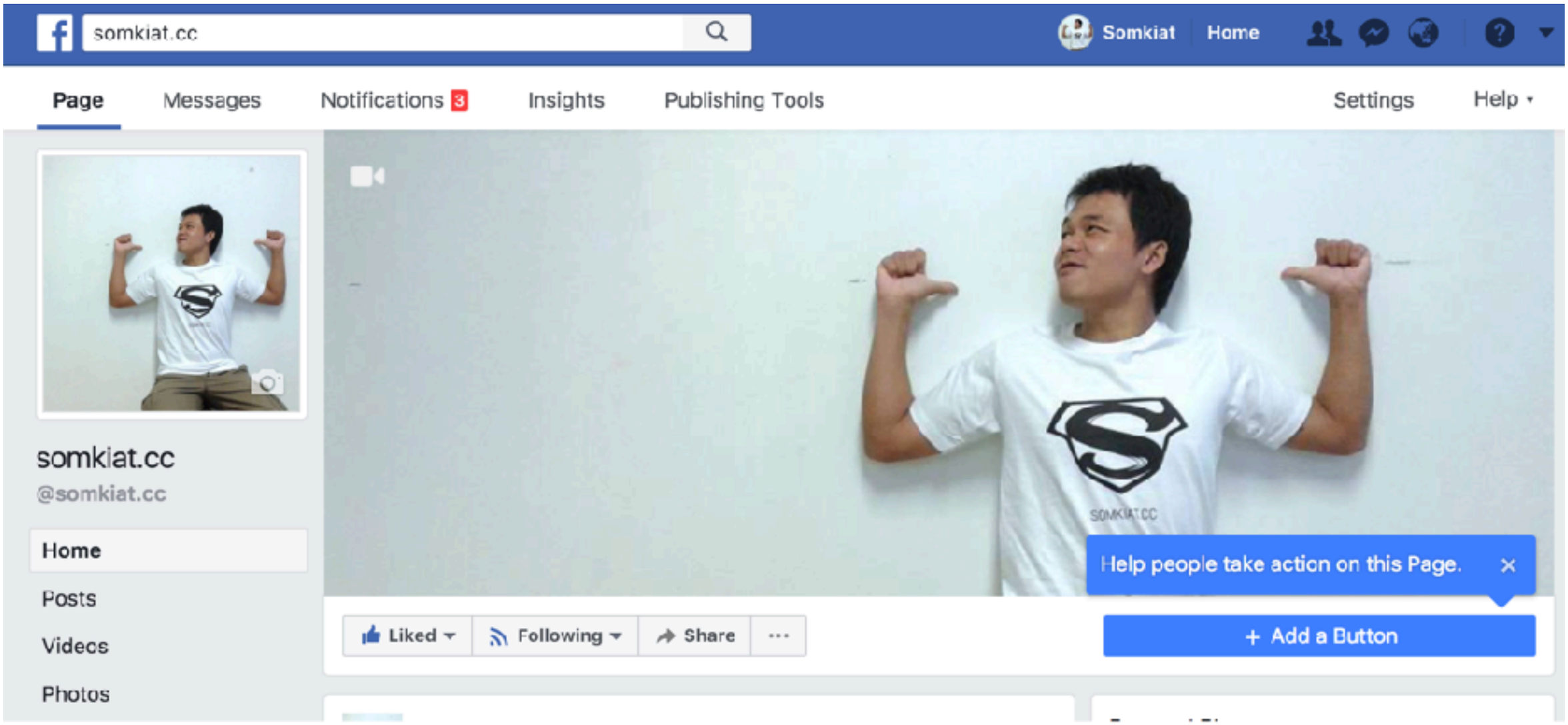
สยามชำนาญกิจ
SIAM CHAMNAN KIT

Redis Workshop



Redis Workshop

https://github.com/up1/course-redis

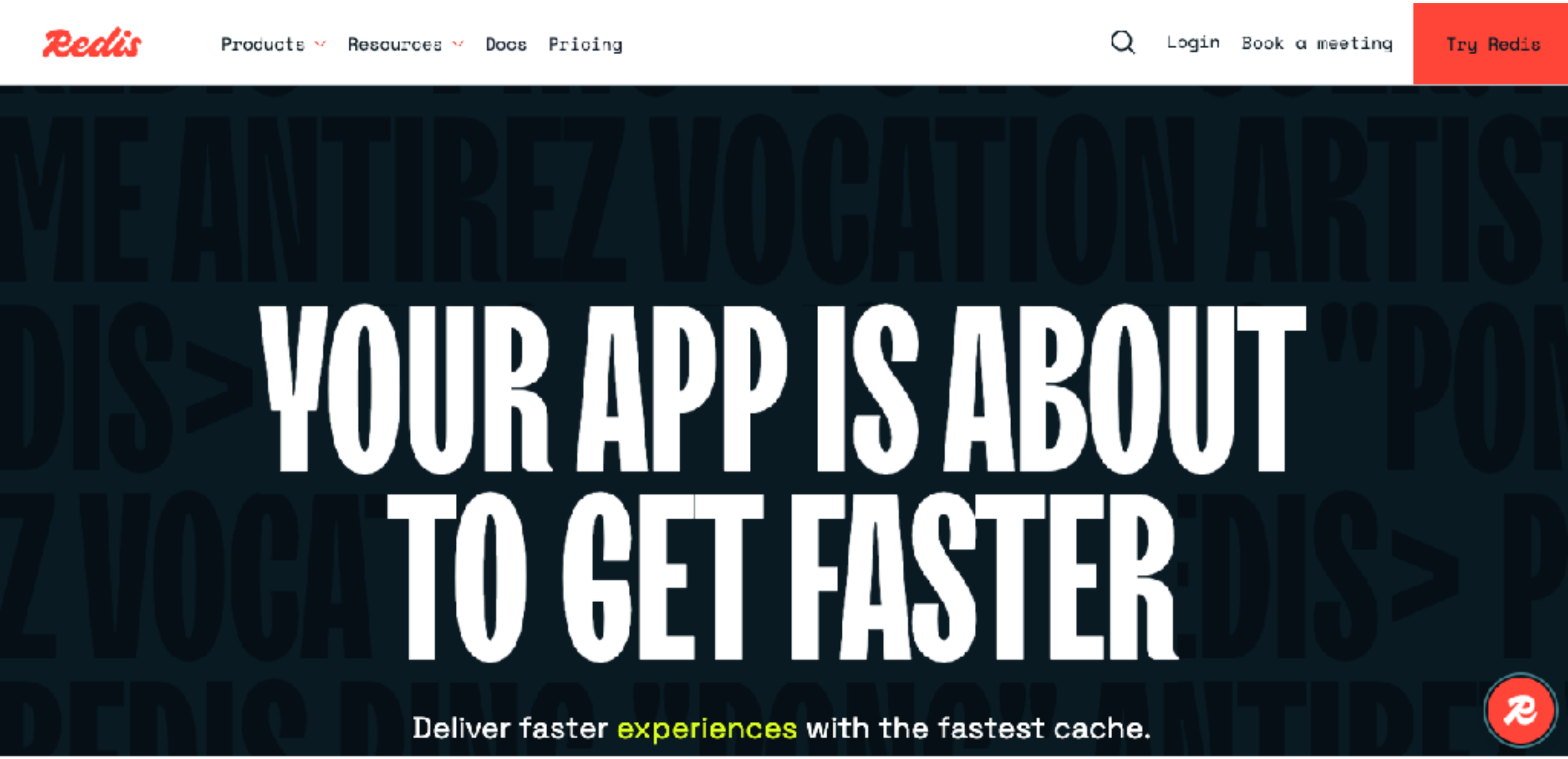
Topics (1)

Database models
Why and what Redis ?
Think about Redis
NoSQL database
Working with Redis
Use cases

Topics (2)

Data types
Data modeling
Redis deployment topologies
Monitoring
Performance tuning and benchmark

Redis



https://redis.io/

Redis

REmote **DI**ctionary **S**erver

Key-value database

Keep data in memory (save to disk)

Expiration data

Data structure server

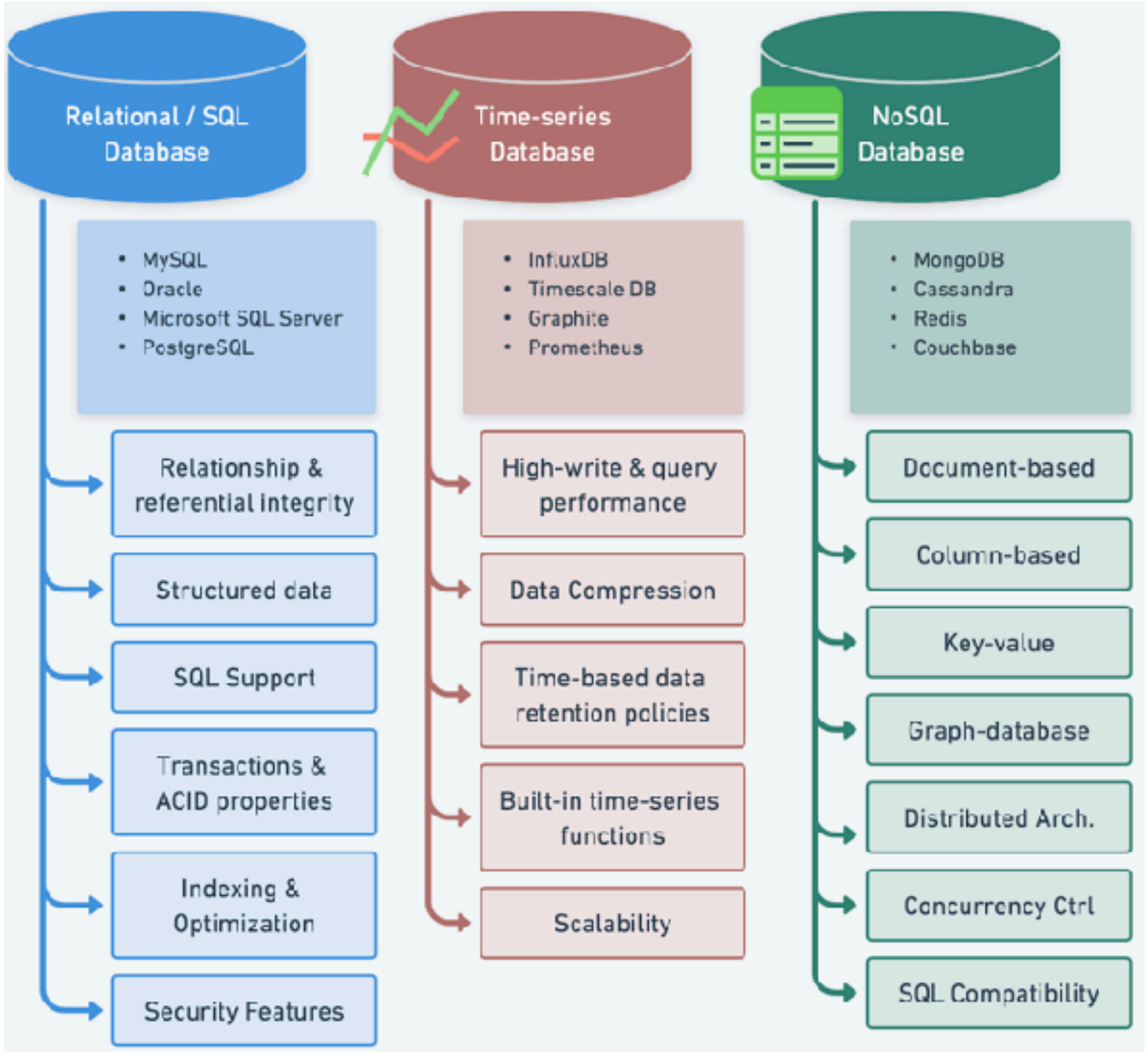
Database Ranking

424 systems in ranking, February 2025

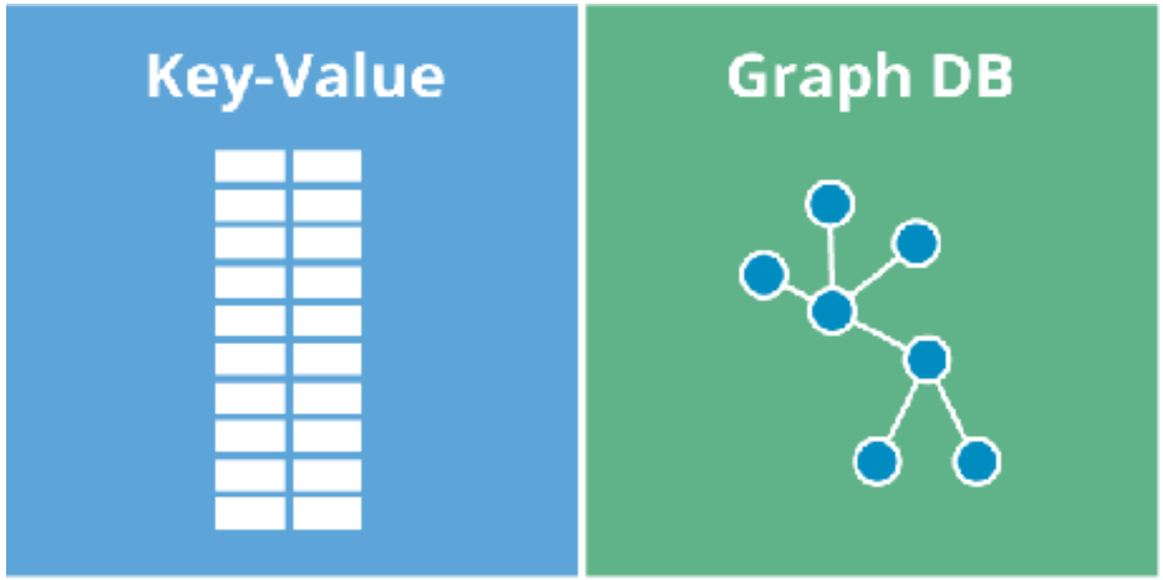
Rank Feb 2025	Rank Jan 2025	Rank Feb 2024	DBMS	Database Model	Score Feb 2025	Score Jan 2025	Score Feb 2024
1.	1.	1.	Oracle	Relational, Multi-model	1254.82	-3.93	+13.38
2.	2.	2.	MySQL	Relational, Multi-model	999.99	+1.84	-106.67
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	786.87	-11.69	-66.70
4.	4.	4.	PostgreSQL	Relational, Multi-model	659.62	-3.79	+30.21
5.	5.	5.	MongoDB	Document, Multi-model	396.63	-5.87	-23.73
6.	↑ 7.	6.	Redis	Key-value, Multi-model	157.91	+4.55	-2.80
7.	↓ 6.	↑ 9.	Snowflake	Relational	155.58	+1.68	+28.13
8.	8.	↓ 7.	Elasticsearch	Multi-model	134.63	-0.29	-1.11
9.	9.	↓ 8.	IBM Db2	Relational, Multi-model	125.44	+2.46	-6.79
10.	10.	10.	SQLite	Relational	113.82	+7.13	-3.47
11.	11.	↑ 12.	Apache Cassandra	Wide column, Multi-model	102.58	+3.39	-6.69
12.	12.	↓ 11.	Microsoft Access	Relational	96.54	+3.84	-16.63
13.	13.	↑ 17.	Databricks	Multi-model	90.03	+2.19	+13.13
14.	14.	↓ 13.	MariaDB	Relational, Multi-model	89.50	+3.92	-7.73
15.	15.	↓ 14.	Splunk	Search engine	80.56	-2.53	-11.09

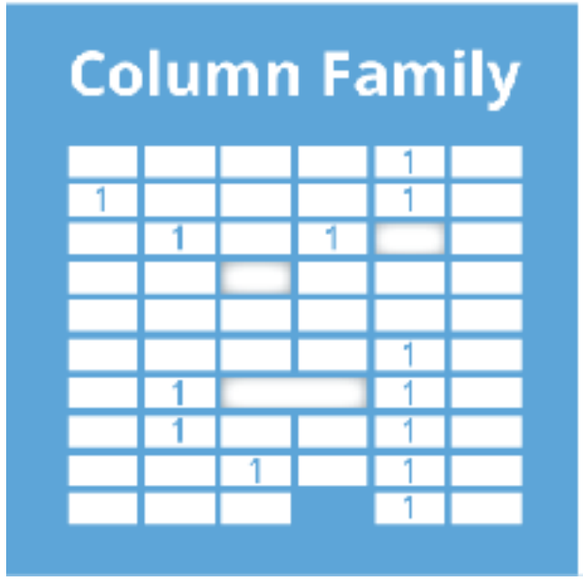
https://db-engines.com/en/ranking

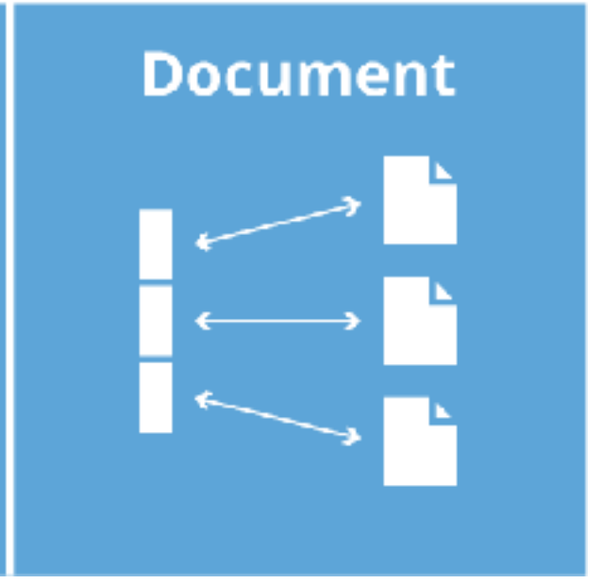
Type of databases



NoSQL database

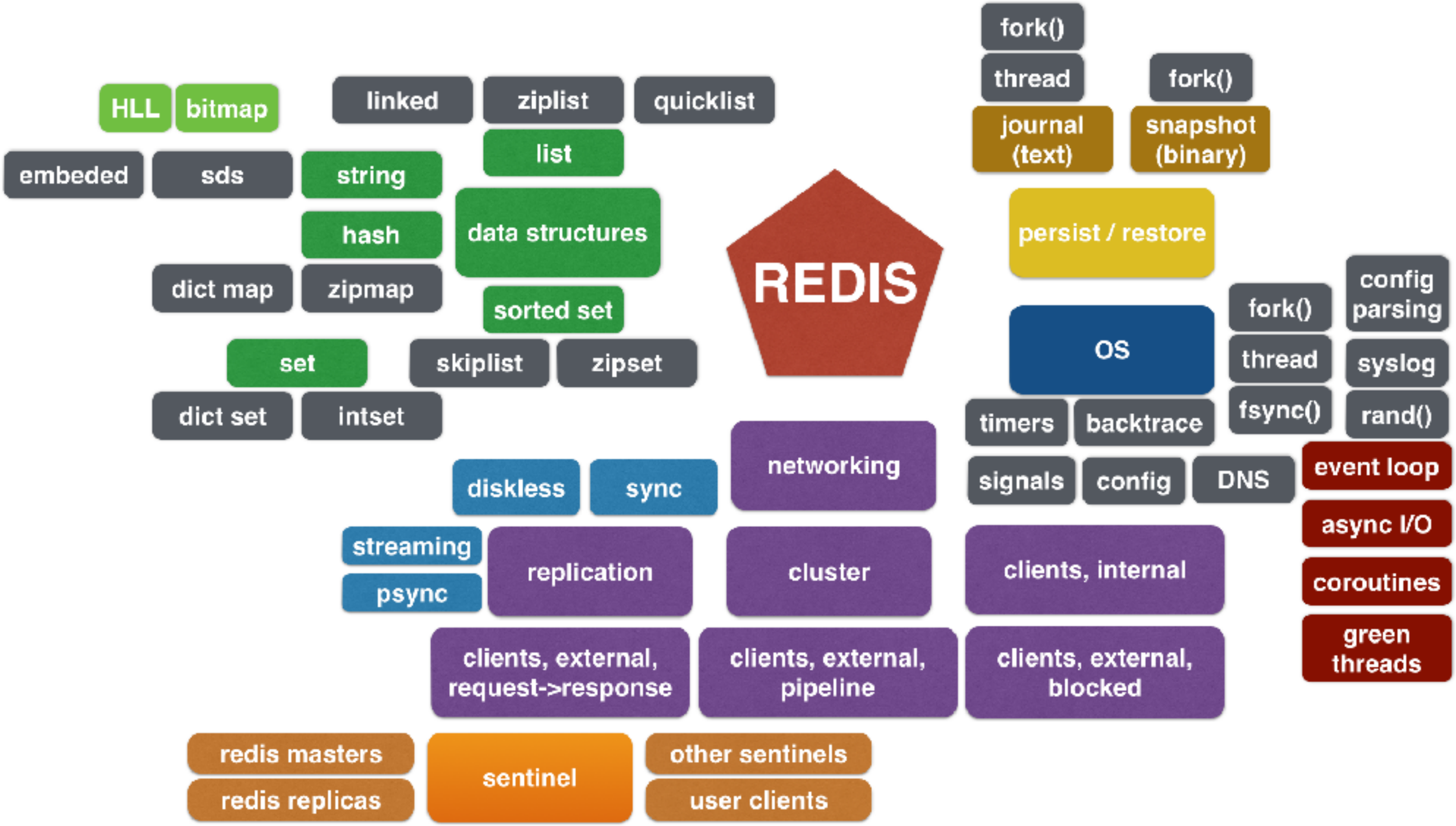






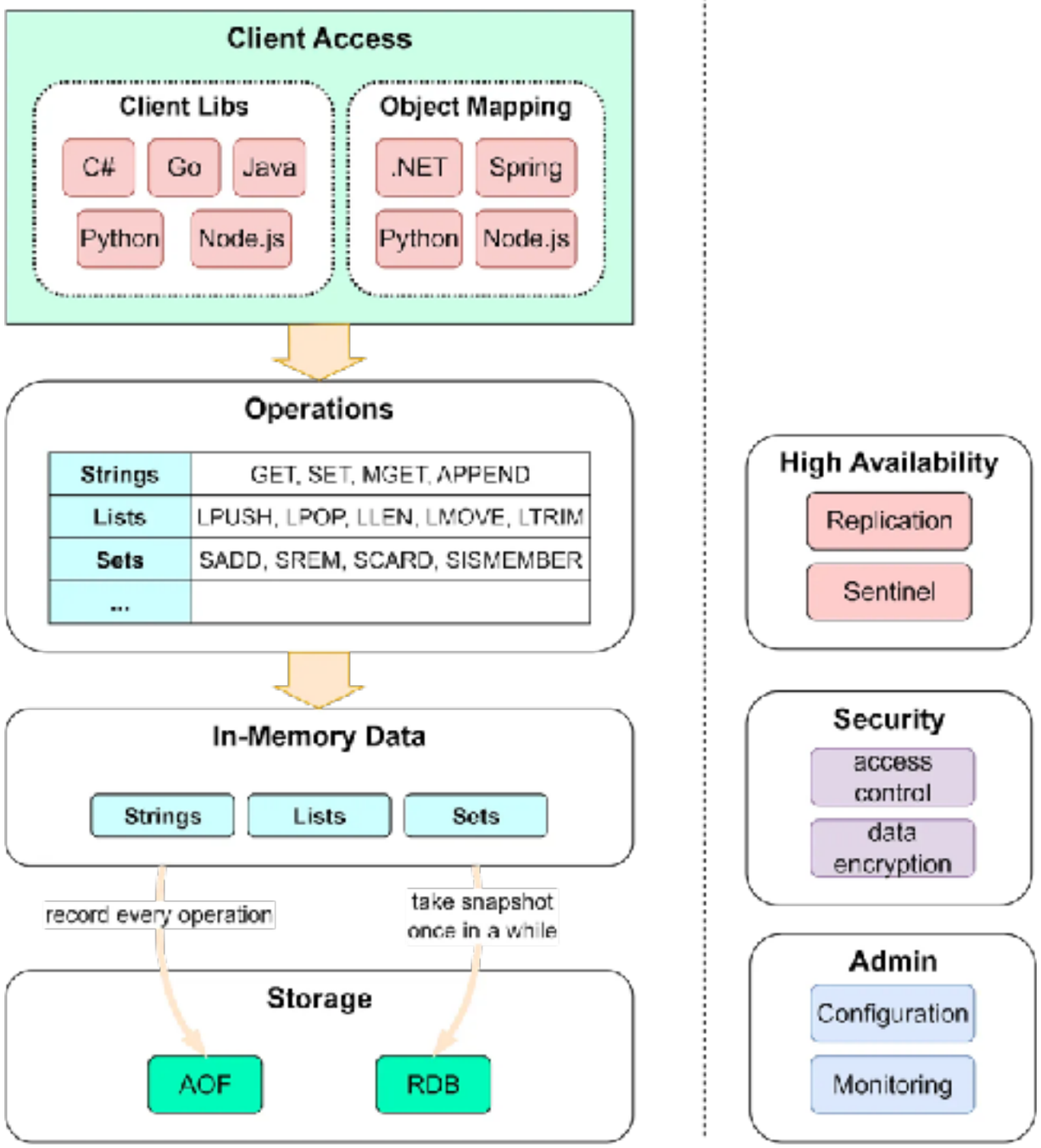
Key-value databases

Learn Redis ?



High level architecture

High level view of Reds



https://blog.bytebytego.com/p/a-crash-course-in-redis

Why Redis Fast ?

Why Redis Fast ?

