**Các lệnh cơ bản trong Redis**

# **Redis Key**

Redis key là các lệnh sử dụng để quản lý các key trong redis. Với cú pháp như sau:

COMMAND KEY\_NAME [VALUE]

Các lệnh thường dùng:

|  |  |  |
| --- | --- | --- |
| **STT** | **Command** | **Ý nghĩa** |
| 1 | DEL key | Xóa key nếu nó tồn tại |
| 2 | EXISTS key | Kiểm tra key có tồn tại không |
| 3 | EXPIRE key n | Đặt expire time cho key sau n giây |
| 4 | PERSIST key | Xóa expire time của key |
| 6 | TTL key | Lấy thời gian sống của key (giây) |
| 7 | RENAME key newkey | Đổi tên key sang newkey, nếu newkey đã tồn tại giá trị của nó sẽ bị ghi đè bởi giá trị của key |
| 8 | RENAMENX key newkey | Đổi tên key sang newkey nếu newkey chưa tồn tại |
| 9 | TYPE key | Lấy loại dữ liệu được lưu trữ bởi key |

# **Redis String**

Redis string là lệnh sử dụng để quản lý các key/value trong đó value có giá trị string trong redis.

Các lệnh thường dùng:

|  |  |  |
| --- | --- | --- |
| **STT** | **Command** | **Ý nghĩa** |
| 1 | SET key value | Đặt giá trị value cho key |
| 2 | GET key | Lấy giá trị lưu trữ bởi key |
| 3 | DEL key | Xoá key và giá trị tương ứng |
| 4 | GETSET key value | Lấy ra giá trị cũ và đặt giá trị mới cho key |
| 5 | MGET key1 key2 ... | Lấy giá trị của nhiều key theo thứ tự |
| 6 | SETEX key seconds value | Đặt giá trị và thời gian expire cho key |
| 7 | SETNX key value | Đặt giá trị cho key nếu key chưa tồn tại |
| 8 | RENAMENX key newkey | Đổi tên key sang newkey nếu newkey chưa tồn tại |
| 9 | STRLEN key | Lấy độ dài giá trị lưu trữ bởi key |
| 9 | APPEND key value | Thêm vào sau giá trị lưu trữ bởi key là value |
| 10 | INCR key | Tăng giá trị lưu trữ của key (số nguyên) 1 đơn vị |
| 11 | INCRBY key n | Tăng giá trị lưu trữ của key (số nguyên) n đơn vị |
| 12 | DECR key | Giảm giá trị lưu trữ của key (số nguyên) 1 đơn vị |
| 11 | DECRBY key n | Giảm giá trị lưu trữ của key (số nguyên) n đơn vị |

# **Redis List**

List trong Redis là linked list, lưu trữ 1 danh sách có thứ tự (trước sau) của các string. Cách lưu trữ này giúp cho thời gian add thêm 1 phần tử vào đầu hoặc cuối list là hằng số, bất kể size của list là bao nhiêu. Lợi thế này cũng có 1 mặt trái là việc truy xuất đến phần tử theo index của linked list là lâu hơn rất nhiều so với array.

Redis list là lệnh sử dụng để quản lý các key/value trong đó value có giá trị là một list (danh sách). List là kiểu dữ liệu khá phổ biến, có 2 kiểu list thường dùng là stack (vào sau ra trước) và queue (vào trước ra trước).

Các lệnh thường dùng:

|  |  |  |
| --- | --- | --- |
| **STT** | **Command** | **Ý nghĩa** |
| 1 | RPUSH | Cho thêm 1 giá trị vào phía bên phải của list |
| 2 | LRANGE | Lấy 1 dải giá trị của list |
| 3 | LPOP key | Lấy phần tử ở đầu danh sách |
| 4 | LINDEX | Lấy ra phần tử của list dựa theo index |
| 5 | LRANGE key start stop | Lấy các phần tử trong list từ vị trí start đến vị trí stop |
| 6 | LSET key index value | Đặt lại giá trị tại index bằng value |
| 7 | RPOP key | Lấy giá trị ở cuối danh sách |
| 8 | RPUSH key value1 value2 ... | Thêm phần tử value1 value2 ... vào cuối danh sách |
| 9 | LINSERT key BEFORE value1 value2 | Thêm phần tử value2 vào trước phần tử value1 trong danh sách |
| 10 | LINSERT key AFTER value1 value2 | Thêm phần tử value2 vào sau phần tử value1 trong danh sách |

Trang redis.io đưa ra 2 trường hợp phổ biến cho việc dùng list. 1 là lưu lại các post mới nhất của users trên mạng xã hội, điển hình là mạng xã hội nổi tiếng Twitter dùng Redis cho việc này (lưu các tweets mới nhất của users). 2 là xây dựng mô hình tương tác giữa consumer và producer, trong đó producer đưa item vào list và consumer dùng các item này theo thứ tự quy định trong list.

