redis-py-examples Documentation

Release 0.1.2

Vladimir Botka

Sep 17, 2019

Contents:

1 Introduction **1**

2 Examples **3**

2.1 Create frequency graph from a log 3

2.2 List 10 most used words in a text 4

3 Indices and tables **7**

CHAPTER 1

Introduction

Redis is popular in-memory data structure store, used as a database, cache and message broker. redis-py is recommended Python client to Redis.

Intention of this documentation is to create examples how to use redis-py.

CHAPTER 2

Examples

2.1 Create frequency graph from a log

Task: Read /var/log/dpkg.log and create a graph to visualize how often packages are installed, upgraded and removed.

Solution: The loop (30) calls function *read_log* which reads the log line by line (13), splits the fields (14) and concatenate date *l[0]* and time *l[1]* in minutes (15). Third field of the log *l[2]* is status of the dpkg operation(install, upgrade, remove ...). Method *zincrby* (16) increments by 1 the score of *word* in the key *l[2]*. As a result the database contains keys(install, upgrade, remove ...) and associated lists of *words* sorted by score. Next loop (33) calls the function *write_csv* with all keys. As a result *status.csv* files are created in the current directory with the *(date;score)* pairs.

[create-graph-01.py]

```
#!/usr/bin/python3
# Tested with python 3.6.3, python-redis 2.10.5 and redis 4.0.1

import redis

LOG_FILES = ['/var/log/dpkg.log', ]
LOG_SEPARATOR = ' '
CSV_SEPARATOR = ';'

def read_log(log_file):
    ''' This function reads log_file and put the status into the database '''
    f = open(log_file, 'r')
    for line in f:
        l = line.split(LOG_SEPARATOR)
        word = l[0] + ' ' + l[1][:-3]
        r.zincrby(l[2], word, 1)
    f.close()

def write_csv(status):
    ''' This function reads the database and writes the status CSV file '''
```

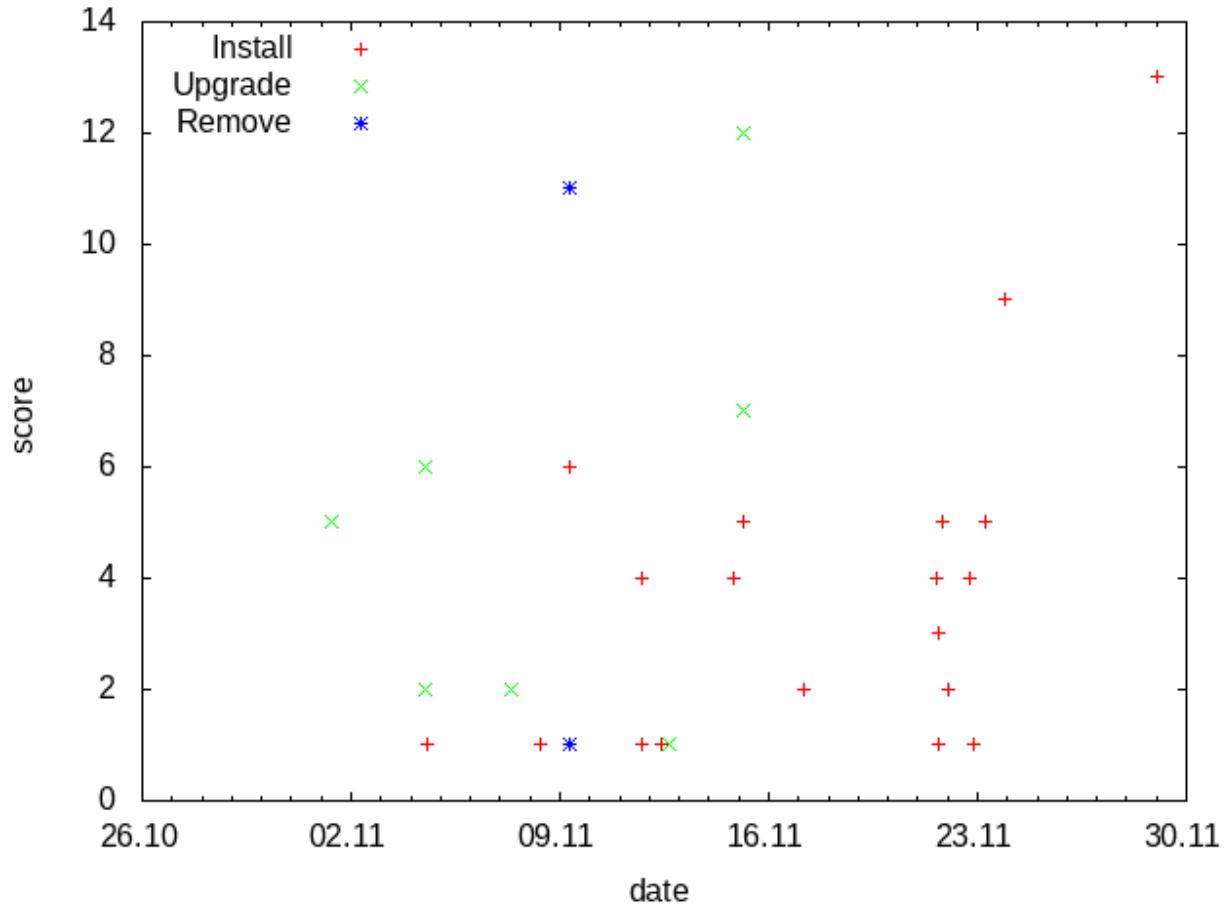
(continues on next page)

