class User(models.Model):

    first\_name= models.CharField(max\_length=255)

    last\_name= models.CharField(max\_length=255)

    email= models.CharField(max\_length=255, unique=True)

    password = models.CharField(max\_length=255)

    created\_at= models.DateTimeField(auto\_now\_add=True)

    updated\_at= models.DateTimeField(auto\_now=True)

class Message(models.Model):

    user\_id= models.ForeignKey(User, related\_name="messages", on\_delete = models.CASCADE)

    message= models.TextField()

    created\_at= models.DateTimeField(auto\_now\_add=True)

    updated\_at= models.DateTimeField(auto\_now=True)

class Comment(models.Model):

    user\_id= models.ForeignKey(User, related\_name="comments\_user", on\_delete = models.CASCADE)

    message\_id= models.ForeignKey(Message, related\_name="comments\_mess", on\_delete = models.CASCADE)

    comment= models.TextField()

    created\_at= models.DateTimeField(auto\_now\_add=True)

    updated\_at= models.DateTimeField(auto\_now=True)

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"

        integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm" crossorigin="anonymous">

    <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js"

        integrity="sha384-KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"

        crossorigin="anonymous"></script>

    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"

        integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"

        crossorigin="anonymous"></script>

    <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"

        integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"

        crossorigin="anonymous"></script>

    {% load static %}

    <link rel="stylesheet" href="{% static 'css/style.css' %}">

class Author(models.Model):

name = models.CharField(max\_length=255)

created\_at = models.DateTimeField(auto\_now\_add=True)

updated\_at = models.DateTimeField(auto\_now=True)

class Book(models.Model):

title = models.CharField(max\_length=255)

author = models.ForeignKey(Author, related\_name="books", on\_delete = models.CASCADE)

created\_at = models.DateTimeField(auto\_now\_add=True)

updated\_at = models.DateTimeField(auto\_now=True)

**CREATING**this\_author = Author.objects.get(id=2) # get an instance of an Author

my\_book = Book.objects.create(title="Little Women", author=this\_author) # set the retrieved author as the author of a new book

# or in one line...

my\_book = Book.objects.create(title="Little Women", author=Author.objects.get(id=2))

#### READING

this\_author = Author.objects.get(id=2)

books = Book.objects.filter(author=this\_author)

# one-line version:

books = Book.objects.filter(author=Author.objects.get(id=2))

class Book(models.Model):

title = models.CharField(max\_length=255)

created\_at = models.DateTimeField(auto\_now\_add=True)

updated\_at = models.DateTimeField(auto\_now=True)

class Publisher(models.Model):

name = models.CharField(max\_length=255)

books = models.ManyToManyField(Book, related\_name="publishers")

created\_at = models.DateTimeField(auto\_now\_add=True)

updated\_at = models.DateTimeField(auto\_now=True)

Adding a relationship between two existing records is simple:

this\_book = Book.objects.get(id=4) # retrieve an instance of a book

this\_publisher = Publisher.objects.get(id=2) # retrieve an instance of a publisher

# 2 options that do the same thing:

this\_publisher.books.add(this\_book) # add the book to this publisher's list of books

# OR

this\_book.publishers.add(this\_publisher)

The syntax to see all books from a given publisher is the same as when doing a reverse look-up on a ForeignKey relationship:

this\_publisher.books.all() # get all the books this publisher is publishing

this\_book.publishers.all() # get all the publishers for this book

And to remove a relationship between two existing records:

this\_book = Book.objects.get(id=4) # retrieve an instance of a book

this\_publisher = Publisher.objects.get(id=2) # retrieve an instance of a publisher

# 2 options that do the same thing:

this\_publisher.books.remove(this\_book) # remove the book from this publisher's list of books

# OR

this\_book.publishers.remove(this\_publisher)