redis

ZADD key score member [score member ...]

> **Available since 1.2.0.**
>
> **Time complexity:** O(log(N)) where N is the number of elements in the sorted set.

Adds all the specified members with the specified scores to the sorted set stored at key. It is possible to specify multiple score / member pairs. If a specified member is already a member of the sorted set, the score is updated and the element reinserted at the right position to ensure the correct ordering.

If key does not exist, a new sorted set with the specified members as sole members is created, like if the sorted set was empty. If the key exists but does not hold a sorted set, an error is returned.

The score values should be the string representation of a double precision floating point number. `+inf` and `-inf` values are valid values as well.

Sorted sets 101

Sorted sets are sorted by their score in an ascending way. The same element only exists a single time, no repeated elements are permitted. The score can be modified both by ZADD that will update the element score, and as a side effect, its position on the sorted set, and by ZINCRBY that can be used in order to update the score relatively to its previous value.

The current score of an element can be retrieved using the ZSCORE command, that can also be used to verify if an element already exists or not.

For an introduction to sorted sets, see the data types page on sorted sets.

Elements with the same score

While the same element can't be repeated in a sorted set since every element is unique, it is possible to add multiple different elements *having the same score*. When multiple elements have the same score, they are *ordered lexicographically* (they are still ordered by score as a first key, however, locally, all the elements with the same score are relatively ordered lexicographically).

The lexicographic ordering used is binary, it compares strings as array of bytes.

If the user inserts all the elements in a sorted set with the same score (for example 0), all the elements of the sorted set are sorted lexicographically, and range queries on elements are possible using the command ZRANGEBYLEX (Note: it is also possible to query sorted sets by range of scores using ZRANGEBYSCORE).

Return value

Integer reply, specifically:

Related commands

ZADD
ZCARD
ZCOUNT
ZINCRBY
ZINTERSTORE
ZLEXCOUNT
ZRANGE
ZRANGEBYLEX
ZRANGEBYSCORE
ZRANK
ZREM
ZREMRANGEBYLEX
ZREMRANGEBYRANK
ZREMRANGEBYSCORE
ZREVRANGE
ZREVRANGEBYSCORE
ZREVRANK
ZSCAN
ZSCORE
ZUNIONSTORE

- The number of elements added to the sorted sets, not including elements already existing for which the score was updated.

History

- `>= 2.4`: Accepts multiple elements. In Redis versions older than 2.4 it was possible to add or update a single member per call.

Examples

```
redis> ZADD myzset 1 "one"
(integer) 1
redis> ZADD myzset 1 "uno"
(integer) 1
redis> ZADD myzset 2 "two" 3 "three"
(integer) 2
redis> ZRANGE myzset 0 -1 WITHSCORES
1) "one"
2) "1"
3) "uno"
4) "1"
5) "two"
6) "2"
7) "three"
8) "3"
redis> zadd myset 1.5 "hi"
(integer) 1
redis> zadd myset 2.2 "hello"
(integer) 1
redis> zadd myset 2.3 "by"
(integer) 1
```

score for all elements (e.g. .0), then these elements will be ordered by their natural binary ordering. So e.g. you don't need to try to compute a score for your strings, in order to have them ordered properly.

23 • Reply • Share ›

Chaoran Yang ypocat • 2 years ago

Can we safely rely on this? I think it is implementation specific. This might not be the case in future when implementation changes.

7 • Reply • Share ›

isteve ypocat • 2 years ago

Thanks a lot for this explanation. I was about to build a string to score algo based on their alphabetical order when I saw your comment. You saved me some work and reduced the complexity of my code. Thanks!

3 • Reply • Share ›

Chris Maddox • 2 years ago

Are there any plans to add TTL expiration to members of sorted sets? This would be very helpful, to add keys who TTL's match the TTL of the corresponding data (e.g. hash)

20 • Reply • Share ›

Mathew_Wang • 3 years ago

what is available range for score? min/max available value for score.

7 • Reply • Share ›

finallygo • a year ago

I think the "Time complexity" is O(log(N) * M) (M is the number of elements will to add) , isn't it??

5 • Reply • Share ›

Aleksandr Conseque • 3 years ago

my redis said second ZADD myzset 1 "uno" from example above has result (integer) 0 is error inside my redis or in example above?

