## **1. What is a Model in Django?**

* A **Model** is a Python class that represents a **table in the database**.
* Each **attribute (variable)** in the model represents a **column** in the table.
* Django automatically **maps your Python class to the database table** using its ORM (Object Relational Mapping).

## **2. Example Model**

student/models.py

**from** django**.**db **import** models

*# Create your models here.*

class **Student**(*models***.***Model*)**:**

    name **=** models**.**CharField(**max\_length=**70)            *# Text column with max length of 70 chars*

    email **=** models**.**EmailField(**max\_length=**255)         *# Stores and validates email addresses*

    city **=** models**.**CharField(**max\_length=**70)            *# Text column for city*

    roll **=** models**.**IntegerField()                      *# Integer column*

**def** \_\_str\_\_(*self*)**:**

**return** f"{*self***.**name} ({*self***.**roll})"

## **3. Commonly Used Field Types**

| **Field Name** | **Description** | **Example** |
| --- | --- | --- |
| CharField | Short text | models.CharField(max\_length=50) |
| TextField | Large text | models.TextField() |
| EmailField | Validates email | models.EmailField() |
| IntegerField | Integer values | models.IntegerField() |
| FloatField | Decimal values | models.FloatField() |
| DateField | Date only | models.DateField(auto\_now\_add=True) |
| DateTimeField | Date + time | models.DateTimeField(auto\_now=True) |
| BooleanField | True/False | models.BooleanField(default=False) |
| ImageField | Image upload | models.ImageField(upload\_to='images/') |
| FileField | File upload | models.FileField(upload\_to='files/') |
| ForeignKey | Many-to-One relationship | models.ForeignKey(OtherModel, on\_delete=models.CASCADE) |

## **4. Steps to Create Database Table from Model**

1️⃣ **Add the app to INSTALLED\_APPS**  
project/settings.py

INSTALLED\_APPS **=** [

    ...

    'student'**,**   *# register the app where model exists*

]

2️⃣ **Create the model** (as shown above).

3️⃣ **Make Migrations**  
Creates migration files based on the models.

python manage**.**py makemigrations

Output example:

Migrations for 'student':

student/migrations/0001\_initial.py

4️⃣ **Apply Migrations to Database**

python manage**.**py migrate

This creates the actual database table.

5️⃣ **Check All Migrations**

python manage**.**py showmigrations

Shows a list of migrations and whether they’ve been applied ([X]).

6️⃣ **View SQL Query of a Migration**

* python manage**.**py sqlmigrate student 0001
* Shows the exact SQL command Django will run for migration 0001 in student app.

## **5. Additional Notes**

* Table name: By default, Django names it as **appname\_modelname** (e.g., student\_student).
* **Model class name** should always be **singular & PascalCase** (Student, not student).
* **\_\_str\_\_ method**: Always define it for better readability in Django admin.
* If you **change the model**, you must run:
* python manage**.**py makemigrations
* python manage**.**py migrate
* You can create **superuser** to manage data in Admin:
* python manage**.**py createsuperuser

## **6. Complete Workflow Diagram**

Model Class **in** models**.**py

       ↓

python manage**.**py makemigrations

       ↓

Migration File (0001\_initial**.**py)

       ↓

python manage**.**py migrate

       ↓

Database Table Created