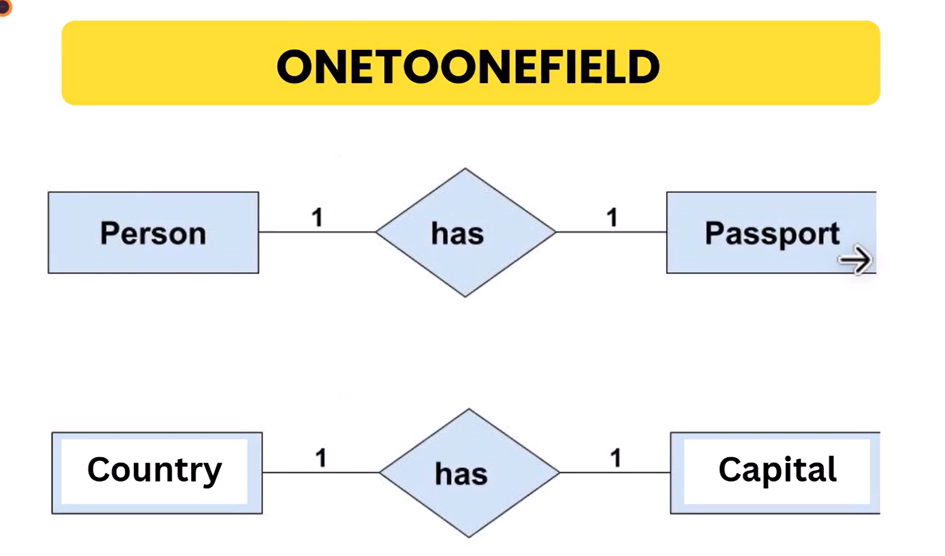
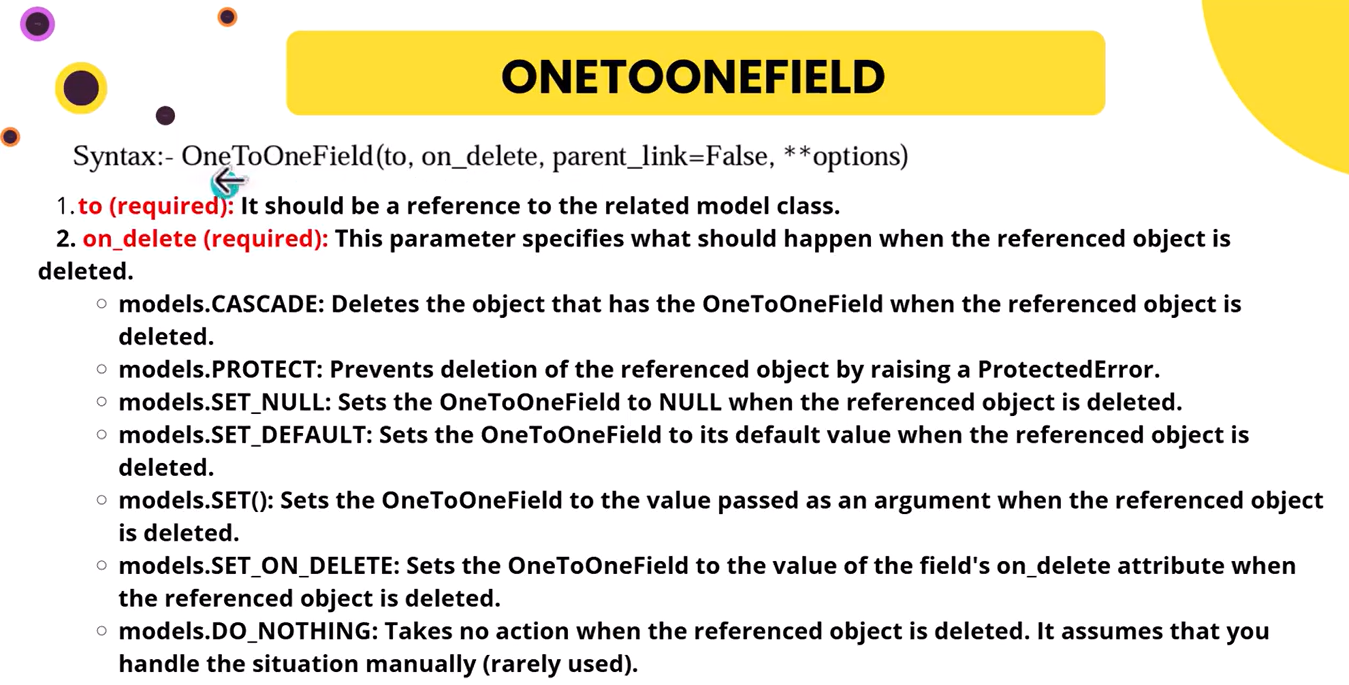


