Redis

AP hogeschool

Philippe Possemiers

ARTESIS PLANTIJN
HOGESCHOOL ANTWERPEN

v 1.2

What is Redis?

- Open-source, BSD license
- Entirely in-memory, blazing fast
- Persistence on disk
- Replication to any number of nodes
- Key-value store aka data structure server
- Keys can contain strings, hashes, lists, sets and sorted sets
- Written in C

What is Redis used for?

- Caching
- Messaging queues (pub sub)
- Session management
- Hit counts
- Big data aggregation
- …

ARTESIS PLANTIJN
HOGESCHOOL ANTWERPEN

Installing Redis

- MacOSX : 'brew install redis'
- Linux : 'sudo apt-get install redis-server'
- Windows : https://github.com/MicrosoftArchive/redis/releases
- Docker image
- https://redislabs.com/redis-enterprise/cloud/

Running Redis

- Four executables :
  - redis-server : the server itself
  - redis-cli : command line interface utility
  - redis-sentinel : monitoring and failover
  - redis-benchmark : check performance
  - redis-check-aof and redis-check-dump : useful if data is corrupt

Redis configuration

- redis.conf in root dir of Redis
- CONFIG command : CONFIG GET * and CONFIG SET key value
- For securing Redis : CONFIG SET requirepass password
- After this, AUTH password is required

Redis data types

- String :
  - SET key value
  - GET key
- Hashes : collection of key/value pairs.
  - HMSET name key value key value …
  - HGETALL name
- Lists : list of strings
  - LPUSH name value, RPUSH name value
  - LRANGE name value

Redis data types

- Sets : unordered collection of <u>unique</u> strings
  - SADD name value
  - SMEMBERS name
- Sorted sets : similar to sets, but with a score
  - ZADD name score value
  - ZRANGEBYSCORE name score1 score2

Redis cli

- Command line interface
- connect : redis-cli host -p port -a password

Redis keys

- SET key value
- GET key
- DEL key : delete
- DUMP key : serialized dump
- EXISTS key
- EXPIRE key seconds : expiration of key, very handy for caching purposes
- PERSIST key : remove expiration

ARTESIS PLANTIJN
HOGESCHOOL ANTWERPEN

Redis keys

- TTL key : time to live
- KEYS pattern : find all keys with pattern
- RANDOMKEY : get random key
- RENAME key newkey
- TYPE key : data type of key
- INCR key : unique counter
- FLUSHALL : clear all

Redis Bitmaps

- From Redis 2.6 on, bitmaps can be used to perform operations on arrays of bits
- Commands :
  - SETBIT key offset value
  - GETBIT key offset
  - BITOP : AND, OR, XOR and NOT
  - BITCOUNT key
- Very cool example : https://tech.bellycard.com/blog/light-speed-analytics-with-redis-bitmaps-and-lua/
- http://blog.getspool.com/2011/11/29/fast-easy-realtime-metrics-using-redis-bitmaps/

Redis commands

- https://redis.io/commands

Redis HyperLogLog

- How to count distinct number of elements in a set?
- Very memory-intensive, proportional to the cardinality
- Possible solution : approximation (standard error of about 0,81% in Redis)
- Basic algorithm calculates maximum number of leading zeroes (n) in the binary representation of each number in the set

ARTESIS PLANTIJN
HOGESCHOOL ANTWERPEN

Redis HyperLogLog

- The cardinality can then be estimated by two to the power of n
- Actually, a hash is also applied to the set and the set is split into several subsets to increase accuracy
- PFADD name elements
- PFCOUNT name

ARTESIS PLANTIJN
HOGESCHOOL ANTWERPEN

Redis PUB / SUB

- Messaging system with publishers and subscribers
- Channel links them together
- Commands :
  - (P)SUBSCRIBE channel
  - (P)UNSUBSCRIBE channel
  - PUBLISH channel message

Redis Transactions

- Atomic execution
- Commands :
  - MULTI
  - EXEC
  - DISCARD

Redis Scripting

- Lua interpreter
- Commands :
  - EVAL script
  - SCRIPT FLUSH
  - SCRIPT KILL
  - SCRIPT LOAD script

Redis and Django

- pip install redis (pip is the python package manager)
- https://pypi.org/project/redis/

```
import redis

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

Redis and Spring

- Pom.xml :

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-redis</artifactId>
</dependency>
```

- In your service :

```
@Autowired
private RedisTemplate<key, value> template;
```

- The template is the central class to interact with Redis

ARTESIS PLANTIJN
HOGESCHOOL ANTWERPEN

Redis and Spring

- Operations :

```
// ValueOperations, BoundValueOperations
template.opsForValue().set(key, value);
template.boundValueOps(key).set(value);

// HashOperations, BoundHashOperations
template.opsForHash().put(key, "hashKey", value);
template.boundHashOps(key).put("hashKey", value);

// ListOperations, BoundListOperations
template.opsForList().leftPush(key, value);
template.opsForList().rightPush(key, value);
template.opsForList().rightPop(key, 1, TimeUnit.SECONDS);
template.opsForList().leftPop(key, 1, TimeUnit.SECONDS);
template.boundListOps(key).leftPush(value);
template.boundListOps(key).rightPush(value);
template.boundListOps(key).rightPop(1, TimeUnit.SECONDS);
template.boundListOps(key).leftPop(1, TimeUnit.SECONDS);

// ZSetOperations, BoundZSetOperations
template.opsForZSet().add(key, "player1", 12.0d);
template.opsForZSet().add(key, "player2", 11.0d);
template.boundZSetOps(key).add("player1", 12.0d);
template.boundZSetOps(key).add("player2", 11.0d);
```

ARTESIS PLANTIJN
HOGESCHOOL ANTWERPEN