from django.db import models  
  
class Author(models.Model):  
 firstname = models.CharField(max\_length=100)  
 lastname = models.CharField(max\_length=100)  
 address = models.CharField(max\_length=200, null=True)  
 zipcode = models.IntegerField(null=True)  
 telephone = models.CharField(max\_length=100, null=True)  
 recommendedby = models.ForeignKey('Author', on\_delete=models.CASCADE, related\_name='recommended\_authors', related\_query\_name='recommended\_authors', null=True)  
 joindate = models.DateField()  
 popularity\_score = models.IntegerField()  
 followers = models.ManyToManyField('User', related\_name='followed\_authors', related\_query\_name='followed\_authors')  
 def \_\_str\_\_(self):  
 return self.firstname + ' ' + self.lastname  
  
class Books(models.Model):  
 title = models.CharField(max\_length=100)  
 genre = models.CharField(max\_length=200)  
 price = models.IntegerField(null=True)  
 published\_date = models.DateField()  
 author = models.ForeignKey('Author', on\_delete=models.CASCADE, related\_name='books', related\_query\_name='books')  
 publisher = models.ForeignKey('Publisher', on\_delete=models.CASCADE, related\_name='books', related\_query\_name='books')  
 def \_\_str\_\_(self):  
 return self.title  
  
class Publisher(models.Model):  
 firstname = models.CharField(max\_length=100)  
 lastname = models.CharField(max\_length=100)  
 recommendedby = models.ForeignKey('Publisher', on\_delete=models.CASCADE, null=True)  
 joindate = models.DateField()  
 popularity\_score = models.IntegerField()  
 def \_\_str\_\_(self):  
 return self.firstname + ' ' + self.lastname  
  
class User(models.Model):  
 username = models.CharField(max\_length=100)  
 email = models.CharField(max\_length=100)  
 def \_\_str\_\_(self):  
 return self.username