**One-to-One**

****

****

**OneToOneField কি?**

* এটা Django মডেলে ব্যবহার করা হয় দুইটা মডেলের মধ্যে **এক-এক সম্পর্ক** বানানোর জন্য।
* মানে, একটা মডেলের এক টা রেকর্ড সরাসরি অন্য মডেলের **এক টা রেকর্ডের সাথে সম্পর্কিত**।

**সিনট্যাক্স:**

OneToOneField(to, on\_delete, parent\_link=False, \*\*options)

**প্রধান অংশগুলো:**

1. **to (প্রয়োজনীয়)**
   * এটা হলো সেই মডেল যার সাথে সম্পর্ক তৈরি করতে চাও।
   * উদাহরণ: যদি তুমি Profile মডেলকে User মডেলের সাথে সম্পর্কিত করতে চাও, তাহলে হবে:

user = models.OneToOneField(User, on\_delete=models.CASCADE)

1. **on\_delete (প্রয়োজনীয়)**
   * যখন to মডেলের রেকর্ড ডিলিট হবে, তখন OneToOneField কী করবে সেটা ঠিক করে।
   * কিছু সাধারণ মান:
     + **models.CASCADE** → সম্পর্কিত রেকর্ডও মুছে যাবে।
     + **models.PROTECT** → ডিলিট হতে দেবে না, ProtectedError দিবে।
     + **models.SET\_NULL** → সম্পর্কটি NULL হয়ে যাবে।
     + **models.SET\_DEFAULT** → ডিফল্ট মানে সেট হয়ে যাবে।
     + **models.SET(value)** → যেই মানটি দেয়া হয়েছে তা হবে।
     + **models.SET\_ON\_DELETE** → on\_delete এর মান অনুযায়ী হবে।
     + **models.DO\_NOTHING** → কিছুই করবে না, তুমি নিজে ম্যানুয়ালি হ্যান্ডেল করতে হবে।

**সহজ উদাহরণ:**

from django.db import models

from django.contrib.auth.models import User

class Profile(models.Model):

user = models.OneToOneField(User, on\_delete=models.CASCADE)

