
redis Cluster

Repositories 49 Projects Packages Stars 678

Grokzen

Grokzen

Edit profile

113 followers · **21** following

Dynamist (https://dynamist.se/)

Sweden

Achievements

x2 x3 x3

Beta Send feedback

Organizations

Grokzen / README.md

Hi there 👋

Github Statistics

Total Stars Earned:	2.8k
Total Commits (2023):	105
Total PRs:	77
Total Issues:	190
Contributed to (last year):	4

A

Most Used Languages

Python 85.66% Makefile 2.58%
Ruby 5.51% Scheme 1.72%
Java 3.56% Visual Basic 0.97%

Pinned

Customize your pins

redis-py-cluster Public
Python cluster client for the official redis cluster. Redis 3.0+.
Python 1.1k 324

docker-redis-cluster Public
Dockerfile for Redis Cluster (redis 3.0+)
Shell 1.4k 547

pykwalify Public
Python YAML/JSON schema validation library
Python 282 85

aic Public
Asset Inventory CLI
Python 2

aic-modules Public
Modules for Asset Inventory CLI [AIC]
Python 1

VBSchell Public archive
Simple REPL for VBScript
Visual Basic 6 3

185 contributions in the last year

Contribution settings

2023

2022

Nov Dec Jan Feb Mar Apr May Jun Jul Aug Sep

Mon

redis

The open source, in-memory data store used by millions of developers as a database, cache, streaming engine, and message broker.

```
redis> PING
"PONG"
redis> HSET user:1 name antirez vocation artist
(integer) 2
redis> SET e 2.71
"OK"
redis> INCRBYFLOAT e 0.43
"3.14"
redis> RENAME e pi
"OK"
redis>
```

The Redis project began when Salvatore Sanfilippo, nicknamed antirez, the original developer of Redis, was trying to improve the scalability of his Italian startup, developing a real-time web log analyzer.

In 2009, the same year Redis was published, Chris Wanstrath, the CEO of a startup called GitHub, used Redis to build a job queue system named Resque for running GitHub's background tasks.

In the Rails world, Resque was the most popular job queue system at the time. A successor, Sidekiq, which emerged in 2012 and was also built on Redis, is now the top choice for Rails application developers.

Use cases

Real-time data store

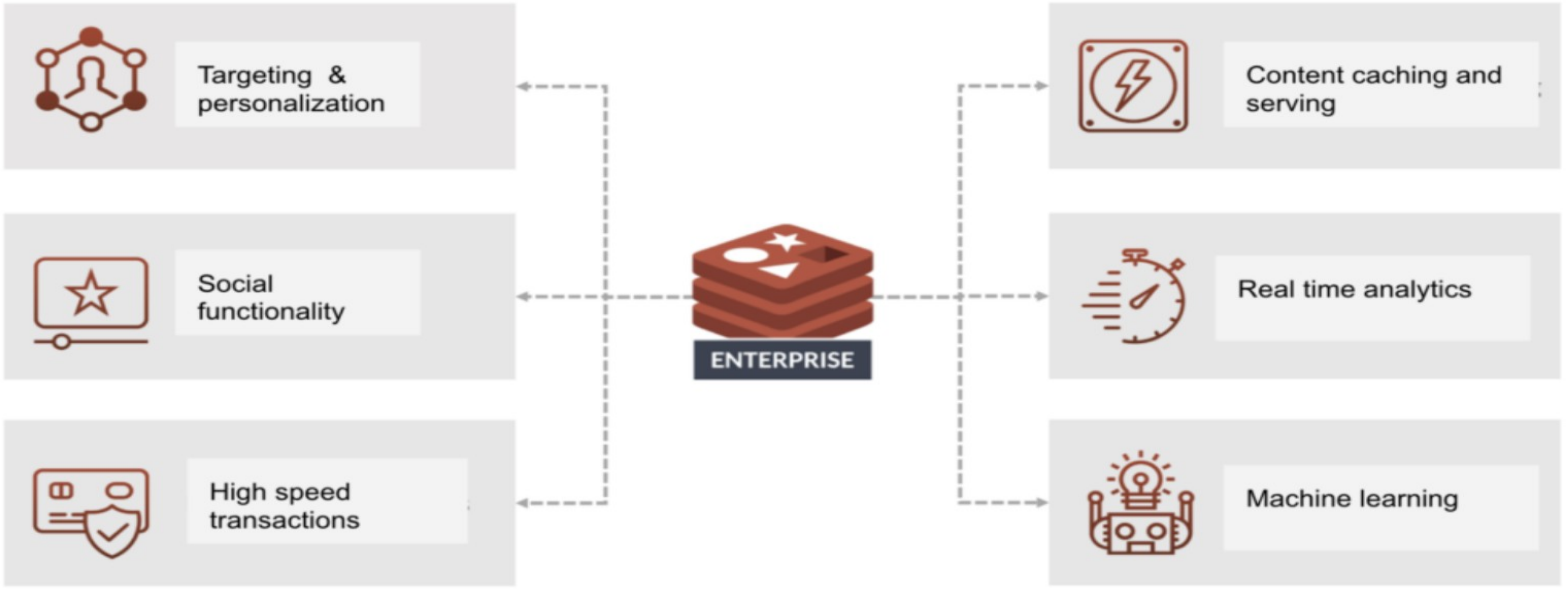
Redis' versatile in-memory data structures enable building data infrastructure for real-time applications that require low latency and high-throughput.

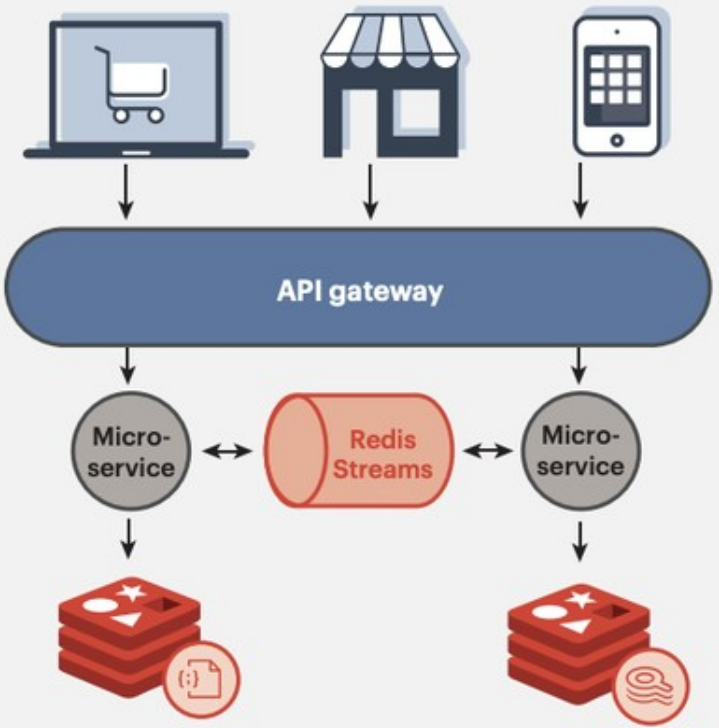
Caching & session storage

Redis' speed makes it ideal for caching database queries, complex computations, API calls, and session state.

Streaming & messaging

The stream data type enables high-rate data ingestion, messaging, event sourcing, and notifications.