**Ejercicios:**

1. Escriba una consulta usando Django ORM para obtener todos los objetos libros de su base de datos.
2. Escriba una consulta usando Django ORM para obtener el título y el published\_date de todos los libros de la base de datos.
3. Obtener el nombre y el apellido de todos los autores nuevos ( Los autores con popularity\_score = 0 son autores nuevos ).
4. Obtenga el nombre y la puntuación de popularidad de todos los autores cuyo nombre comience con A y la puntuación de popularidad sea mayor o igual a 8.
5. Obtener el nombre de todos los autores con un nombre que no distingue entre mayúsculas y minúsculas.
6. Obtener la lista de todos los autores cuyos identificadores están en la lista = [1, 3, 23, 43, 134, 25].
7. Obtener la lista de todos los editores que se unieron después o en septiembre de 2012, la lista de salida solo debe contener el nombre y la fecha de unión del editor. Ordenar por fecha de unión.
8. Obtener la lista ordenada de los 10 primeros apellidos de los editores, la lista no debe contener duplicados.
9. Obtén la fecha de registro del último autor y editor que se unieron.
10. Obtenga el nombre, apellido y fecha de incorporación del último autor que se unió.
11. Obtener la lista de todos los autores que se unieron después o en el año 2013
12. Obtén el precio total de todos los libros escritos por autores con una puntuación de popularidad de 7 o superior.
13. Obtener la lista de títulos de todos los libros escritos por autores cuyo nombre comienza con 'A'. El resultado debe contener una lista de los títulos de cada libro. No es una lista de tuplas.
14. Obtener el precio total de todos los libros escritos por el autor con pk en la lista [1, 3, 4]
15. Elabore una lista de todos los autores junto con su recomendador.
16. Producir lista de todos los autores que publicaron su libro por editorial pk = 1, la lista de salida debe ordenarse por nombre.
17. Cree tres nuevos usuarios y agregue los seguidores del autor con pk = 1.
18. Establezca la lista de seguidores del autor con pk = 2, con un solo usuario.
19. Añade nuevos usuarios en seguidores del autor con pk = 1.
20. Eliminar un usuario de los seguidores del autor con pk = 1.
21. Obtenga los nombres de pila de todos los autores, cuyo usuario con pk = 1 está siguiendo. ( sin acceder al administrador de author.objects )
22. Obtener la lista de todos los autores que escribieron un libro con "tle" como parte del título del libro.
23. Obtenga la lista de autores cuyos nombres comienzan con 'A' sin distinción entre mayúsculas y minúsculas, y su puntuación de popularidad es superior a 5 o se han unido después de 2014. con objetos Q.
24. Recupere un objeto específico con primary key= 1 de la tabla Author.
25. Recupere los primeros N=10 registros de una tabla Author.
26. Recupere registros de una tabla que coincidan con esta condición, puntuación de popularidad = 7. Y obtener el primer y último registro de esa lista.
27. Recupere todos los autores que se unieron después o en el año 2012, la puntuación de popularidad mayor o igual a 4, la fecha de incorporación posterior o con la fecha 12 y el nombre comienza con 'a' (no distingue entre mayúsculas y minúsculas) sin usar objetos Q.
28. Recuperar todos los autores que no se unieron en 2012.
29. Recuperar el autor más antiguo, el autor más reciente, la puntuación media de popularidad de los autores, la suma del precio de todos los libros de la base de datos.
30. Recuperar todos los autores que no tienen ningún recomendador, el campo recomendado por es nulo.
31. Recupere los libros que no tienen ningún autor, donde el autor es nulo. Además, recupere los libros cuyos autores están presentes, pero no tienen un recomendador, donde el autor no es nulo y el recomendador del autor es nulo. (Tenga en cuenta que si no se especifica la condición para que el autor no sea nulo y solo se menciona la condición para que el recomendador sea nulo, se recuperarán todos los libros con el autor nulo y el recomendador del autor nulo).
32. Precio total de los libros escritos por el autor con clave principal = 1. ( Agregación sobre modelo relacionado ), libro más antiguo escrito por el autor con pk = 1, último libro escrito por el autor con pk = 1.
33. Entre las editoriales de la tabla Editoriales, ¿cuál es el libro más antiguo que ha publicado una editorial?
34. Precio medio de todos los libros de la base de datos.
35. Puntuación máxima de popularidad de la editorial entre todas las editoriales que publicaron un libro para el autor con pk = 1. (Salto de clave externa inversa)
36. Cuente el número de autores que han escrito un libro que contiene la frase 'ab' sin distinción entre mayúsculas y minúsculas.
37. Consigue que todos los autores tengan más de 216 seguidores.
38. Obtén la puntuación media de popularidad de todos los autores que se unieron después del 20 de septiembre de 2014.
39. Genere una lista de libros cuyo autor haya escrito más de 10 libros.
40. Obtén la lista de libros con títulos duplicados.

