django-sabridge Documentation

Release 0.0.1

John Paulett

Nov 09, 2017

Contents

CHAPTER 1

Motivation

CHAPTER 2

Usage

To demonstrate sabridge, we will access `django.contrib.auth.models.User` through SQLAlchemy.

Import and initialize the *`sabridge.Bridge`*:

```
>>> from sabridge import Bridge
>>> bridge = Bridge()
```

We use the model's class to obtain the SQLAlchemy version of the table:

```
>>> from django.contrib.auth.models import User
>>> table = bridge[model]
```

The *`sabridge.Bridge`* returns an instance of `sqlalchemy.schema.Table`. If we write data in Django, we can then view that data via SQLAlchemy:

```
>>> User.objects.create(username='alice')
>>> result = list(table.select().execute())
>>> len(result)
1
>>> result[0][table.c.username]
u'alice'
```

CHAPTER 3

Caveats

3.1 Transactions

sabridge does not re-use Django's connection to the database, thus if executing in a transaction, any data modified by either Django or SQLAlchemy will not be visible to the other, until the transaction is committed.

Practically, this means that any test cases that uses both Django and SQLAlchemy will have to inherit from `django.test.TransactionTestCase` instead of the more typical `django.test.TestCase`. The TransactionTestCase does not wrap each test in a transaction, thus the data modified by SQLAlchemy and Django is not isolated. Unfortunately, the TransactionTestCase is significantly slower than the normal TestCase. Refer to the TransactionTestCase documentation.

3.2 Performance

sabridge uses SQLAlchemy's reflection (`autoload=True`) to discover the schema of the requested Django model.

CHAPTER 4

Contents

4.1 sabridge API

class `sabridge.`**`Bridge`**

`__getitem__`(*model_cls*)

Returns the `sqlalchemy.schema.Table` representation of `model_cls`, a `django.db.models.Model` subclass.

Use dict-notation to obtain the `Table`:

```
>>> from myapp.models import mymodel
>>> brige = Bridge()
>>> mytable = bridge[mymodel]
>>> print type(mytable)
<class 'sqlalchemy.schema.Table'>
```

`Bridge` stores the `Table` for the lifetime of the `Bridge`, thus table reflection only occurs once per model for the `Bridge`.

`connection_url`()

Build a URL for `sqlalchemy.create_engine()` based upon the database defined by `django.db.connection`

`meta`

`sqlalchemy.schema.MetaData` instance bound to the current Django database connection.

4.2 License

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of django-sabridge nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

Index

Symbols

B

C

M