Performance

Consistency

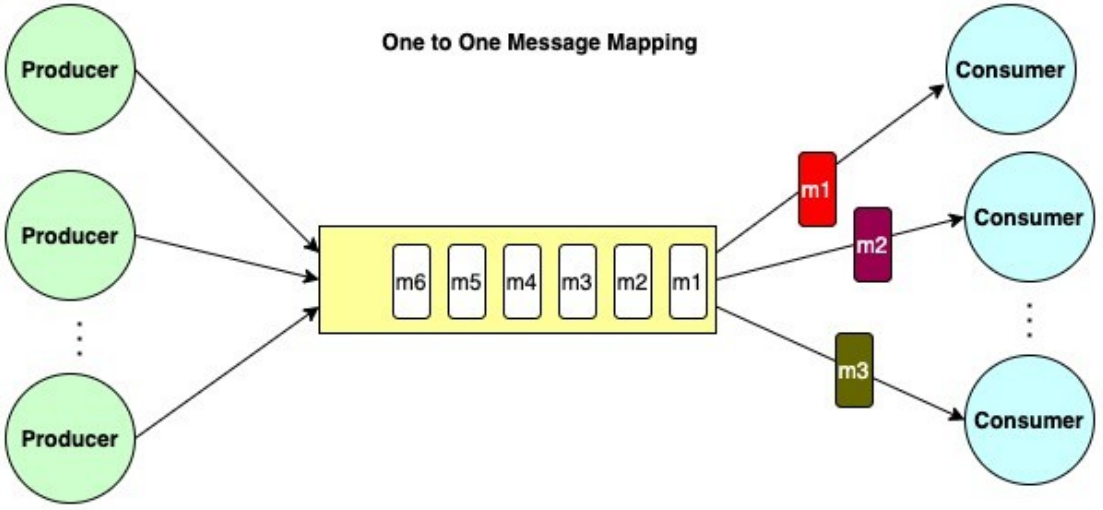
Vendor/tech sprawl

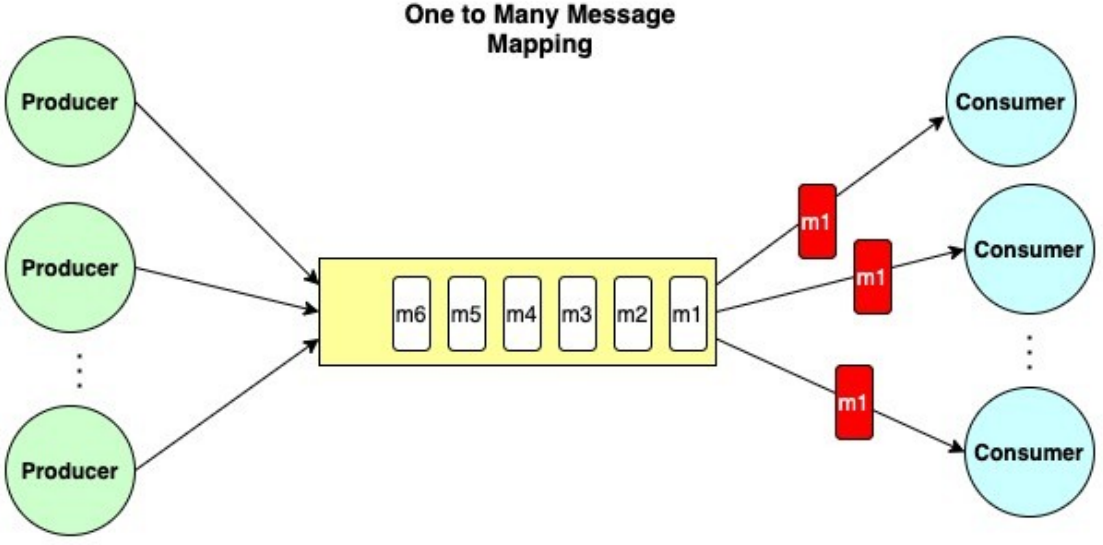
Cache

Chat, messages, queues

Session storage för webb

Geospatial indexes


One to One Message Mapping
Producer
Producer
Producer
m6
m5
m4
m3
m2
m1
m1
m2
m3
Consumer
Consumer
Consumer


One to Many Message Mapping
Producer
Producer
Producer
m6
m5
m4
m3
m2
m1
m1
m1
m1
Consumer
Consumer
Consumer

Performance

All Redis data resides in memory, which enables low latency and high throughput data access. Unlike traditional databases, In-memory data stores don’t require a trip to disk, reducing engine latency to microseconds. Because of this, in-memory data stores can support an order of magnitude more operations and faster response times. The result is blazing-fast performance with average read and write operations taking less than a millisecond and support for millions of operations per second.

Flexible data structures

Unlike other key-value data stores that offer limited data structures, Redis has a vast variety of data structures to meet your application needs. Redis data types include:

Simplicity and ease-of-use

Redis enables you to write traditionally complex code with fewer, simpler lines. With Redis, you write fewer lines of code to store, access, and use data in your applications. The difference is that developers who use Redis can use a simple command structure as opposed to the query languages of traditional databases. For example, you can use the Redis hash data structure to move data to a data store with only one line of code. A similar task on a data store with no hash data structures would require many lines of code to convert from one format to another. Redis comes with native data structures and many options to manipulate and interact with your data. Over a hundred open source clients are available for Redis developers. Supported languages include Java, Python, PHP, C, C++, C#, JavaScript, Node.js, Ruby, R, Go, and many others.

High availability and scalability

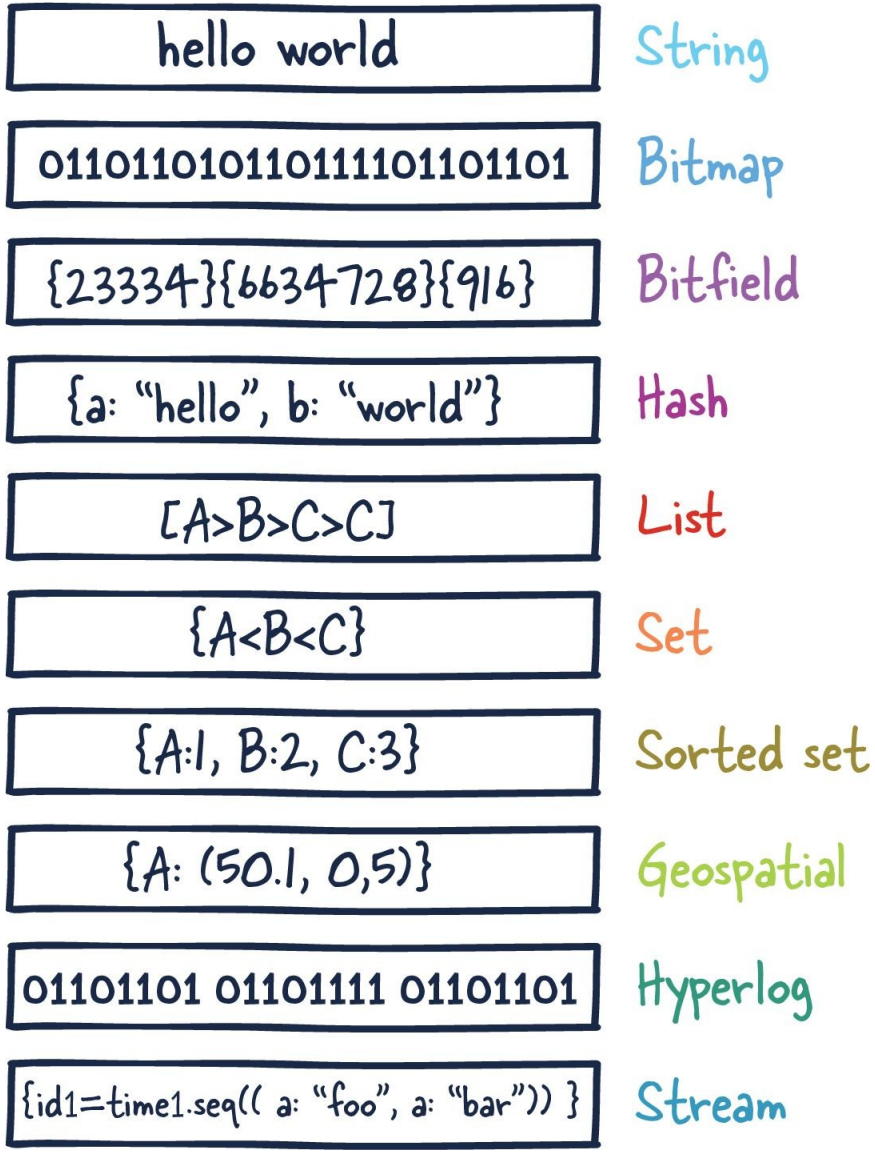
Redis offers a primary-replica architecture in a single node primary or a clustered topology. This allows you to build highly available solutions providing consistent performance and reliability. When you need to adjust your cluster size, various options to scale up and scale in or out are also available. This allows your cluster to grow with your demands.

Open Source

Redis is an open source project supported by a vibrant community, including AWS. There’s no vendor or technology lock in as Redis is open standards based, supports open data formats, and features a rich set of clients.

Here is a current breakdown of capabilities between these two caches.

	Memcached	Redis
Sub-millisecond latency	Yes	Yes
Developer ease of use	Yes	Yes
Data partitioning	Yes	Yes
Support for a broad set of programming languages	Yes	Yes
Advanced data structures	-	Yes
Multithreaded architecture	Yes	-
Snapshots	-	Yes
Replication	-	Yes
Transactions	-	Yes
Pub/Sub	-	Yes
Lua scripting	-	Yes
Geospatial support	-	Yes




```
127.0.0.1:6379> multi
127.0.0.1:6379> set key_MeaningOfLife 1
127.0.0.1:6379> incr key_MeaningOfLife
127.0.0.1:6379> incrby key_MeaningOfLife 40
127.0.0.1:6379> get key_MeaningOfLife
127.0.0.1:6379> exec
```

Redis Modules

RediSearch - A query and indexing engine for Redis, providing secondary indexing, full-text search and aggregations.

RedisJSON - a JSON data type for Redis

RedisGraph - A graph database as a Redis module

RedisBloom - Probabilistic Datatypes Module for Redis

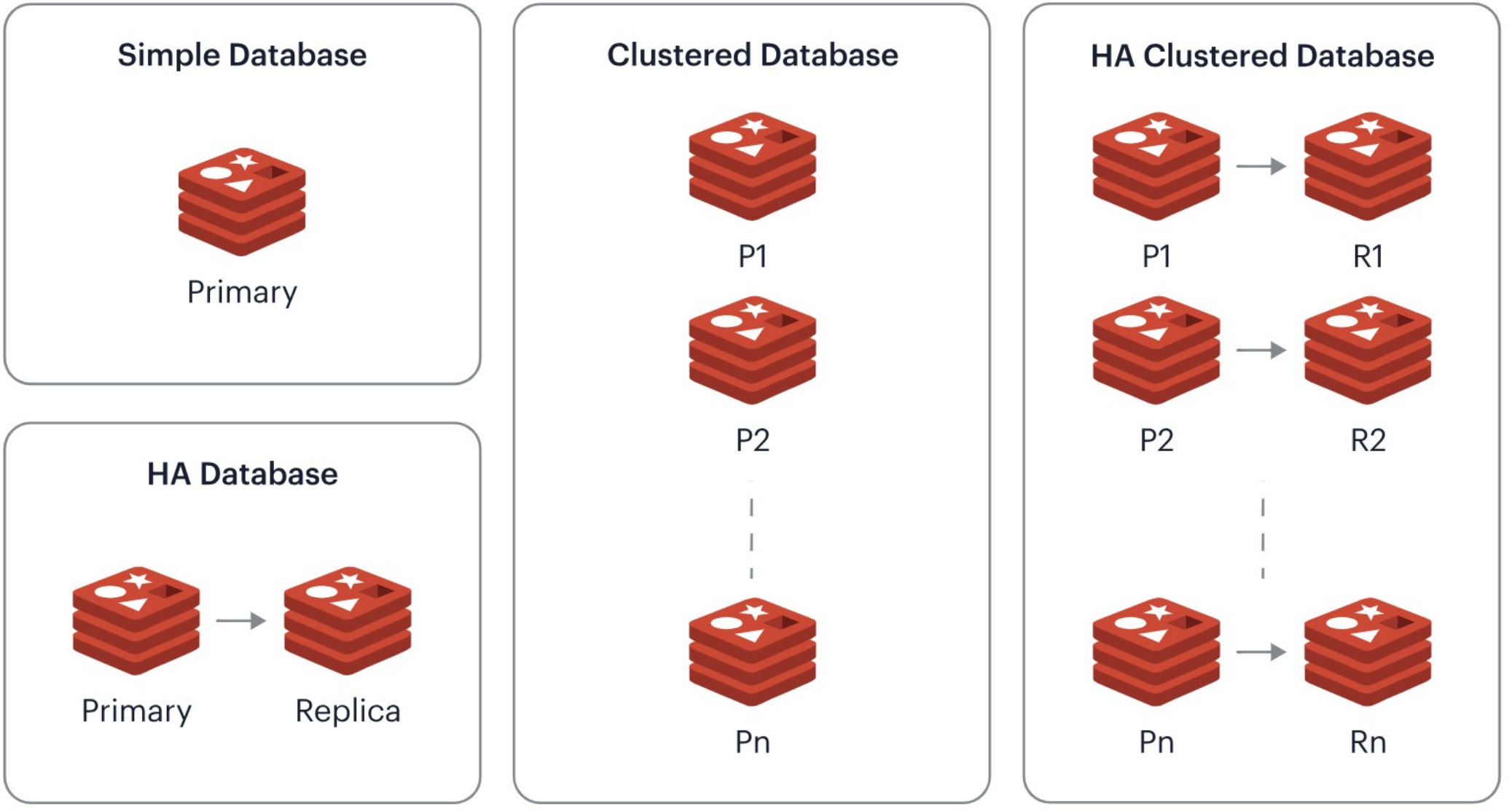
redis-cell - A Redis module that provides rate limiting in Redis as a single command.