4 • Reply • Share ›

Joe Niedzwiecki Aleksandr Conseque • 8 months ago

I thought this for awhile also Alek. So after looking at this again and again take a look again at the sample. We try to add "two" twice. It seems this redis command sees that we already have "two" in there and instead get the return value of 0 meaning "two" was not added again. Notice when we print out the results at the end we only have "two" once.

1 • Reply • Share ›

Yifei Teng Aleksandr Conseque • 4 months ago

Thiel Yang ➚ Aleksandr Conseque • 4 months ago

Same problem here. I am using redis 2.4.6. It just won't behave like the docs said to update the score.

^ | v • Reply • Share ›

kaulie_gl • 2 years ago

how to control the elements' sequence in redis,which all share with same score ?
i hope zadd can add a param to indicate varied stretagy ,like lpush (to head), random (default), rpush (to tail).
for example:
zadd mykey elem1 100,
zadd mykey elem2 100,
zadd mykey elme3 100,
i hope i can control where elem3 and elme2 be placed in ,
elem1,elem2,elem3
elme2,elem1,elem3
elme2,elem3,elem1
...
...

6 ^ | v • Reply • Share ›

Krotton • a year ago

ZADDNX would be a wonderful addition. I'm in a situation when I should only add new values to the set and never update their scores. Instead of just using a conditional add in my transaction, I have to run ZRANK before it and only execute ZADD if the former returns nil. As an effect there is a possibility for a (luckily not dangerous in my case) race condition to occur.

3 ^ | v • Reply • Share ›

Thanasis Polychronakis ➚ Krotton • a year ago

+1 here's a typical use case:

I want to keep ordered sets and use a timestamp as ranking. If the member already exists i dont want to update the rank (timestamp) as i need to know when the member was first added so i can maintain FIFO.

1 ^ | v • Reply • Share ›

Vasya ➚ Krotton • a year ago

You can use Lua scripting to add ZADDNX yourself. And it's much simpler and leads to less complex code than using transactions.

1 ^ | v • Reply • Share ›

John Crepezzi ➚ Krotton • a year ago

Also - Krotton, you should be using ZSCORE instead of ZRANK I think

The performance characteristics are O(1) if all you're checking is that the key is around

1 • Reply • Share ›

Krotton ➚ John Crepezzi • a year ago

You're definitely right, thanks! I didn't spot the difference in complexity when browsing through the docs.

• Reply • Share ›

John Crepezzi ➚ Krotton • a year ago

+1 on this - same situation for me

• Reply • Share ›

Jurjen • a year ago

Is it possible to use ZADD without updating the score if the given member already exists?

2 • Reply • Share ›

chaoticinferno ➚ Jurjen • a year ago

Why would you want to do that? You'd be effectively doing a relatively expensive noop.

2 • Reply • Share ›

Дмитрий Шалашов ➚ chaoticinferno • a year ago

Because he might be adding not one element but portions of data every time.

1 • Reply • Share ›

tama dinata • 2 years ago

xzxcvxcvxcv

1 • Reply • Share ›

Kishore Relangi • 7 months ago

How to get score of a sorted set member ?

something like
zadd myzset 10 key1

now I need to get the score of key1 of myzset (sorted set name)
zget myzset key1 (something like this will be helpfull)

The workaround I am using is
zincrby myzset 0 key1
--- (which will return the score of the key1, but there is no way to know whether the key1

is a member of the set or not). So, next approach is to use zrank to know whether the given key is exists or not before using zincrby to get the value

zrank myzset key1 -- if returns an integer then
zincrby myzset 0 key1 -- to get actual score

But, it is a good idea to have single command to get the score.

• Reply • Share ›

Shelfy App • 3 months ago

"When multiple elements have the same score, they are ordered lexicographically"

But how are non ASCII members(strings) sorted when they have equal score?

Would like to have some feedback please.
Thakns

• Reply • Share ›

Bojan Jovanovic • 10 months ago

Is it possible to have Long only score and Long only members? Both would speed up everything considerably, and there are no unpredictable floating point issues. That would be a great enhancement.

• Reply • Share ›

Vinodkumar Saravana • 2 years ago

Is it possible to insert a members at a given index...just like LINDEX doing...