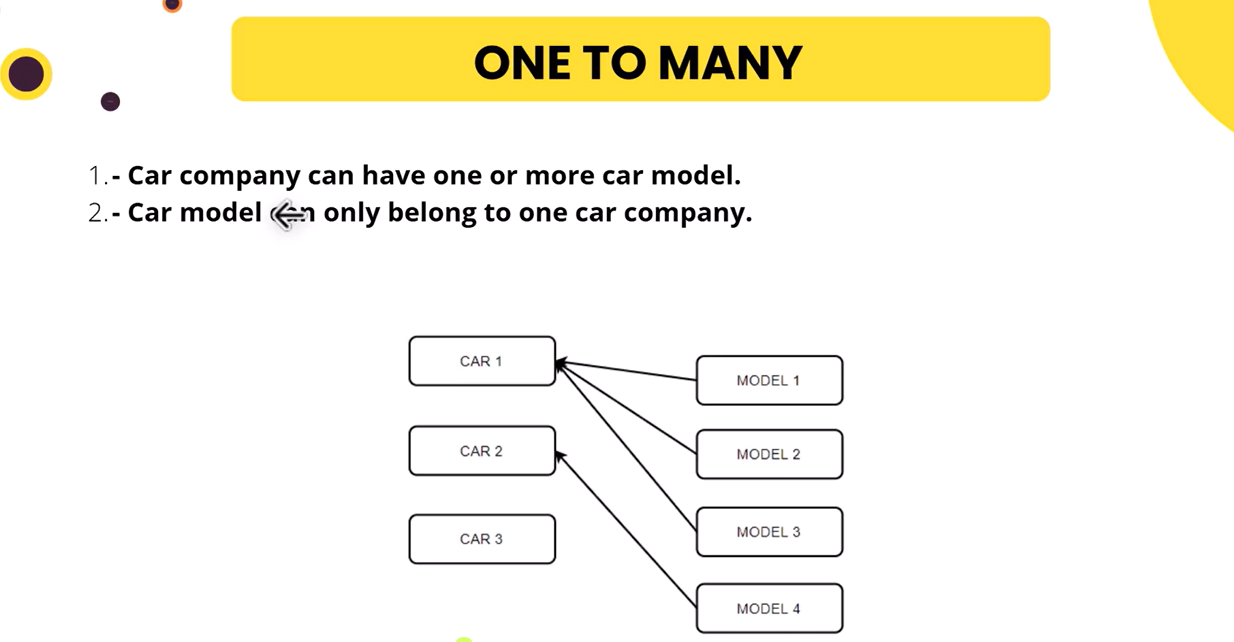
bio = models.TextField()

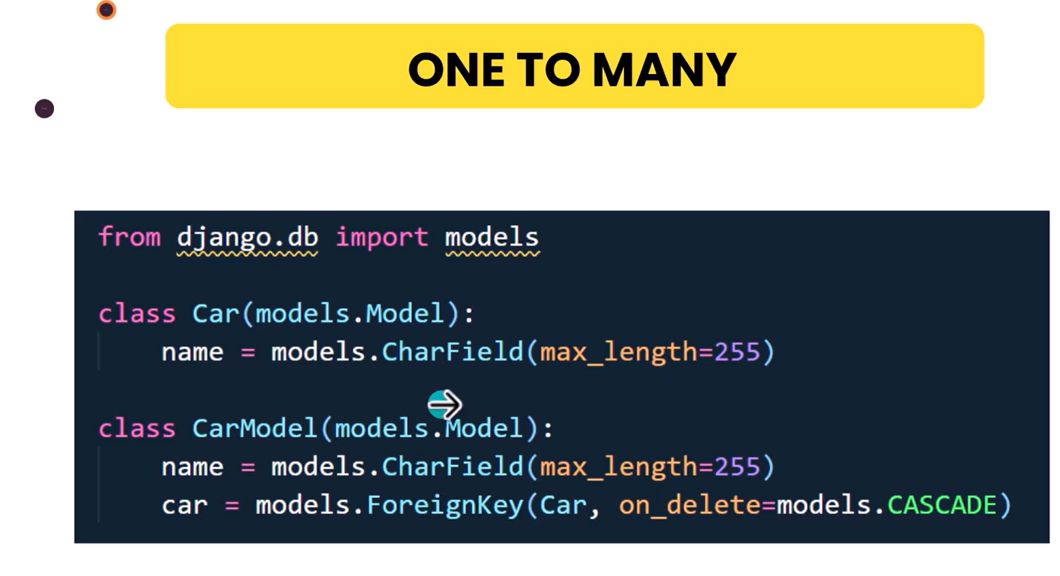
* এখানে **প্রতি User এর জন্য শুধু একটি Profile থাকবে**।
* যদি User মুছে যায়, Profile ও মুছে যাবে (CASCADE)।

💡 সংক্ষেপে:

* **OneToOneField = একেকটি রেকর্ড শুধু একেকটি রেকর্ডের সাথে সম্পর্কিত**।
* **on\_delete = যখন সম্পর্কিত রেকর্ড ডিলিট হবে তখন কী হবে**।