RedisTimeSeries - Time series data structure for Redis

RedisAI - A Redis module for serving tensors and executing deep learning graphs

redis-roaring - Roaring Bitmaps for Redis

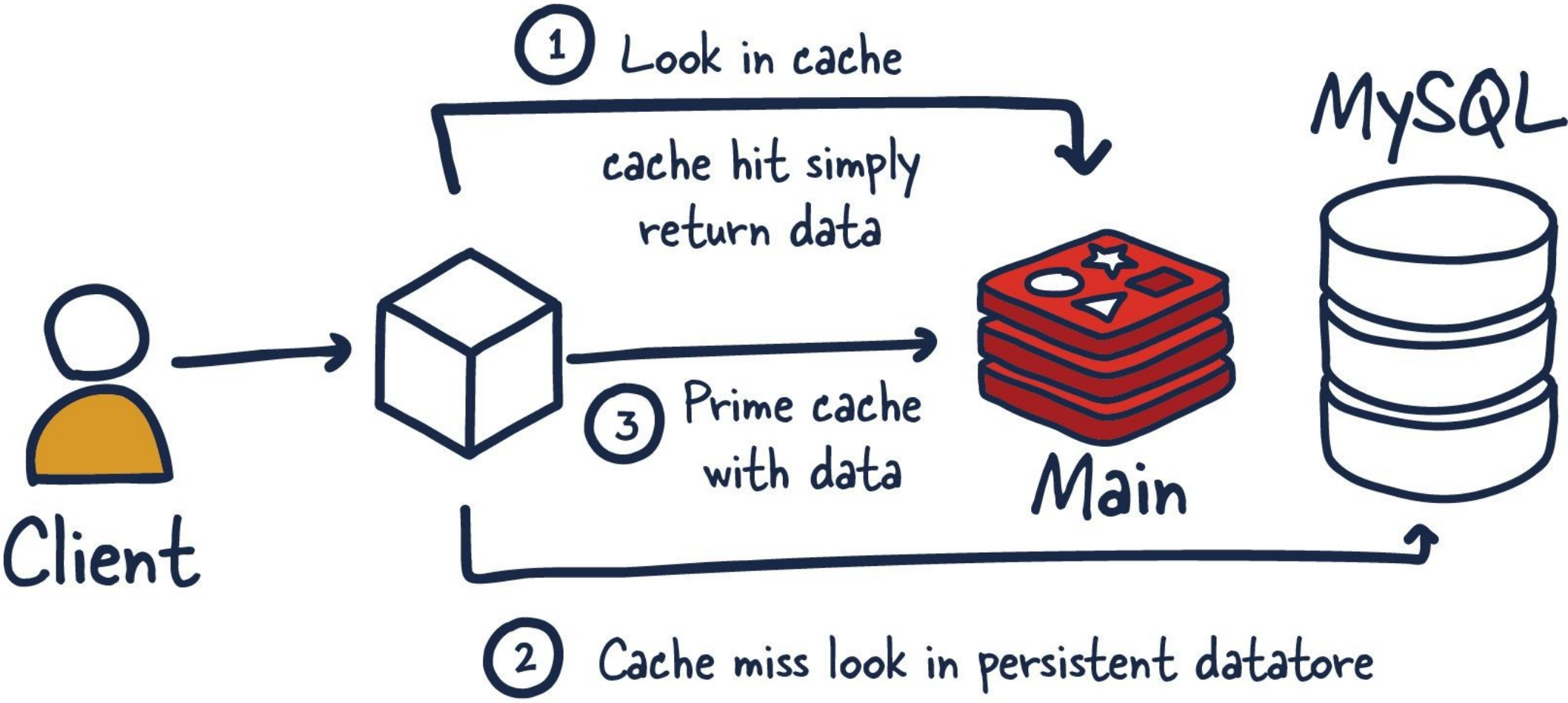
RedisGears - Dynamic execution framework for your Redis data

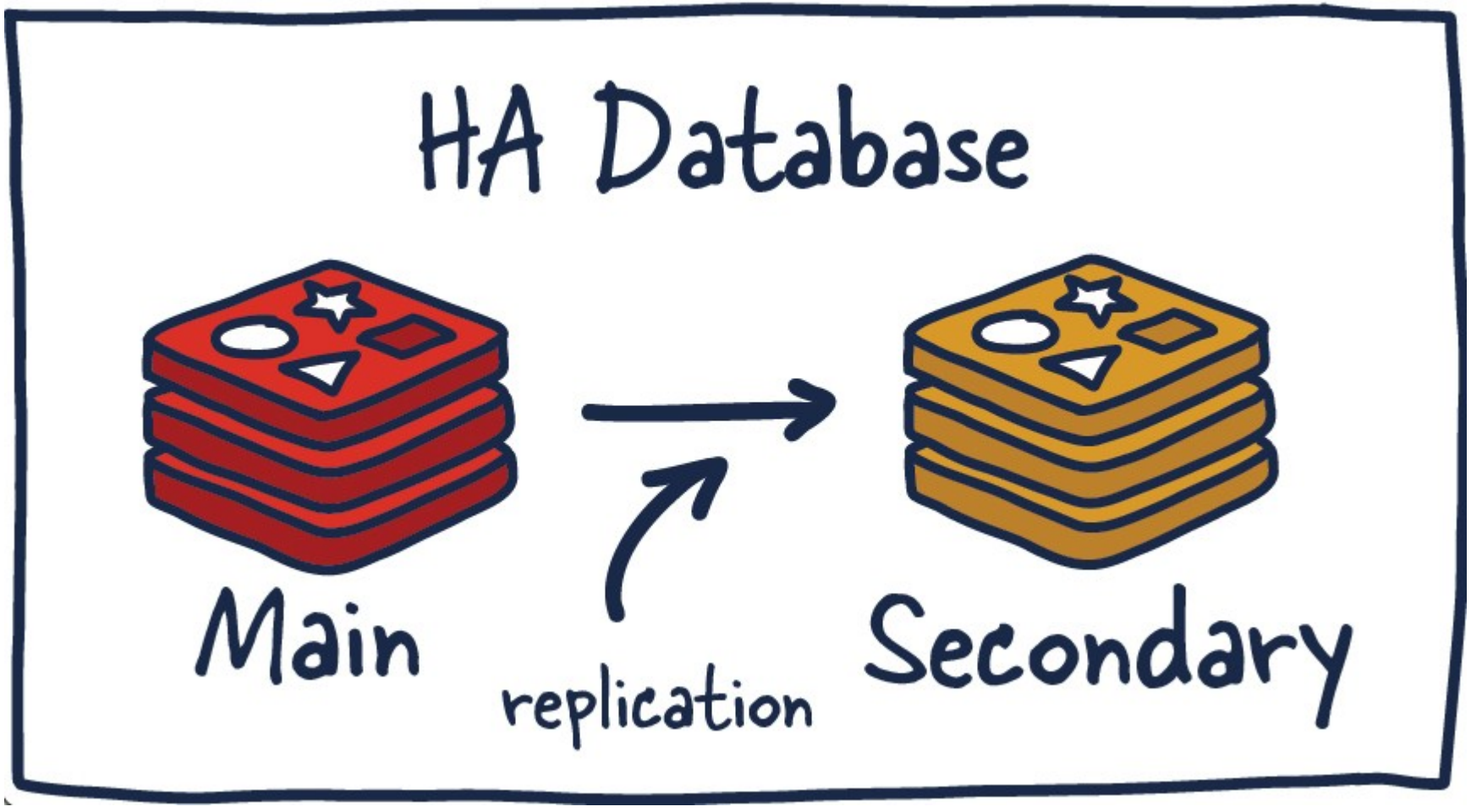

Simple Database
Primary
HA Database
Primary
Replica
Clustered Database
P1
P2
Pn
HA Clustered Database
P1
R1
P2
R2
Pn
Rn

