“Redis Functions and Data Structures”

redislabs

Dave Nielsen, Developer Advocate
dave@redislabs.com @davenielsen
Redis Labs @redislabs

Redis = A Unique Database

Redis is an open source (BSD licensed), **in-memory**, **data structure store**, used as **database**, **cache AND message broker**

About Redis

# 1

NoSQL in User Satisfaction and Market Presence
[@G2CROWD]

# 1

In growth among NoSQL databases
[@DB-Engines]

# 1

NoSQL database on containers
[@DevOps.com & ClusterHQ]

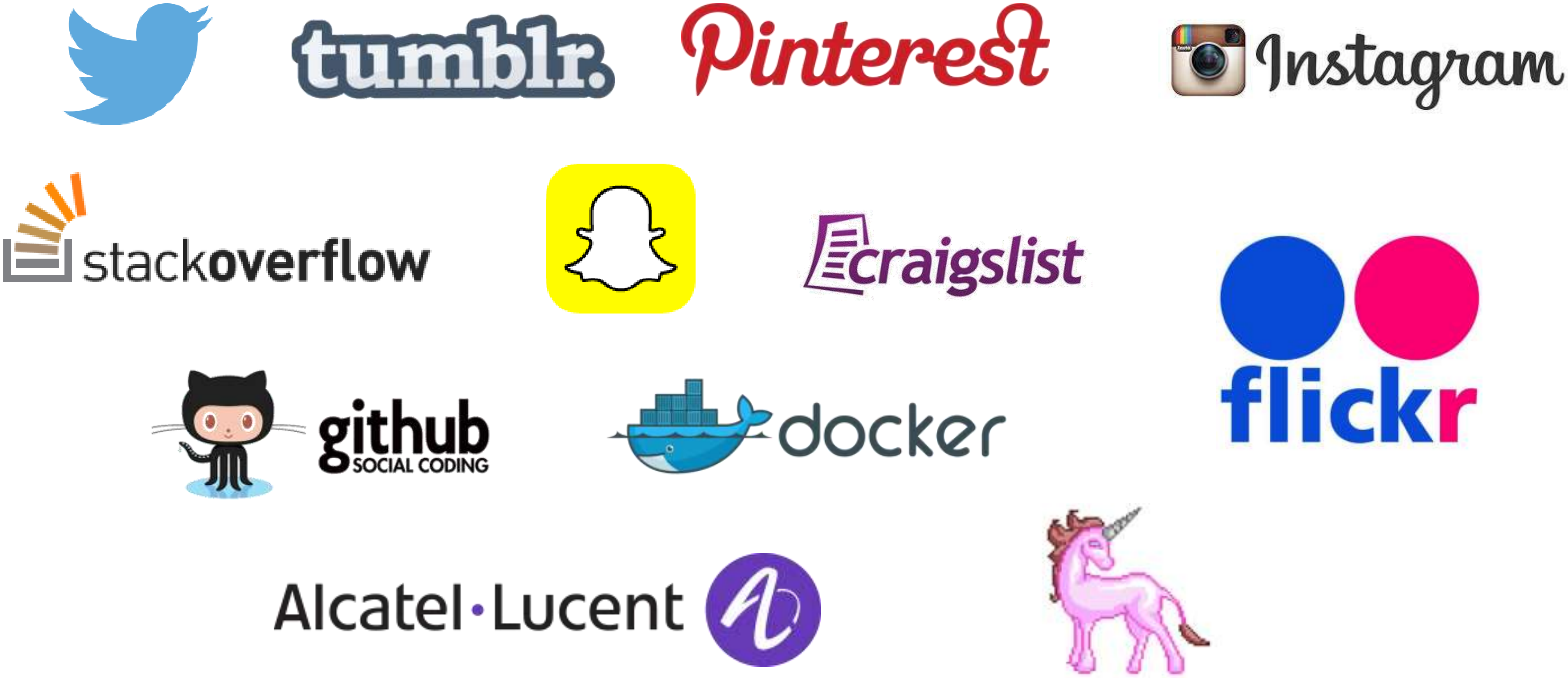
# 12

Out of 50 tools developers love to use
[@Stackshare]

Redis has the **largest open source community** among the NoSQL databases

Created by Salvatore Sanfilippo (@antirez)

Redis Helps the Web Scale!



