (ex : `2 hours 30 minutes`).
- **Usage courant :** durée d'un événement.
- **Exemple :**

```
duration = models.DurationField()
```

`JSONField()`

- **Description :** Champ stockant du JSON (dictionnaire / liste).
- **Usage courant :** données semi-structurées.
- **Exemple :**

```
metadata = models.JSONField(null=True, blank=True)
```

En résumé

Type	Usage typique	Exemple rapide
`CharField`	Texte court	`title = models.CharField(max_length=100)`
`TextField`	Texte long	`description = models.TextField()`
`IntegerField`	Nombres entiers	`age = models.IntegerField()`
`DecimalField`	Montants précis	`price = models.DecimalField(max_digits=6, decimal_places=2)`
`BooleanField`	True / False	`is_active = models.BooleanField(default=True)`
`ForeignKey`	Clé étrangère	`user = models.ForeignKey(User, on_delete=models.CASCADE)`
`ManyToManyField`	Relation plusieurs-à-plusieurs	`tags = models.ManyToManyField(Tag)`