Simple Database

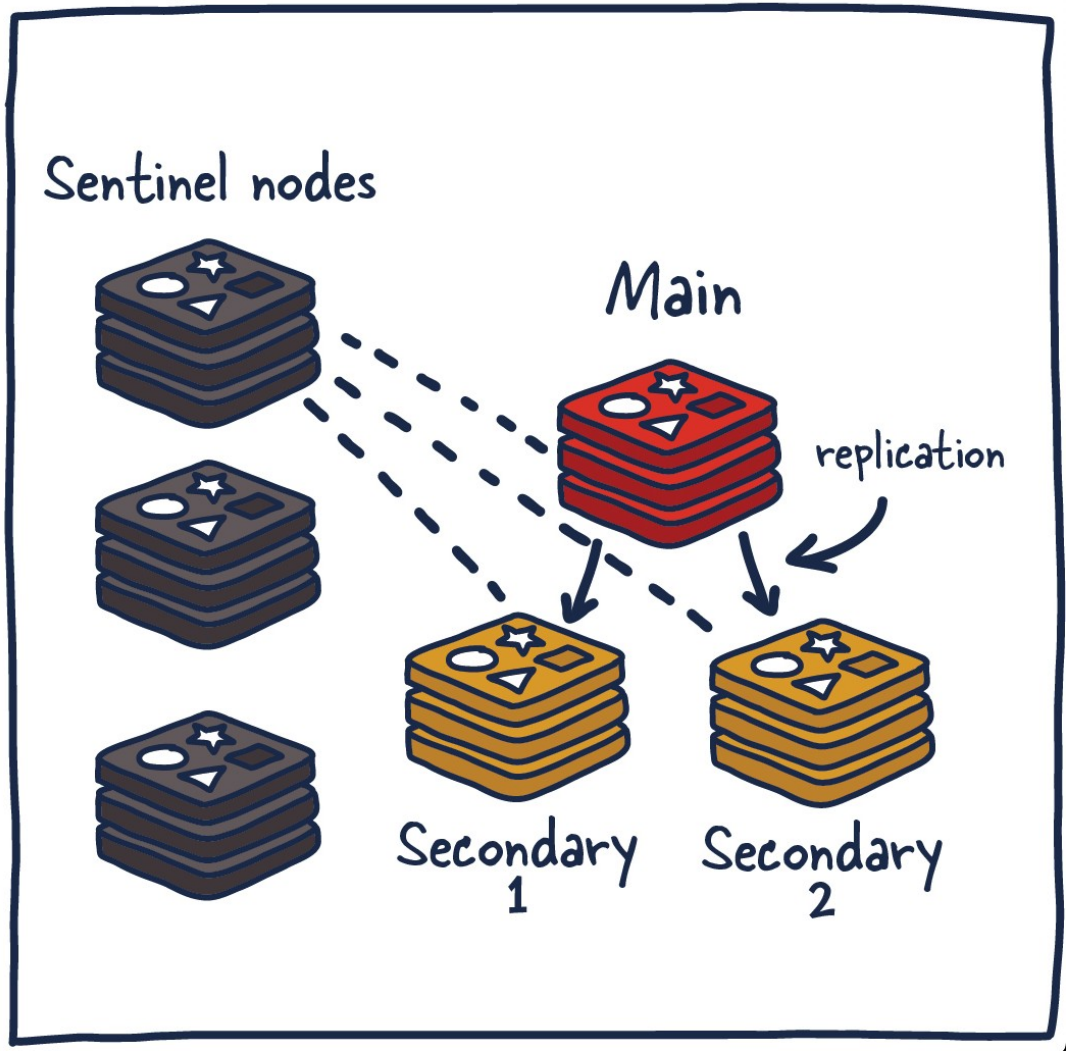
Main

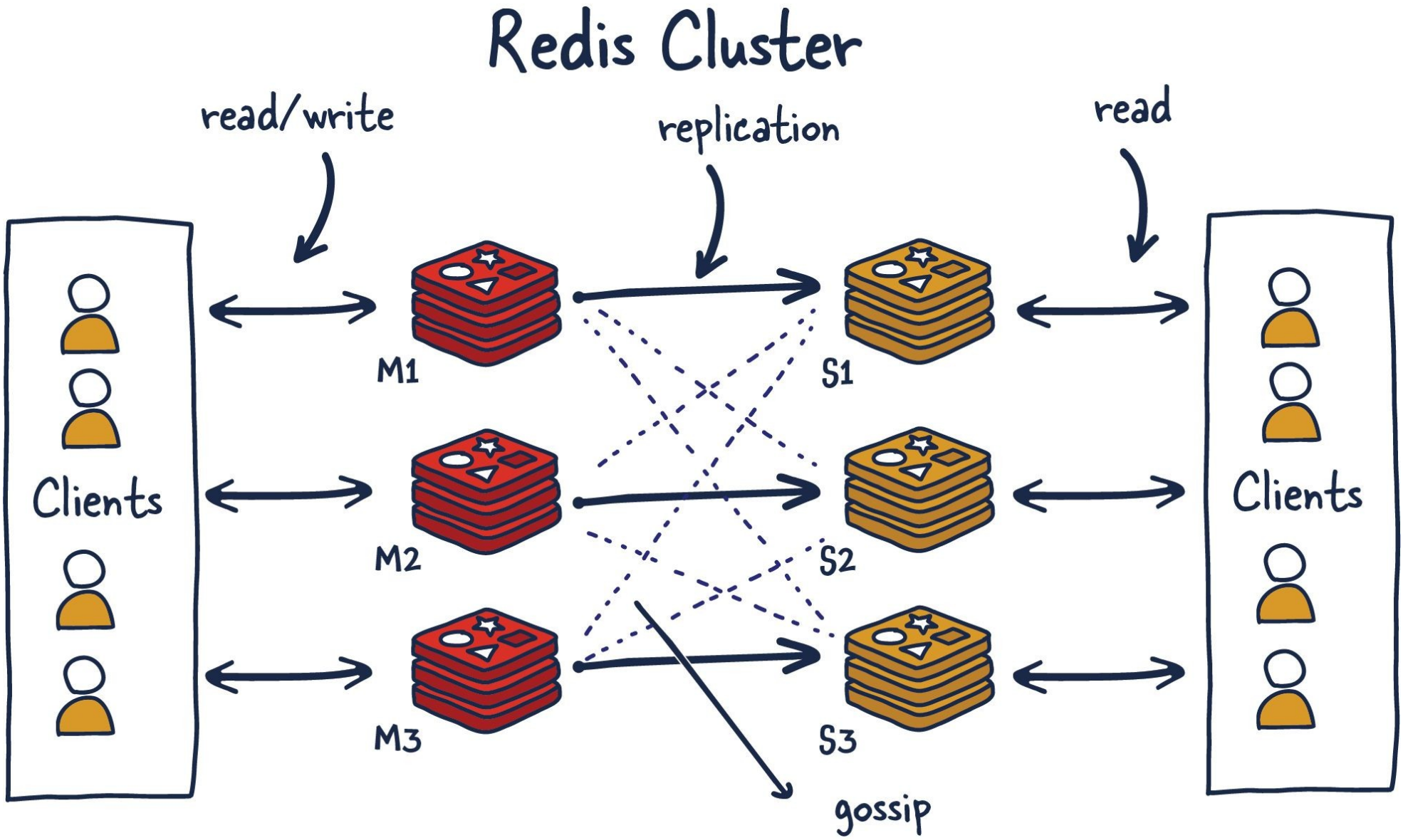
How is redis traditionally used




HA Database
Main
replication
Secondary

Redis sentinel




Redis Cluster
read/write
replication
read
Clients
M1
M2
M3
S1
S2
S3
Clients
gossip

