django-pgcrypto Documentation

Release 1.4.0

Dan Watson

Sep 27, 2017

Contents

CHAPTER 1

Quickstart

There are several encrypted versions of Django fields that you can use (mostly) as you would use a normal Django field:

```
from django.db import models
import pgcrypto

class Employee (models.Model):
    name = models.CharField(max_length=100)
    ssn = pgcrypto.EncryptedTextField()
    pay_rate = pgcrypto.EncryptedDecimalField()
    date_hired = pgcrypto.EncryptedDateField(cipher='Blowfish', key='datekey')
```

If not specified when creating the field (as in the `date_hired` field above), fields are encrypted according to the following settings:

`PGCRYPTO_VALID_CIPHERS` (default: `('AES', 'Blowfish')`): A list of valid PyCrypto cipher names. Currently only AES and Blowfish are supported, so this setting is mostly for future-proofing.

`PGCRYPTO_DEFAULT_CIPHER` (default: `'AES'`): The PyCrypto cipher to use when encrypting fields.

`PGCRYPTO_DEFAULT_KEY` (default: `''`): The default key to use for encryption.

CHAPTER 2

Querying

With Django 1.7, it is possible to filter on encrypted fields as you would normal fields via `exact`, `gt`, `gte`, `lt`, and `lte` lookups. For example, querying the model above is possible like so:

```
Employee.objects.filter(date_hired__gt='1981-01-01', salary__lt=60000)
```

CHAPTER 3

Indices and tables

- genindex
- modindex
- search