**One-to-Many**

****



**🔹 One-to-Many কী?**

One-to-Many মানে হলো:

* একদিকের একটি রেকর্ড (Parent)
* অপর দিকে একাধিক রেকর্ড (Child) এর সাথে যুক্ত হতে পারে।

👉 উদাহরণ: **একজন Teacher → অনেকগুলো Student**  
একজন শিক্ষক অনেক শিক্ষার্থী পড়াতে পারেন। কিন্তু প্রতিটি শিক্ষার্থী একজন শিক্ষকের অধীনে থাকবে (যদি আমরা এভাবে relation define করি)।

**🔹 Django তে কিভাবে করা হয়?**

Django তে **ForeignKey** ব্যবহার করলে One-to-Many relation তৈরি হয়।

from django.db import models

class Teacher(models.Model):

name = models.CharField(max\_length=100)

class Student(models.Model):

name = models.CharField(max\_length=100)

teacher = models.ForeignKey(Teacher, on\_delete=models.CASCADE)

**🔹 ব্যাখ্যা**

* Student মডেলে teacher ফিল্ডটা আছে।
* প্রতিটি Student কেবলমাত্র **একজন Teacher** এর সাথে যুক্ত।
* কিন্তু একজন Teacher এর সাথে **অনেক Student** থাকতে পারবে।

**🔹 Access করার নিয়ম**

*# 1) Student থেকে Teacher বের করা*

student = Student.objects.get(name="Rahim")

print(student.teacher.name)

*# 2) Teacher থেকে Student গুলো বের করা*

teacher = Teacher.objects.get(name="Mr. Karim")

students = teacher.student\_set.all() *# Django by default "\_set" যোগ করে*

for s in students:

print(s.name)

