pghstore Documentation

Release 2.0.2

Hong Minhee

Oct 12, 2021

Contents

pghstore provides PostgreSQL hstore formatting for Python.

This small module implements a formatter and a loader for hstore, one of PostgreSQL supplied modules, that stores simple key-value pairs.

```
>>> simple_dictionary = {u'a': u'1'}
>>> dumps(simple_dictionary) == b'"a"=>"1"'
True
>>> loads('"a"=>"1"') == simple_dictionary
True
>>> src = [('pgsql', 'mysql'), ('python', 'php'), ('gevent', 'nodejs')]
>>> loads(dumps(src), return_type=list) == src
True
```

You can easily install the package from PyPI by using **pip** or **easy_install**:

```
$ pip install pghstore
```

pghstore.**dump**(*obj*, *file*, *key_map=None*, *value_map=None*, *encoding='utf-8'*)

Write the object to the specified file in HStore format.

```
>>> import io
>>> f = io.BytesIO()
>>> dump({u'a': u'1'}, f)
>>> f.getvalue() == b'"a"=>"1"'
True
```

Parameters

- **obj** – a mapping object to dump
- **file** – a file object to write into
- **key_map** – an optional mapping function that takes a non-string key and returns a mapped string key
- **value_map** – an optional mapping function that takes a non-string value and returns a mapped string value
- **encoding** – a string encode to use

pghstore.**dumps**(*obj*, *key_map=None*, *value_map=None*, *encoding='utf-8'*, *return_unicode=False*)

Convert a mapping object as PostgreSQL `hstore` format.

```
>>> dumps({u'a': u'1 "quotes"'}) == b'"a"=>"1 \\"quotes\\""'
True
>>> dumps([('key', 'value'), ('k', 'v')]) == b'"key"=>"value","k"=>"v"'
True
```

It accepts only strings as keys and values except `None` for values. Otherwise it will raise `TypeError`:

```
>>> dumps({'null': None}) == b'"null"=>NULL'
True
>>> dumps([('a', 1), ('b', 2)])
Traceback (most recent call last):
  ...
TypeError: value 1 of key 'a' is not a string
```

Or you can pass `key_map` and `value_map` parameters to workaround this:

```
>>> dumps([('a', 1), ('b', 2)], value_map=str) == b'"a"=>"1","b"=>"2"'
True
```

By applying these options, you can store any other Python objects than strings into `hstore` values:

```
>>> import json
>>> dumps([('a', list(range(3))), ('b', 2)], value_map=json.dumps) == b'"a"=>"[0,␣
↪1, 2]","b"=>"2"'
True
>>> import pickle
>>> result = dumps([('a', list(range(3))), ('b', 2)],
...        value_map=pickle.dumps)

...'"a"=>"...","b"=>"..."'
```

It returns a UTF-8 encoded string, but you can change the `encoding`:

```
>>> dumps({u'surname': u'\ud64d'}) == b'"surname"=>"\xed\x99\x8d"'
True
>>> dumps({u'surname': u'\ud64d'}, encoding='utf-32') == b'
↪"\xff\xfe\x00\x00s\x00\x00\x00u\x00\x00\x00r\x00\x00\x00n\x00\x00\x00a\x00\x00\x00m\x00\x00\x0
↪"=>"\xff\xfe\x00\x00M\xd6\x00\x00"'
True
```

If you set `return_unicode` to `True`, it will return `six.text_type` instead of `str` (byte string):

```
>>> dumps({'surname': u'\ud64d'}, return_unicode=True) == u'"surname"=>"\ud64d"'
True
```

Parameters

- **obj** – a mapping object to dump
- **key_map** – an optional mapping function that takes a non-string key and returns a mapped string key
- **value_map** – an optional mapping function that takes a non-string value and returns a mapped string value
- **encoding** – a string encode to use
- **return_unicode** (`bool`) – returns an `six.text_type` string instead byte `str`. `False` by default

Returns a `hstore` data

Return type `six.string_types`

pghstore.**load**(*file*, *encoding='utf-8'*)

Load the contents of file in HStore into a Python object.

pghstore.**loads**(*string*, *encoding='utf-8'*, *return_type=<class 'dict'>*)

Parse the passed hstore format `string` to a Python mapping object.

```
>>> loads('"a"=>"1"') == {u'a': u'1'}
True
```

If you want to load a hstore value as any other type than `dict` set `return_type` parameter. Note that the constructor has to take an iterable object.

```
>>> loads('"a"=>"1", "b"=>"2"', return_type=list) == [(u'a', u'1'), (u'b', u'2')]
True
>>> loads('"return_type"=>"tuple"', return_type=tuple) == ((u'return_type', u
↪'tuple'),)
True
```

Parameters

- **string** (six.string_types) – a hstore format string
- **encoding** – an encoding of the passed string. if the string is an six.text_type string, this parameter will be ignored
- **return_type** – a map type of return value. default is dict

Returns a parsed map. its type is decided by return_type parameter

Version data for the pghstore module.

You can find the current version in the command line interface:

```
$ python -m pghstore.version
2.0.2
```

New in version 1.0.0.

pghstore.version.**VERSION = '2.0.2'**
(six.string_types) The version string e.g. '0.9.2'.

pghstore.version.**VERSION_INFO = (2, 0, 2)**
(tuple) The version tuple e.g. (0, 9, 2).

This module is initially written by Hong Minhee (pghstore._native — original Python version) and Robert Kajic (pghstore._speedups — optimized C version).

Distributed under MIT License.

You can find the source code from the GitHub repository:

```
$ git clone git://github.com/heroku/pghstore.git
```

To report bugs or request features use the issue tracker.

Python Module Index

p

Index