RAM-based database

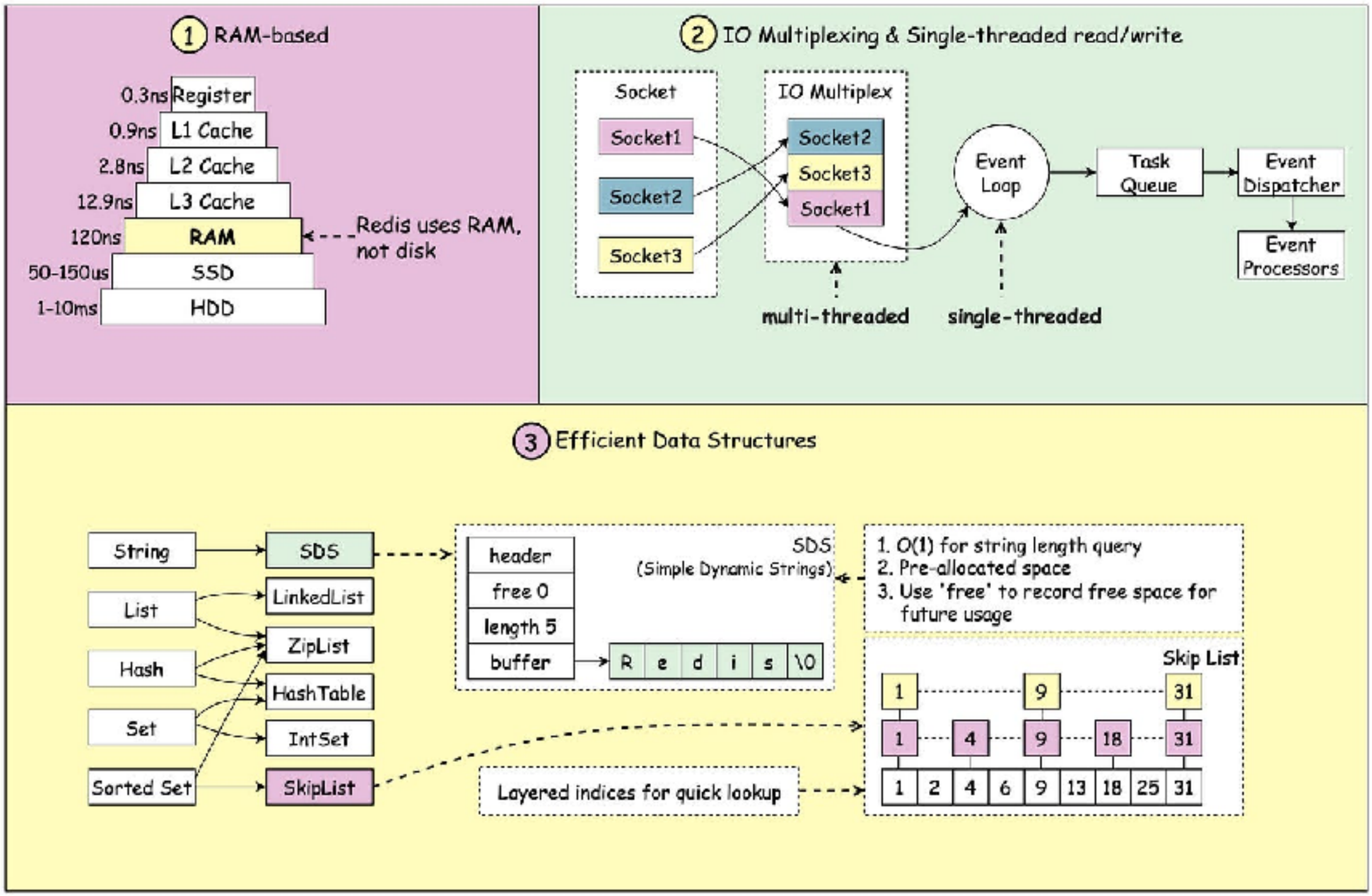
I/O multiplexing and single-thread read/write

Efficient low-level data structure

Why is Redis so fast?

blog.bytebytego.com

100k QPS



https://blog.bytebytego.com/p/why-is-redis-so-fast

Advantage of Single thread

No context switching

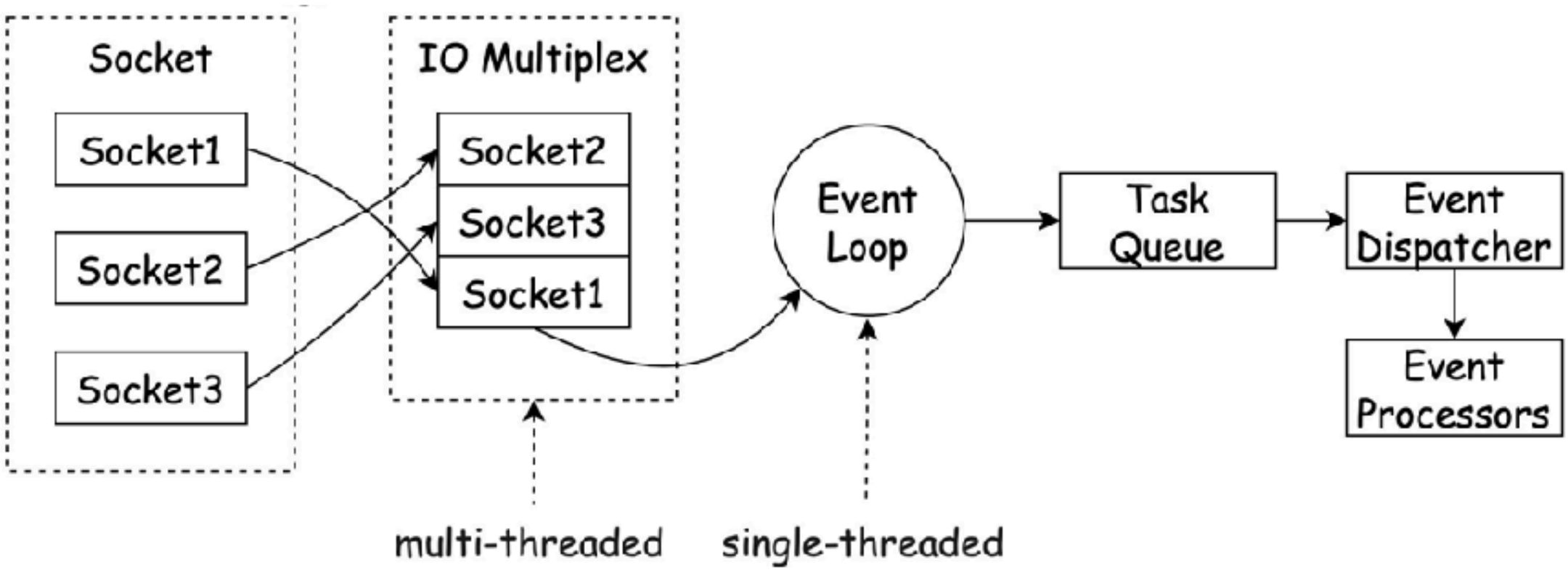
No synchronization required

Minimal overhead

Disadvantage of Single thread

Poor CPU utilization
Scalability limitation
Unresponsiveness for blocking operation

I/O multiplexing and single-thread read/write



Atomic command and Data consistency

Each command is **atomic**

Eliminate the risk of race condition

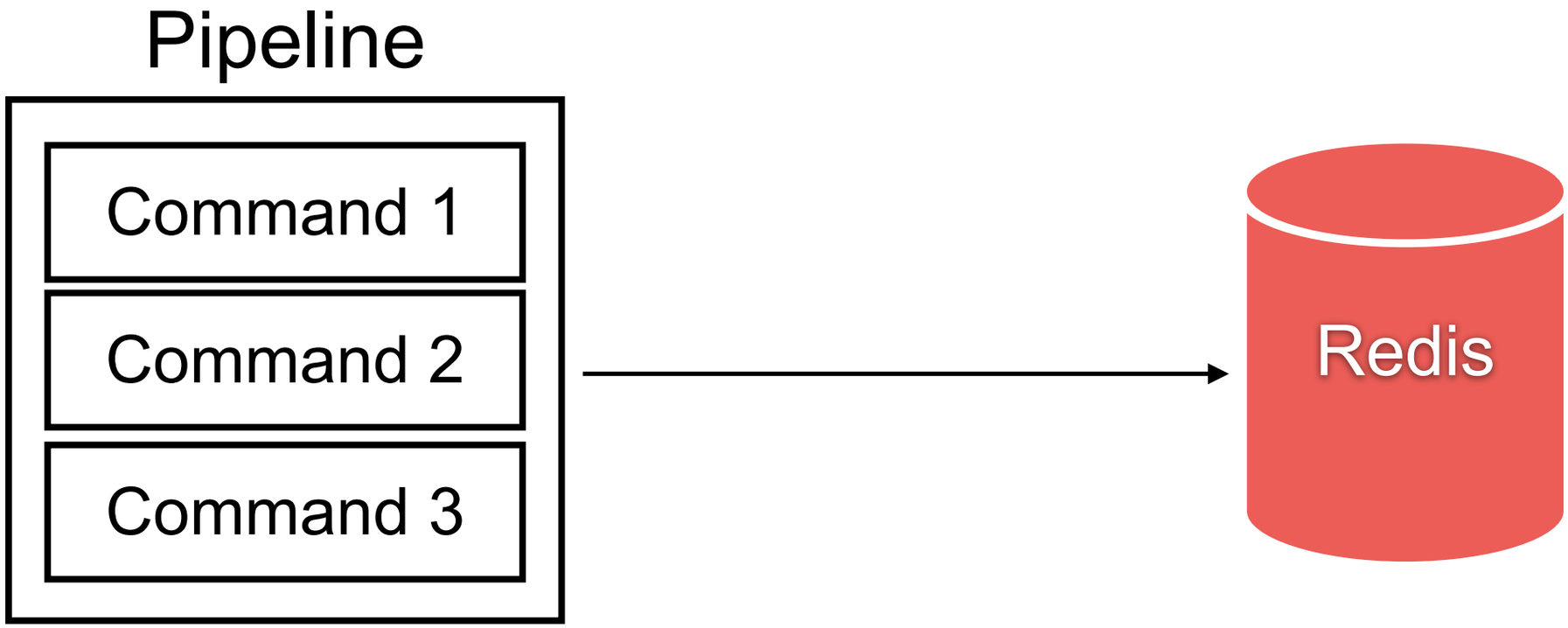
Reduce network latency

Working with pipeline in Redis

https://redis.io/docs/latest/develop/use/pipelining/

Reduce network latency

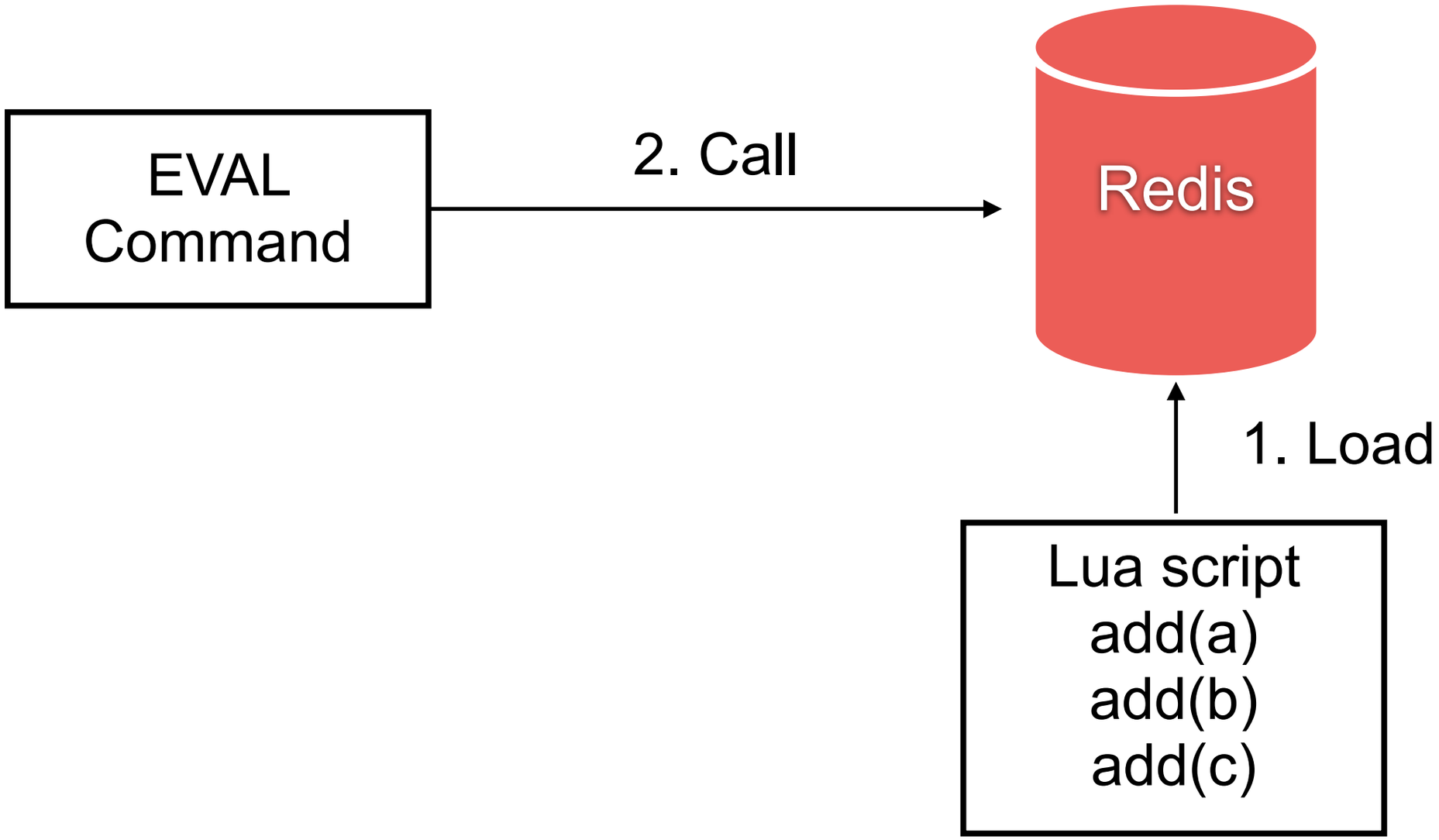
Working with pipeline in Redis
Reduce network overhead



https://redis.io/docs/latest/develop/use/pipelining/

Reduce network latency

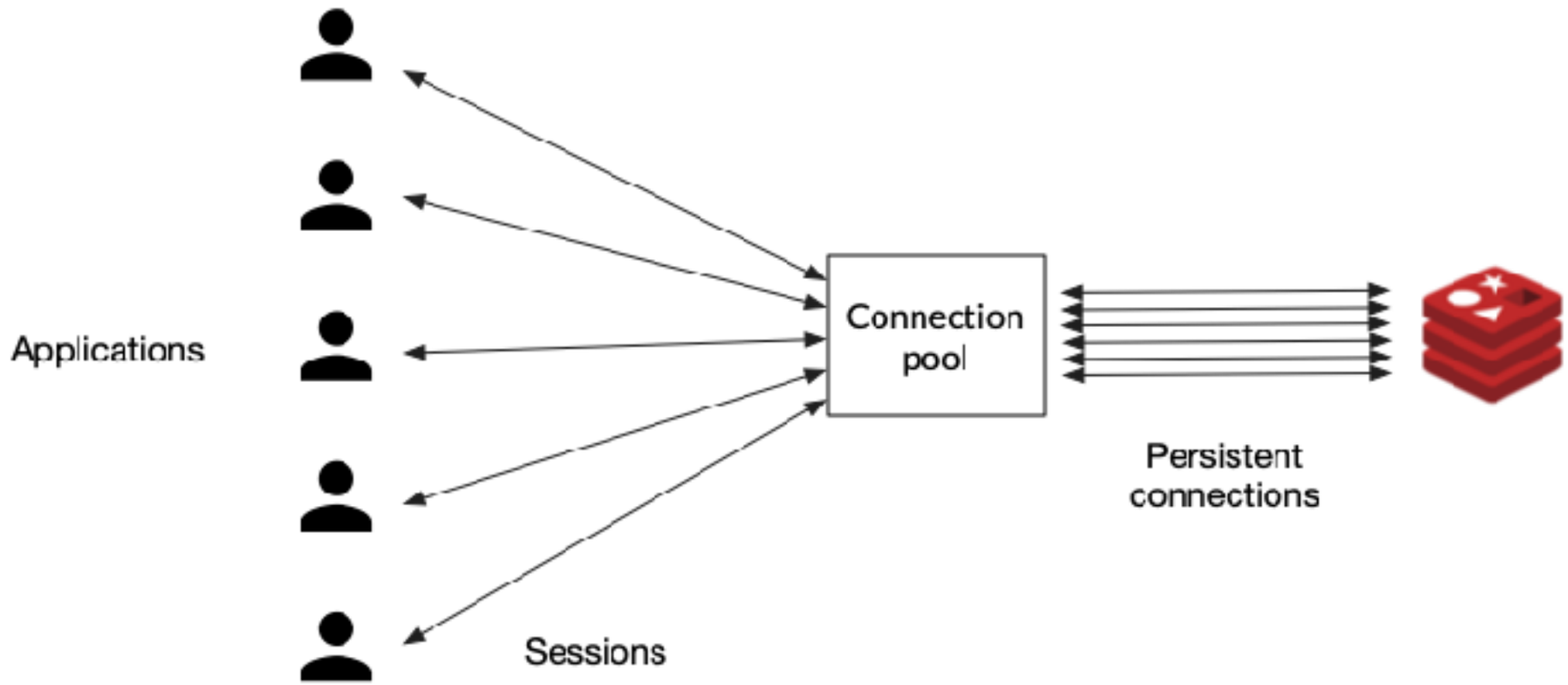
Write Lua script in Redis (like store procedure)



https://redis.io/docs/latest/develop/interact/programmability/eval-intro/

Use connection pool in client

Open and close connections cause overhead !!



Redis connection management

Unmanaged

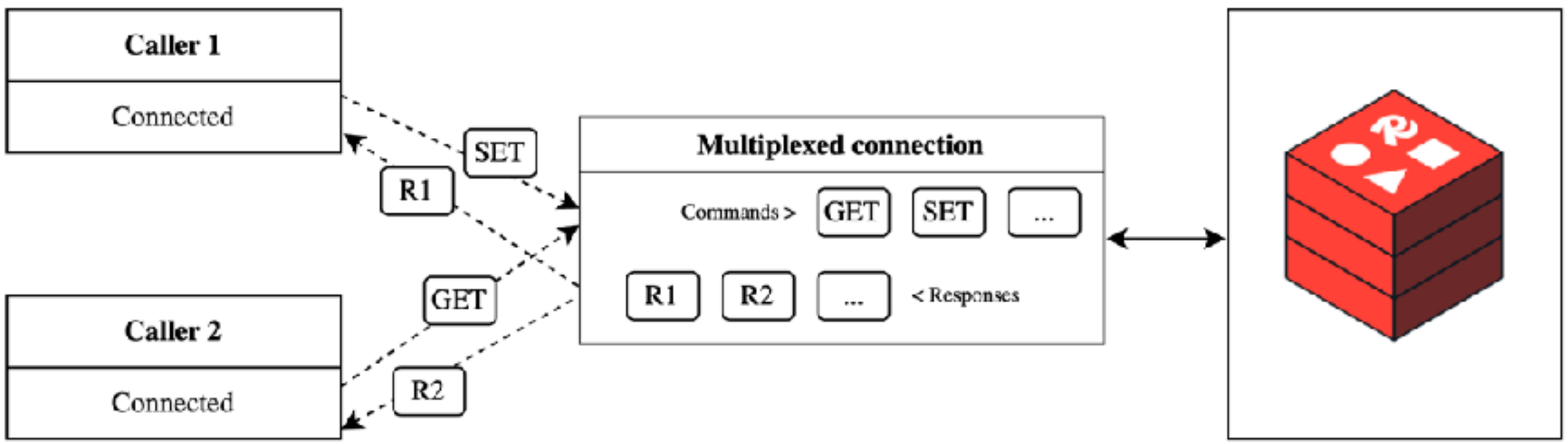
Connection Pooling

Multiplexing

https://redis.io/blog/multiplexing-explained/

Multiplexing

Keep a single connection open and use for all traffic

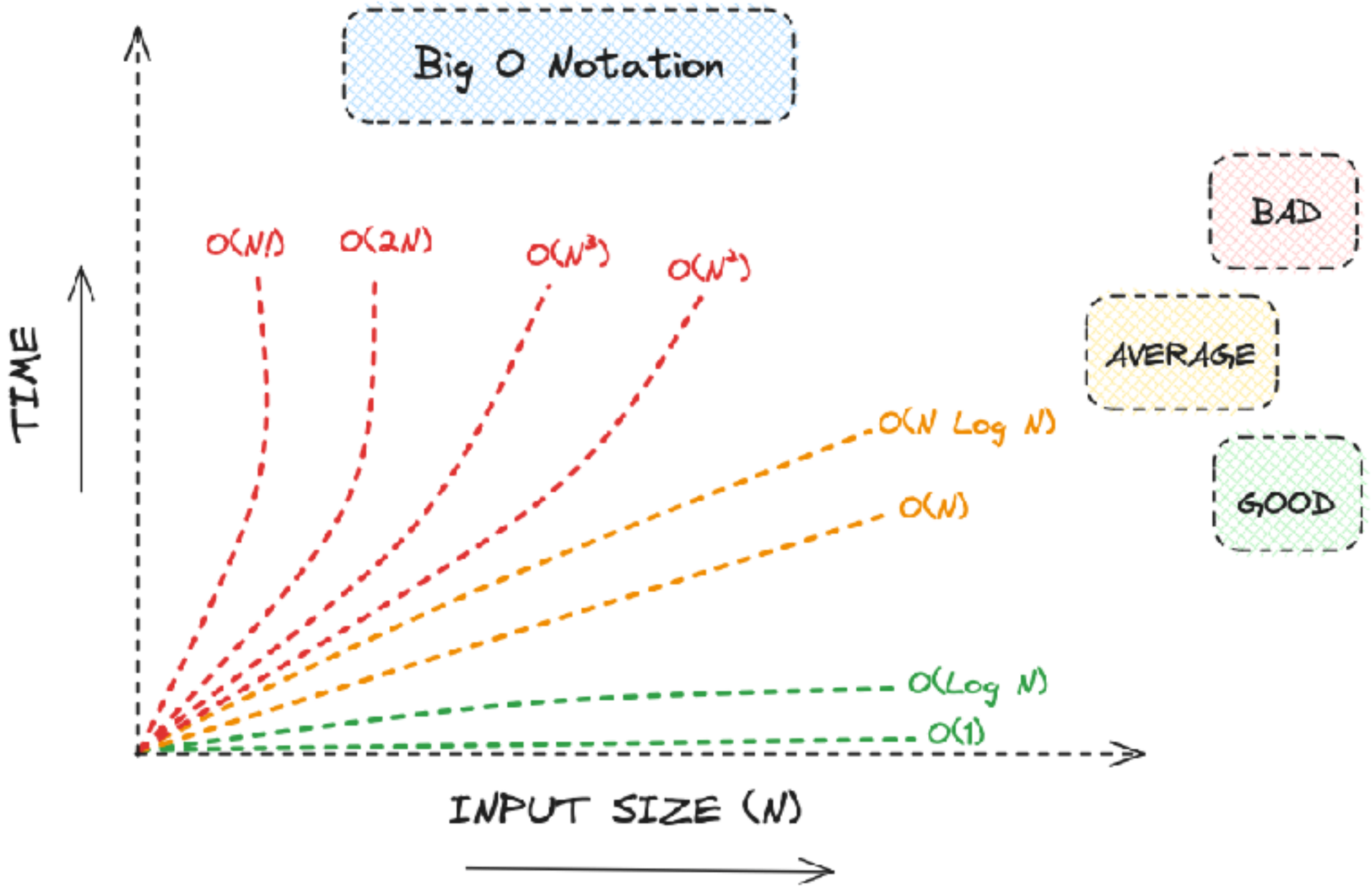


https://redis.io/docs/latest/develop/clients/pools-and-muxing/

Efficient low-level data structure

https://redis.io/technology/data-structures/

Big O Notation

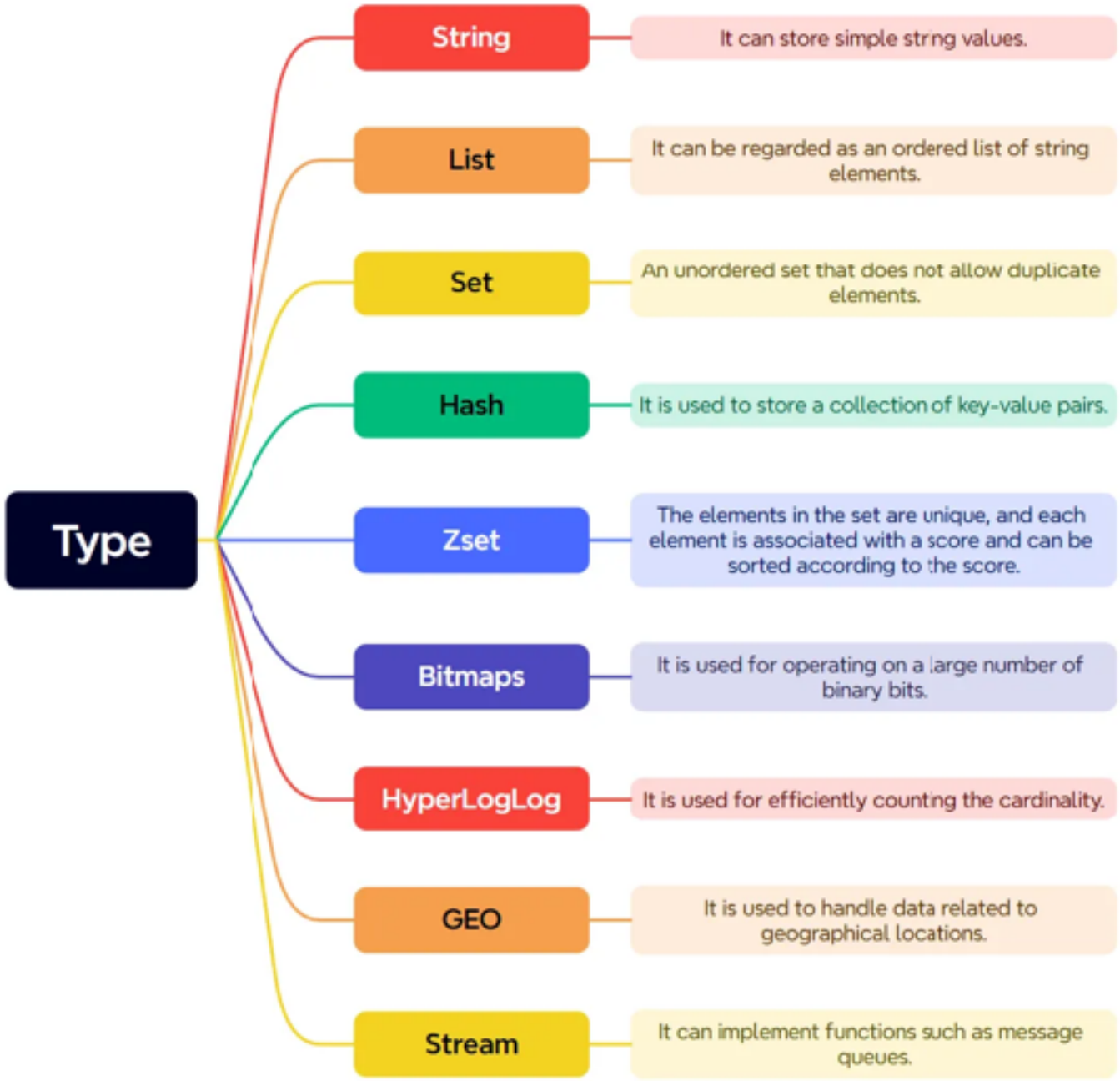


https://www.linkedin.com/pulse/big-o-notation-critical-tool-modern-software-micael-santana-8uaxf/

