* 1. Создать двух пользователей (с помощью метода User.objects.create\_user('username')).

>>> user1 = User.objects.create\_user(username='Dima')

>>> user2 = User.objects.create\_user(username='Makar')

>>> user3 = User.objects.create\_user(username='Ivan')

>>> user4 = User.objects.create\_user(username='Roman')

* 1. Создать два объекта модели *Author*, связанные с пользователями.

>>> Author.objects.create(author\_user=user1)

<Author: Author object (1)>

>>> Author.objects.create(author\_user=user2)

<Author: Author object (2)>

* 1. Добавить 4 категории в модель *Category.*

>>> Category.objects.create(name='Sport')

<Category: Category object (1)>

>>> Category.objects.create(name='Politics')

<Category: Category object (2)>

>>> Category.objects.create(name='Education')

<Category: Category object (3)>

>>> Category.objects.create(name='Finance')

<Category: Category object (4)>

* 1. Добавить 2 статьи и 1 новость.

>>> author1 = Author.objects.get(id=1)

>>> author2 = Author.objects.get(id=2)

>>> Post.objects.create(author=author1, category\_type='AR', heading='The impact of the financial sector on sports.', content='Today there will be an article no longer about sport')

<Post: Post object (1)>

>>> Post.objects.create(author=author2, category\_type='NW', heading='Sanctions in sports', content='Athletes from Russia were banned from competing at the Olympics.')

<Post: Post object (3)>

>>> Post.objects.create(author=author2, category\_type='AR', heading='Political literacy', content='Politc it is a ddsdd vfvfgf ekofjefjksf')

<Post: Post object (2)>

* 1. Присвоить им категории (как минимум в одной статье/новости должно быть не меньше 2 категорий).

>>> Post.objects.get(id=1).\_category.add(Category.objects.get(id=1))

>>> Post.objects.get(id=1).\_category.add(Category.objects.get(id=4))

>>> Post.objects.get(id=2).\_category.add(Category.objects.get(id=2))

>>> Post.objects.get(id=2).\_category.add(Category.objects.get(id=3))

>>> Post.objects.get(id=3).\_category.add(Category.objects.get(id=1))

>>> Post.objects.get(id=3).\_category.add(Category.objects.get(id=2))

* 1. Создать как минимум 4 комментария к разным объектам модели *Post* (в каждом объекте должен быть как минимум один комментарий).

>>> Comment.objects.create(comment\_post=Post.objects.get(id=1), comment\_user=User.objects.get(id=1), content\_comment='Some nonsense!')

<Comment: Comment object (1)>

>>> Comment.objects.create(comment\_post=Post.objects.get(id=2), comment\_user=User.objects.get(id=2), content\_comment='I agree with the author')

<Comment: Comment object (2)>

>>> Comment.objects.create(comment\_post=Post.objects.get(id=3), comment\_user=User.objects.get(id=1), content\_comment="I'm not sure that's true!")

<Comment: Comment object (3)>

>>> Comment.objects.create(comment\_post=Post.objects.get(id=3), comment\_user=User.objects.get(id=3), content\_comment='Fake!')

<Comment: Comment object (4)>

* 1. Применяя функции like() и dislike() к статьям/новостям и комментариям, скорректировать рейтинги этих объектов.

>>> Comment.objects.get(id=3).like()

>>> Comment.objects.get(id=3).like()

>>> Comment.objects.get(id=3).like()

>>> Comment.objects.get(id=3).like()

>>> Comment.objects.get(id=3).like()

>>> Comment.objects.get(id=3).like()

>>> Comment.objects.get(id=4).like()

>>> Comment.objects.get(id=3).dislike()

>>> Comment.objects.get(id=3).dislike()

>>> Comment.objects.get(id=1).dislike()

>>> Comment.objects.get(id=4).dislike()

>>> Comment.objects.get(id=4).dislike()

>>> Post.objects.get(id=1).like()

>>> Post.objects.get(id=1).like()

>>> Post.objects.get(id=1).like()

>>> Post.objects.get(id=1).like()

>>> Post.objects.get(id=1).like()

>>> Post.objects.get(id=1).like()

>>> Post.objects.get(id=2).like()

>>> Post.objects.get(id=2).like()

>>> Post.objects.get(id=2).like()

>>> Post.objects.get(id=3).dislike()

>>> Post.objects.get(id=3).dislike()

>>> Post.objects.get(id=3).dislike()

>>> Post.objects.get(id=3).dislike()

* 1. Обновить рейтинги пользователей.

>>> author1 = Author.objects.get(id=1)

>>> author2 = Author.objects.get(id=2)

>>> author1.update\_rating()

>>> author2.update\_rating()

* 1. Вывести username и рейтинг лучшего пользователя (применяя сортировку и возвращая поля первого объекта).

>>> a = Author.objects.order\_by('-ratting\_author')[:1]

>>> for i in a:

... i.author\_user.username

... i.ratting\_author

* 1. Вывести дату добавления, username автора, рейтинг, заголовок и превью лучшей статьи, основываясь на лайках/дислайках к этой статье.

>>> post = Post.objects.order\_by('-ratting\_post')[:1]

>>> for i in post:

... i.date\_time\_create.date()

... i.author.author\_user.username

... i.ratting\_post

... i.heading

... i.preview()

* 1. Вывести все комментарии (дата, пользователь, рейтинг, текст) к этой статье.

>>> Comment.objects.filter(comment\_post = post).values('date\_time\_create', 'comment\_user', 'ratting\_comment', 'content\_comment')