1. Придумать 10 разных фильтров или эксклюдов, с разными лукапами и заходом в связанные модели.

>>> from django.contrib.auth.models import User

>>> from myapp.models import Article, Author, Comment, Like

1. Простой фильтр:

>>> lam = Author.objects.filter(name='Lam').first()

1. Фильтр с заходом в связанные модели:

>>> Comment.objects.filter(user=lam, article\_\_author\_\_name='Linda')

<QuerySet [<Comment: Hi Linda! Let's dance!!! by Lam>]>

1. Фильтр с лукап icontains:

>>> Comment.objects.filter(text\_\_icontains='some text for').first()

<Comment: some text for a comment by Lam>

1. Фильтр с лукап year: отобрать все статьи 2019 года.

>>> Article.objects.filter(created\_at\_\_year=2019)

1. Получить 5 последних написанных комментариев (именно текст).

>>> q = Comment.objects.order\_by('-created\_on').all()[:5]

Или

>>> q = Comment.objects. order\_by('-created\_on').values('body')[:5]

1. Создать 5 комментариев с разным текстом, Хотя бы один должен начинаться со слова "Start", хоть один в середине должен иметь слово "Middle", хоть один должен заканчиваться словом "Finish".

**Через .create():**

**Start:**

>>> Comment.objects.create(text='Start comment',

... article=article,

... user=user)

<Comment: Start comment by Lam>

**Middle:**

>>> Comment.objects.create(text='Comment Middle comment.',

... article=article,

... user=user)

<Comment: Comment Middle comment. by Lam>

**Finish:**

>>> Comment.objects.create(text='Comment Finish.',

... article=article,

... user=user)

<Comment: Comment Finish. by Lam>

**Через .save():**

>>> comment4 = Comment(text='some text for a comment',

... article=article,

... user=user)

>>> comment4.save()

>>> comment5 = Comment(text='some new text for a comment 5',

... article=article,

... user=user)

>>> comment5.save()

1. Переписать сейв комментария так, чтобы при создании дата менялась бы на год назад (если сегодня 20 декабря 2019, должна выставляться 20 декабря 2018), изменение комментариев не затрагивать.

**Документация:** <https://docs.djangoproject.com/en/3.0/topics/db/models/#overriding-model-methods>

class Article(models.Model):  
 author = models.ForeignKey(Author, on\_delete=models.CASCADE, null=True, related\_name='articles')  
 text = models.TextField(max\_length=10000, null=True)  
 created\_at = models.DateTimeField(default=timezone.now)  
 updated\_at = models.DateTimeField(default=timezone.now)  
 genre = models.IntegerField(choices=GENRE\_CHOICES, default=1)  
  
 **def save(self, \*args, \*\*kwargs):  
 self.created\_at = timezone.now() - timedelta(years=1)  
 super().save(\*args, \*\*kwargs)**

1. Изменить комментарии со спец словами "Start", "Middle", "Finish".

1) Отобрать все комментарии:

>>> q = Comment.objects.filter(text\_\_icontains='Start') | Comment.objects.filter(text\_\_icontains='Middle') | Comment.objects.f

ilter(text\_\_icontains='Finish')

2) Update эти комментарии:

>>> q.update(text='new text for the comment')

3

1. Удалить все комментарии, у которых в тексте есть буква "a", но не удалять если есть буква "с".

1) Отобрать все комментарии:

>>> q = Comment.objects.filter(text\_\_icontains='a').exclude(text\_\_icontains='c').all()

2) Удалить комментарии:

>>> q.delete()

(3, {'myapp.Comment': 3})

1. Получить первые 2 комментария по дате создания к статье, у которой имя автора последнее по алфавиту.

>>> Article.objects.order\_by('-author\_\_name', '-created\_at')[:2]

1. Практика с annotate():

>>> from django.db.models import Count

>>> q = Author.objects.annotate(comments\_count=Count('comments'))

>>> for author in q:

... print(author.name, author.comments\_count)

...

Lam 4

Nastya 0

Linda 0

Khuyen 0

Dong 0