Data Structure in Redis

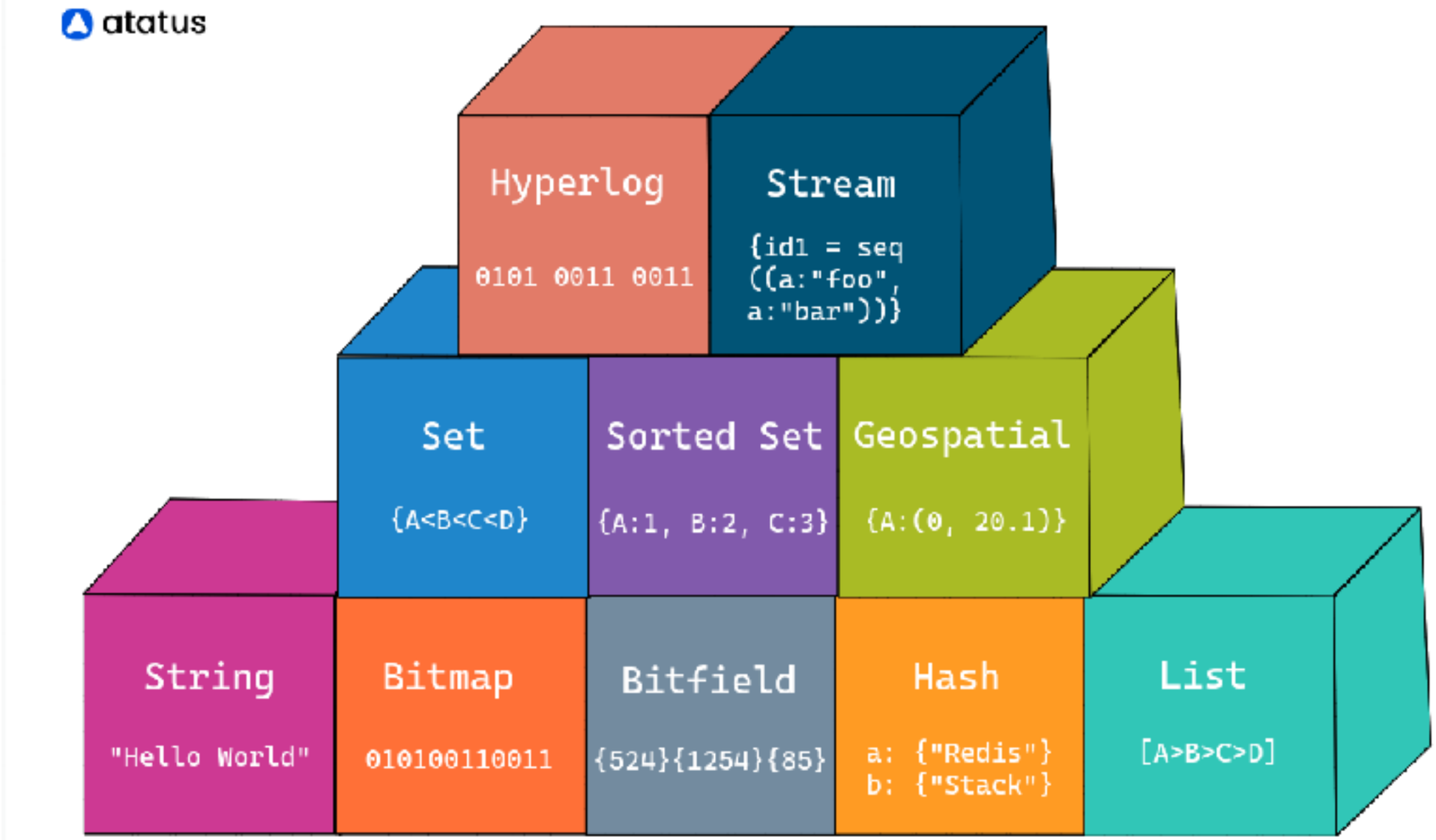
Basic Data Structures	Strings	Used for cache, counter, distributed locks, sessions	GET, SET, MGET, APPEND, SUBSTR, STRLEN
	Lists	Used for message queues	LPUSH, LPOP, LLEN, LMOVE, LTRIM
	Sets	Used for intersections, unions etc	SADD, SREM, SCARD, SISMEMBER, SINTER
	Hashes	Used for caches	HGET, HSET, HMGET, HINCRBY
	Sorted Sets (ZSet)	Used for ranking	ZADD, ZRANGE, ZREVRANGE, ZRANGEBYSCORE, ZREMRANGEBYSCORE, ZRANK
Added After Redis 2.2	Streams	Append only log for event sourcing, sensor monitoring etc	
	Geospatial	Store and query coordinates	
	Bitmaps	User daily login status	
Added Later - Probabilistic Data Structures for **Big Data**	HyperLogLog	Cardinality for big data	
	Bloom Filter	Check presence of an element in a set	
	Cuckoo Filter	Check presence of an element in a set	
	t-digest	Estimate the percentile of a data stream	
	Top-k	Find the most frequent items in a data stream	
	Count-Min Sketch	Estimate the frequency of an element in a data stream	

Data Structure in Redis



https://redis.io/docs/latest/develop/data-types/

Data Structure in Redis



https://www.atatus.com/blog/redis-tutorial-exploring-data-types-architecture-and-key-features/

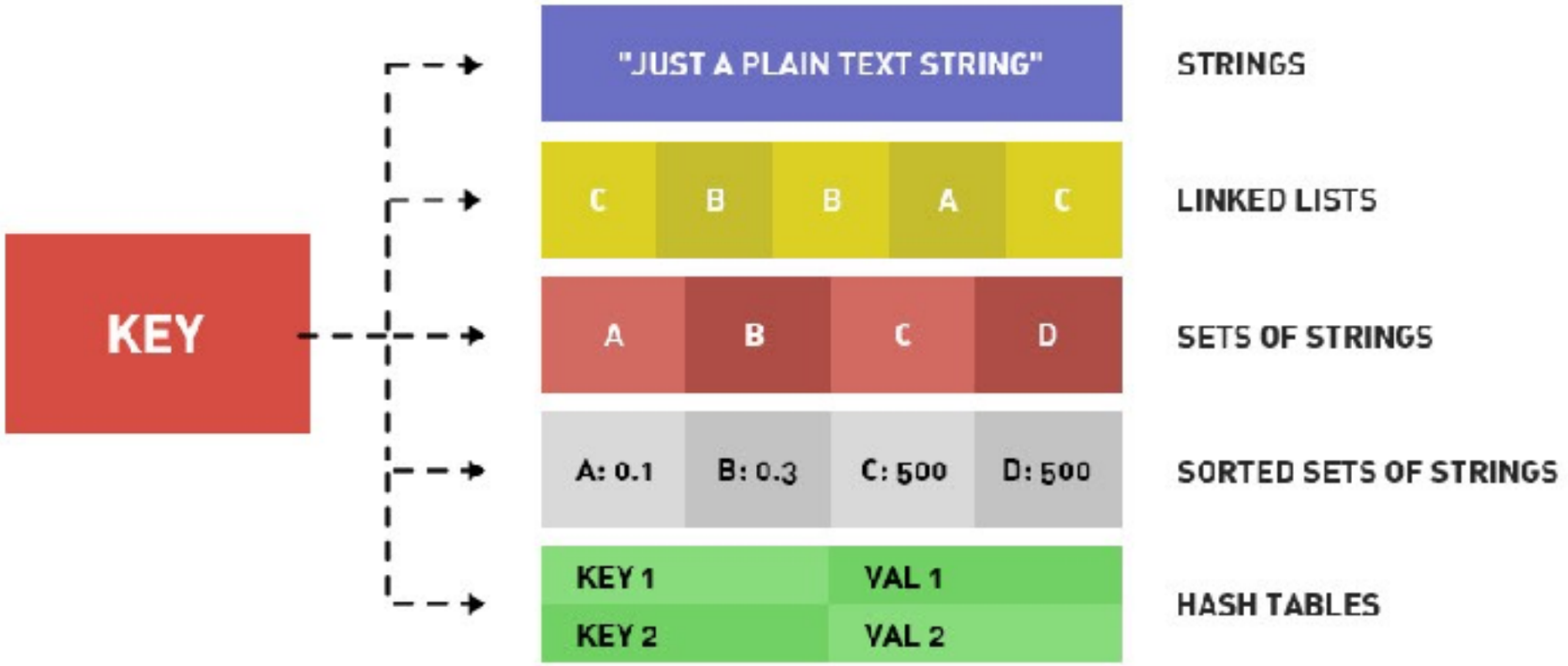
More ..

RedisJSON

Vector Database

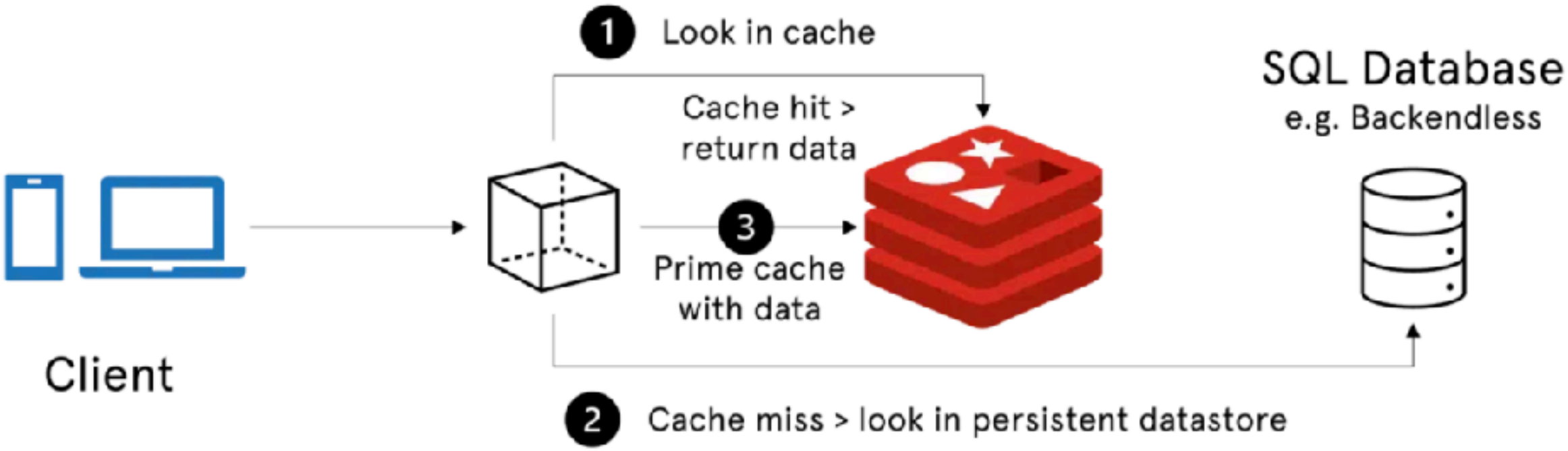
RedisSearch

Caching data ?



Simple Use Case

Caching Data for applications
Reduce workload of database



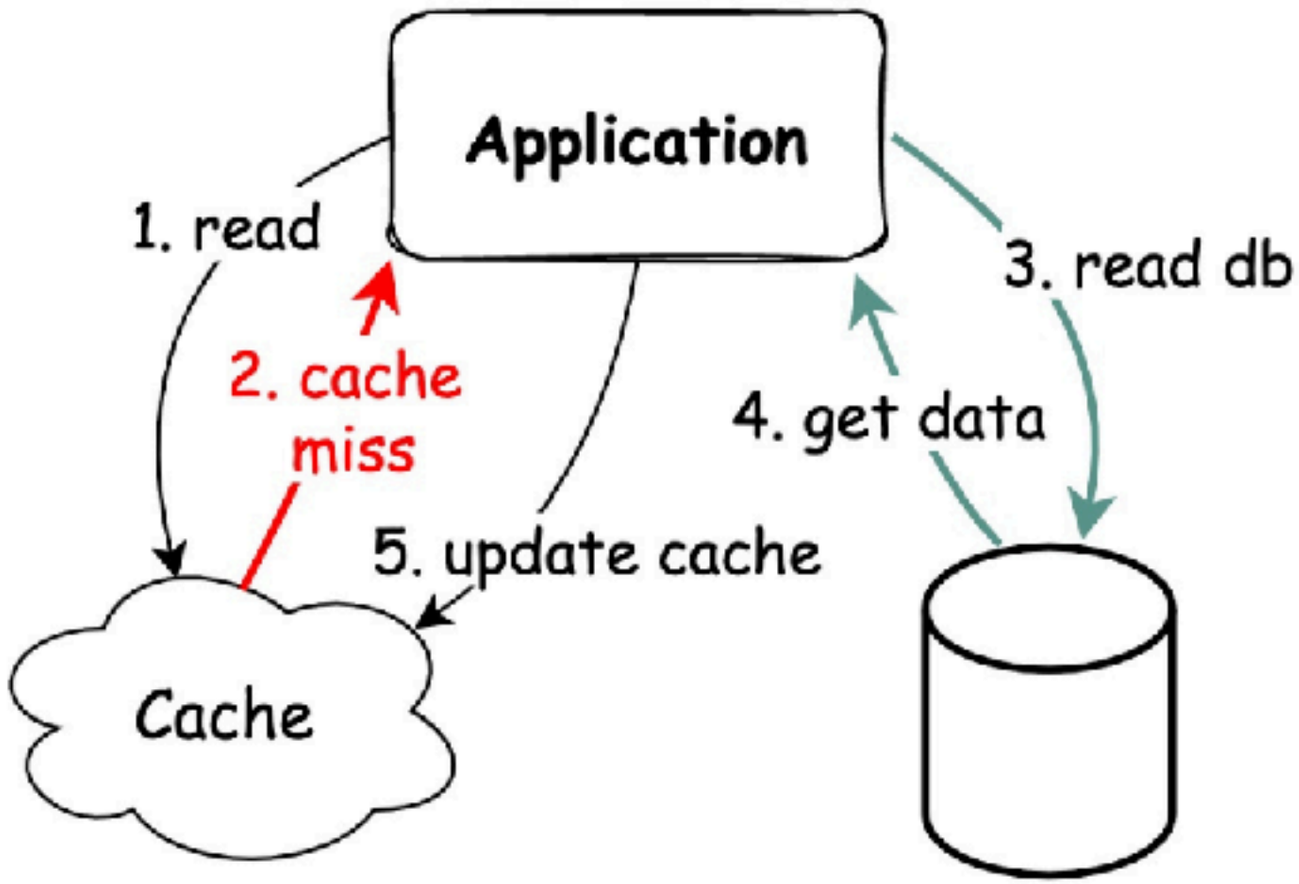
Caching Strategies ?

Read

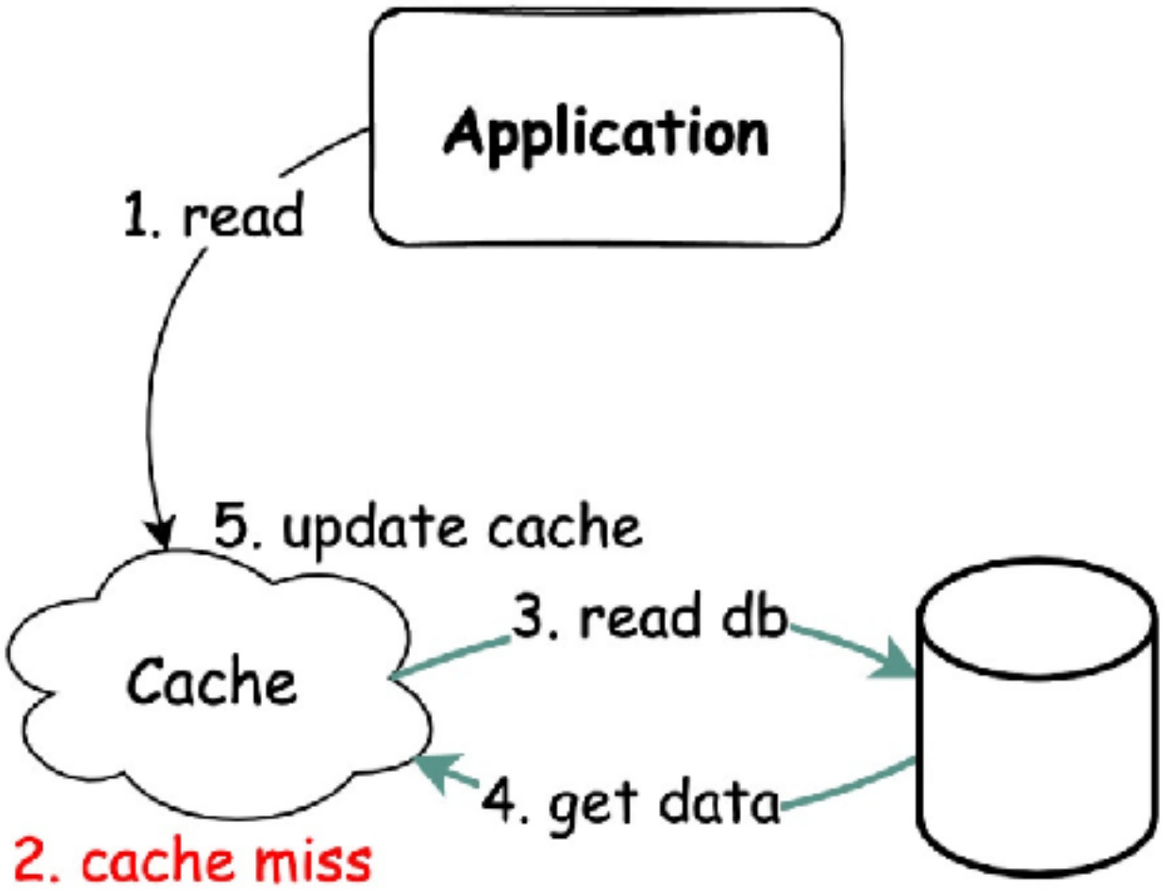
Write

https://blog.bytebytego.com/p/top-caching-strategies
https://blog.algomaster.io/p/top-5-caching-strategies-explained