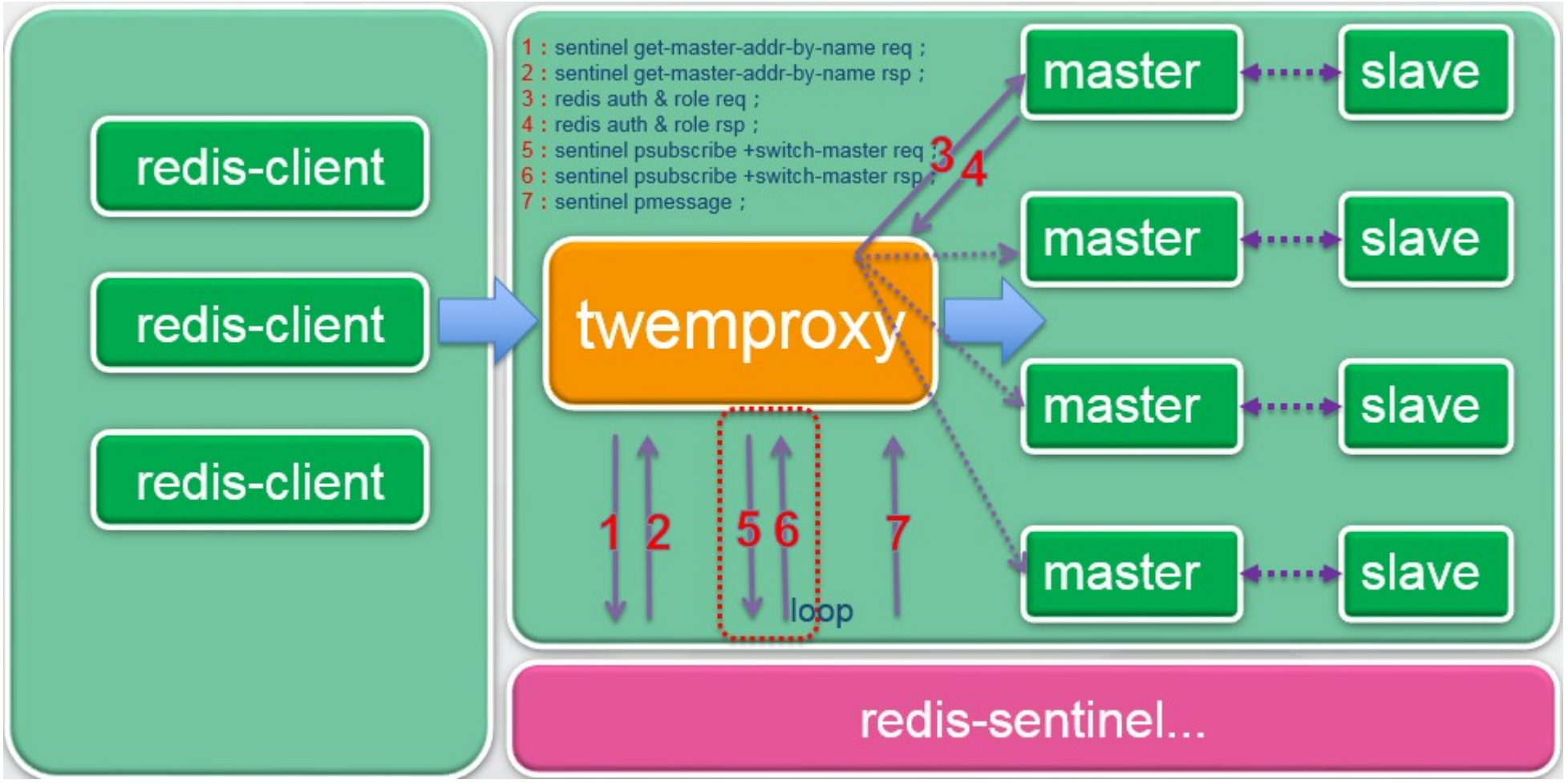
https://github.com/twitter/twemcache

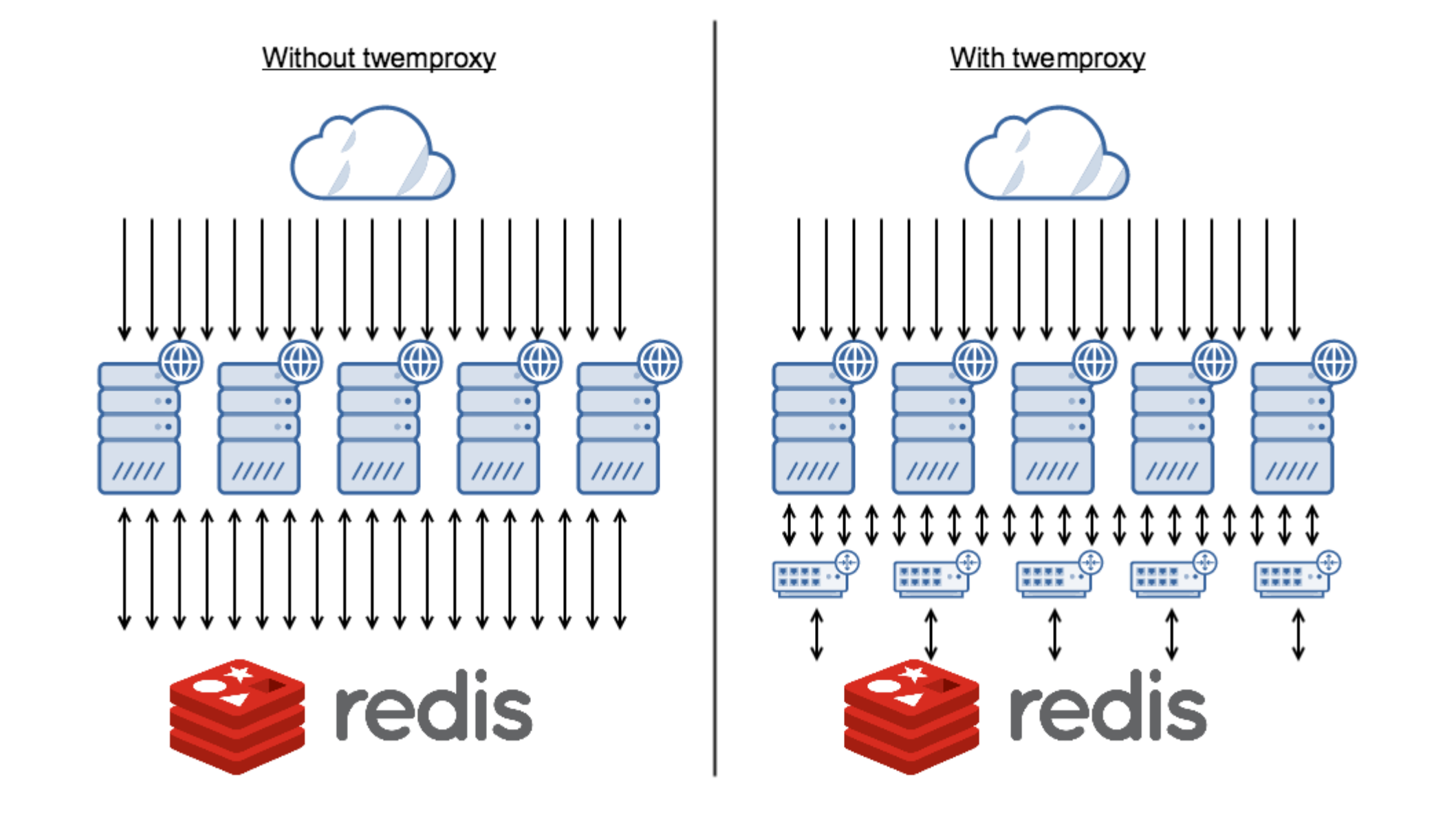
Twemproxy, a lightweight proxy for Memcache & Redis protocol, enables the caching layer to scale horizontally minimizing the connections to the backend caching servers.

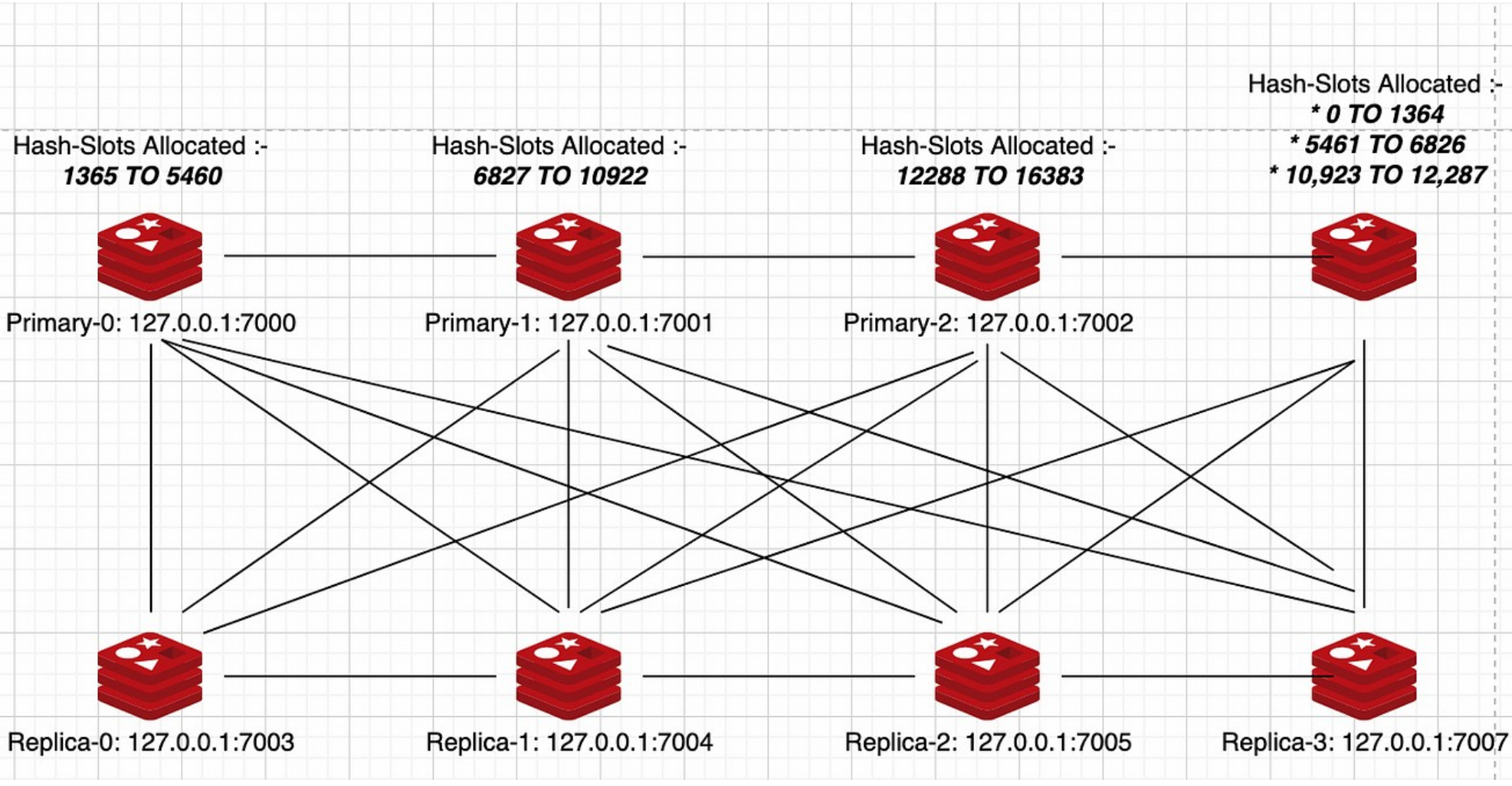
Twitter internally runs many caching services & one of them is powered by Redis. Redis clusters cache direct user messages, ad spends, impressions & engagement.

There is another caching service, Haplo which acts as a primary cache for the Twitter Timeline. It’s backed by Redis.