Data Structure Store

MISSION STATUS:

COMMANDER BOWMAN LEFT DISCOVERY TO INVESTIGATE. LAST TRANSMISSION FROM COMMANDER BOWMAN:

"MY GOD, IT'S FULL OF STARS."

“MY GOD, IT’S FULL OF STRUCTURES.”

Redis : A Data Structure Store

Strings

Hashes

Lists

Sets

Sorted Sets

Bit Arrays

Hyper-LogLogs

Geospatial indexes

Data structures are used like “Lego” building blocks, saving developers coding effort and time

What Can You Do With Redis?

Use as in-memory database, cache or message broker

Common Uses

- User Sessions
- Message Brokers/Queues
- Real-time Recommendation Engine
- Leaderboads
- …More

User Sessions

The Problem

- Maintain session state across multiple servers
- Multiple session variables
- High speed/low latency required

Why Redis Rocks

- **Hashes** are perfect for this!
- HSET lets you save session variables as key/value pairs
- HGET to retrieve values
- HINCRBY to increment any field within the hash structure

Redis Hashes for User Sessions

hash key: usersession:1

userid	8754
name	dave
ip	10:20:104:31
hits	1
lastpage	home

```
HMSET usersession:1 userid 8754 name dave ip 10:20:104:31 hits 1
HMGET usersession:1 userid name ip hits
HINCRBY usersession:1 hits 1
```

```
HSET usersession:1 lastpage “home”
HGET usersession:1 lastpage
HDEL usersession:1 lastpage
```

```
DEL usersession:1
```

Hashes store a mapping of keys to values – like a dictionary or associative array – but faster

Managing Queues of Work

The Problem

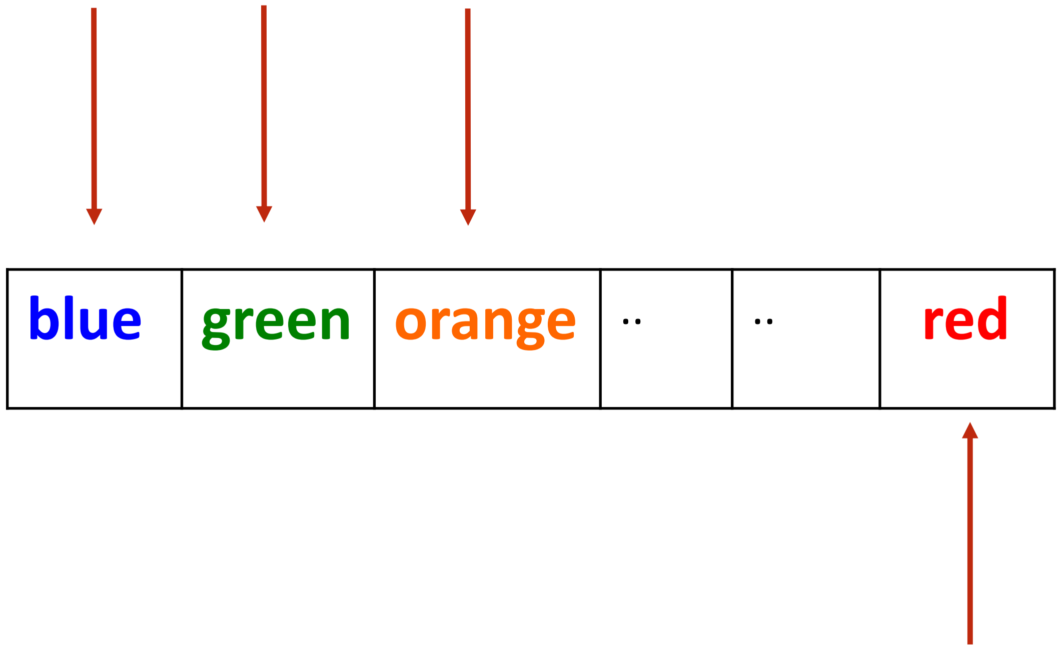
- Tasks need to be worked on async~~h~~ to reduce block/wait times
- Lots of items to be worked on
- Assign items to worker process and remove from queue at the same time
- Similar to buffering high speed data-ingestion

Why Redis Rocks

- **Lists** are perfect for this!
- LPUSH, RPUSH add values at beginning or end of queue
- RPOPLPUSH – pops an item from one queue and pushes it to another queue