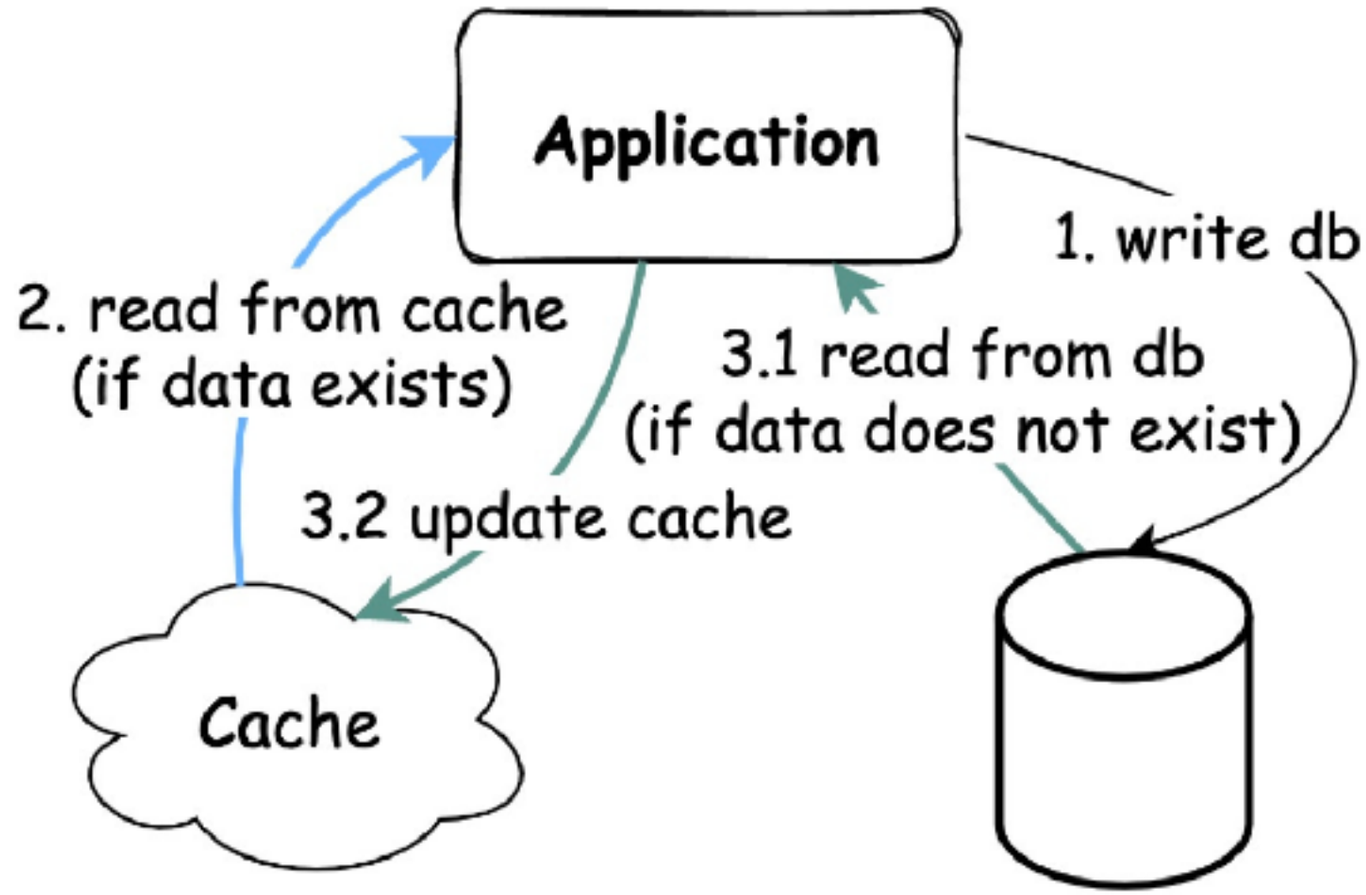
Read strategy - Cache Aside



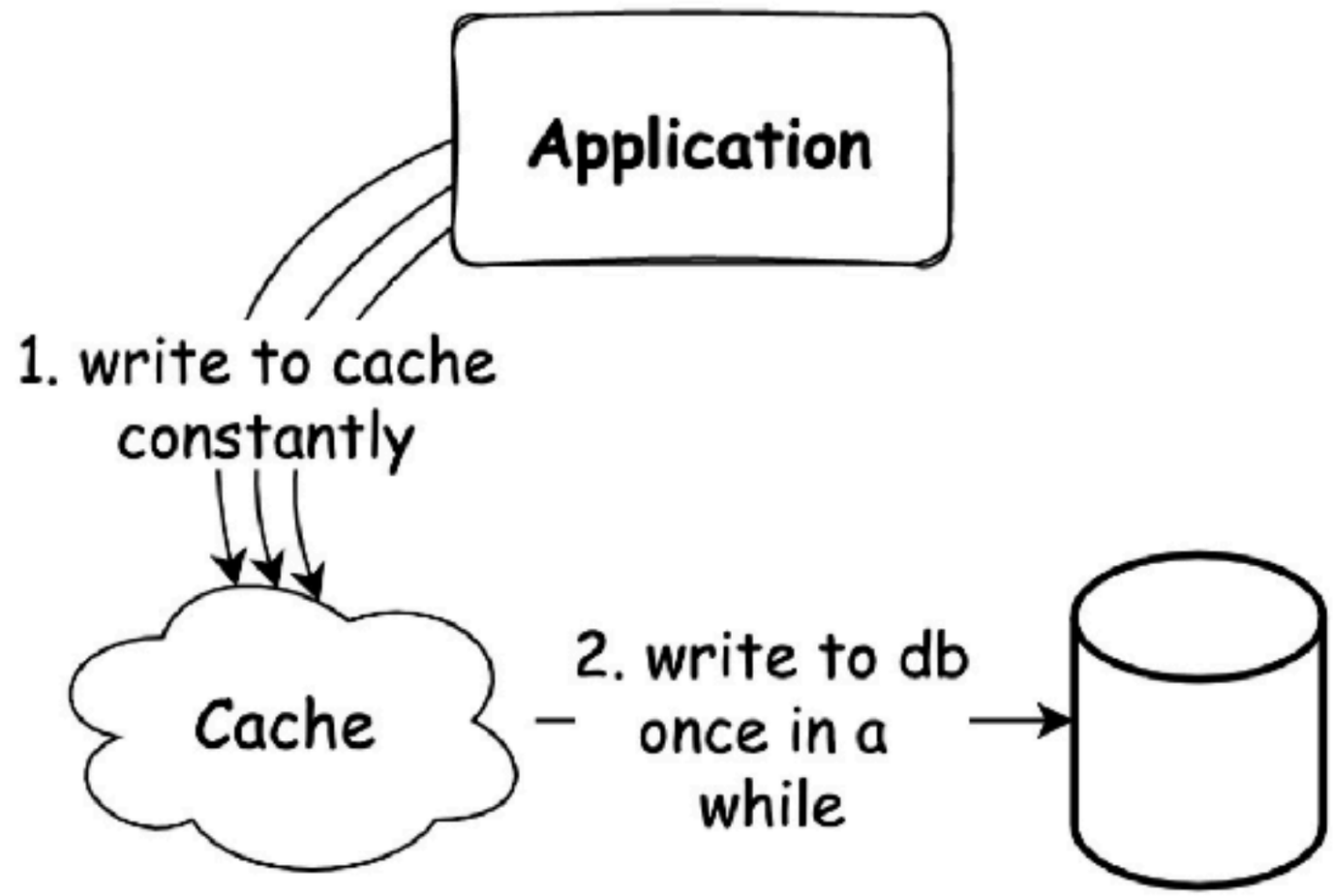
Read strategy - Read through



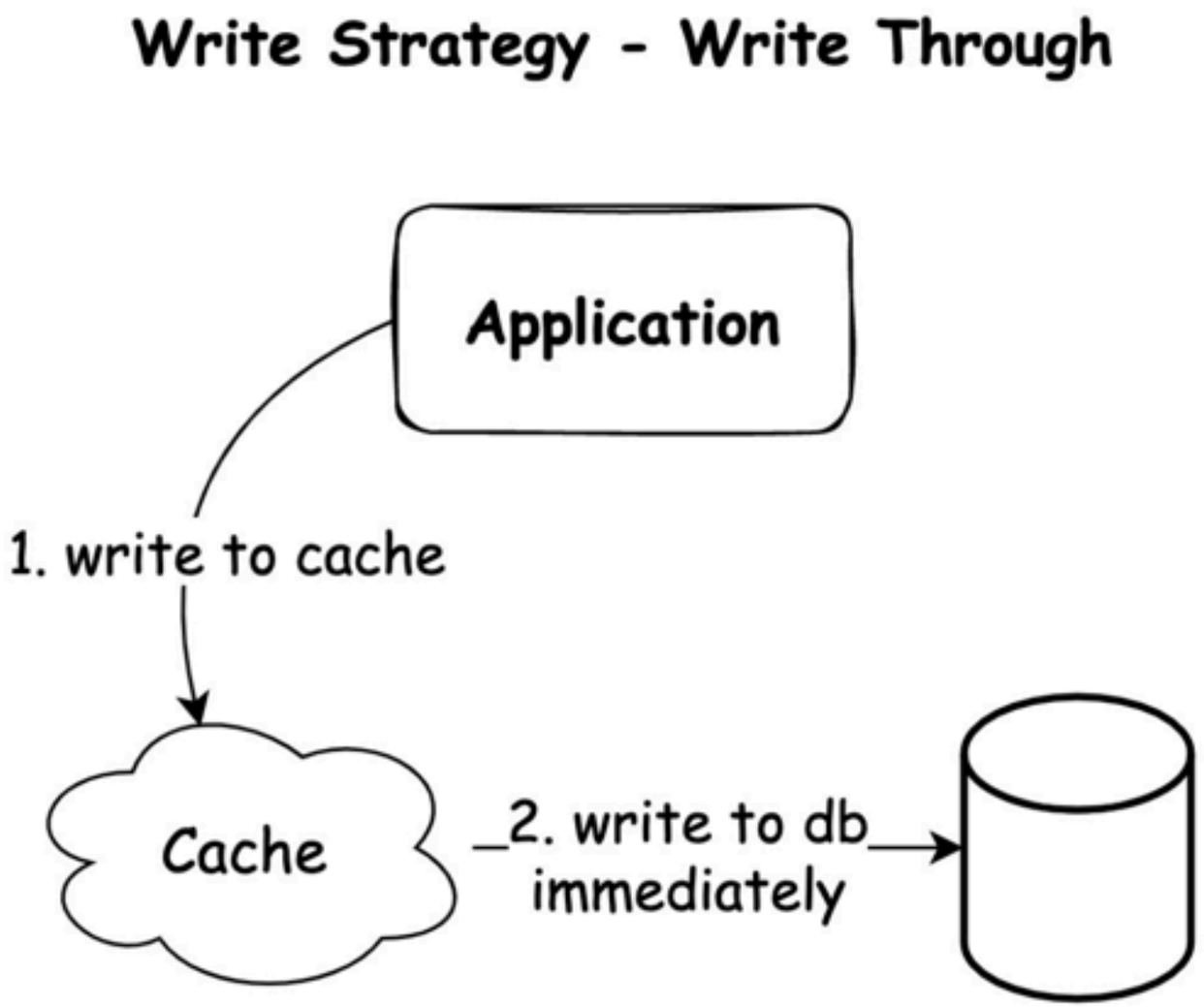
Write strategy - Write around



Write strategy - Write back



Write strategy - Write through

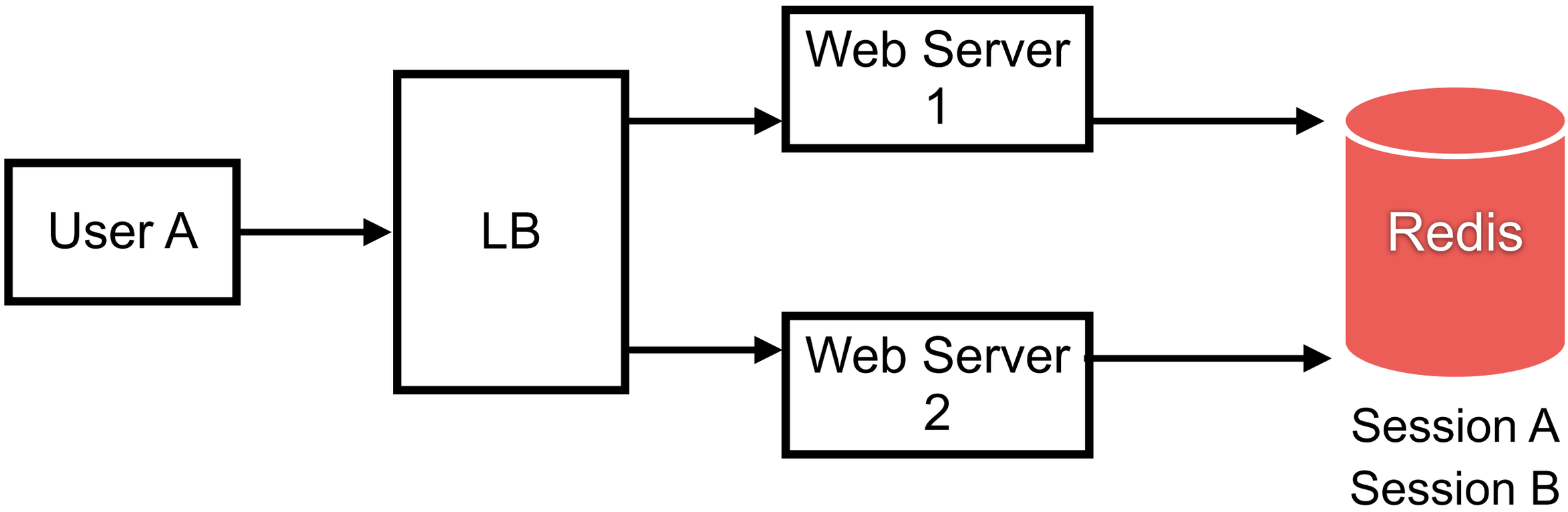


Use Cases

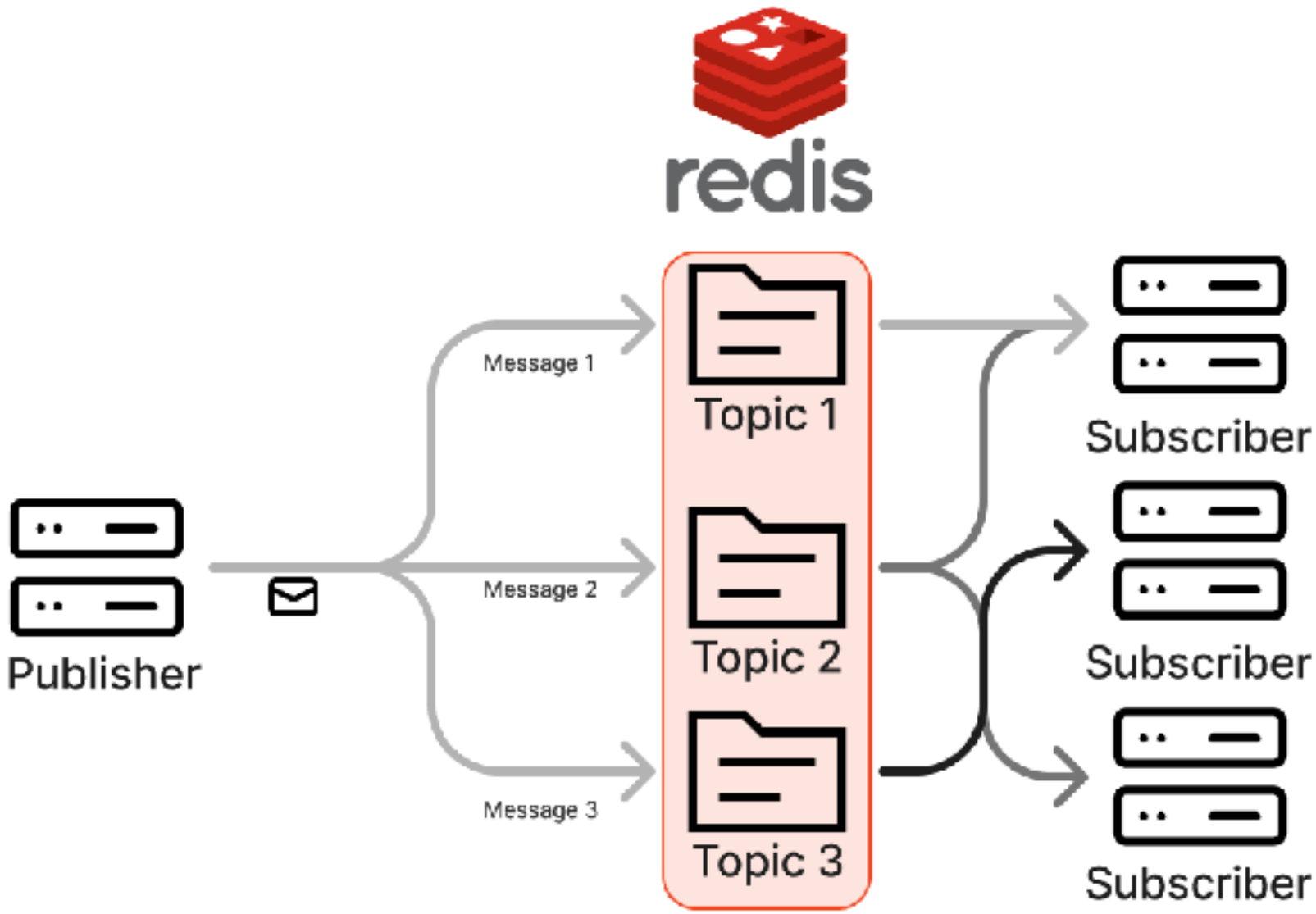
Session or Token data
Message broker (pub/sub)
Job queue
Realtime analytic
Fraud detection
Gaming

Session or Token data

Data with expire time
key=value(String)



Pub/Sub



https://redis.io/docs/latest/develop/interact/pubsub/

Sales leaderboard

BY WIN RATE | BY LEADS WON | BY VALUE WON | **BY ACTIVITIES**

Sales Reps


Enrique Romero — 46


Clayton Clark — 45


Andrew McGary — 21


Michael Francis — 19


Kurt Deruiter — 8


Tiffany Falls — 5


Brian Harvey — 5

Redis key naming !!

objectType:objectId:field

SET **user:1000:name** “Somkiat”

Data modeling with Redis

https://redis.io/blog/nosql-data-modeling/

How do I structure my data ?

Start with questions !!

Requirement
Read vs Write
User experience of your application
Data that user need (schema of data)
Size of data
Performance
Scalability

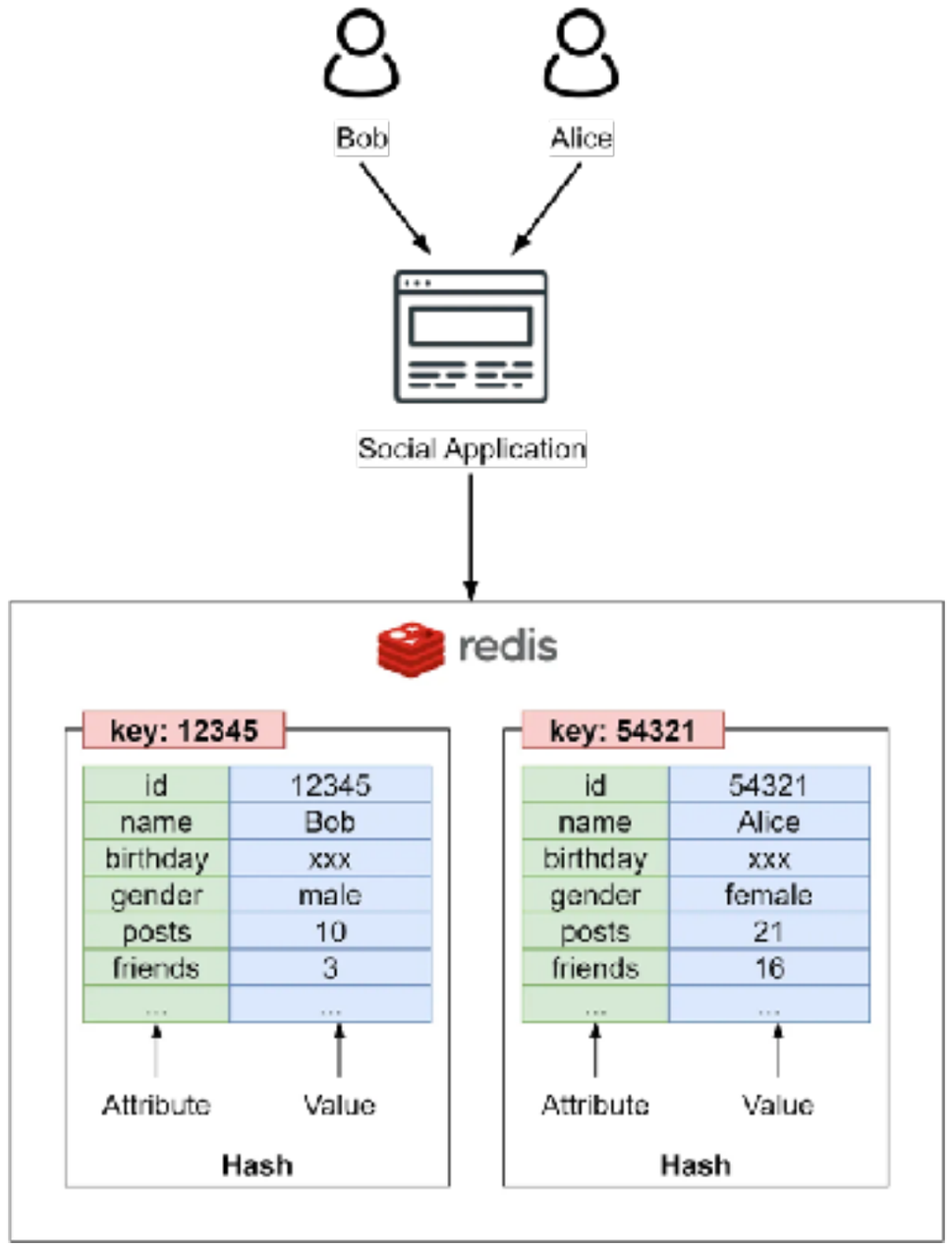
Schemaless !== Not design schema

Data modeling with Redis ?

Data structure
Relationships

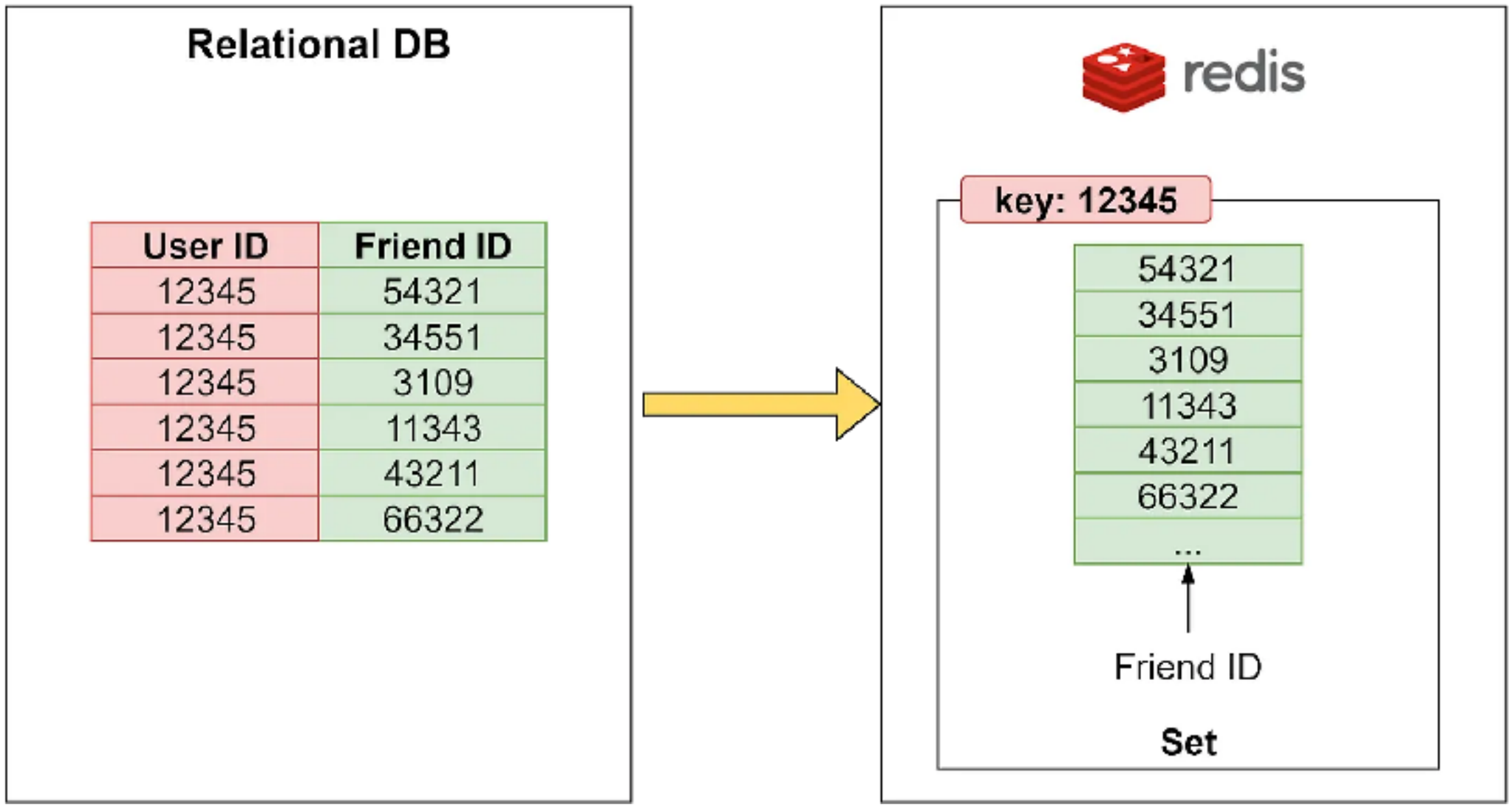
Data Structure

User profile



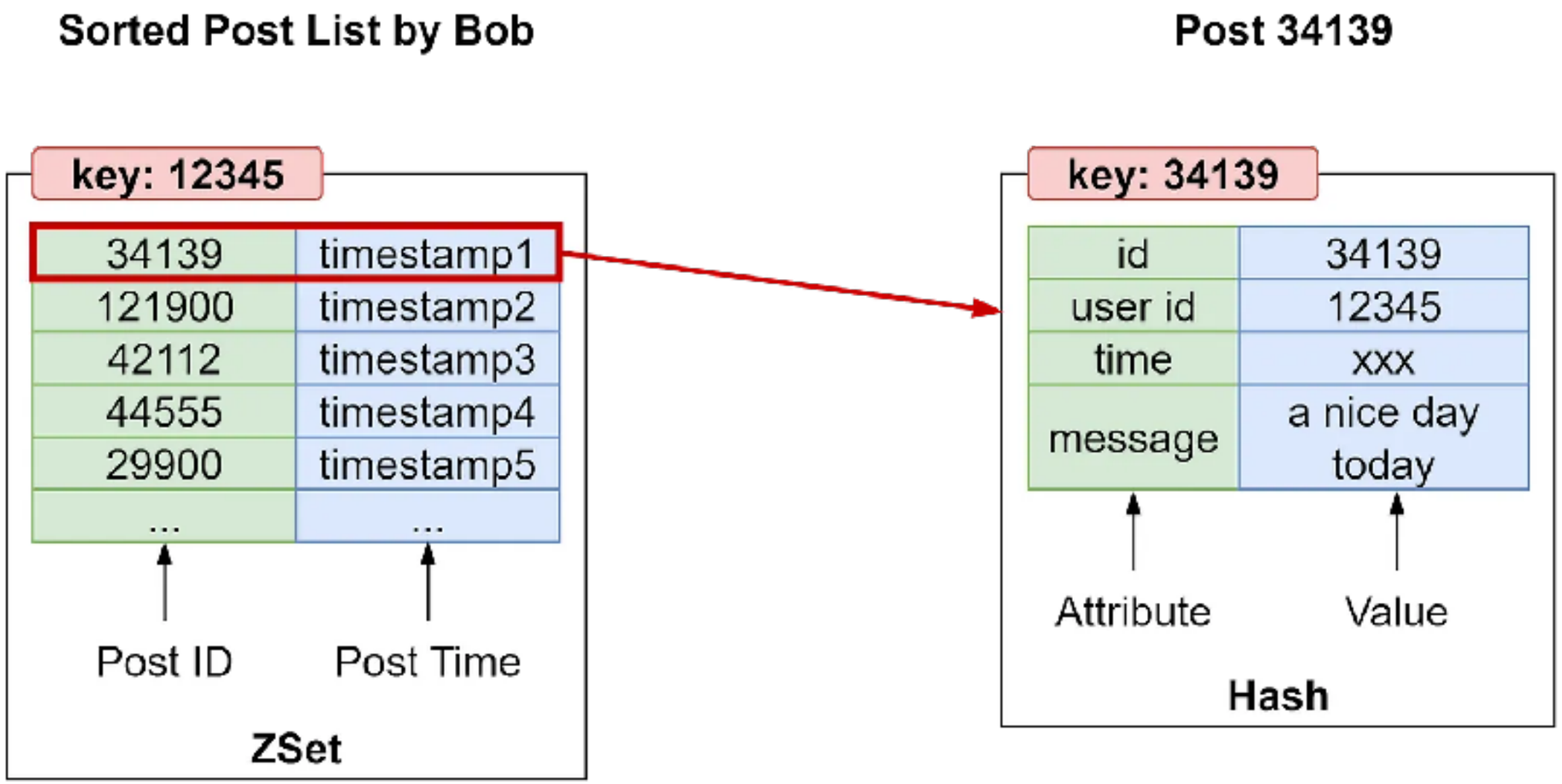
https://blog.bytebytego.com/p/redis-can-do-more-than-caching

User’s friend



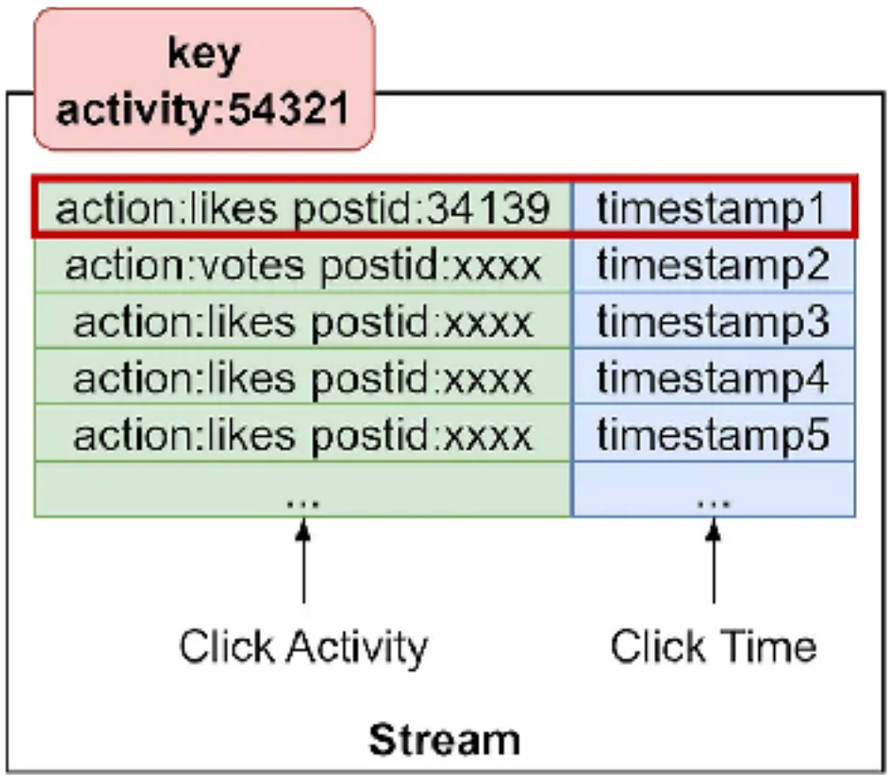
https://blog.bytebytego.com/p/redis-can-do-more-than-caching

