

Tokyo {Cabinet, [Py]Tyrant}

Portland Python Users Group

Author: Michael Schurter <michael@susens-schurter.com>

Date: March 10th, 2009

Is there a Godzilla joke in here somewhere?

- Tokyo Cabinet is a key/value database meant to be the successor of {G,Q}DBM.
- Tokyo Tyrant is the remote service wrapper for Tokyo Cabinet.
- PyTyrant is a Python client for Tokyo Tyrant.

Tokyo Cabinet supports 4 types of databases:

- Hash table (`tch`) - like a Python dictionary
- B+ tree (`tcb`) - non-unique keys, comparison & cursor functionality
- Fixed-length array (`tcf`) - each key is a unique natural number, each value is of fixed length
- Table (`tct`) - variant of Hash table where volumes can have named columns

There are in-memory versions of the Hash table and B+ tree databases as well.

Installing Tokyo Cabinet

Packages are for sissies, lets build some source:

```
$ sudo aptitude install libbz2-dev zlib1g-dev # YMMV
$ wget http://tokyocabinet.sourceforge.net/tokyocabinet-1.4.10.tar.gz
$ tar xzf tokyocabinet-1.4.10.tar.gz
$ cd tokyocabinet-1.4.10/
$ ./configure && make && make install
$ cd ~/tokyo # virtualenv
$ tchmgr create test.tch # Create a new hash database
$ tchmgr put test.tch foo bar
$ tchmgr get test.tch foo
bar
```

pytc - Tokyo Cabinet Python Bindings

Lets quick test Tokyo Cabinet with `pytc`, the Python bindings for TC.

<http://pypi.python.org/pypi/pytc/>

```
(tokyo):~/tokyo$ easy_install pytc
```

```
>>> import pytc
>>> db = pytc.BDB()
>>> db.open('test.db', pytc.BDBOWRITER | pytc.BDBOREADER | pytc.BDBOCREAT)
>>> db.put('foo', 'bar')
>>> db.get('foo')
'bar'
```

pytc Hash table Performance

```
import pytc
db = pytc.HDB()
db.open('test.tch', pytc.BDBOWRITER | pytc.BDBOREADER | pytc.BDBOCREAT)
for i in range(256):
    v = chr(i)
    for x in range(256):
        db.put(chr(x), v)
        db.get(chr(x))
```

```
$ time python test.py
```

```
real    0m0.168s
user    0m0.157s
sys     0m0.010s
```

pytc B+ Tree Performance

```
import pytc
db = pytc.BDB()
db.open('test.db', pytc.BDBOWRITER | pytc.BDBOREADER | pytc.BDBOCREAT)
for i in range(256):
    v = chr(i)
    for x in range(256):
        db.put(chr(x), v)
        db.get(chr(x))
```

```
$ time python test.py
```

```
real    0m0.169s
user    0m0.157s
sys     0m0.011s
```

Same thing with bsddb

```
import bsddb
db = bsddb.btopen('test2.db')

for i in range(256):
    v = chr(i)
    for x in range(256):
        db[chr(x)] = v
        db[chr(x)]
```

```
$ time python test2.py
```

```
real    0m1.382s
user    0m1.370s
sys     0m0.006s
```

Network support with Tokyo Tyrant

Tokyo Tyrant is a daemon which acts as a database server. Its much more than just a network TC RPC though:

- High concurrency (multi-threaded and uses epoll/kqueue)
- 3 Protocol Options: Binary, memcached, and HTTP
- Hot backup
- Update logging
- Replication (master/slave, master/master)
- Lua Extensions

Installing Tokyo Tyrant

```
# Lua 5.0 did *not* work for me
$ sudo aptitude install liblua5.1-0 liblua5.1-0-dev lua5.1 # YMMV
$ wget http://tokyocabinet.sourceforge.net/tyrantpkg/tokyotyrant-1.1.11.tar.gz
$ tar xzf tokyotyrant-1.1.11.tar.gz; cd tokyotyrant-1.1.11
$ ./configure --enable-lua && make && sudo make install # Note --enable-lua
$ cd ~/tokyo
$ ttserver test.tch # Suffix must be a type of TC database
----- logging started [17486] -----
server configuration: host=(any) port=1978
database configuration: name=test.tch
service started: 17486
timer thread 1 started
worker thread 1 started
# etc
worker thread 8 started
listening started
```

PyTyrant: Pure Python Tokyo Tyrant Client

Two annoying little caveats:

- PyTyrant SVN Trunk is the best to use (added TCP_NODELAY)
- Even trunk only supports Tokyo Tyrant 1.1.11 (.17 is latest)

```
(tokyo):~/tokyo$ svn export http://pytyrant.googlecode.com/svn/trunk/ pytyrant
(tokyo):~/tokyo$ cd pytyrant && python setup.py install
```

Previous example code with PyTyrant

```
import pytyrant

t = pytyrant.PyTyrant.open('127.0.0.1', 1978)

for i in range(256):
    v = chr(i)
    for x in range(256):
        t[chr(x)] = v
        t[chr(x)]
```

```
$ time python pyt-test.py
```

```
real    0m11.151s  
user    0m1.317s  
sys     0m1.653s
```

Resources

- Tokyo Cabinet Specifications <http://tokyocabinet.sourceforge.net/spex-en.html>
- Tokyo Tyrant Documentation <http://tokyocabinet.sourceforge.net/tyrantdoc/>
- PyTyrant <http://code.google.com/p/pytyrant/>

Related Resources:

- Lightcloud - fancy distributed key-value database built on Tokyo Tyrant
<http://opensource.plurk.com/LightCloud/>