django-ormex Documentation

Release 0.2.1

Artur Barseghyan <artur.barseghyan@gmail.com>

Jan 22, 2020

Contents

1 Prerequisites 3

2 Supported databases 5

3 Installation 7

4 Usage 9

4.1 Aggregations 9

5 Demo 13

5.1 Run demo locally 13

6 Testing 15

7 License 17

8 Support 19

9 Author 21

Django ORM extensions.

CHAPTER 1

Prerequisites

- Django 1.11, 2.2 and 3.0.
- Python 2.7, 3.5, 3.6, 3.7 and 3.8

CHAPTER 2

Supported databases

PostgreSQL, MySQL, SQLite

CHAPTER 3

Installation

Install in your virtual environment.

Latest stable version from PyPI:

```
pip install django-ormex
```

Latest stable version from GitHub:

```
pip install https://github.com/barseghyanartur/django-ormex/archive/stable.tar.gz
```

CHAPTER 4

Usage

Common usage examples.

4.1 Aggregations

Contains various modules for aggregations.

4.1.1 GroupConcat

Works like Concat, but for concatenating field values of related ManyToMany model. For instance, you may use it if you have an `Author` model as ManyToMany relation in the `Book` model (`Book.authors = ManyToManyField(Author)`) and you want to have concatenated list of all authors coupled to a given book.

Given the following models:

```
class Publisher(models.Model):
    """Publisher."""

    name = models.CharField(max_length=255)
    address = models.CharField(max_length=255)
    city = models.CharField(max_length=255)
    state_province = models.CharField(max_length=255)
    country = models.CharField(max_length=255)
    website = models.URLField(max_length=255)


class Author(models.Model):
    """Author."""

    salutation = models.CharField(max_length=255)
    name = models.CharField(max_length=255)
    email = models.EmailField(max_length=255)
```

(continues on next page)

(continued from previous page)

```
    headshot = models.ImageField(upload_to='authors', null=True, blank=True)


class Book(models.Model):
    """Book."""

    title = models.CharField(max_length=255)
    authors = models.ManyToManyField('books.Author', related_name='books')
    publisher = models.ForeignKey(Publisher, related_name='books')
    publication_date = models.DateField()
    isbn = models.CharField(max_length=255, unique=True)
    price = models.DecimalField(max_digits=10, decimal_places=2)
    pages = models.PositiveIntegerField(default=200)
    stock_count = models.PositiveIntegerField(default=30)
```

We could use GroupConcat as follows:

```
from ormex.aggregations import GroupConcat

book = Book.objects.all() \
        .values(
            'id',
            'title',
            'pages',
            'price',
            'publisher__id',
            'publisher__name'
        ).annotate(
            authors__name=GroupConcat('authors__name', separator=', ')
        ).first()
```

Output would look as follows:

```
{
    'authors__name': 'Finn Janssen, Dan Dijkman, Merel Wolf, Evy de Jong',
    'id': 14,
    'pages': 83,
    'price': Decimal('62.13'),
    'publisher__id': 19,
    'publisher__name': 'Rijn, de Bruyn and Verharen',
    'title': 'Laboriosam officia temporibus facere omnis odit.'
}
```

`GroupConcat` accepts an optional argument `order_by` which can be used for tuning the sorting order of the resulted list of strings. In case if `self` is given as value, sorted by the same field. In order to sort the list of authors by name from the example above, do:

```
book = Book.objects.all() \
        .values(
            'id',
            'title',
            'pages',
            'price',
            'publisher__id',
            'publisher__name'
        ).annotate(
```

(continues on next page)

(continued from previous page)

```
            authors__name=GroupConcat(
                'authors__name',
                separator=', ',
                order_by='self'
            )
        ).first()
```

Output would look as follows:

```
{
    'authors__name': 'Dan Dijkman, Evy de Jong, Finn Janssen, Merel Wolf',
    'id': 14,
    'pages': 83,
    'price': Decimal('62.13'),
    'publisher__id': 19,
    'publisher__name': 'Rijn, de Bruyn and Verharen',
    'title': 'Laboriosam officia temporibus facere omnis odit.'
}
```

CHAPTER 5

Demo

5.1 Run demo locally

In order to be able to quickly evaluate the *django-ormex*, a demo app (with a quick installer) has been created (works on Ubuntu/Debian, may work on other Linux systems as well, although not guaranteed). Follow the instructions below to have the demo running within a minute.

Grab the latest `ormex_demo_installer.sh`:

```
wget -O - https://raw.github.com/barseghyanartur/django-ormex/stable/examples/ormex_
↪demo_installer.sh | bash
```

Open your browser and test the app.

- URL: http://127.0.0.1:8001/

If quick installer doesn't work for you, see the manual steps on running the example project.

CHAPTER 6

Testing

Simply type:

```
./runtests.py
```

or use tox:

```
tox
```

or use tox to check specific env:

```
tox -e py38
```

or run Django tests:

```
./manage.py test ormex --settings=settings.testing
```

CHAPTER 7

License

GPL-2.0-only OR LGPL-2.1-or-later

CHAPTER 8

Support

For any issues contact me at the e-mail given in the *Author* section.

CHAPTER 9

Author

Artur Barseghyan <artur.barseghyan@gmail.com>