Post or Topic in Social media

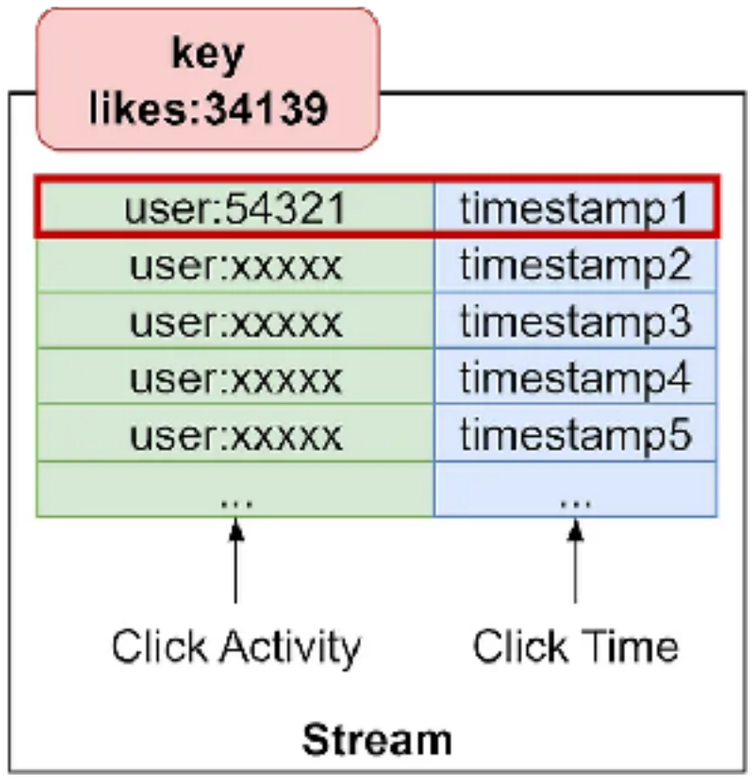


Activity in Social media

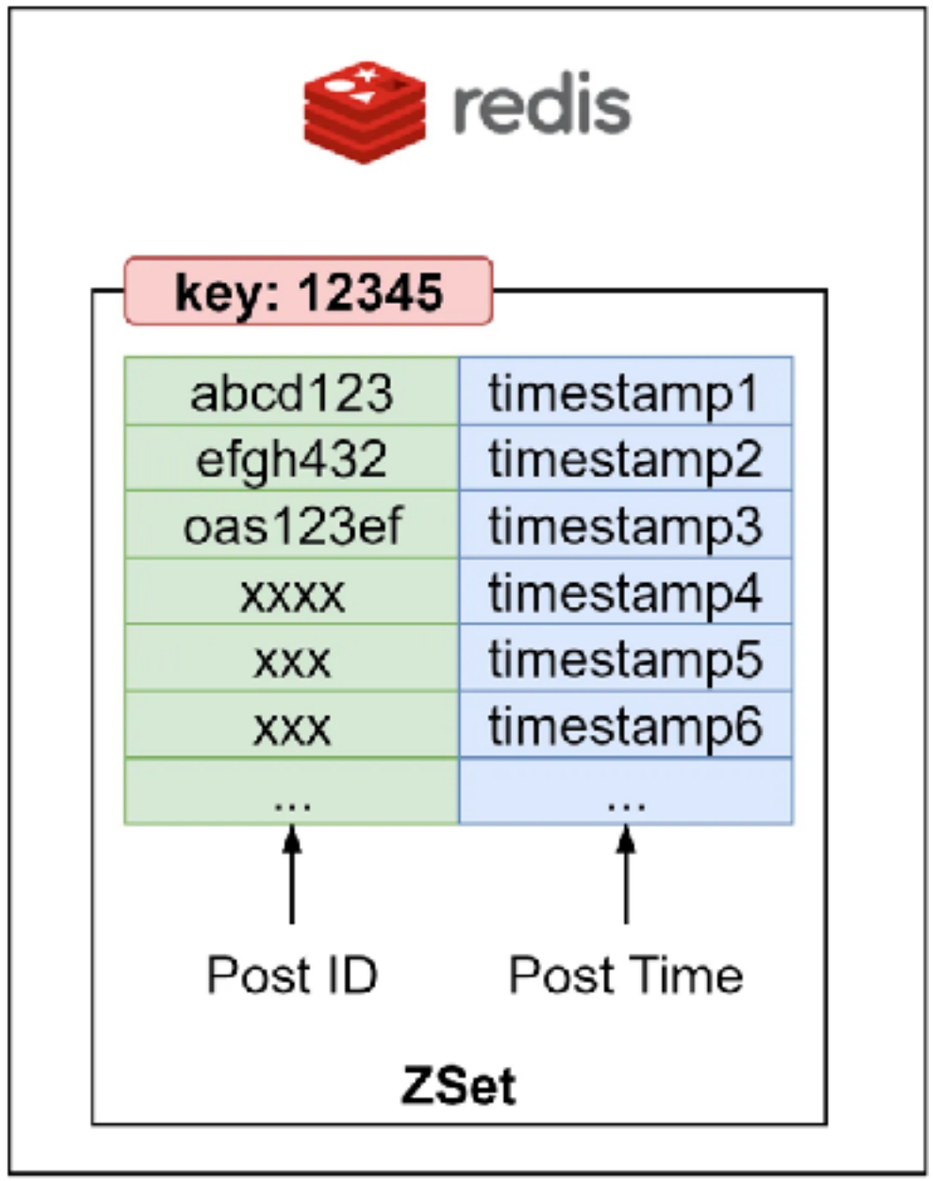
Alice's Activities



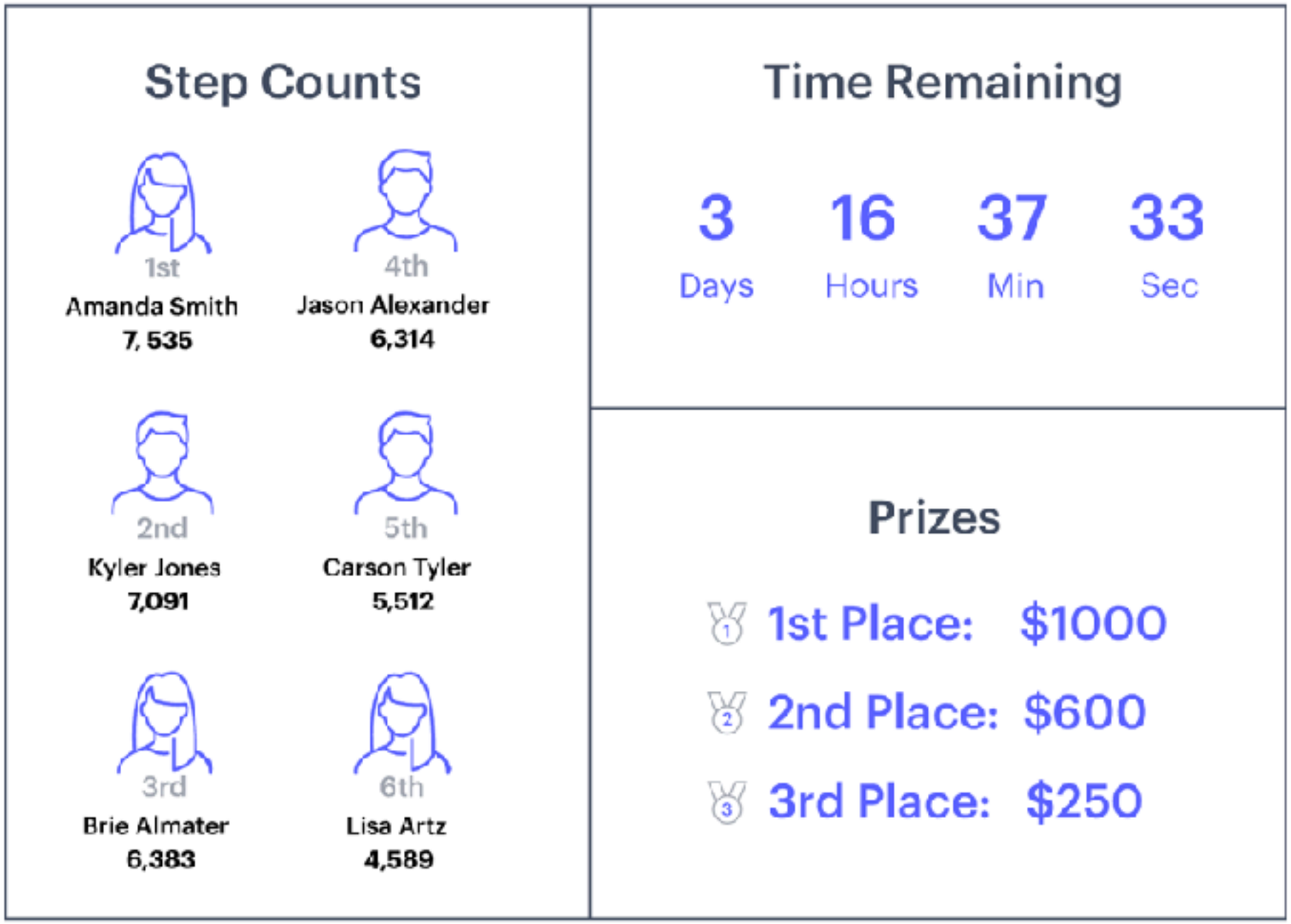
"Likes" Action on Post 34139



My feed



Leaderboard for gamification



https://redis.io/solutions/leaderboards/

More data structure

Location-based service
Tracking user state with bitmap
Probabilistic

Probabilistic

Bloom filter

Hyperloglog

Top-K

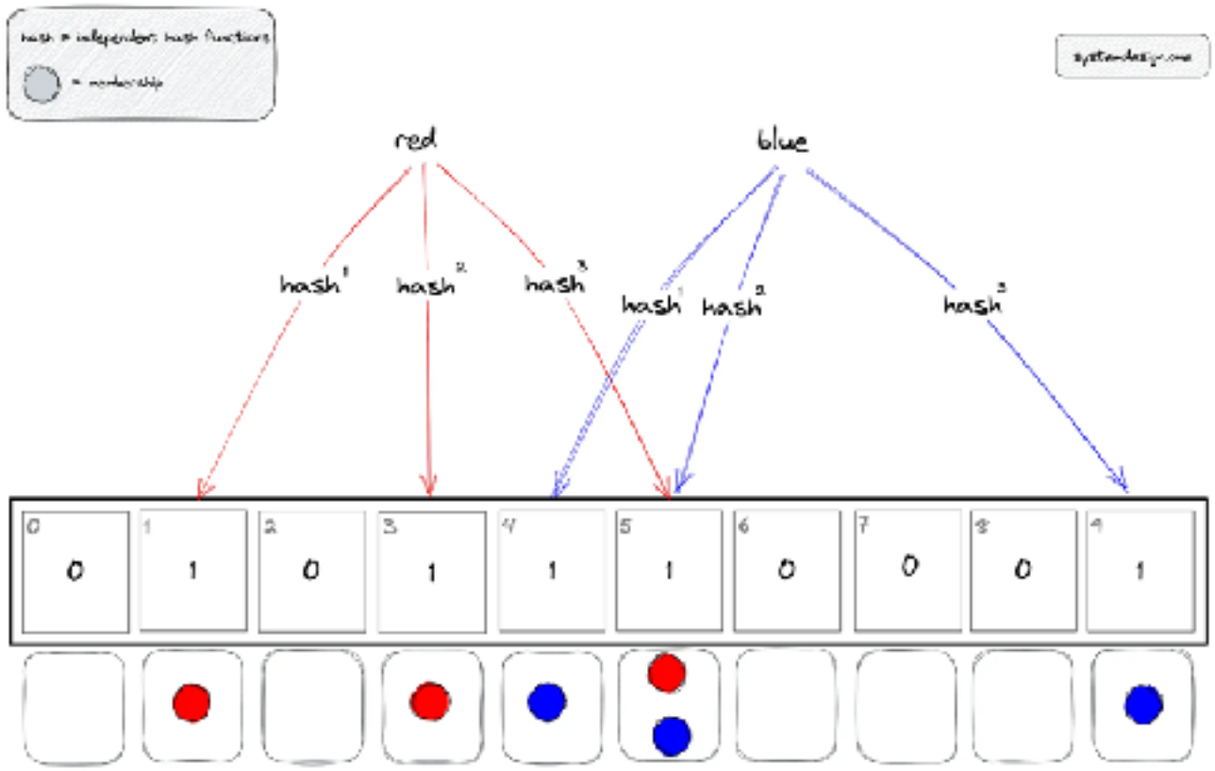
https://redis.io/docs/latest/develop/data-types/probabilistic/

Bloom filter

Check data in redis

Reduce size of data in memory (but have error rate)

Balance between memory and error rate



https://redis.io/docs/latest/develop/data-types/probabilistic/bloom-filter/

Relationships of Data

Relationships

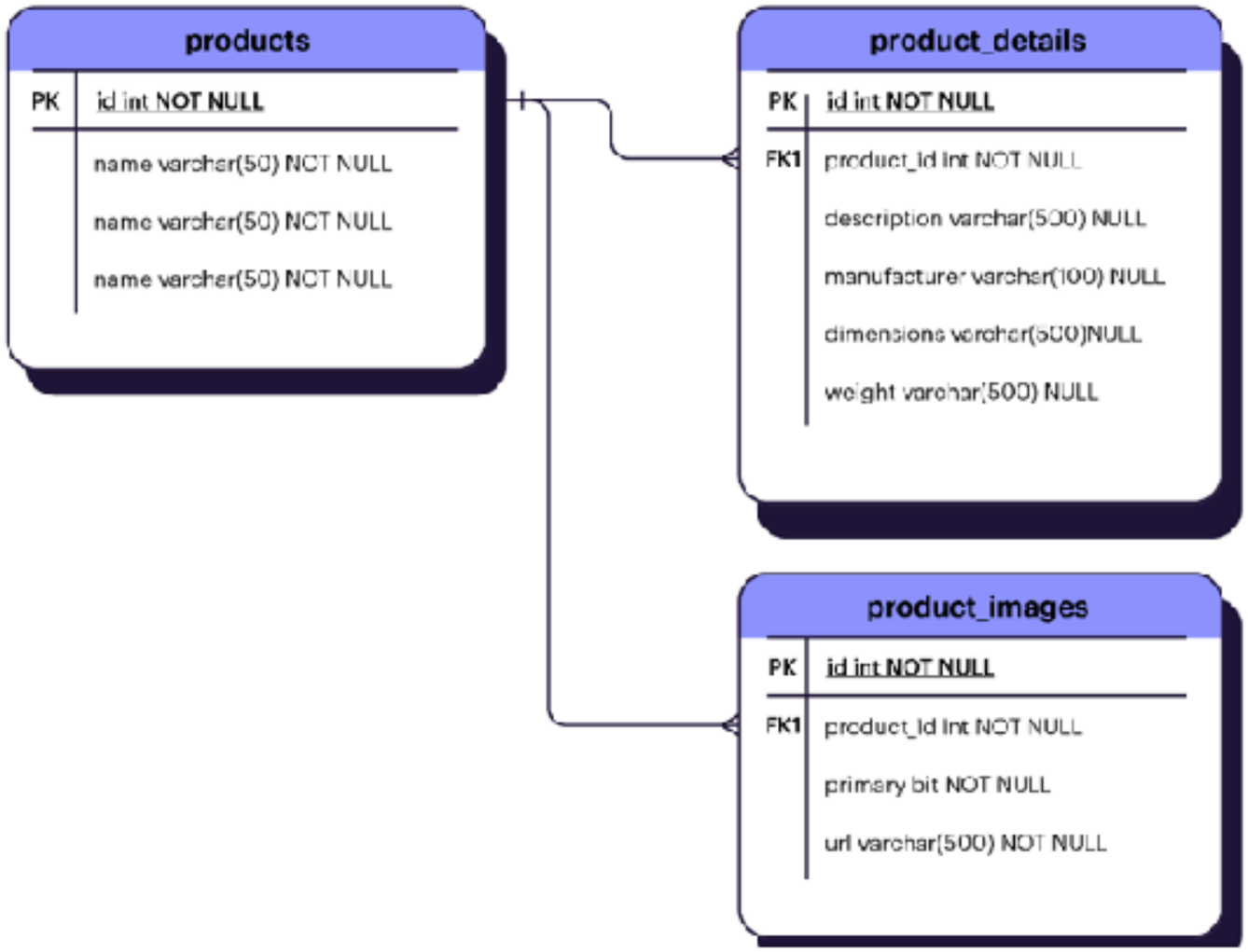
One-to-one

One-to-many

Many-to-many

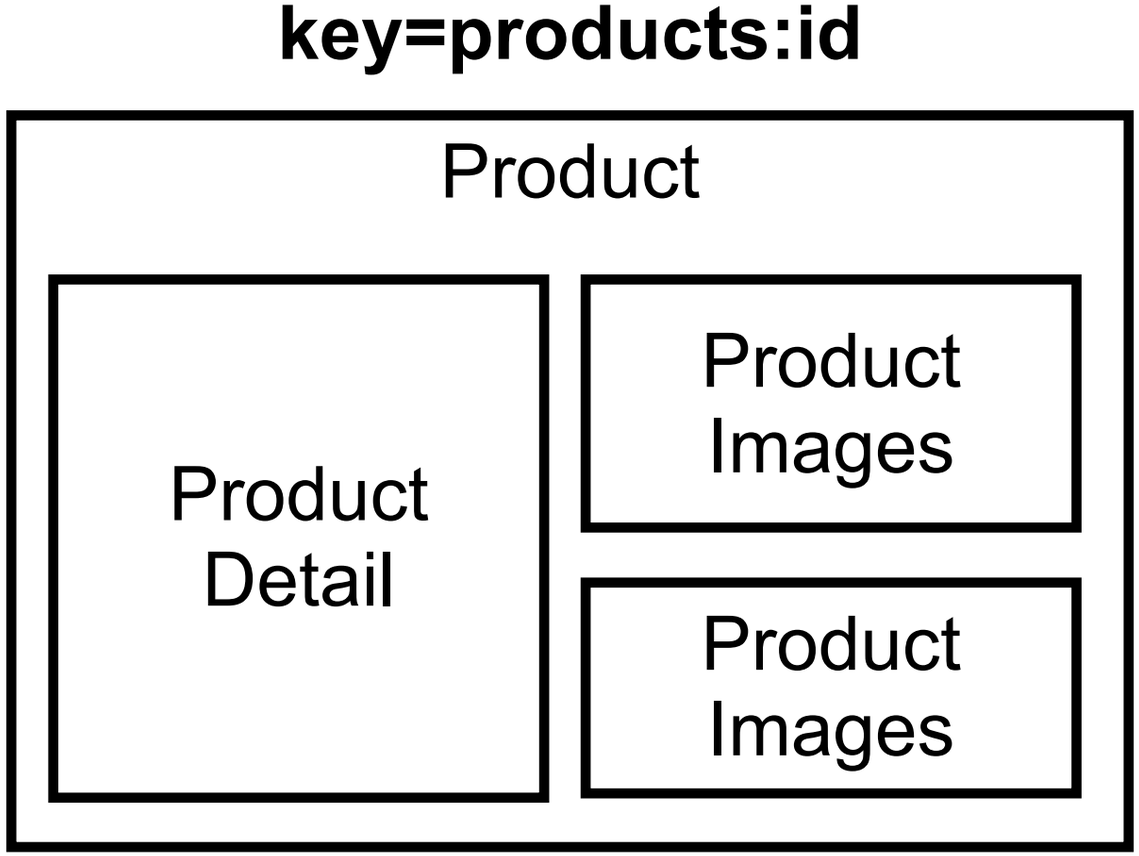
One-to-one

Product => Product detail



One-to-one in Redis

Embedded pattern for get product details by id



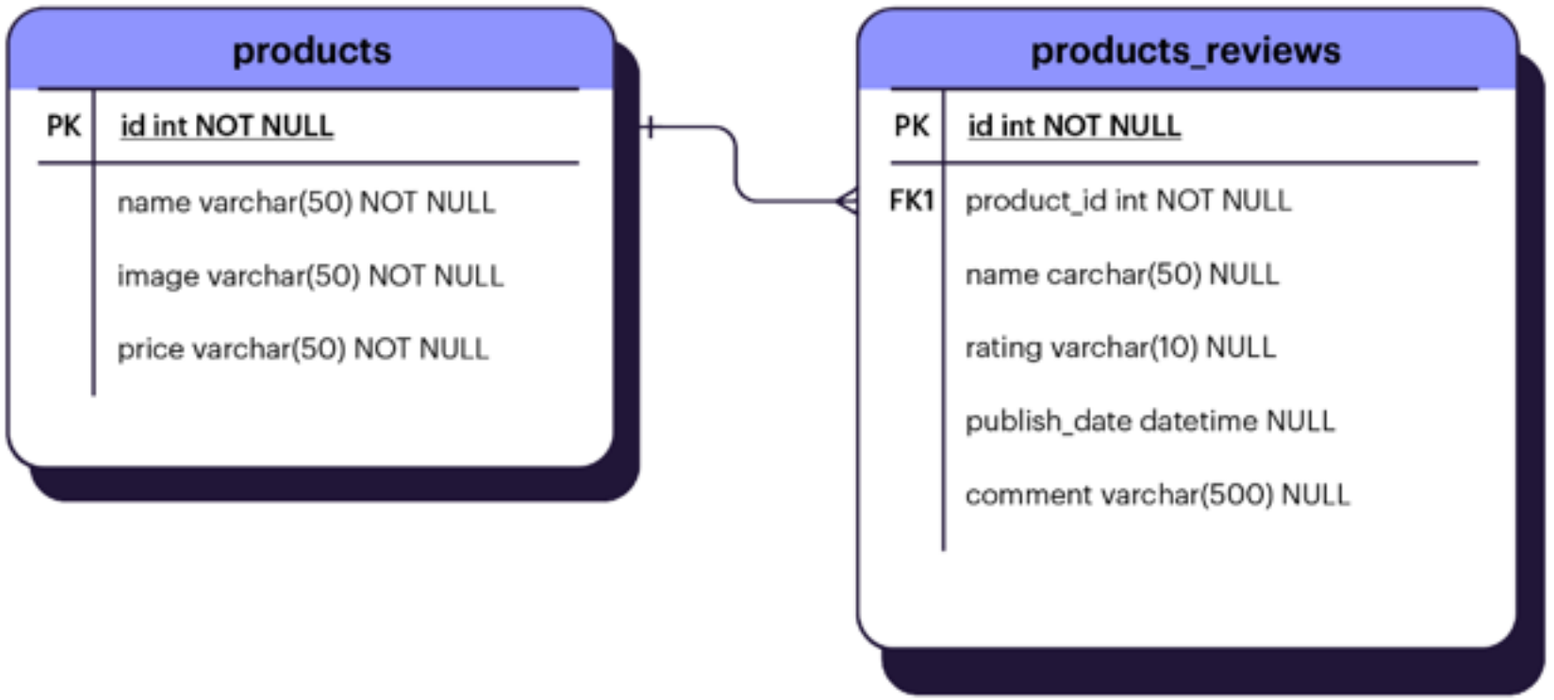
Search products ?