Redis Lists for Managing Queues

LPUSH adds values to head of list



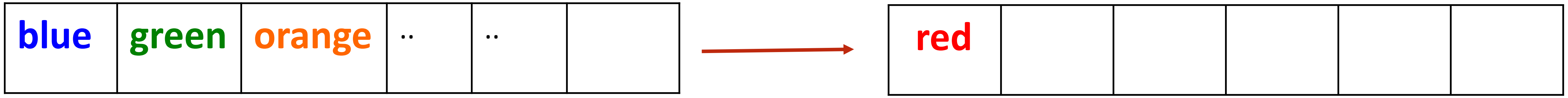
blue	green	orange	..	..	red

RPUSH adds value to tail of list

```
LPUSH queue1 orange
LPUSH queue1 green
LPUSH queue1 blue
RPUSH queue1 red
```

Redis Lists for Managing Queues

```
LPUSH queue1 orange
LPUSH queue1 green
LPUSH queue1 blue
RPUSH queue1 red
```



```
RPOPLPUSH queue1 queue2
```

RPOPLPUSH pops a value from one list and pushes it to another list

Real-time Recommendation Engine

The Problem

- People who read this article also read these other articles
- Want real time not data mining

Also used for:

- Recommending Similar Purchases
- Identifying Fraud

Why Redis Rocks

- **SETS** are unique collections of strings
- SADD to add tags to each article
- SISMEMBER to check if an article has a given tag
- SMEMBERS to get all the tags for an article
- use SINTER to find similar articles tagged with the same tags

Redis Sets for Recommendations

Set: tag:1

article 1	**article 3**	….		

Set: tag:2

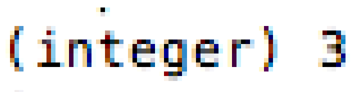
article 3	article 14	Article 22	..	

Set: tag:3

article 2	**article 3**	article 9	..	

Add values (articles) to Sets (tags)

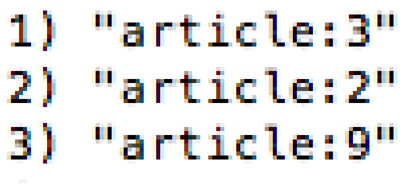
```
SADD tag:1 article:3 article:1
SADD tag:2 article:22 article:14 article:3
SADD tag:3 article:9 article:3 article:2
```



```
(integer) 3
```

Confirm the values have been added

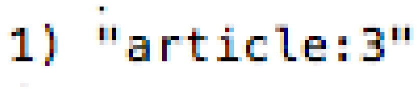
```
SMEMBERS tag:3      (also tag:1 & tag:2)
```



```
1) "article:3"
2) "article:2"
3) "article:9"
```

Find values that exist in all three Sets

```
SINTER tag:1 tag:2 tag:3
```



```
1) "article:3"
```