from main.models import \*  
import datetime  
from django.db.models import Count, Avg, Sum, Max, Min  
from django.db.models import Q, F  
ans1 = Books.objects.all()  
ans2 = Books.objects.all().values\_list('title', 'published\_date')  
ans3 = Authors.objects.all().filter(popularity\_score=0).values\_list('firstname', 'lastname')  
ans4 = Authors.objects.all().filter(firstname\_\_startswith='a', popularity\_score\_\_gte=8).values\_list('firstname', 'popularity\_score')  
ans5 = Authors.objects.all().filter(firstname\_\_icontains='aa').values\_list('firstname')  
ans6 = Authors.objects.all().filter(pk\_\_in=[1, 3, 23, 43, 134, 25])  
ans7 = Authors.objects.all().filter(joindate\_\_gte=datetime.date(year=2012, month=9, day=1)).order\_by('joindate').values\_list('firstname', 'joindate')  
ans8 = Publishers.objects.all().order\_by('lastname').values\_list('lastname').distinct()[:10]  
ans9 = [Authors.objects.all().order\_by('joindate').last(),  
Publishers.objects.all().order\_by('-joindate').first()]  
ans10 = Authors.objects.all().order\_by('-joindate').values\_list('firstname', 'lastname', 'joindate').first()  
ans11 = Authors.objects.all().filter(joindate\_\_year\_\_gte=2013)  
ans12 = Books.objects.all().filter(author\_\_popularity\_score\_\_gte=7).aggregate(total\_book\_price=Sum('price'))  
ans13 = Books.objects.all().filter(author\_\_firstname\_\_contains='a').values\_list('title', flat=True)  
ans14 = Books.objects.all().filter(author\_\_pk\_\_in=[1, 3, 4]).aggregate('price')  
ans15 = Authors.objects.all().values\_list('firstname', 'recommendedby\_\_firstname')  
ans16 = Authors.objects.all().filter(books\_\_publisher\_\_pk=1)  
user1 = Users.objects.create(username='user1', email='user1@test.com')  
user2 = Users.objects.create(username='user2', email='user2@test.com')  
user3 = Users.objects.create(username='user3', email='user3@test.com')  
ans17 = Authors.objects.get(pk=1).followers.add(user1, user2, user3)  
ans18 = Authors.objects.get(pk=2).followers.set(user1)  
ans19 = Authors.objects.get(pk=1).followers.add(user1)  
ans20 = Authors.objects.get(pk=1).followers.remove(user1)  
ans21 = Users.objects.get(pk=1).followed\_authors.all().values\_list('firstname', flat=True)  
ans22 = Authors.objects.all().filter(books\_\_title\_\_icontains='tle')  
ans23 = Authors.objects.all().filter(Q(firstname\_\_istartswith='a') and ( Q(popularity\_score\_\_gt=5) or Q(joindate\_\_year\_\_gt=2014)))  
ans24 = Authors.objects.all().get(pk=1)  
ans25 = Authors.objects.all()[:10]  
qs = Authors.objects.all().filter(popularity\_scre=7)  
author1 = qs.first()  
author2 = qs.last()  
ans26 = [author1, author2]  
ans27 = Authors.objects.all().filter(joindate\_\_year\_\_gte=2012, popularity\_score\_\_gte=4, joindate\_\_day\_\_gte=12, firstame\_\_istartswith='a')  
ans28 = Authors.objects.all().exclude(joindate\_\_year=2012)  
oldest\_author = Authors.objects.all().aggregate(Min('joindate'))  
newest\_author = Authors.objects.all().aggregate(Max('joindate'))  
avg\_pop\_score = Authors.objects.all().aggregate(Avg('popularity\_score'))  
sum\_price = Books.objects.all().aggregate(Sum('price'))  
ans29 = [oldest\_author, newest\_author, avg\_pop\_score, sum\_price]  
ans30 = Authors.objects.all().filter(recommendedby\_\_isnull=True)  
one = Books.objects.all().filter(author\_\_isnull=False)  
two = Books.objects.all().filter(author\_\_isnull=False, author\_\_recommender\_\_isnull=True)  
ans31 = [one, two]  
ans32 = Books.objects.all().filter(author\_\_pk=1).aggregate(Sum('price'))  
ans33 = Books.objects.all().order\_by('published\_date').last().title  
ans34 = Books.objects.all().aggregate(Avg('price'))  
ans35 = Publishers.objects.filter(books\_\_author\_\_pk=1).aggregate(Max('popularity\_score'))  
ans36 = Authors.objects.filter(books\_\_title\_\_icontains='ab').count()  
ans37 = Authors.objects.annotate(f\_count=Count('followers')).filter(f\_count\_\_gt=216)  
ans38 = Authors.objects.filter(joindate\_\_gt=datetime.date(year=2014, month=9, day=20)).aggregate(Avg('popularity\_score'))  
ans39 = Books.objects.all().annotate(bk\_count=Count('author\_\_books')).filter(bk\_count\_\_gt=10).distinct()  
ans40 = Books.objects.all().annotate(count\_title=Count('title')).filter(count\_title\_\_gt=1)