RediSearch

https://redis.io/docs/latest/operate/oss_and_stack/stack-with-enterprise/search/

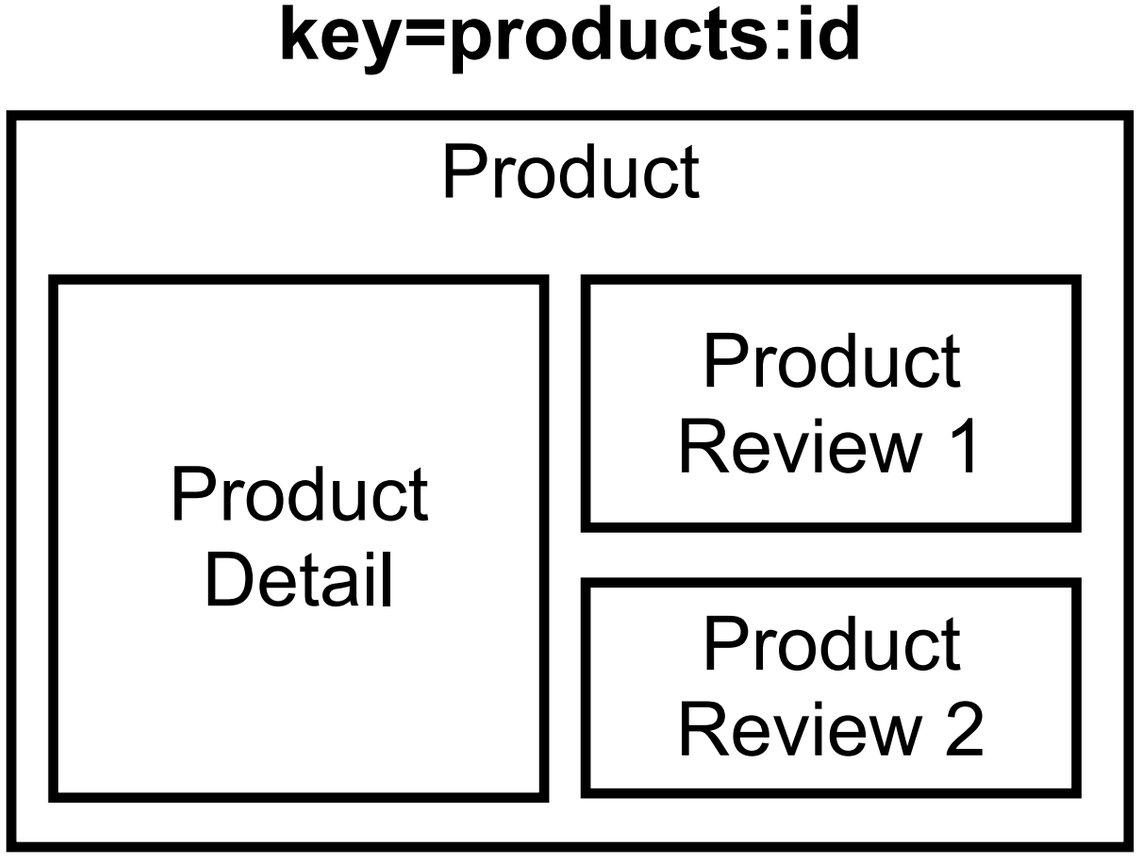
One-to-many

Product => Product reviews



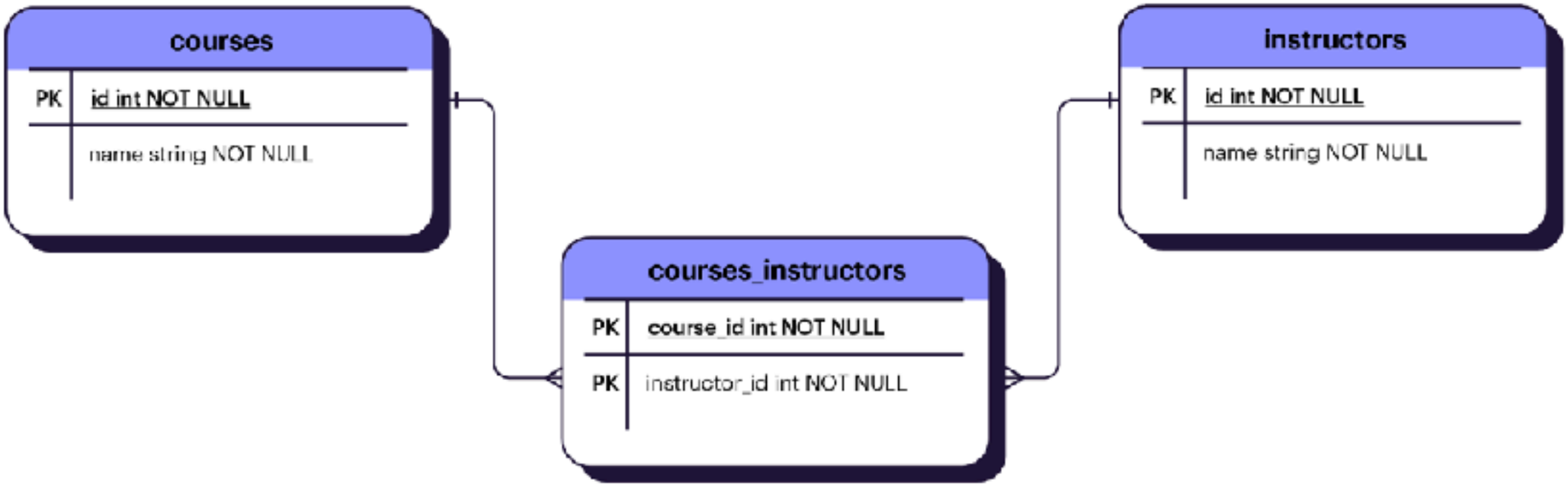
One-to-many in Redis

Partial Embedded pattern for recently review (newest)



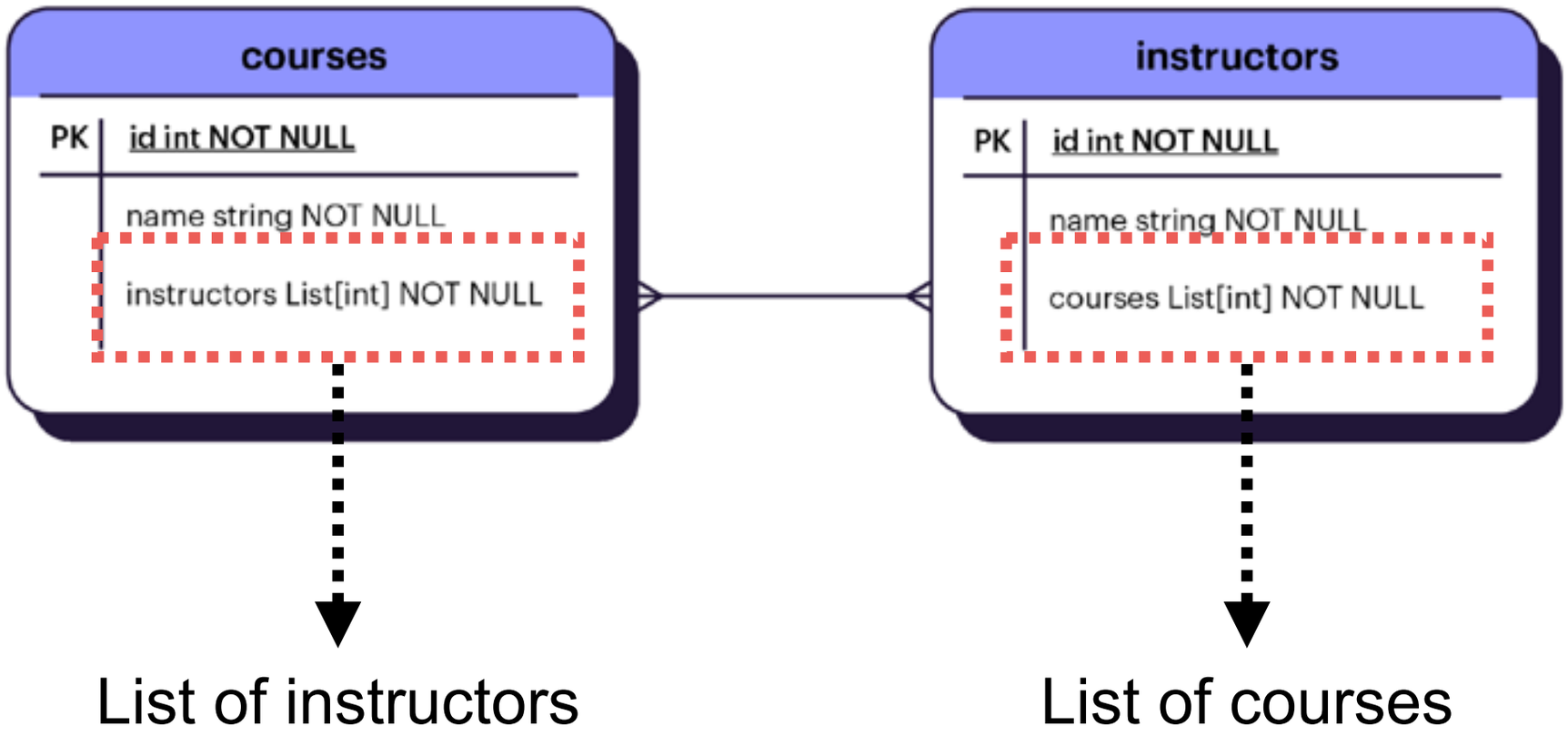
Many-to-many

Courses and Instructors



Many-to-many in Redis

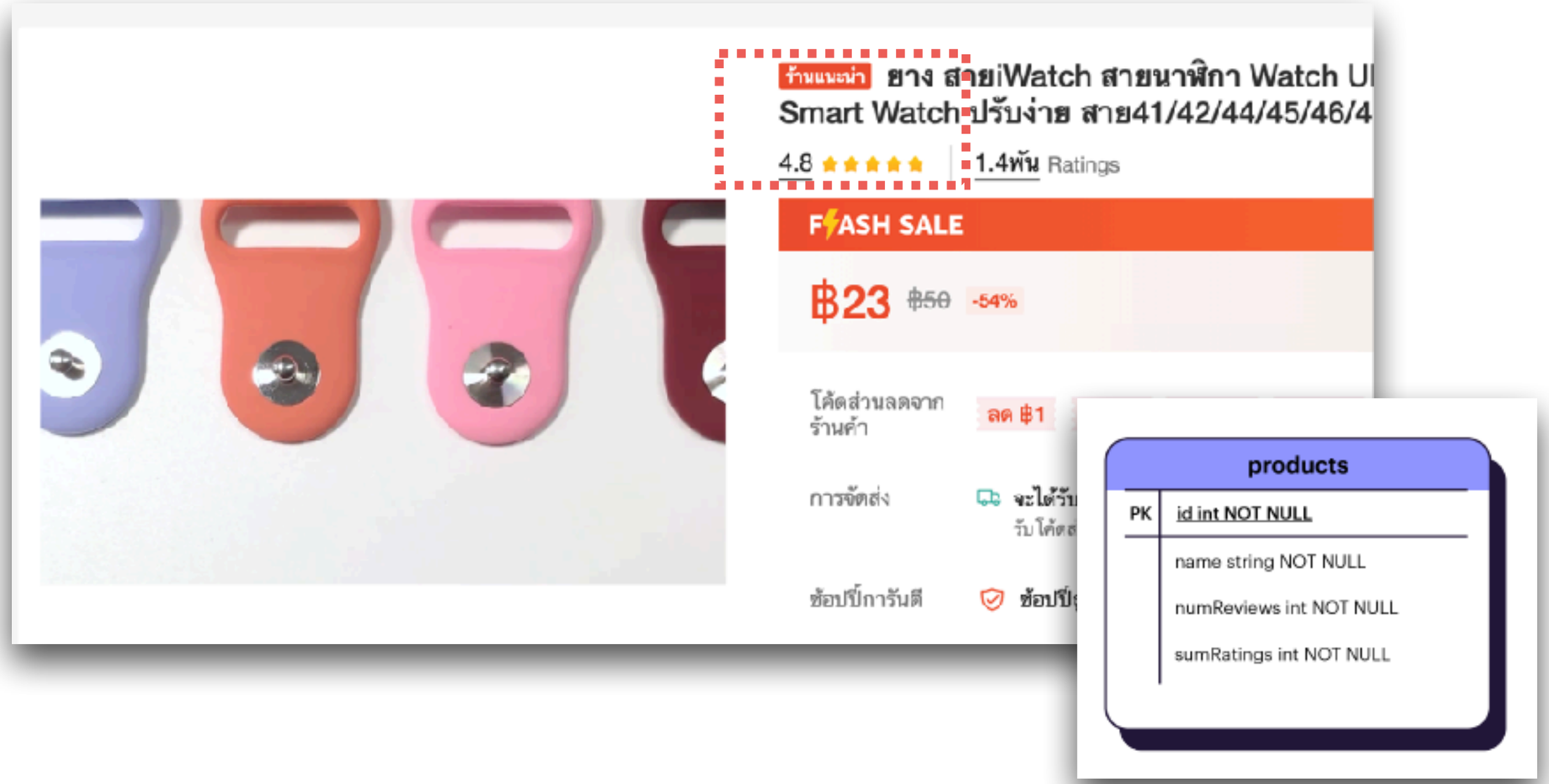
Courses and Instructors (many but small !!)



More patterns

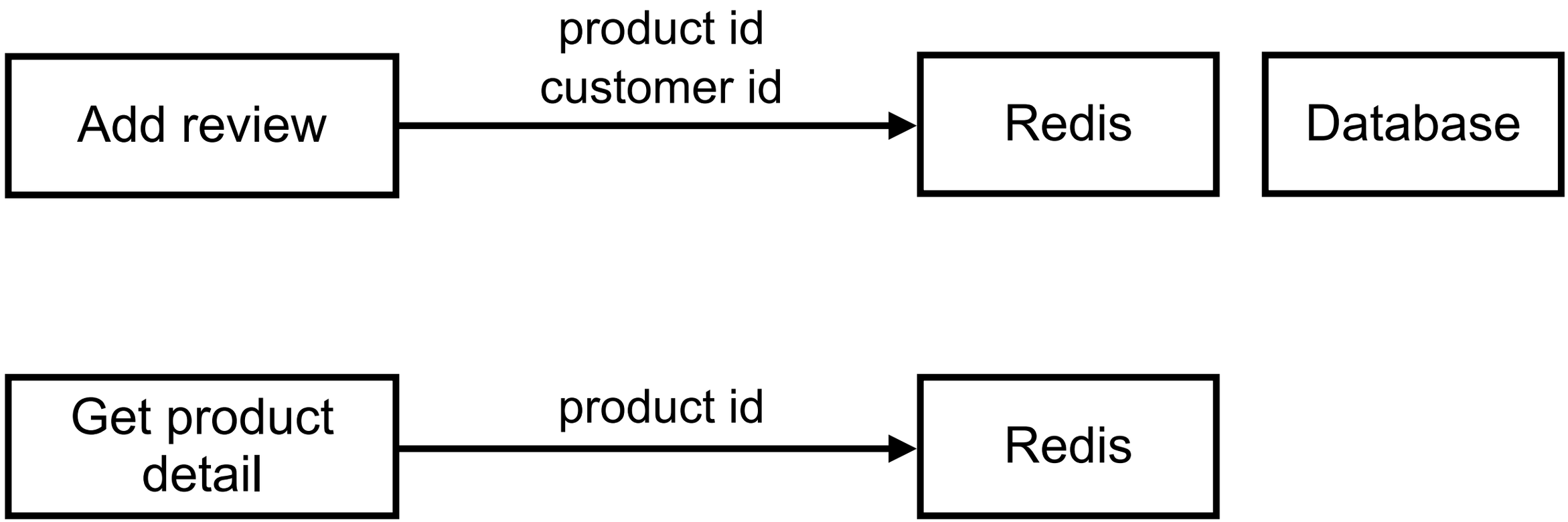
Aggregate pattern

Product rating !!



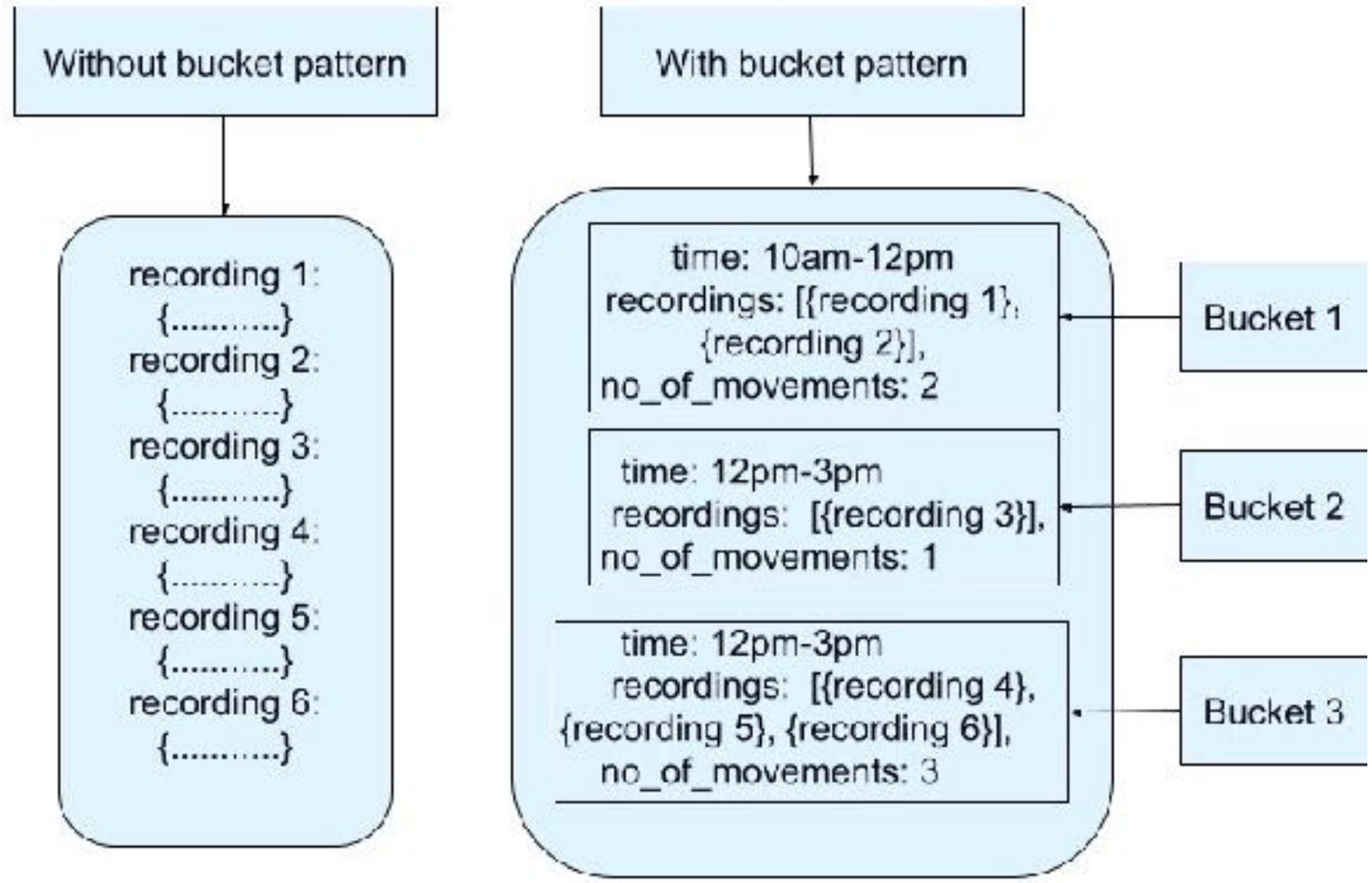
Aggregate pattern in Redis

Product rating !!



Bucket pattern

Working with time series database



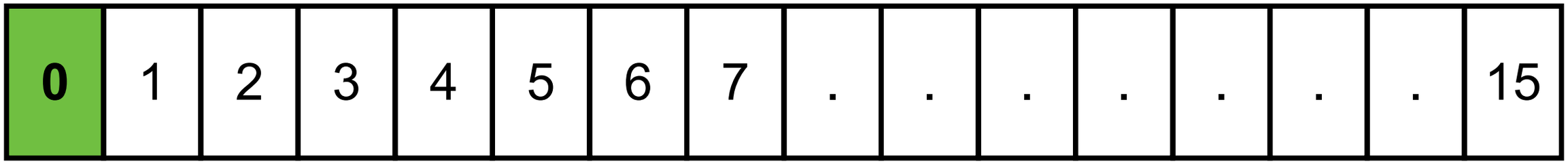
Configurations !!

redis.conf

https://www.dragonflydb.io/guides/redis-best-practices

Redis databases

Default = 16



Redis databases

Isolation the data
Easy data separation
Flexible in configuration
Selective data management

Memory eviction

FLUSHALL

Max memory

Default = none
Allow size of memory to Redis
Working with memory eviction policies

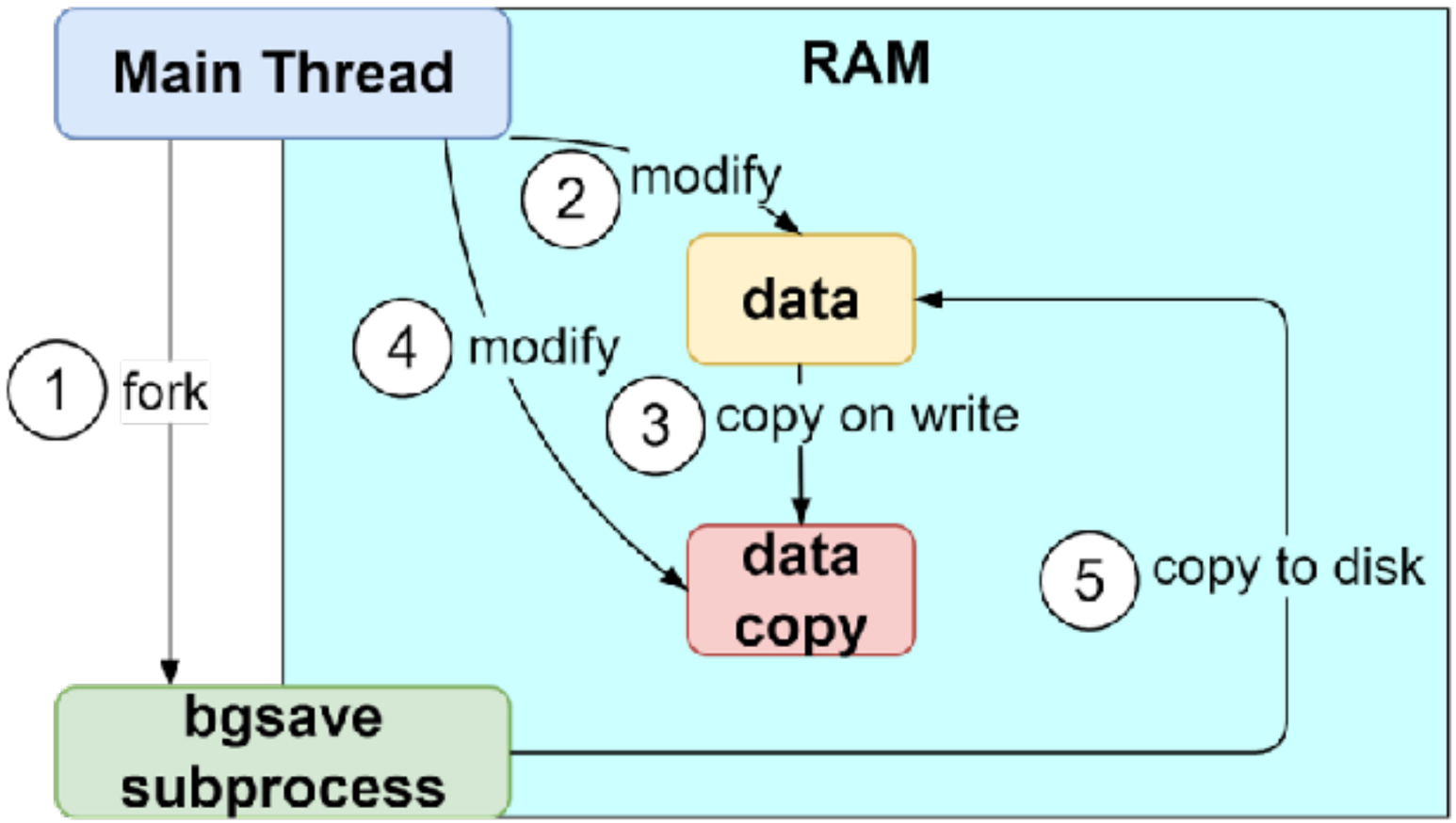
Redis Persistence

RDB (Redis Database)
AOF (Append Only File)
No persistence
RDB + AOF

https://redis.io/docs/latest/operate/oss_and_stack/management/persistence/

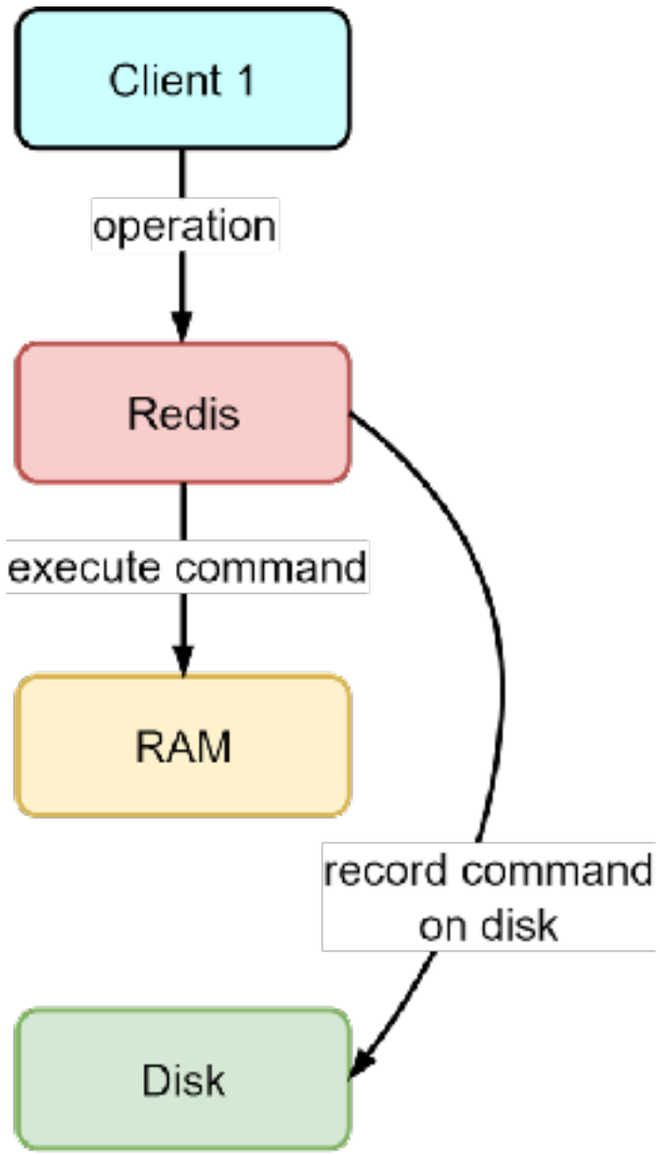
RDB (Redis Database)

Snapshot data from point-in-time
Copy-on-write solution, don't block on write



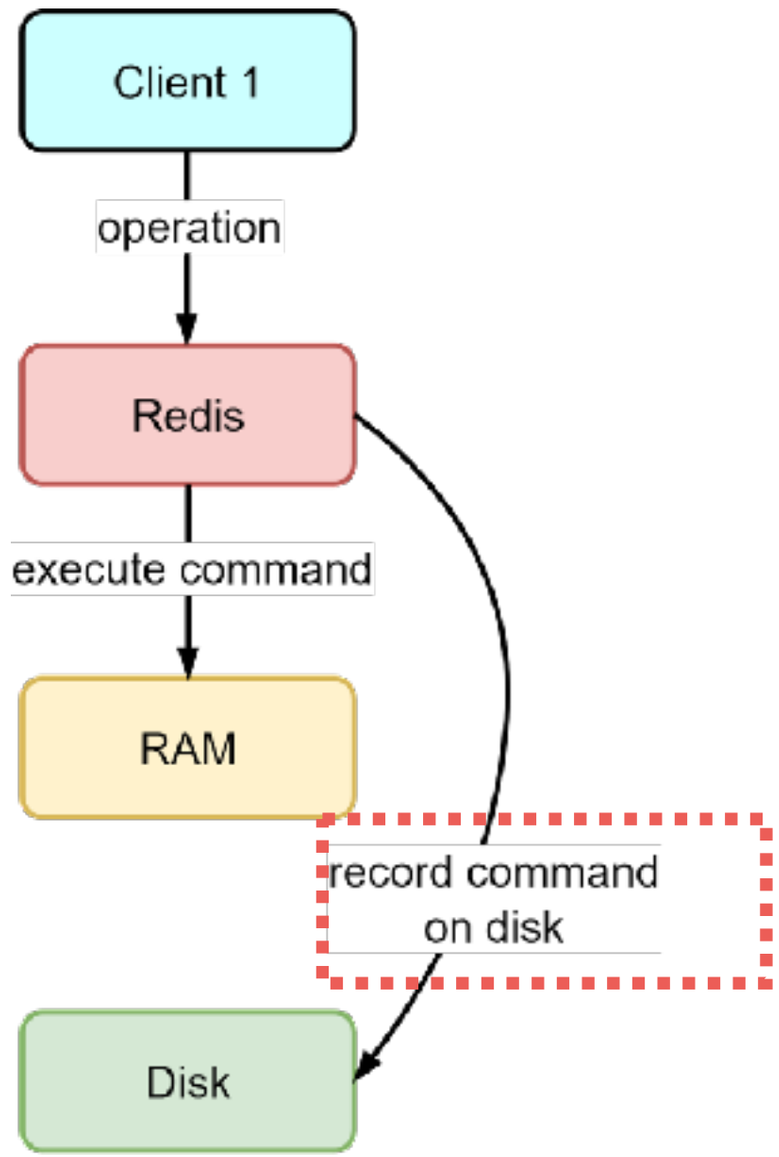
AOF (Append Only File)

Write-after log



AOF (Append Only File)

Every commands write data on disk !!



