partitions Documentation

Release 0.1

Eldarion

January 03, 2015

Contents

1 Development **3**
1.1 Contents 3

an app for partitions querysets based on indexed Q expressions in settings

CHAPTER 1

Development

The source repository can be found at https://github.com/eldarion/django-partitions

1.1 Contents

1.1.1 ChangeLog

0.1

- initial release

1.1.2 Installation

- To install django-partitions:

```
pip install django-partitions
```

- Add `'partitions'` to your `INSTALLED_APPS` setting:

```
INSTALLED_APPS = (
    # other apps
    "partitions",
)
```

1.1.3 Template Tags

partition

This will add additional filtering to the query given a string used to lookup the registery Q expression for the key and model (same model as queryset):

```
{% partition queryset using request.get_host %}
```

1.1.4 Usage

Using `django-partitions` starts with registering the models that you want to apply partitioning, or filtering, to. Somewhere that will get executed when the server startups up (like urls.py or a models.py) should import and use the register function:

```
from django.db.models import Q

from partitions.registry import register


register("partner-a.mycompany.com", "products.Product", Q(partner__slug="partner-a"))
register("partner-a.mycompany.com", "blog.Post", Q(tags__name__in=["partner", "widget"]))
register("partner-b.mycompany.com", "products.Product", Q(partner__slug="partner-b"))
register("partner-b.mycompany.com", "blog.Post", Q(tags__name__in=["partner", "foo"]))
```

Then to consume this data we can either use a template tag (ideal for external apps that are not under your control) or call the `chop` function with the same parameters in your `views.py`.

First the templatetag:

```
{% load partitions_tags %}
...
{% partition posts using request.get_host %}
```

If you control the app, it makes the templates a bit cleaner to use the util function directly:

```
from products.models import Product

from partitions.utils import chop


def product_list(request):
    products = Product.objects.all()
    products = chop(products, by=request.get_host())
    ...
```