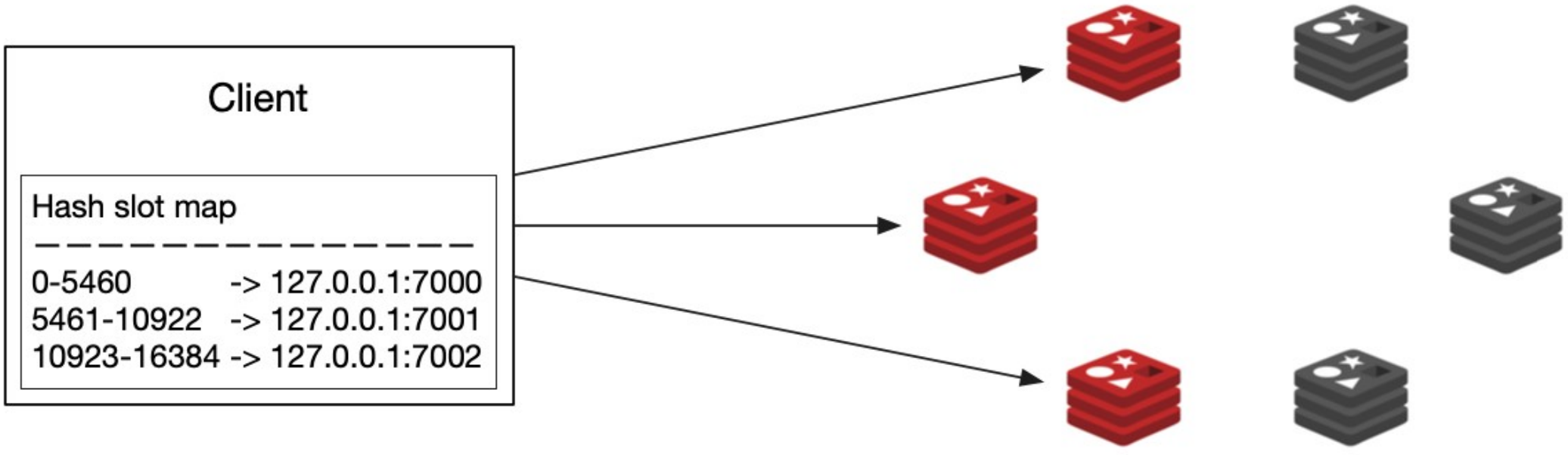
How Twitter Uses Redis To Scale - 105TB RAM, 39MM QPS, 10,000+ Instances


redis-client
redis-client
redis-client
1 : sentinel get-master-addr-by-name req ;
2 : sentinel get-master-addr-by-name rsp ;
3 : redis auth & role req ;
4 : redis auth & role rsp ;
5 : sentinel psubscribe +switch-master req ;
6 : sentinel psubscribe +switch-master rsp ;
7 : sentinel pmessage ;
twemproxy
3
4
1
2
5
6
7
loop
master
slave
master
slave
master
slave
master
slave
redis-sentinel...

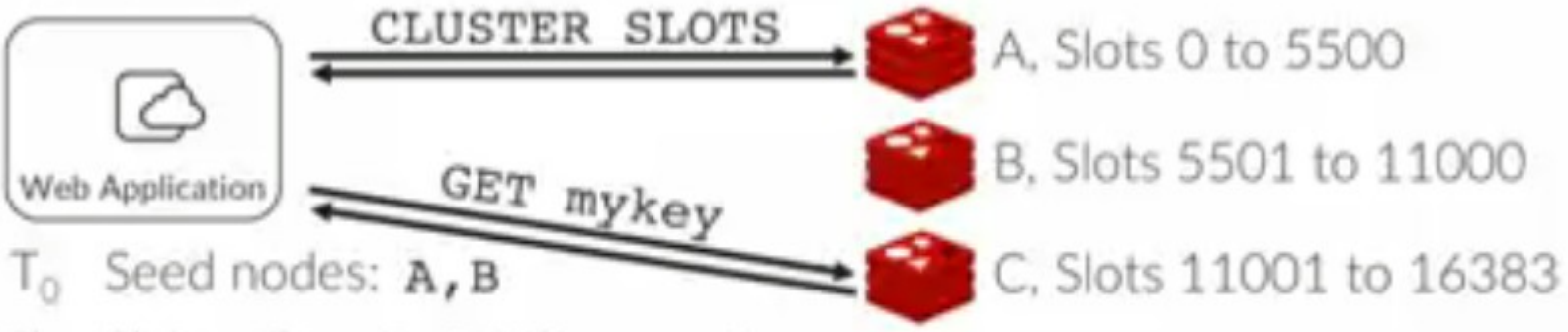

Without twemproxy
redis
With twemproxy
redis


Hash-Slots Allocated :-
1365 TO 5460
Hash-Slots Allocated :-
6827 TO 10922
Hash-Slots Allocated :-
12288 TO 16383
Hash-Slots Allocated :-
* 0 TO 1364
* 5461 TO 6826
* 10,923 TO 12,287
Primary-0: 127.0.0.1:7000
Primary-1: 127.0.0.1:7001
Primary-2: 127.0.0.1:7002
Replica-0: 127.0.0.1:7003
Replica-1: 127.0.0.1:7004
Replica-2: 127.0.0.1:7005
Replica-3: 127.0.0.1:7007

