

# Django – Les Champs des Modèles

## Définition

Chaque modèle Django est une **classe** qui hérite de `django.db.models.Model`.

Les variables de classe définissent les **champs de la base de données**.

Chaque champ est une **instance d'une classe Field** (exemple : `CharField`, `IntegerField`, etc.).

👉 Le nom de la variable devient le **nom du champ en interne** (et le nom de colonne dans la base).

---

## Types de Champs

- **AutoField** → incrémente automatiquement sa valeur.
  - **BinaryField** → stocke des données binaires (bytes).
  - **BooleanField** → champ booléen (True / False).
  - **CharField** → chaîne de caractères courte (`max_length` requis).
  - **TextField** → texte long.
  - **CommaSeparatedIntegerField** → liste d'entiers séparés par virgule.
  - **EmailField** → vérifie une adresse email valide.
  - **SlugField** → slug (alphanumérique + tirets).
  - **URLField** → URL valide.
  - **DateField** → date (`datetime.date`).
  - **DateTimeField** → date et heure (`datetime.datetime`).
  - **DecimalField** → nombre décimal précis (`Decimal`).
  - **FloatField** → nombre flottant (`float`).
  - **FileField** → fichier téléversé.
  - **ImageField** → fichier image (vérifie le format).
  - **FilePathField** → chemin vers un fichier.
  - **GenericIPAddressField** → adresse IP (IPv4 ou IPv6).
  - **IPAddressField** → ancienne version (IPv4 uniquement).
  - **IntegerField** → entier 32 bits (-2 147 483 648 à 2 147 483 647).
  - **BigIntegerField** → entier 64 bits.
  - **PositiveIntegerField** → entier positif (0 à 2 147 483 647).
  - **SmallIntegerField** → entier 16 bits (-32 768 à 32 767).
  - **PositiveSmallIntegerField** → entier positif (0 à 32 767).
  - **NullBooleanField** → booléen + `Null` (déprécié).
  - **TimeField** → heure (`datetime.time`).
- 

## Relations

- **ForeignKey** → relation plusieurs-à-un.
  - **ManyToManyField** → relation plusieurs-à-plusieurs.
  - **OneToOneField** → relation un-à-un.
-

## Paramètres communs

- `db_column` → nom de la colonne en base.
  - `db_index` → création d'un index.
  - `default` → valeur par défaut.
  - `editable` → champ modifiable dans l'admin ( `False` = non modifiable).
  - `help_text` → texte d'aide dans les formulaires.
  - `primary_key` → définit la clé primaire.
  - `unique` → valeur unique (pas de doublon).
  - `verbose_name` → nom plus explicite.
  - `validators` → liste de validateurs.
- 


## Paramètres spécifiques

- `blank` → champ peut être laissé vide dans un formulaire.
  - `null` → autorise les valeurs NULL en base.
  - `unique_for_date` → unique pour une date donnée.
  - `unique_for_month` → unique pour un mois donné.
  - `unique_for_year` → unique pour une année donnée.
  - `choices` → liste de choix possibles.
- 

## Exemple

```
class Produit(models.Model):
    intitule = models.CharField(max_length=200, unique=True)
    prix = models.DecimalField(max_digits=10, decimal_places=2, default=0.0)
    disponible = models.BooleanField(default=True)
    date_ajout = models.DateTimeField(auto_now_add=True)

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

 Avec Django ORM, vous manipulez vos objets Python, et Django traduit automatiquement en SQL adapté au SGBD utilisé.