Bad performance !!

High I/O on server

AOF (Append Only File)

Try to configuration in Redis

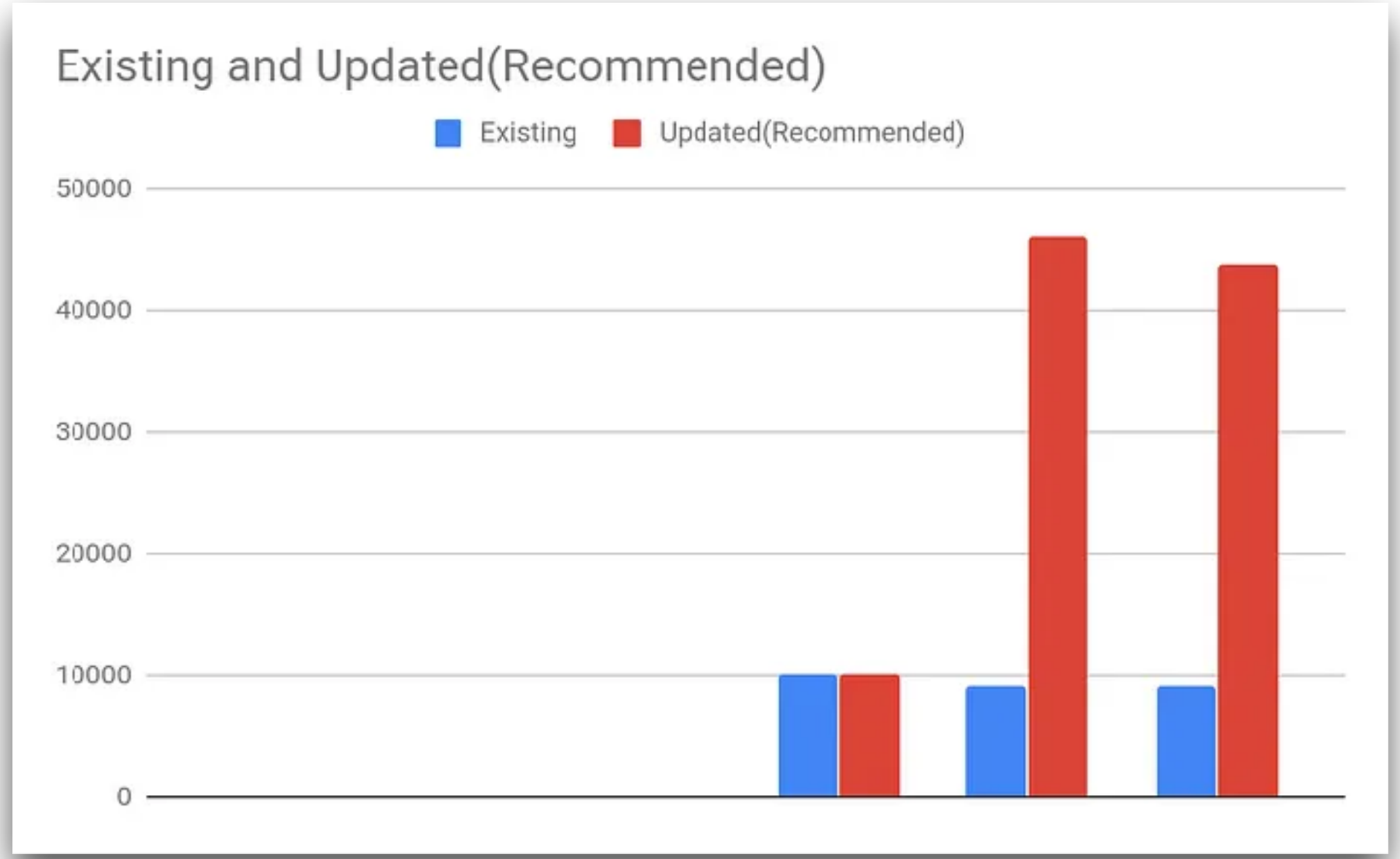
Solution	Description
Always	Write all commands to disk
Everysec	Write all commands in memory buffer first, write to disk every second
No	Write all commands in memory buffer first, decision to flush by OS !!

Keep-Alive

tcp-keepalive 0

Allow the same TCP connection
for HTTP conversation instead of
opening a new one with each new request

Keep-Alive



https://iamabhishek-dubey.medium.com/redis-best-practices-and-performance-tuning-c48611388bbc

Security

Add password to authorized

https://www.dragonflydb.io/guides/redis-best-practices

Limitation of Redis !!

Limitation of Redis !!

Memory size with large data
OOM (Out of Memory)

Redis provides evection policy

https://redis.io/docs/latest/operate/rs/databases/memory-performance/eviction-policy/

Redis eviction policy

Eviction Policy	Description
noeviction	New values aren't saved when memory limit is reached When a database uses replication, this applies to the primary database
allkeys-lru	Keeps most recently used keys; removes least recently used (LRU) keys
allkeys-lfu	Keeps frequently used keys; removes least frequently used (LFU) keys
allkeys-random	Randomly removes keys
volatile-lru	Removes least recently used keys with `expire` field set to true
volatile-lfu	Removes least frequently used keys with `expire` field set to true
volatile-random	Randomly removes keys with `expire` field set to true
volatile-ttl	Removes least frequently used keys with `expire` field set to true and the shortest remaining time-to-live (TTL) value

https://redis.io/docs/latest/operate/rs/databases/memory-performance/eviction-policy/

Limitation of Redis

Single thread limitation with high throughput

Scaling Redis with Clustering

Partition workload by data type in each instance

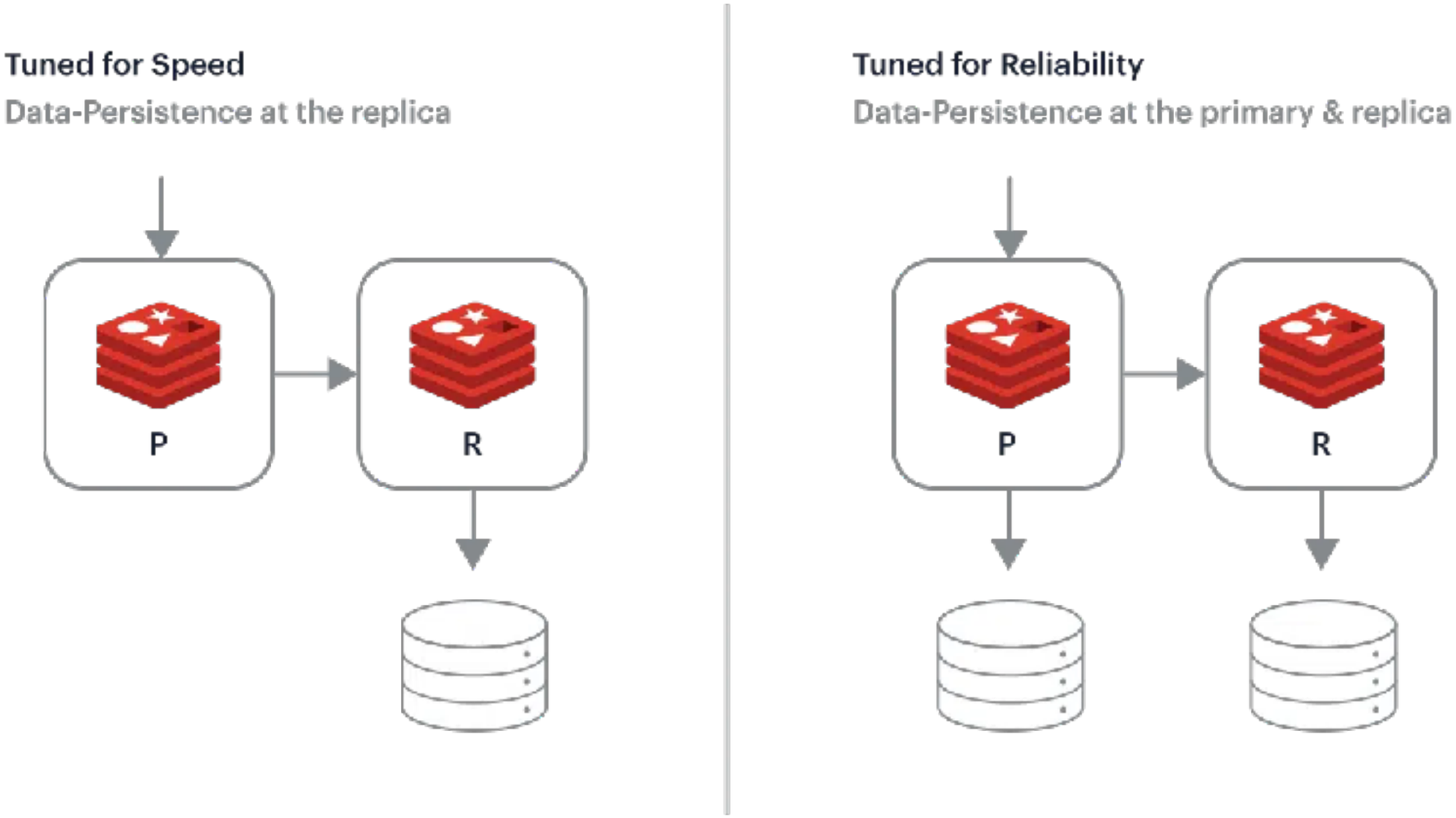
Redis 6+ supports Multi-thread I/O

https://redis.io/docs/latest/operate/rs/databases/memory-performance/eviction-policy/

Redis Topologies

Scale up	Scale out
Read/Write	High availability

Speed vs Reliability

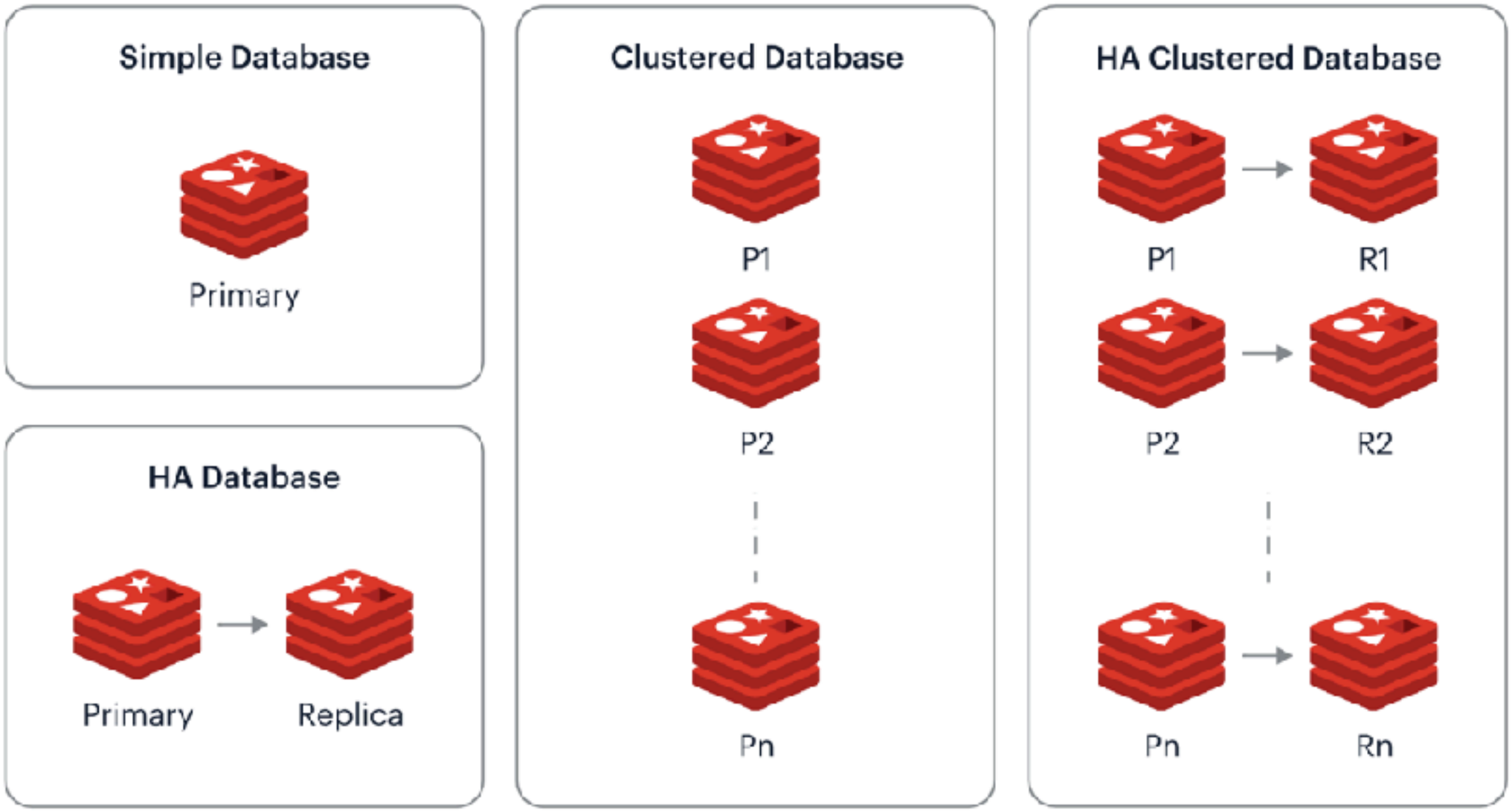


Redis Topologies

Single node
Multi-nodes (replication)
Master-slaves
Sentinel
Cluster + Sharing

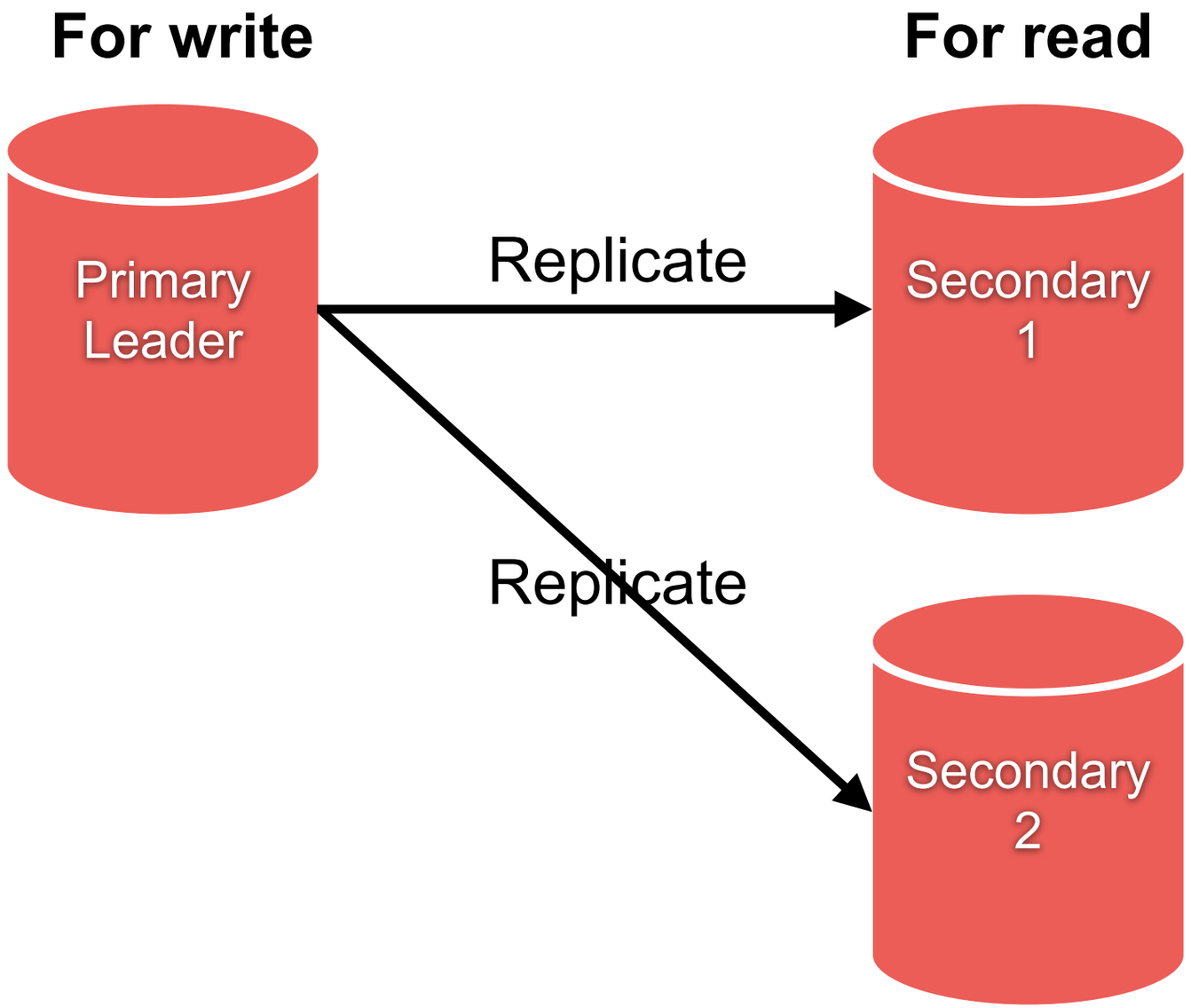
https://architecturenotes.co/p/redis

Redis Topologies



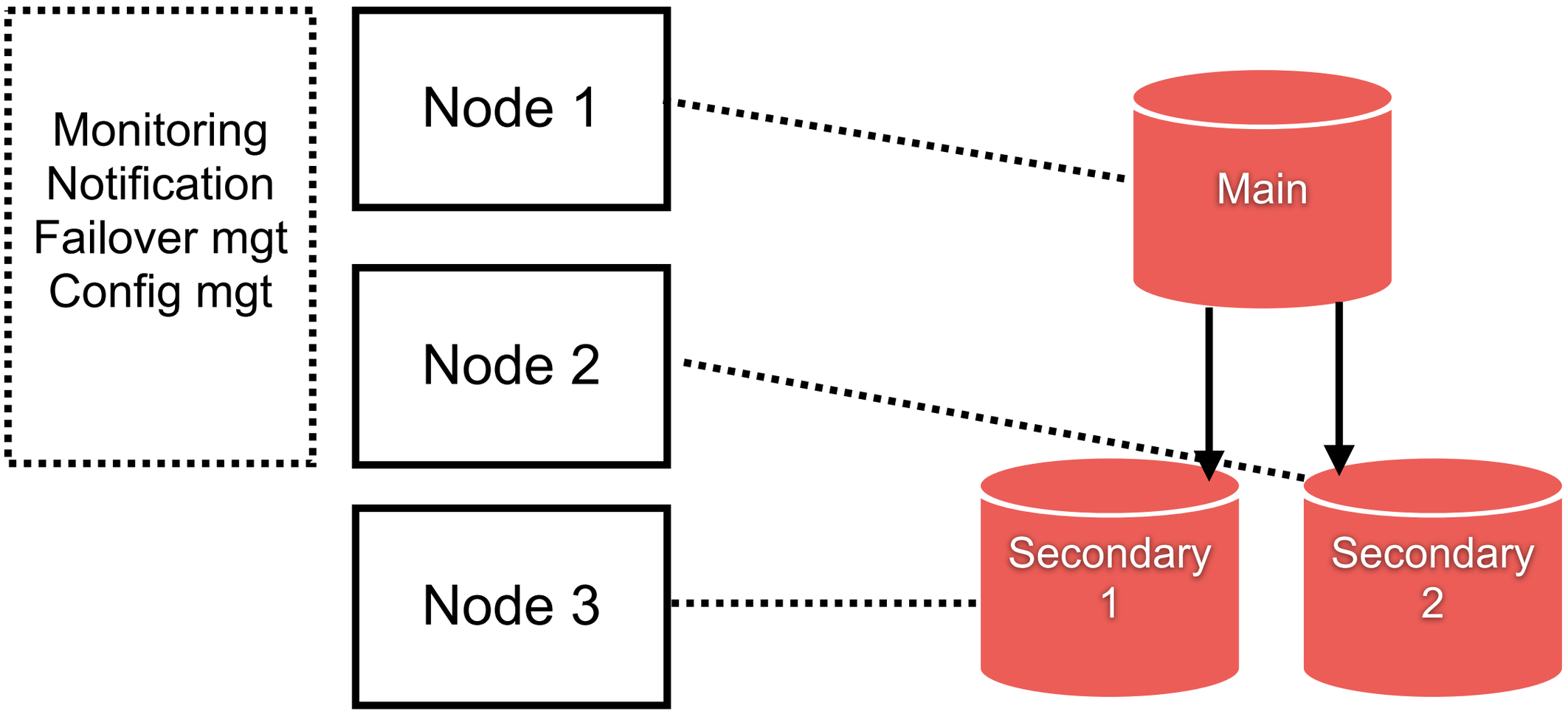
https://redis.io/technology/redis-enterprise-cluster-architecture/

Replication



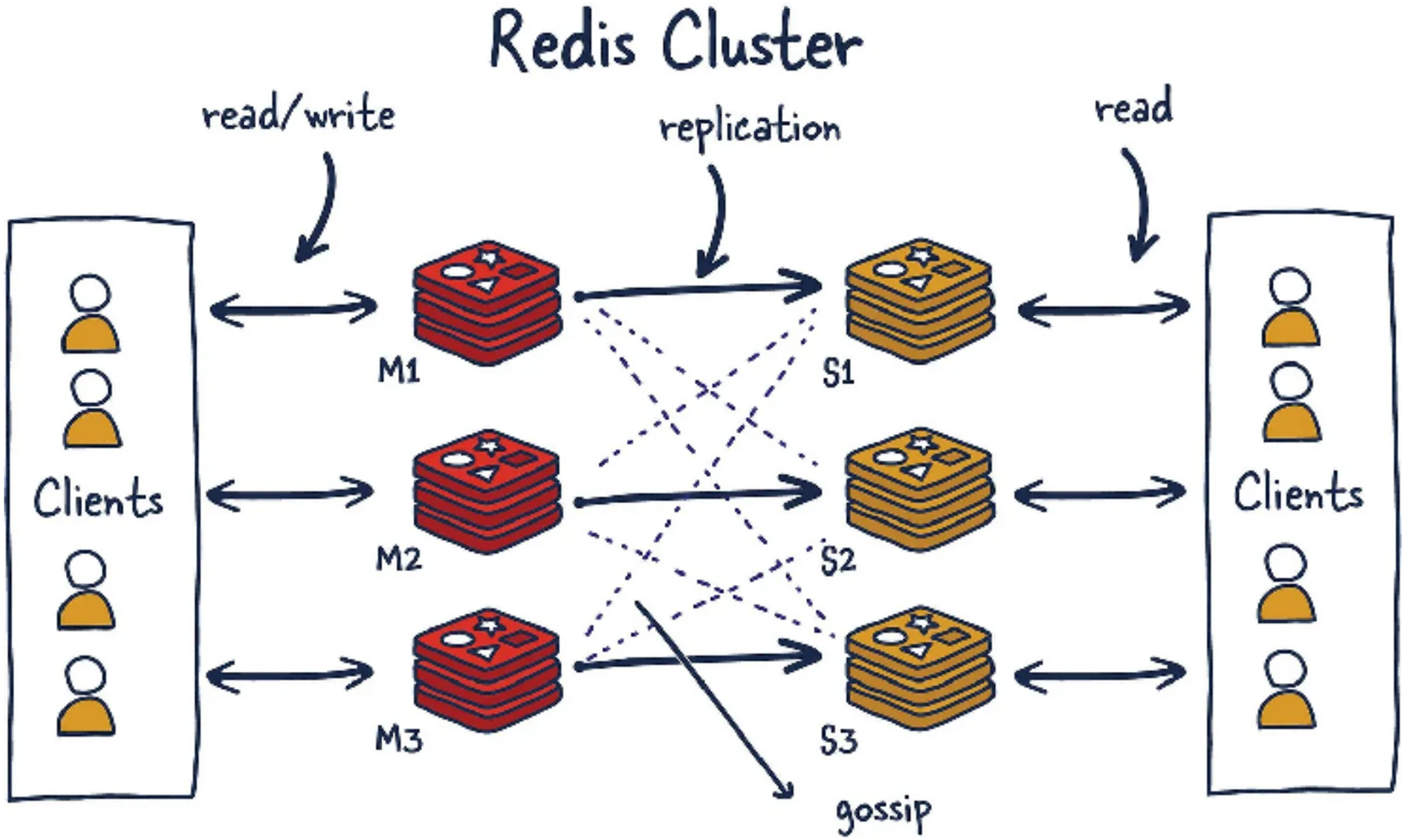
Redis sentinel (HA)

Sentinel nodes (odd number)



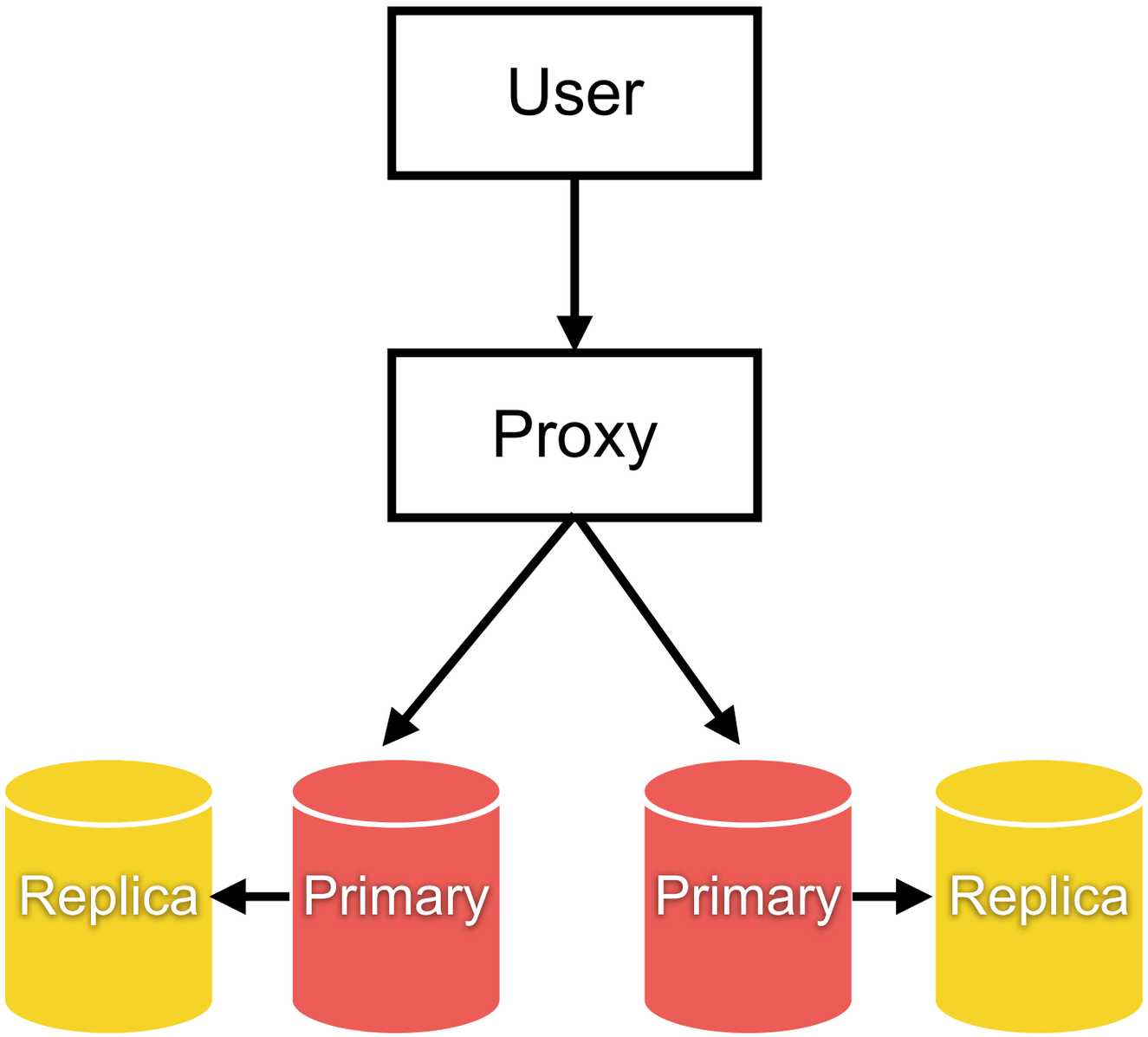
Redis cluster

Minimum 6 nodes !!