(continued from previous page)

```
    f = open(status.decode() + '.csv', 'w')
    l = r.zrange(status, 0, -1, 'DESC', 'WITHSCORES')
    for x in l:
        f.write(x[0].decode() + CSV_SEPARATOR + str(int(x[1])) + '\n')
    f.close()

r = redis.StrictRedis(host='localhost', port=6379, db=0)
r.flushdb()

for log_file in LOG_FILES:
    read_log(log_file)

for status in r.keys():
    write_csv(status)
```

Result: The *status.csv* files can be used to create a graph with *gnuplot*.

[create-graph-01.gnuplot]



2.2 List 10 most used words in a text

Task: Read text from a file and list 10 most frequently used words in it.

Solution: Let's use article about Redis at wikipedia.org as a text.

[create-topchart-text.bash]

```
#!/bin/bash
lynx -dump -nolist https://en.wikipedia.org/wiki/Redis > redis.txt
```

To tokenize words from the text we use NLTK. NLTK data must be installed by *nltk.download()* (8) before *word_tokenize* (18) and *wordnet.synsets* (21) can be used. Complete NLTK data is over 3GB, hence the download (8) is commented. *zincrby* (22) increments by 1 the score of *word* in the key *topchart* and *zrange* (24) returns top 10 words with scores.

[create-topchart.py]

```
1  #!/usr/bin/python3
2  # Tested with python 3.6.3, python-redis 2.10.5 and redis 4.0.1
3
4  import redis
5  import nltk
6  from nltk.corpus import wordnet
7  from nltk.tokenize import word_tokenize
8  # nltk.download()
9
10 file = 'redis.txt'
11
12 r = redis.StrictRedis(host='localhost', port=6379, db=0)
13 r.flushdb()
14
15 f = open(file, 'r')
16 text = f.read()
17 f.close()
18 words = word_tokenize(text)
19
20 for word in words:
21     if wordnet.synsets(word):
22         r.zincrby('topchart', word, 1)
23
24 ranking = r.zrange('topchart', 0, 10, 'DESC', 'WITHSCORES')
25 for x in ranking:
26     print ("%d\t%s" % (x[1], x[0].decode()))
```

Result:

```
> ./create-topchart.py
24   is
23   a
19   in
13   edit
11   Retrieved
10   by
9    database
9    Labs
8    are
7    on
7    data
```

CHAPTER 3

Indices and tables

- genindex
- modindex
- search