Sorted Sets for Leaderboards

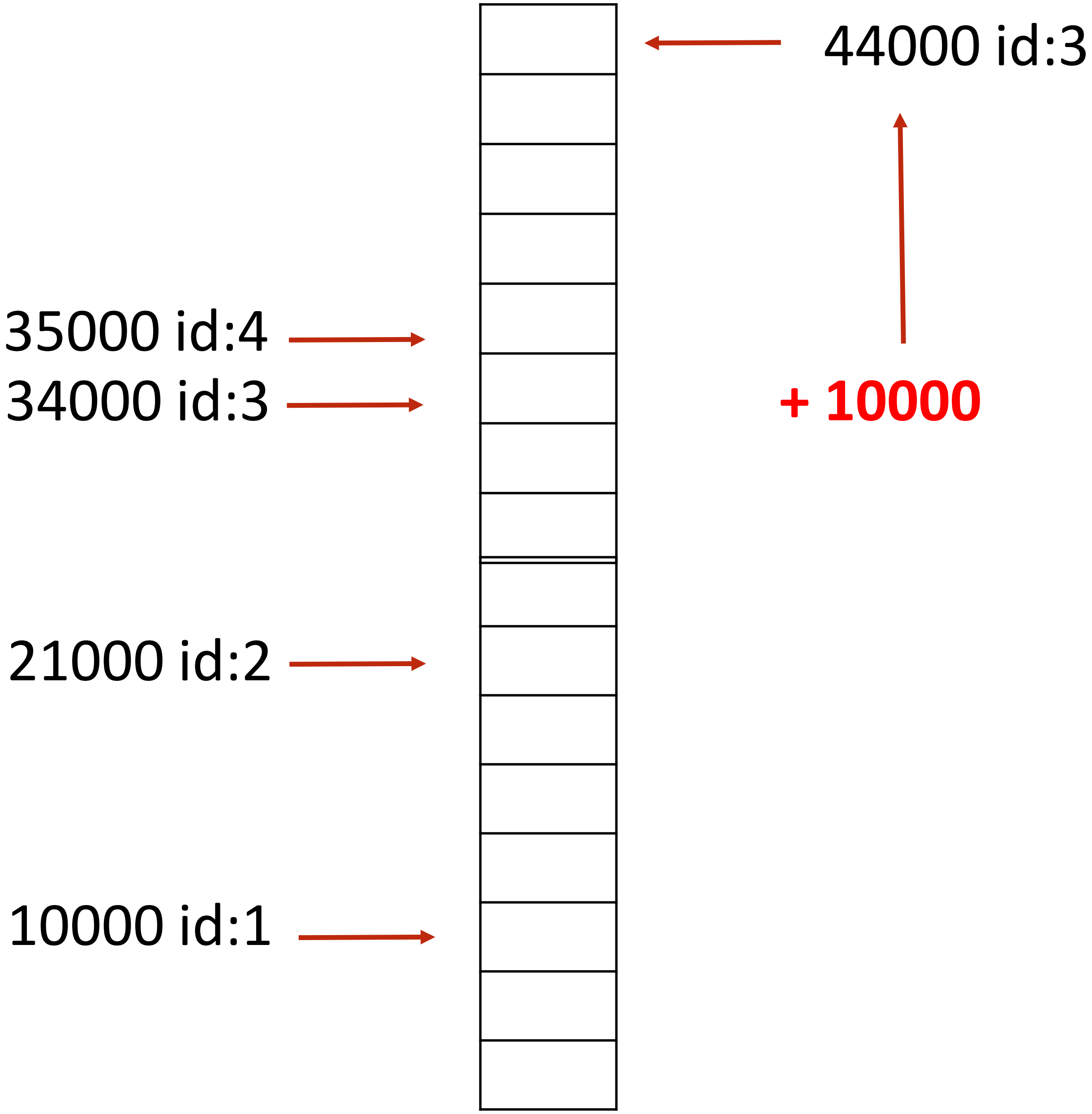
The Problem

- MANY users playing a game or collecting points
- Display real-time leaderboard.
- Who is your nearest competition
- Disk-based DB is too slow

Why Redis Rocks

- Sorted Sets are perfect!
- Automatically keeps list of users sorted by score
- ZADD to add/update
- ZRANGE, ZREVRANGE to get user
- ZRANK will get any users rank instantaneously

Redis Sorted Sets



```
ZADD game:1 10000 id:1
ZADD game:1 21000 id:2
ZADD game:1 34000 id:3
ZADD game:1 35000 id:4
ZADD game:1 44000 id:3
  or
ZINCRBY game:1 10000 id:3
```

```
ZREVRANGE game:1 0 0
ZREVRANGE game:1 0 1 WITHSCORES
```

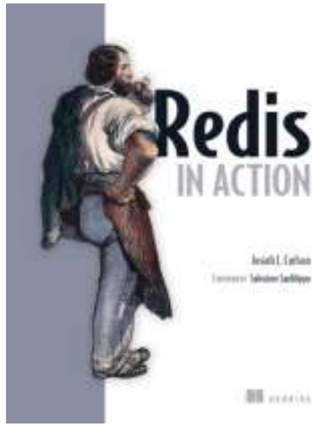
So What?

- Redis Data Structures are entirely in memory … and blazingly fast!
- Simplicity and speed make each Data Structure easy to use
- Combine Data Structures with Functions like Lego building blocks
- Clustering, Persistence, High Availability are now standard

Learn More ...

Open Source Redis → redis.io

Free “Redis in Action” eBook → redislabs.com/ebook



Free 30mb Redis Cloud → redislabs.com

Download RLEC Trial → redislabs.com/redis-enterprise redislabs

Home of Redis

Thank You!

Dave Nielsen, Developer Advocate
dave@redislabs.com @davenielsen
Redis Labs @redislabs