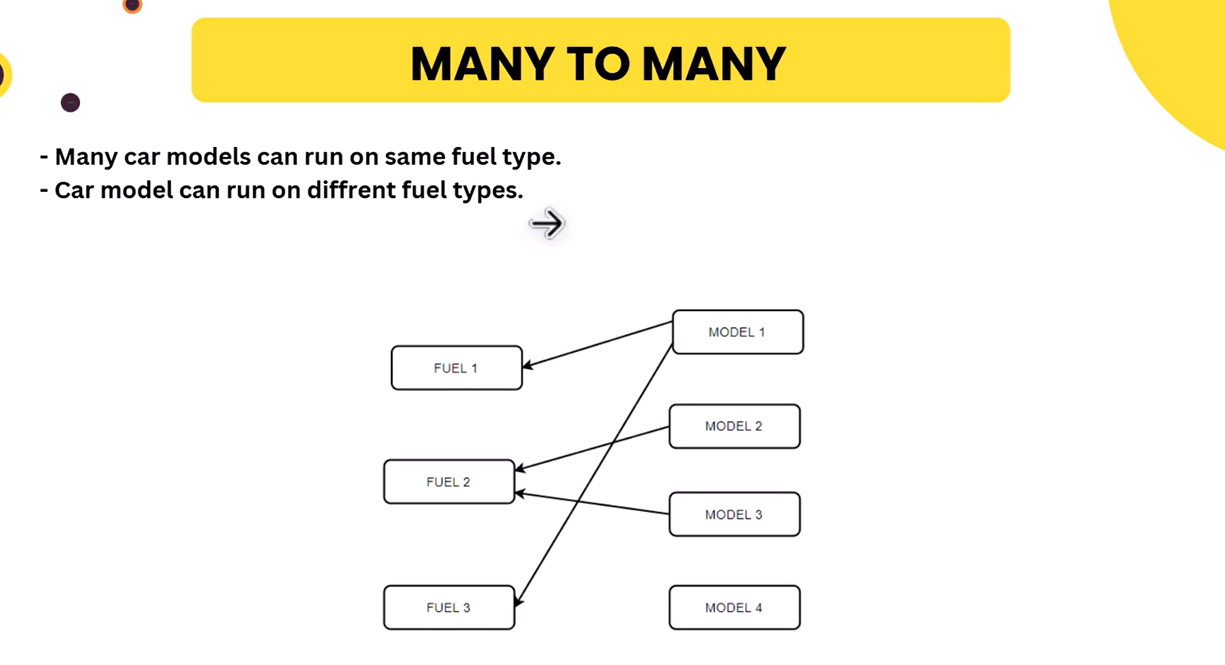
**🔹 বাস্তব উদাহরণ**

* **Author ↔ Books** → একজন Author অনেকগুলো বই লিখতে পারেন।
* **Company ↔ Employees** → একটি কোম্পানির অনেক কর্মচারী থাকতে পারে।
* **Category ↔ Products** → একটি ক্যাটাগরিতে অনেক পণ্য থাকতে পারে।

👉 সংক্ষেপে:

* **OneToOneField → একে অপরের সাথে শুধু একবার সম্পর্ক।**
* **ForeignKey (One-to-Many) → একদিকে একটি, অন্যদিকে অনেক।**

**Many-TO-Many**

****



অসাধারণ প্রশ্ন 👍

এখন আমরা **Many-to-Many** relation শিখব (Django তে ManyToManyField)।

**🔹 Many-to-Many কী?**

**Many-to-Many** মানে হলো:

* একদিকের একটি রেকর্ড → অপর দিকে অনেকগুলোর সাথে যুক্ত হতে পারে।
* আবার অন্যদিক থেকেও → অনেকগুলো রেকর্ড ওই একটার সাথে যুক্ত হতে পারে।

👉 উদাহরণ:

* **Student ↔ Course**  
  একজন Student অনেকগুলো Course নিতে পারে, আবার একটি Course অনেক Student এর থাকতে পারে।

**🔹 Django তে কিভাবে করা হয়?**

from django.db import models

class Student(models.Model):

name = models.CharField(max\_length=100)

class Course(models.Model):

title = models.CharField(max\_length=100)

students = models.ManyToManyField(Student)

**🔹 ব্যাখ্যা**

* এখানে Course এর সাথে অনেকগুলো Student থাকতে পারে।
* আবার একজন Student অনেকগুলো Course এ enroll হতে পারবে।
* Django ভেতরে ভেতরে **একটা intermediate table** বানিয়ে রাখে (যেটা তুমি আলাদা না বানালেও Django তৈরি করে নেয়)।

**🔹 Access করার নিয়ম**

*# 1) Student থেকে Course পাওয়া*

student = Student.objects.get(name="Rahim")

courses = student.course\_set.all()

for c in courses:

print(c.title)

*# 2) Course থেকে Student পাওয়া*

course = Course.objects.get(title="Math")

students = course.students.all()

for s in students:

print(s.name)

*# 3) নতুন সম্পর্ক যোগ করা*

course.students.add(student)

*# 4) সম্পর্ক মুছে ফেলা*

course.students.remove(student)

**🔹 বাস্তব উদাহরণ**

* **Doctors ↔ Patients** → একজন ডাক্তার অনেক রোগী দেখতে পারেন, আর একজন রোগী অনেক ডাক্তার দেখাতে পারে।
* **Movies ↔ Actors** → একটি সিনেমায় অনেক অভিনেতা থাকতে পারে, আবার একজন অভিনেতা অনেক সিনেমায় অভিনয় করতে পারে।
* **Users ↔ Groups (Permissions system)** → একজন ইউজার অনেক গ্রুপে থাকতে পারে, আবার এক গ্রুপে অনেক ইউজার থাকতে পারে।

**🔹 তুলনামূলক টেবিল**

| **Relationship** | **Django Field** | **উদাহরণ** |
| --- | --- | --- |
| One-to-One | OneToOneField | Person ↔ Passport |
| One-to-Many | ForeignKey | Teacher ↔ Students |
| Many-to-Many | ManyToManyField | Student ↔ Course |