11 Common Web Use Cases Solved in Redis

Wednesday, July 6, 2011 at 9:16AM

General Chicken in Strategy

In How to take advantage of Redis just adding it to your stack Salvatore 'antirez' Sanfilippo shows how to solve some common problems in Redis by taking advantage of its unique data structure handling capabilities. Common Redis primitives like LPUSH, and LTRIM, and LREM are used to accomplish tasks programmers need to get done, but that can be hard or slow in more traditional stores. A very useful and practical article. How would you accomplish these tasks in your framework?

1. **Show latest items listings in your home page**. This is a live in-memory cache and is very fast. LPUSH is used to insert a content ID at the head of the list stored at a key. LTRIM is used to limit the number of items in the list to 5000. If the user needs to page beyond this cache only then are they sent to the database.
2. **Deletion and filtering**. If a cached article is deleted it can be removed from the cache using LREM.
3. **Leaderboards and related problems**. A leader board is a set sorted by score. The ZADD commands implements this directly and the ZREVRANGE command can be used to get the top 100 users by score and ZRANK can be used to get a users rank. Very direct and easy.
4. **Order by user votes and time**. This is a leaderboard like Reddit where the score is formula the changes over time. LPUSH + LTRIM are used to add an article to a list. A background task polls the list and recomputes the order of the list and ZADD is used to populate

the list in the new order. This list can be retrieved very fast by even a heavily loaded site. This should be easier, the need for the polling code isn't elegant.

5. **Implement expires on items**. To keep a sorted list by time then use unix time as the key. The difficult task of expiring items is implemented by indexing current_time+time_to_live. Another background worker is used to make queries using ZRANGE ... with SCORES and delete timed out entries.
6. **Counting stuff**. Keeping stats of all kinds is common, say you want to know when to block an IP addresss. The INCRBY command makes it easy to atomically keep counters; GETSET to atomically clear the counter; the *expire* attribute can be used to tell when an key should be deleted.
7. **Unique N items in a given amount of time**. This is the unique visitors problem and can be solved using SADD for each pageview. SADD won't add a member to a set if it already exists.
8. **Real time analysis of what is happening, for stats, anti spam, or whatever**. Using Redis primitives it's much simpler to implement a spam filtering system or other real-time tracking system.
9. **Pub/Sub**. Keeping a map of who is interested in updates to what data is a common task in systems. Redis has a pub/sub feature to make this easy using commands like SUBSCRIBE, UNSUBSCRIBE, and PUBLISH.
10. **Queues**. Queues are everywhere in programming. In addition to the push and pop type commands, Redis has blocking queue commands so a program can wait on work being added to the queue by another program. You can also do interesting things implement a rotating queue of RSS feeds to update.
11. **Caching**. Redis can be used in the same manner as memcache.

The take home is to not endlessly engage in model wars, but see what can be accomplished by composing powerful and simple primitives together.

Certainly you can write specialized code to do all these operations, but Redis makes it much easier to implement and reason about.

Please read the original article for more details.

Related Articles

- Resque - Redis-backed library for creating background jobs, placing those jobs on multiple queues, and processing them later.
- Hacker News thread.