Cluster discovery



Cluster Slots



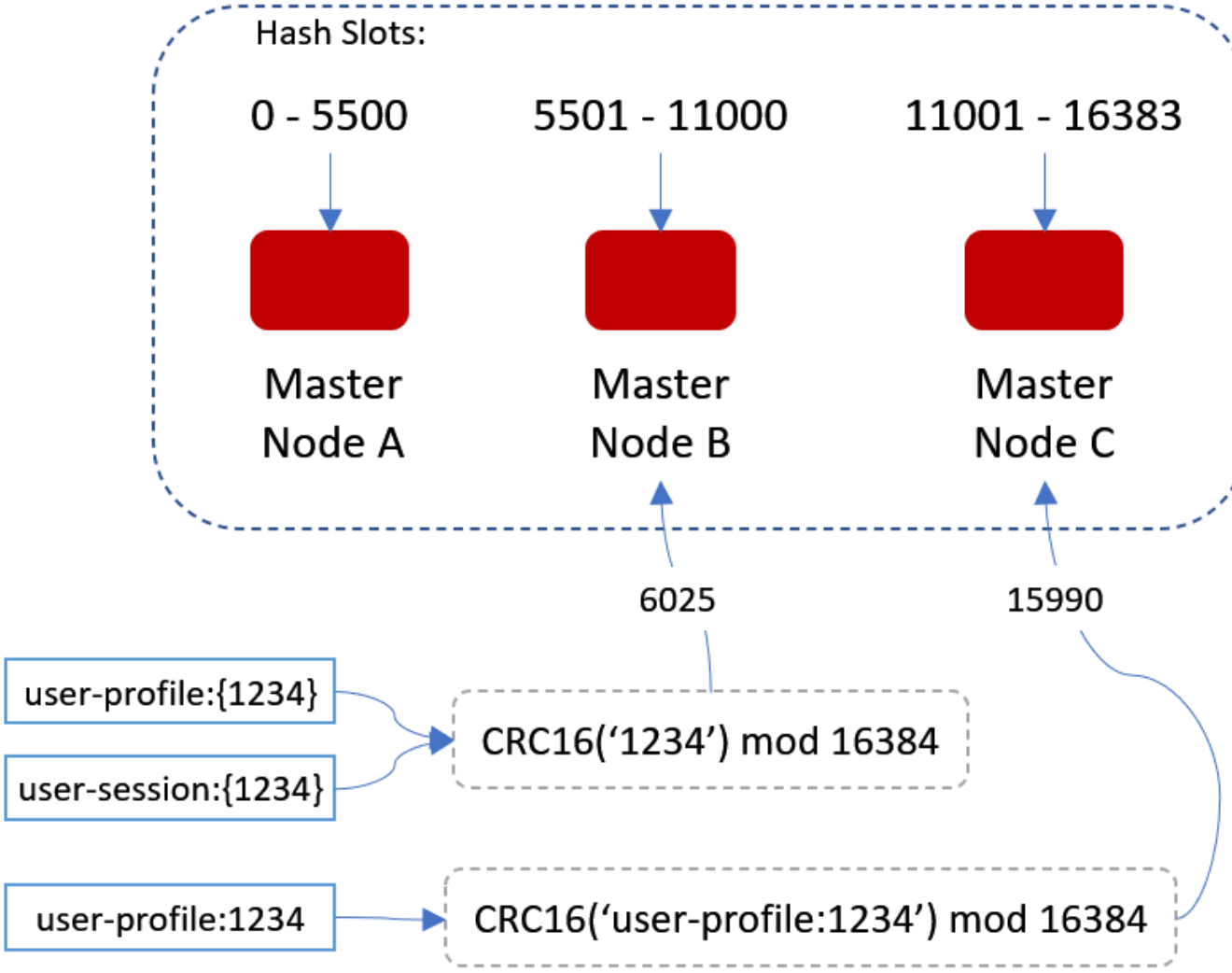
T_0 Seed nodes: **A,B**

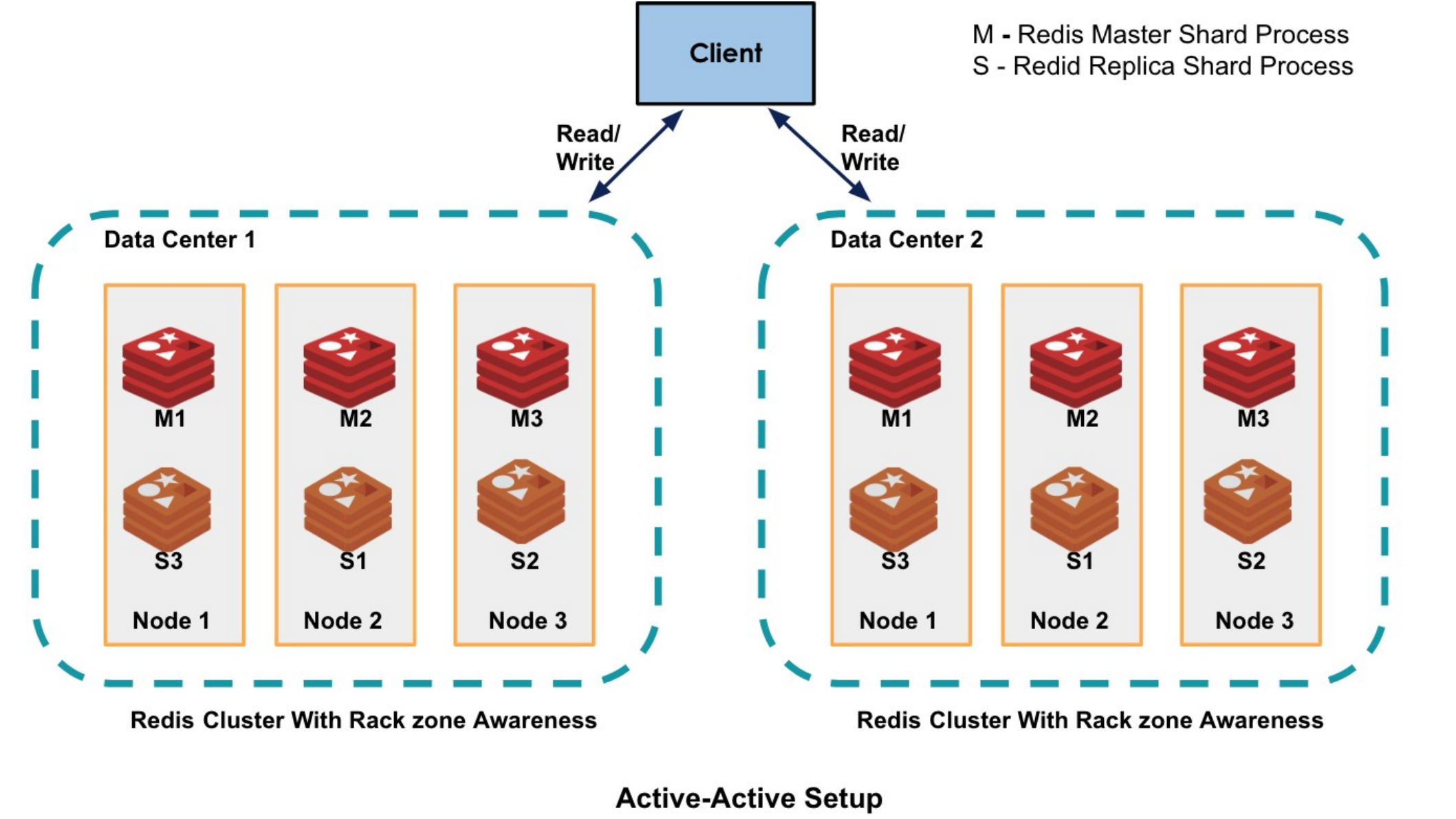
T_1 Slots: **{A: 0-5500, ...}**

T_2 Fetch mykey

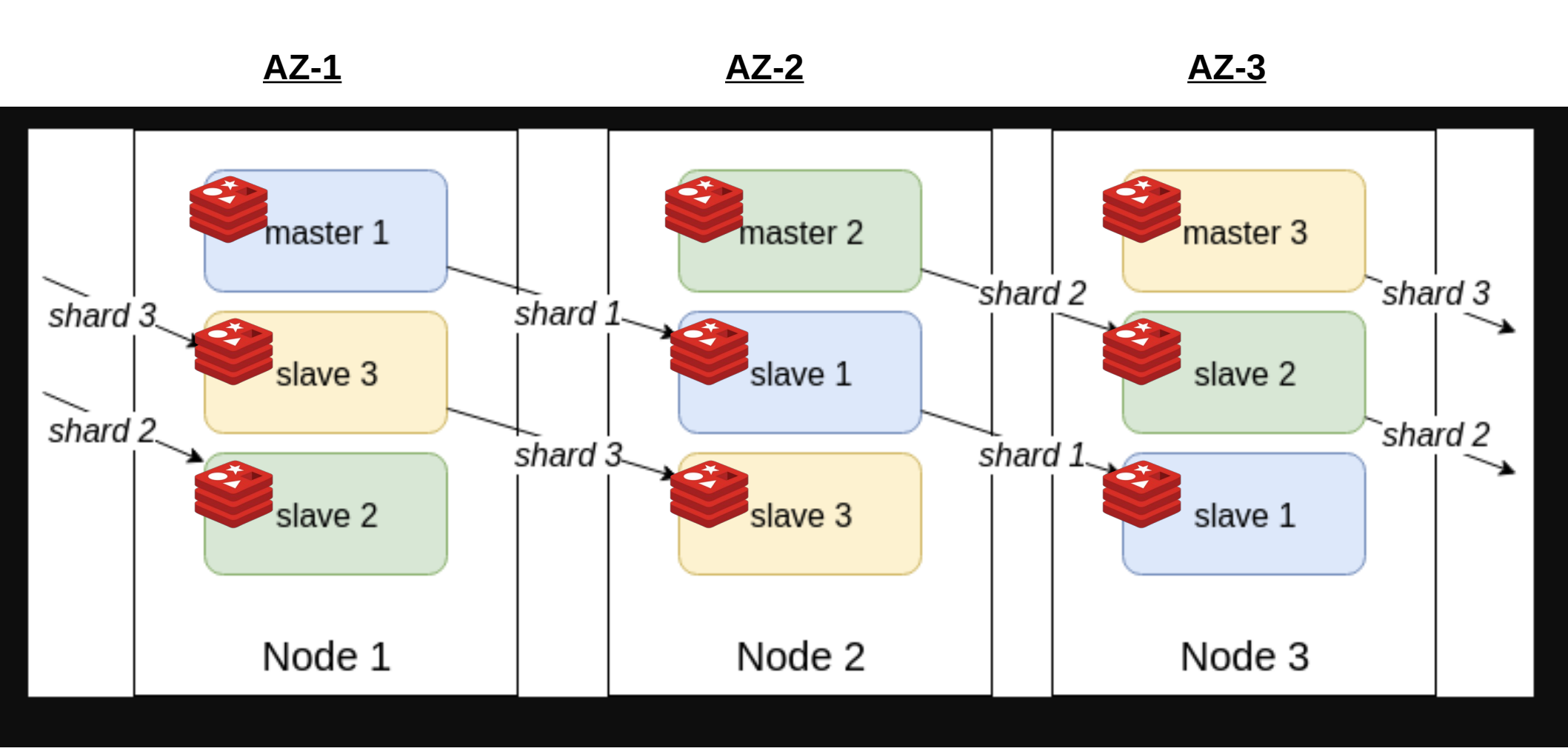
```
HASH_SLOT = CRC16("mykey") mod 16384
HASH_SLOT = 14687
```

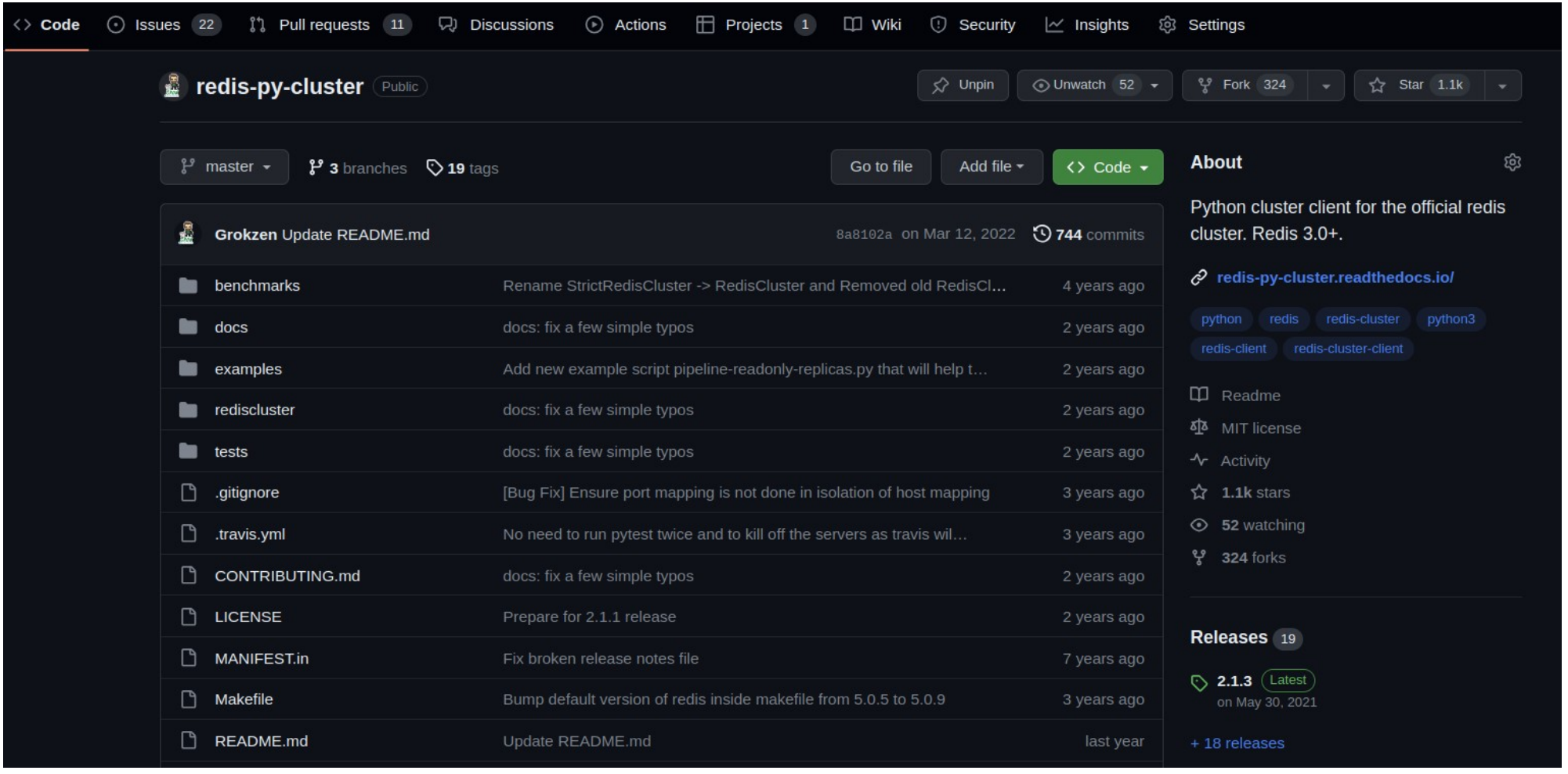
Redis Cluster Data Sharding – Hash tags





Active-Active Setup


AZ-1
AZ-2
AZ-3
master 1
slave 3
slave 2
master 2
slave 1
slave 3
master 3
slave 2
slave 1
shard 3
shard 2
shard 1
shard 3
shard 2
shard 1
shard 3
shard 2
Node 1
Node 2
Node 3


