**Redis: Hash and Sorted Set Values**

**Recap:**

In previous article we learned about:

* Basic Redis CLI commands for List Values
* Basic Redis CLI commands for Set Values

In this article we will see basic commands used to store Hash and Sorted Set Values.

**Redis: Hash Values**

Hashes are a field-value pairs that are used to store objects. They are quiet handy as it has an additional attribute associated with the value. Here are basic Add, Read and Delete operations on Hash values using Redis CLI commands.

1. HMSET – Sets the hashes.

**Syntax –** HMSET key-name Field Value [Field Value] [Field Value]

* + This saves the value in Field Value pair. If the field already has any value, it gets overridden if reassigned with different value.
  + Alternatively to set only one field value, HSET can be used.
  + Moreover you can use HSETNX which Sets the value for a field if the field does not already exists.

1. HMGET -- Gets the hash values

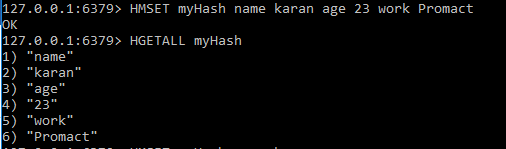
**Syntax –** HMGET key-name Field1 [Field2] [Field3]

* + If field is not present then it returns nil.
  + Alternatively to get only one field value, HGET can be used or to get all field value pairs, HGETALL command can also be used.
  + To get all field names only, you can fire HKEYS command and to get all values only, you can fire HVALS

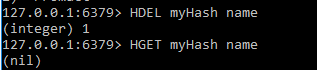
1. HDEL – Removes the fields from Hash

**Syntax –** HDEL key—name Field1 [Field2] [Field3]

* + It returns the number of fields removed.



1 HMSET and HGETALL commands



2 HDEL command

These were basic CLI commands for Hashes. You can checkout full list of commands [here](https://redis.io/commands#hash).

**Redis: Sorted Sets**

Sorted sets as the name says are ordered collection of values. Moreover, sorted sets have a floating point number called ‘Score’ which is mapped with the value (somewhat resembling hashes, just hashes have fields, but sorted sets have scores). For ordering following rule is applied:

* If score1 and score2 in a set with element1 and element2 values are saved then,
  + Element1 will precede Element2 if value of score1 is more than value of score2. For ex.

Score value

123 34

134 12

* + In above case even if 34 value is greater than 12 value or is entered first in the set, the value 12 will precede over 34 because score of value 12 is more than score of value 34.
* If score1 and score2 are equal, then the values’ lexicography will be taken into account. For ex.

Score Value

123 defg

123 abcd

* + In above case scores of both values are equal, however abcd value will be given precedence since ‘a’ comes first and‘d’ comes after in English alphabets (dictionary). Values can never be same because ‘Sets’ intrinsic property is to save unique elements.

Having understood the properties of sorted sets, let’s have a look on basic redis CLI commands to store Sorted Sets values.

1. ZADD – Adds Score-Value pairs in sorted set

**Syntax --** ZADD key [NX|XX] [CH] [INCR] score member [score member]

* + NX -> add new elements despite having that element in set.
  + XX -> Updates existing elements and does not add new elements
  + The default return value after successful execution of this command is total number of elements added in set. I used the word ‘default’ because using CH you can change the return type to return the total number of elements changed.
  + If INCR is used, only one Score Member pair can be added into the set at a time.

1. ZRANGE – Prints out set elements

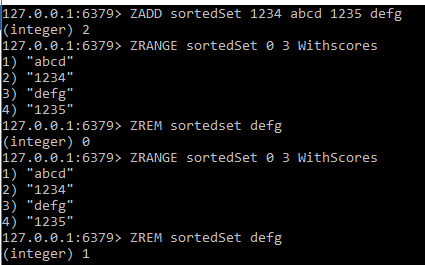
**Syntax –** ZRANGE key StartIndex EndIndex [WithScores]

* + If WithScores is used, then scores will also be printed, else only the values will be printed between start and end indices.

1. ZREM – Removes set element

**Syntax -** ZREM key member [member]

* + Returns number of members removed from set



3 ZADD, ZRANGE and ZREM commands

These were basic introduction and commands of sorted sets. To play with more commands and learn more, you can checkout [here](https://redis.io/commands#sorted_set).

Thus, this was complete tutorial series of basic CLI commands used for values of different data types. I hope that these posts would have help you getting started with redis with less resistance.