• Reply • Share ›

Vinodkumar Saravana ➚ Vinodkumar Saravana • 2 years ago

sorry just like LINSERT

1 • Reply • Share ›

primijos • 2 years ago

It would be great to allow multiple values for the score (a tuple of values) and to allow not only integer but also alphabetic (utf-8) values. This would allow, for example, to keep sorted sets for more than one hashes values (emulating composite indexes, a la SQL)

I'm struggling on how to achieve that only with the available tools/abstractions right now on redis, but I can't manage to get a nice scenario to be able to simultaneously apply multiple filteres on hashes without the need to zrangebyscore + zinterstore continuosly (with the added overhead). That ability (composite + alphabetic scoring) would help in this case.

On the other hand, maybe some kind of pipelining between operations (being able to use the results of a zrangebyscore as a sorted set directly, without the need to store it, even in Lua) or "virtual zsets" just for on-time use could help here

in Lua) or "virtual zsets" just for on-time use could help here.

Just some ideas... :)

• Reply • Share ›

Denis K primijos • a year ago

That would be great! I have encountered the same problem and the solution is to use very complex hasing algorithm. It would be great to have multikey, which allows strings.

• Reply • Share ›

forest • 2 years ago

sorted set can bind some extra info to every element? this is very useful to some cases such as sns mail. for example, uid:$uid:mailers store the mailers, and bind the last mail-id or mail-count to every element is very useful.

• Reply • Share ›

张涛 • 2 years ago

good job

• Reply • Share ›

brycebaril • 2 years ago

Did some playing around with using ZADD to create a new ZSET of 10000 elements (using 2.6.0_rc3):

I created the data like so:
perl -le 'for (1..10000) { my $r = rand() * 1000; print "zadd testzset $r \"$_\"" }' > single

Then created three copies and modified them in vim:
"multi" (added multi to the top and exec to the bottom)
"combined" (block deleted the 'zadd testzset ' from each line, joined all lines and added 'zadd testzset ' back to the first line

(All time commands were done multiple times and timing was pretty stable per run)

Single ZADD statements:
time cat single | redis-cli
(integer) 1
(integer) 1
(integer) 1
...

see more

• Reply • Share ›

亮 于 • 2 years ago

very good

• Reply • Share ›

Kien Nguyen • 2 years ago

How many elements does ZADD accept per call?

• Reply • Share ›

lala • 3 years ago

is there a way to insert a member that has more than 1 score (the other 1 as a tiebreaker)? e.g member1 has score 2 and 5, and member 2 has score 2 and 3... and it should be returned as "member2", "member1" when we're using the ZRANGEBYSCORE..

• Reply • Share ›

Rfid • 3 years ago

A note to state it is Zero based (starting index value is 0) is useful. Others ate 1 based.

• Reply • Share ›

pskirko • 3 years ago

Hi, I posted the following question on Stack Overflow as well:

http://stackoverflow.com/quest...

The redis documentation for ZADD states the operation is O(logN).

However, does anyone know if ZADD is better than O(logN) when the inserted element is at the beginning or end of the sort order?

E.g. for certain implementations this could be O(1).

Specifically, the redis tutorial states that:

Sorted sets are implemented via a dual-ported data structure containing
both a skip list and an hash table, so every time we add an element
Redis performs an O(log(N)) operation.

It seems plausible to modify a skip list to support O(k) insert at beginning and end, where k is the max level of the skip list.

• Reply • Share ›

Pieter Noordhuis Mod ➚ pskirko • 3 years ago

That is correct. The sorted set relies on an RNG to determine the number of levels per node (it's a probabilistic data structure). Inserting/deleting an element

at the beginning of the skiplist can be O(1), while the theoretical worst case performance is O(N) (with every node having the same level). However, the amortized time complexity is O(log N) when you take in account the distribution of the levels among the nodes.

2 • Reply • Share ›

kumar • 3 years ago

Is it O(log(N)) for multiple member inserts

• Reply • Share ›

Pieter Noordhuis Mod ➦ kumar • 3 years ago

It is O(log N) for every insert/update.

1 • Reply • Share ›

Dan ➦ Pieter Noordhuis • 3 years ago

To clarify, say you're inserting M members into a much larger sorted set of size N...
Are you saying this operation is O(M log(N)) or O(log N)?
In other words, is there any advantage to batching ZADDs together?
Thank you.

• Reply • Share ›

Peter Scott ➦ Dan • 2 years ago

It's O(M log(N+M)), no matter what approach you take.

• Reply • Share ›

Noah ➦ Dan • 3 years ago

I'd like to know this also??? Is there any advantage to batching ZADDs together?

• Reply • Share ›

yoav ➦ Noah • 2 years ago