Code
Issues 22
Pull requests 11
Discussions
Actions
Projects 1
Wiki
Security
Insights
Settings
redis-py-cluster
Public
Unpin
Unwatch 52
Fork 324
Star 1.1k
master
3 branches
19 tags
Go to file
Add file
Code
Grokzen Update README.md
8a8102a on Mar 12, 2022
744 commits
benchmarks Rename StrictRedisCluster -> RedisCluster and Removed old RedisCl... 4 years ago
docs docs: fix a few simple typos 2 years ago
examples Add new example script pipeline-readonly-replicas.py that will help t... 2 years ago
rediscluster docs: fix a few simple typos 2 years ago
tests docs: fix a few simple typos 2 years ago
.gitignore [Bug Fix] Ensure port mapping is not done in isolation of host mapping 3 years ago
.travis.yml No need to run pytest twice and to kill off the servers as travis wil... 3 years ago
CONTRIBUTING.md docs: fix a few simple typos 2 years ago
LICENSE Prepare for 2.1.1 release 2 years ago
MANIFEST.in Fix broken release notes file 7 years ago
Makefile Bump default version of redis inside makefile from 5.0.5 to 5.0.9 3 years ago
README.md Update README.md last year
About
Python cluster client for the official redis cluster. Redis 3.0+.
redis-py-cluster.readthedocs.io/
python
redis
redis-cluster
python3
redis-client
redis-cluster-client
Readme
MIT license
Activity
1.1k stars
52 watching
324 forks
Releases 19
2.1.3 Latest
on May 30, 2021
+ 18 releases

Usage example

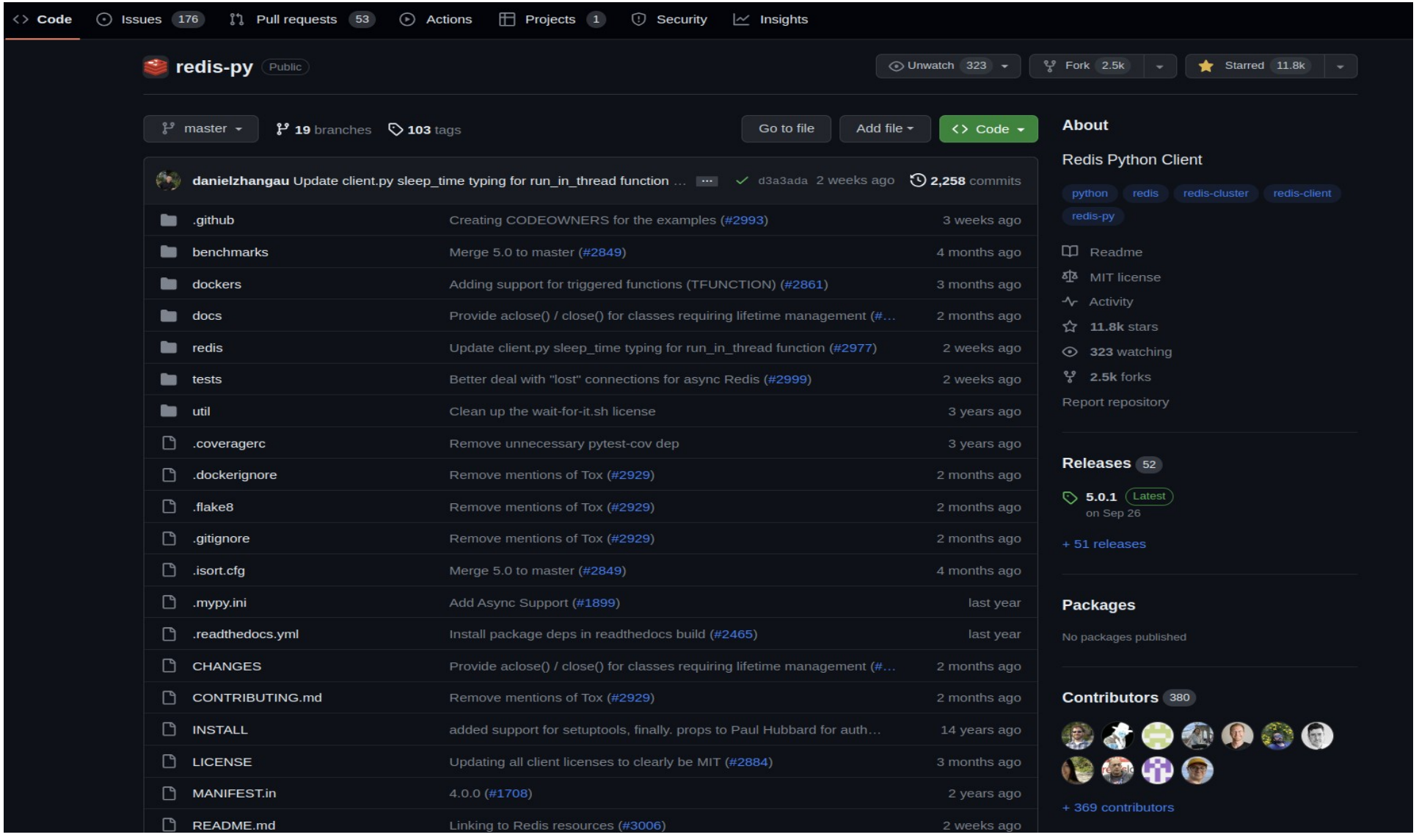
Small sample script that shows how to get started with RedisCluster. It can also be found in examples/basic.py

```
>>> from rediscluster import RedisCluster

>>> # Requires at least one node for cluster discovery. Multiple nodes is recommended.
>>> startup_nodes = [{"host": "127.0.0.1", "port": "7000"}, {"host": "127.0.0.1", "port": "7001"}]
>>> rc = RedisCluster(startup_nodes=startup_nodes, decode_responses=True)

# Or you can use the simpler format of providing one node same way as with a Redis() instance
<<< rc = RedisCluster(host="127.0.0.1", port=7000, decode_responses=True)

>>> rc.set("foo", "bar")
True
>>> print(rc.get("foo"))
'bar'
```


Code
Issues 176
Pull requests 53
Actions
Projects 1
Security
Insights
redis-py
Public
Unwatch 323
Fork 2.5k
Starred 11.8k
master
19 branches
103 tags
Go to file
Add file
Code
danielzhangau Update client.py sleep_time typing for run_in_thread function ...
d3a3ada 2 weeks ago
2,258 commits
.github Creating CODEOWNERS for the examples (#2993) 3 weeks ago
benchmarks Merge 5.0 to master (#2849) 4 months ago
dockers Adding support for triggered functions (TFUNCTION) (#2861) 3 months ago
docs Provide aclose() / close() for classes requiring lifetime management (#... 2 months ago
redis Update client.py sleep_time typing for run_in_thread function (#2977) 2 weeks ago
tests Better deal with "lost" connections for async Redis (#2999) 2 weeks ago
util Clean up the wait-for-it.sh license 3 years ago
.coveragerc Remove unnecessary pytest-cov dep 3 years ago
.dockerignore Remove mentions of Tox (#2929) 2 months ago
.flake8 Remove mentions of Tox (#2929) 2 months ago
.gitignore Remove mentions of Tox (#2929) 2 months ago
.isort.cfg Merge 5.0 to master (#2849) 4 months ago
.mypy.ini Add Async Support (#1899) last year
.readthedocs.yml Install package deps in readthedocs build (#2465) last year
CHANGES Provide aclose() / close() for classes requiring lifetime management (#... 2 months ago
CONTRIBUTING.md Remove mentions of Tox (#2929) 2 months ago
INSTALL added support for setuptools, finally. props to Paul Hubbard for auth... 14 years ago
LICENSE Updating all client licenses to clearly be MIT (#2884) 3 months ago
MANIFEST.in 4.0.0 (#1708) 2 years ago
README.md Linking to Redis resources (#3006) 2 weeks ago
About
Redis Python Client
python
redis
redis-cluster
redis-client
redis-py
Readme
MIT license
Activity
11.8k stars
323 watching
2.5k forks
Report repository
Releases 52
5.0.1 Latest
on Sep 26
+ 51 releases
Packages
No packages published
Contributors 380
+ 369 contributors

DYNAMIST OFFER TRAINING IN

REDIS

Dynamist offers training opportunities with a maximum of 10 participants at a time. Each training session takes about 5 hours and is divided into four different sections. Each block is estimated at about 1 hour. However, the sections may change during the training itself based on your needs.

Course Overview

1. Basics
2. Cluster
3. Metrics
4. Operations

Length: 1 day

https://redis.readthedocs.io/en/stable/clustering.html

https://redis.io/

https://aws.amazon.com/redis/

https://twitter.com/antirez

https://github.com/redis/redis-py

https://github.com/Grokzen/redis-py-cluster

https://github.com/Grokzen/docker-redis-cluster

https://redis.com/redis-enterprise/technology/redis-enterprise-cluster-architecture/