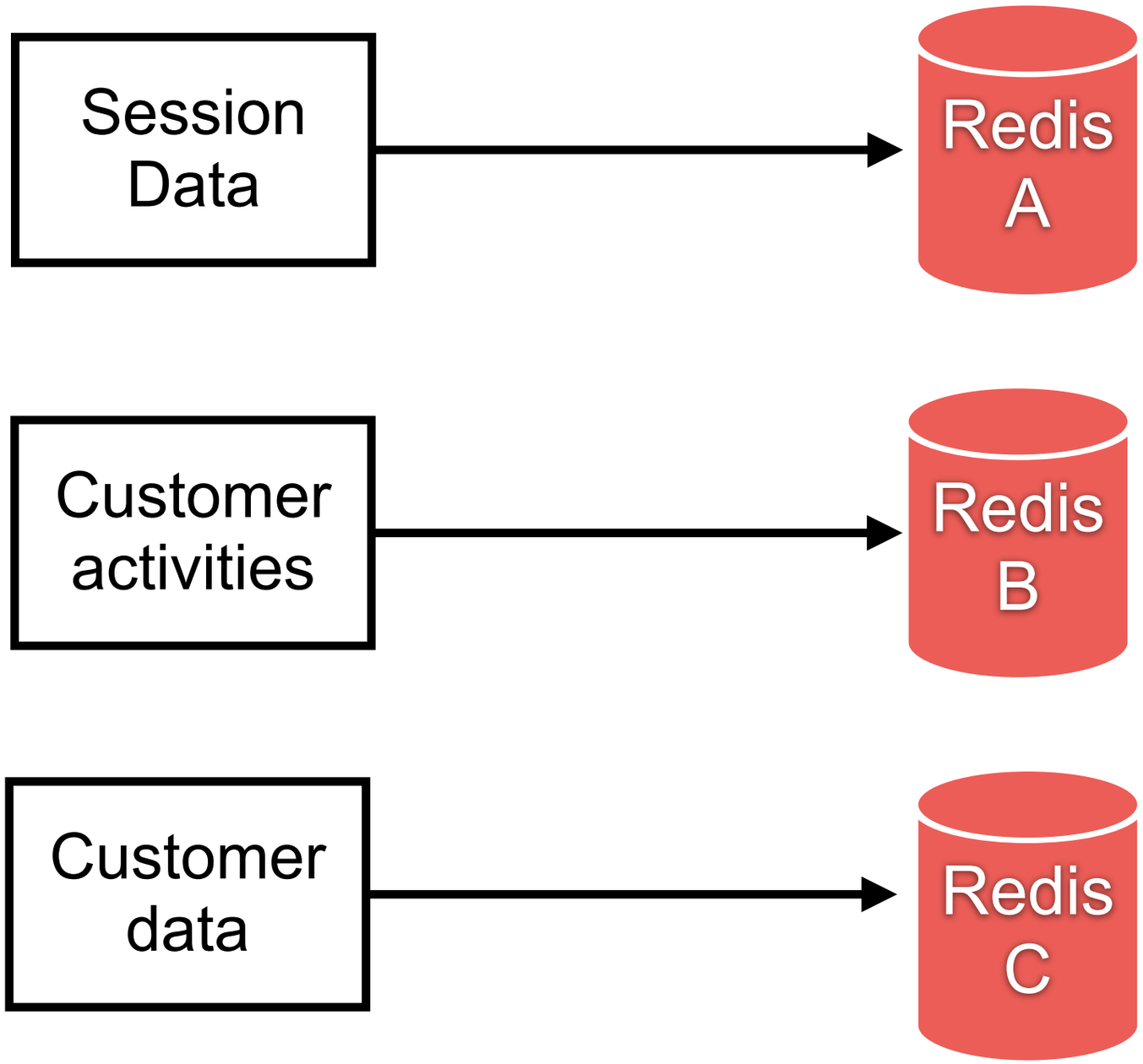
https://redis.io/docs/latest/operate/oss_and_stack/management/scaling/

Proxy-based with HA



https://semaphoreci.com/blog/redis-architectures

Partition Data into nodes



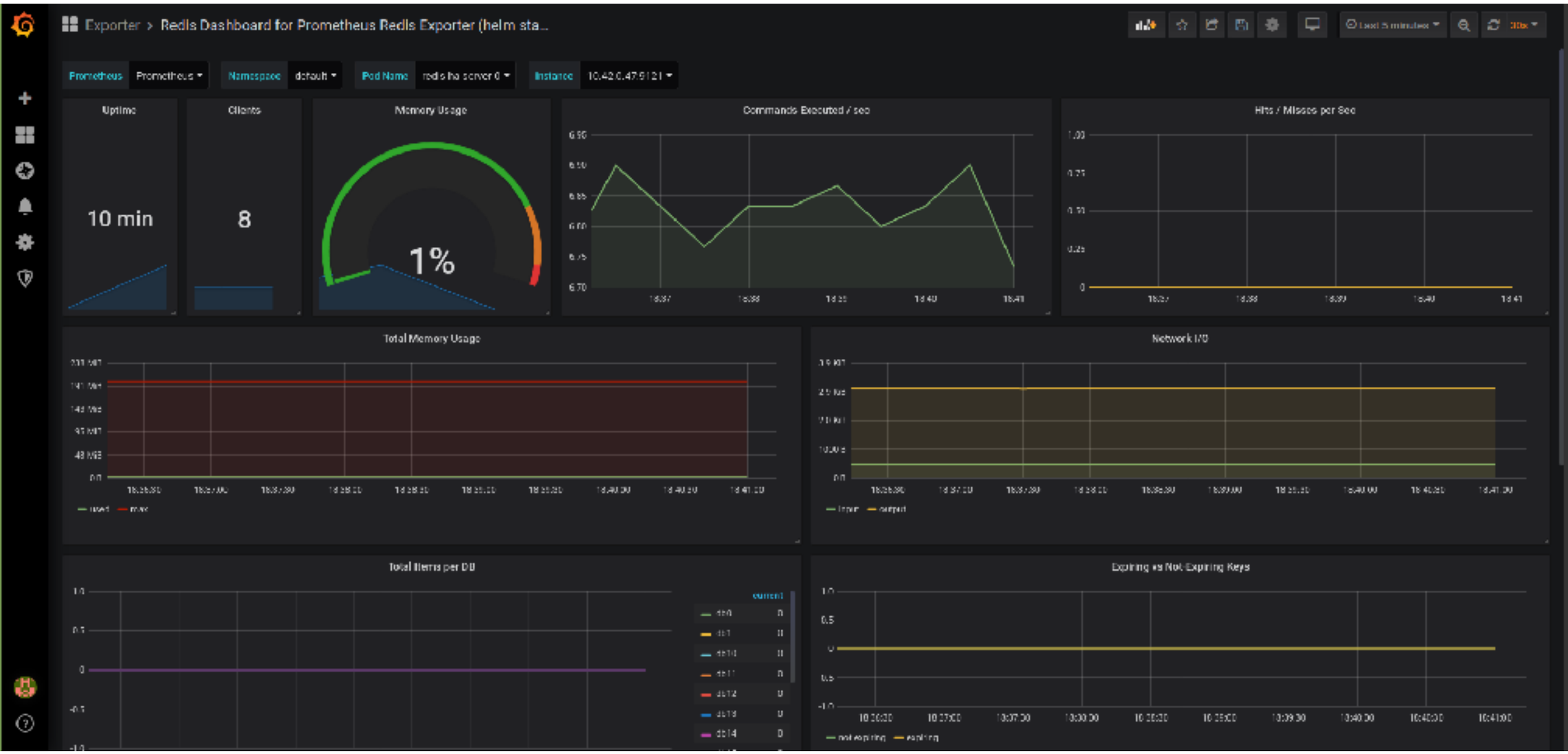
Monitoring

Redis Monitoring ?

Latency	Memory usage	CPU utilization
Throughput	Network	Keyspace
	Logging	

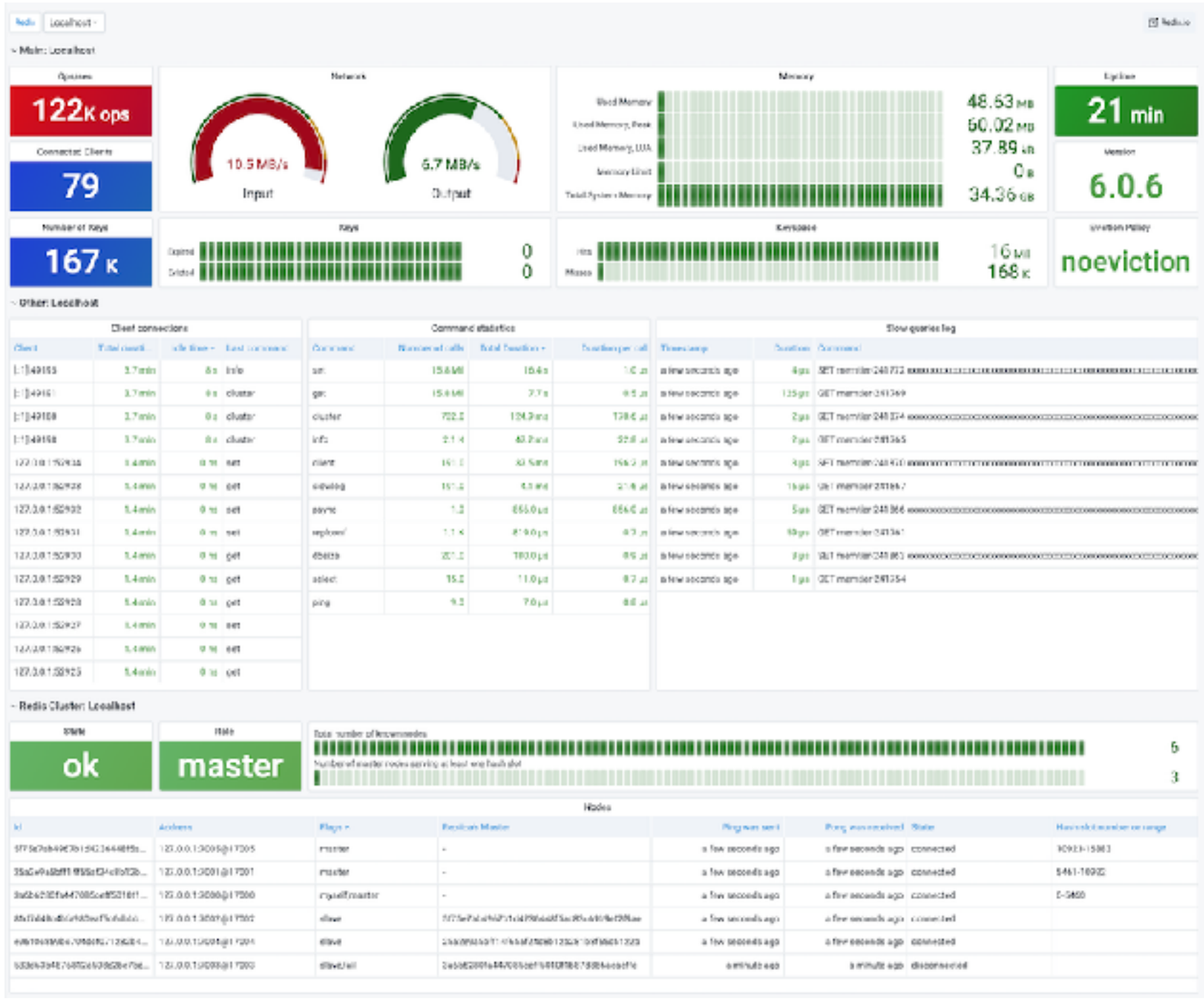
https://uptrace.dev/blog/redis-monitoring

Grafana Stack



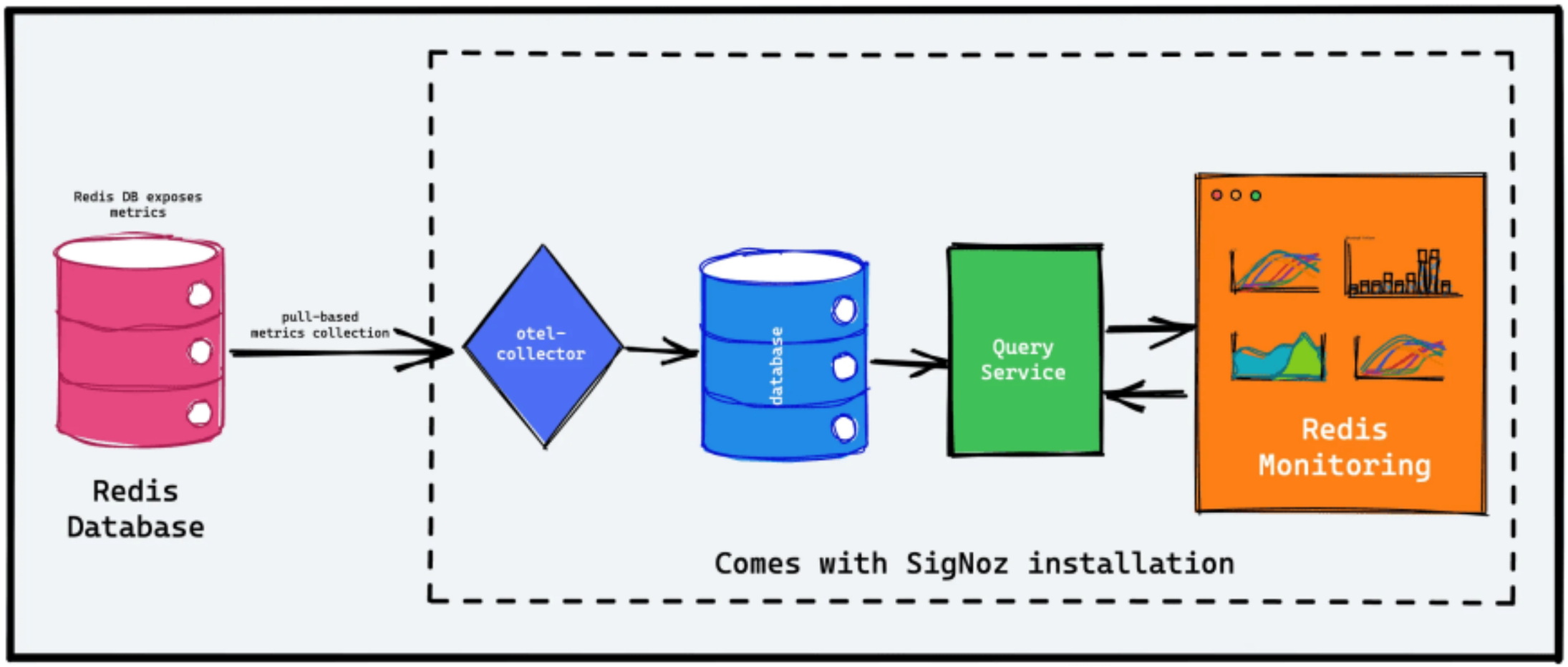
https://redis.io/docs/latest/integrate/prometheus-with-redis-cloud/

Grafana Stack



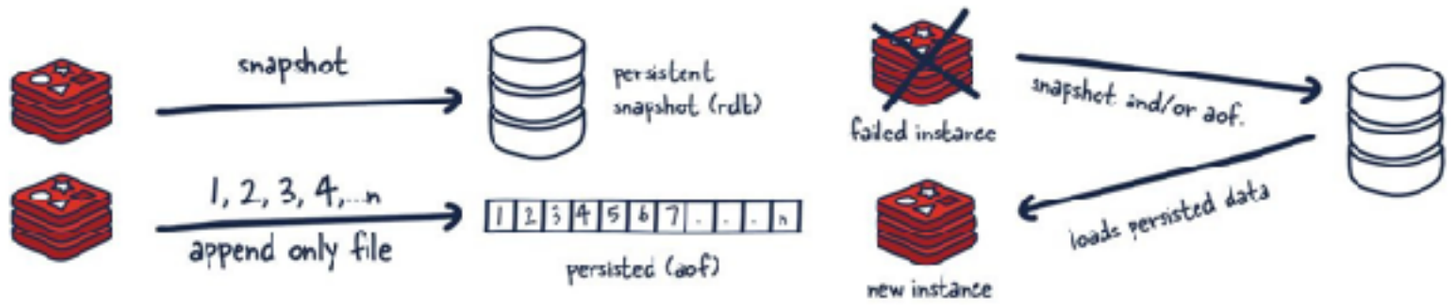
https://grafana.com/grafana/dashboards/12776-redis/

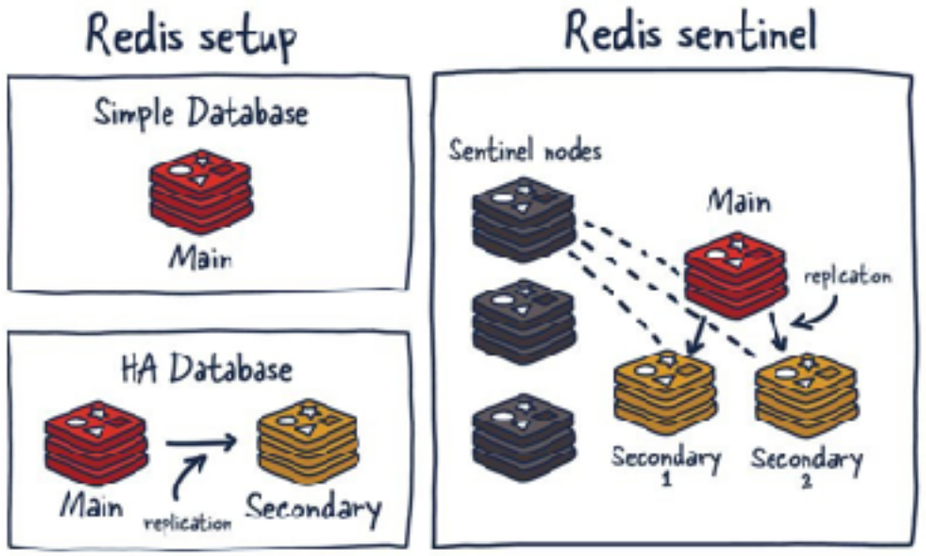
OpenTelemetry

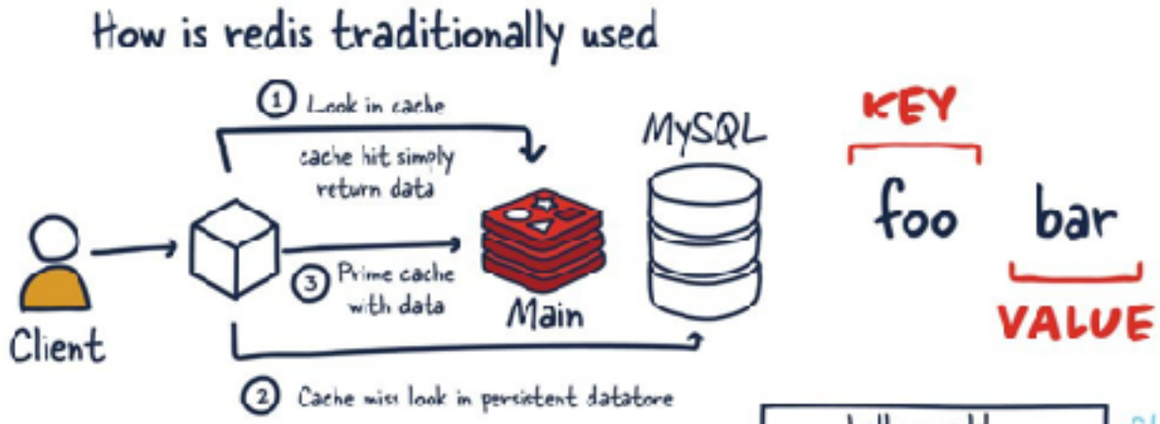


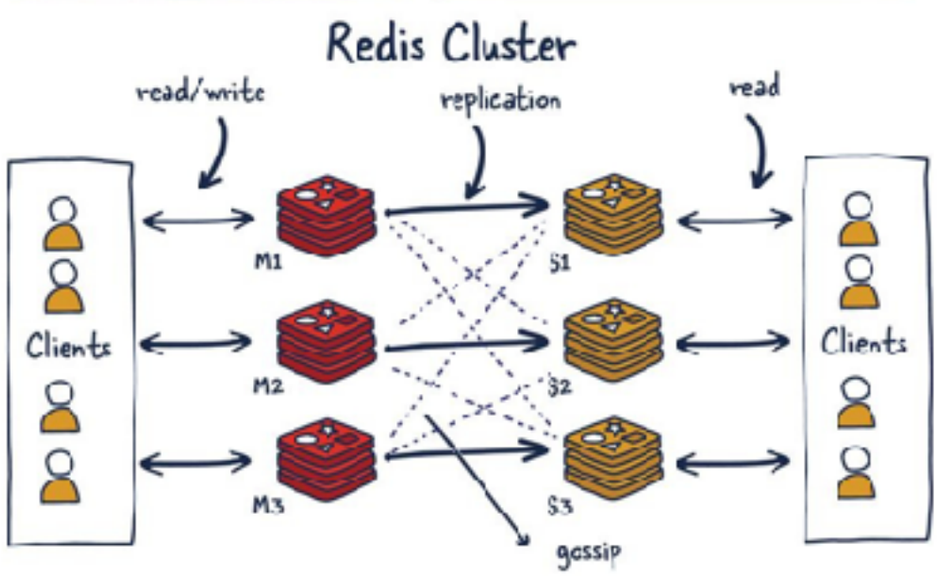
https://signoz.io/blog/redis-opentelemetry/

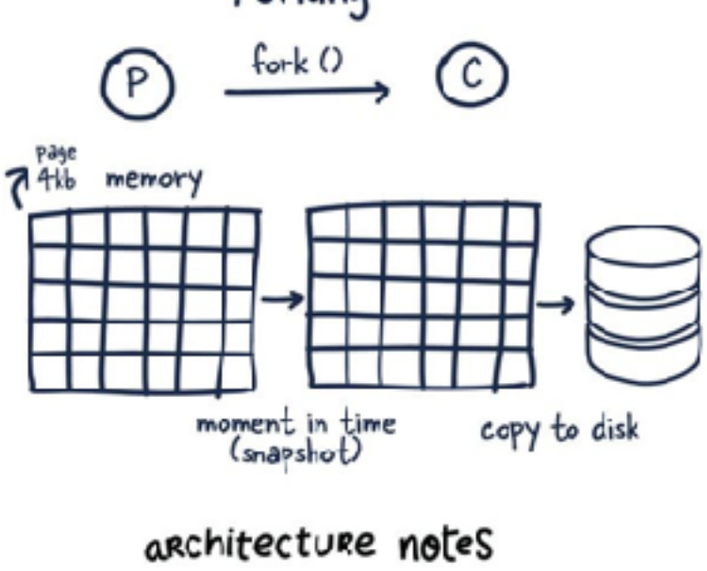
Summary











https://architecturenotes.co/p/redis

Q/A