Key notes of implementation of Redis in Java

About Redis:

Redis is an open source (BSD licensed), in-memory data structure store, used as a database, cache and message broker. It supports data structures such as strings, hashes, lists, sets, sorted sets with range queries, bitmaps, etc.

Functionalities which I used during implementation of Redis:

- Programming language: Java.

  Q. Why Java?

  Java is a widely used programming language expressly designed for use in the distributed environment of the internet. The main purpose of redis as in the memory caching system which makes the operations much faster than simple database queries. To satisfy the purpose Java provides in built data structures. To satisfy time constraints as Redis provides operations in a much faster way than conventional sql database queries, I used some of the data structures like Maps based and Trees based collection class and to fulfil other features Java multi threading concepts is been used.

  Q. What are the improvements that can be made and how?

  Problem: In the current execution once the expire command is executed for some key and before given time pass away, again if expire command is been called for the same key, two threads created to remove the same data entry, which will creates problem in execution.

  Solution: problem can be handled by killing the current allocated thread and recreating thread for the remaining updated time. (If we want to consider latest EXPIRE command).

  Q. What data structures are used and why?

  IIn the implementation I mainly used HashSets and TreeSets. As HashMap supports O(1) time for insertion, deletion and finding operations.

  For Z functions I used TreeSets. As we want to store data in sorted manner treesets are the best option, as they support basic operations in O(logn) time.

  I have used HashMap for normal GET,SET,EXPIRE operation, other HashMap storing sorted set key and SortedSet object.

  Note: If we want to allow duplication of data then we need to use Map data structure. For the purpose of redis as nosql database duplication of data should not be allowed so, for this purpose usage of set data structure is preferable.

  Q. Does this implementation support multi-thread operations?

  Yes, because for EXPIRE commands if we use a single thread then we need to wait for that milliseconds pause for system response as for that milliseconds main thread sleeps but this is not practical scenario. Hence i am creating a seperate thread for executing the EXPIRE commands.

Nilay Shah
+91 8347773231.
201601230@daiict.ac.in