Natural Key Cache Documentation

Release 0.1

Timothy Messier

May 16, 2016

Contents

Natural Key Cache (NKC) is a custom Django Manager for easily caching objects by their natural keys. If you have code that often calls `Model.objects.get`, this will halp make those lookups faster. However if you often call `Models.objects.filter`, it will not help. In the latter case use a more robust caching Managers like Django Cache Manager.

The NKC aims to be more efficient by sacrificing the features and robustness of other cache Managers. In order to cache any and all database lookups for a model, other cache Managers incur additional overhead. While still being faster than a database query, this overhead causes lookups to be slower than a simple `cache.get`. By only preforming single object lookups based on natural keys, the NKC can avoid much of this overhead.

Simple Example

To start using the NKC, define a new manager on a model.

```python
from django.db import models
from natural_key_cache.cache_manager import NaturalKeyCacheManager

class MyModel(models.Model):
    # Define natural keys, this can be ommited if id/pk is the natural key
    natural_keys = (
        'nat_key',
    )
    cache = NaturalKeyCacheManager(natural_keys)
    nat_key = models.CharField(max_length=32, unique=True)
```

Then access the cached model via the new manager.

```python
MyModel.cache.get(nat_key='unique_instance')
```

CHAPTER 2

Indices and tables

- genindex
- modindex
- search