Relationship Fields

Relationship Fields

- ForeignKey()

- ManyToManyField()

- OneToOneField()

- Clearly, the power of relational databases lies in relating tables to each other.

- Django offers ways to define the three most common types of database relationships:

  - many-to-one,

  - many-to-many

  - one-to-one

Many to One

- To define a many-to-one relationship, use django.db.models.ForeignKey.

- You use it just like any other Field type: by including it as a class attribute of your model.

- ForeignKey requires a positional argument: the class to which the model is related.

- Example

  ```
  from django.db import models

  class Course(models.Model):
      ...

  class Batch(models.Model):
      course = models.ForeignKey(Course, on_delete=models.CASCADE)
  ```

-

Many to Many

- To define a many-to-many relationship, use ManyToManyField.

- You use it just like any other Field type: by including it as a class attribute of your model

- ManyToManyField requires a positional argument: the class to which the model is related.

```python
from django.db import models

class Student(models.Model):
    # ...
    pass

class Batch(models.Model):
    # ...
    students = models.ManyToManyField(Student)
```

•

- It doesn't matter which model has the ManyToManyField, but you should only put it in one of the models – not both

- Sometimes you want to add extra fields to the relationship

- In Student Batch example, you may want to store extra information like join date

```python
from django.db import models

class Student(models.Model):
    name = models.CharField(max_length=128)

class Batch(models.Model):
    students = models.ManyToManyField(Student, through='StudentBatch')

class StudentBatch(models.Model):
    student = models.ForeignKey(Student, on_delete=models.CASCADE)
    batch = models.ForeignKey(Batch, on_delete=models.CASCADE)
    date_joined = models.DateField()
```

•

One to One

- To define a one-to-one relationship, use OneToOneField.

- You use it just like any other Field type: by including it as a class attribute of your model.

```python
from django.db import models

class StudentBatch(models.Model):
    ...

class Certificate(models.Model):
    studentBatch = models.OneToOneField(StudentBatch, on_delete=models.CASCADE )
    issue_date=models.DateField()
```

•