# **Redis Set**

Set trong Redis khá giống với list, nhưng khác 1 điều là các phần tử trong set không được sắp xếp theo thứ tự nào cả. Tuy nhiên, Redis đã tăng performance khi làm việc với set bằng cách sử dụng 1 bảng băm (hash table) để lưu trữ các phần tử của set. Hiểu đơn giản thì mỗi item được add vào set sẽ là 1 key trong bảng băm, còn value thì không có. Việc làm này giúp theo tác truy xuất dữ liệu trên SET nhanh hơn nhiều (do tận dụng ưu thế về tốc độ tìm kiếm trên bảng băm), nhất là khi muốn đảm bảo không bị trùng lặp phần tử trong set.

Redis set là lệnh sử dụng để quản lý các key/value trong đó value có giá trị là một set (tập hợp). Các giá trị trong tập hợp là duy nhất không bị trùng lặp

Các lệnh thường dùng:

|  |  |  |
| --- | --- | --- |
| **STT** | **Command** | **Ý nghĩa** |
| 1 | SADD key value1 value2 .. | Thêm các giá trị value1 value2 ... vào tập hợp |
| 2 | SCARD key | Lấy số lượng phần tử trong tập hợp |
| 3 | SMEMBERS key | Lấy các phần tử trong tập hợp |
| 4 | SPOP key | Xóa bỏ ngẫu nhiên một phần tử trong tập hợp và trả về giá trị phần tử đó |

Cũng cần nói thêm rằng độ phức tạp tính toán của SISMEMBER là O(1), đây là ưu thế rất lớn của set so với dùng list khi sử dụng bảng băm. Chi tiết hơn về độ phức tạp tính toán trong Redis sẽ được nhắc đến ở phần sau của bài viết này.

# **Redis Hash**

Không giống như LIST và SET lưu trữ 1 tập dữ liệu là các string, HASH lưu trữ tập các map của key và value. Key vẫn là string, còn value có thể là string hoặc số. Nếu là số thì chúng ta có thể làm các thao tác tăng, giảm giá trị 1 cách đơn giản. HASH được coi là 1 mô hình thu nhỏ của Redis, khi dữ liệu được tổ chức dạng key-value.

Redis hash là lệnh sử dụng để quản lý các key/value trong đó value có giá trị là hash. Hash là kiểu dữ liệu khá phổ biến, thường được dùng để lưu trữ các object.

|  |  |  |
| --- | --- | --- |
| **STT** | **Command** | **Ý nghĩa** |
| 1 | HSET key field value | Đặt giá trị cho field là value trong hash |
| 2 | HGET key field | Lấy giá trị của field trong hash |
| 3 | HDEL key field1 field2 ... | Xóa field1, field2 ... trong hash |
| 4 | HEXISTS key field | Kiểm tra file có tồn tại trong hash không |
| 5 | HGETALL key | Lấy tất cả các field và value của nó trong hash |
| 6 | HINCRBY key field n | Tăng giá trị của field (số nguyên) lên n đơn vị |
| 7 | HDECRBY key field n | Giảm giá trị của field (số nguyên) lên n đơn vị |
| 8 | HINCRBYFLOAT key field f | Tăng giá trị của field (số thực) lên f |
| 9 | HDECRBYFLOAT key field n | Giảm giá trị của field (số thực) f |
| 10 | HKEYS key | Lấy tất cả các field của hash |
| 11 | HVALS key | Lấy tất cả các value của hash |
| 12 | HLEN key | Lấy số lượng field của hash |
| 13 | HMSET key field1 value1 field2 value2 ... | Đặt giá trị cho các field1 giá trị value1 field2 giá trị value2 ... |
| 14 | HMGET key field1 field2 ... | Lấy giá trị của các field1 field2 ... |

# **Redis Sorted Set**

Sorted Set (ZSET) là 1 phiên bản đầy đủ của set, khi mà phần value của item được thiết lập, và bắt buộc là 1 số (float number) được gọi là score. Ở điểm này thì zset khá giống với hash khi lưu trữ 1 cặp key, value (trong zset gọi là member và score). Và vì là “sorted”, nên các cặp member-score được add vào sorted set sẽ được sắp xếp theo thứ tự của các score, nếu score trùng nhau thì tiếp tục sắp xếp theo member. Ngoài ra cũng cần chú ý là không cho phép 2 phần tử khác nhau của zset có member trùng nhau.

Redis sorted set là lệnh sử dụng để quản lý các key/value trong đó value có giá trị là một sorted set (tập hợp được sắp xếp theo điểm/độ ưu tiên từ thấp đến cao). Các giá trị trong sorted set là duy nhất không bị trùng lặp.

Các lệnh thường dùng:

|  |  |  |
| --- | --- | --- |
| **STT** | **Command** | **Ý nghĩa** |
| 1 | ZADD key score1 value1 score2 value2 .. | Thêm các phần tử value1 value2 vào sorted set với độ ưu tiên tương ứng là score1 và score2 |
| 2 | SCARD key | Lấy số lượng phần tử trong sorted set |
| 3 | ZRANGE key start stop | Lấy các phần tử trong tập hợp từ start đến stop |
| 4 | ZRANGE key start stop WITHSCORES | Lấy các phần tử trong tập hợp từ start đến stop kèm theo giá trị score của chúng |
| 5 | ZSCORE key member | Lấy giá trị score của member |
| 6 | ZRANK key member | Lấy vị trí của member trong sorted set |
| 7 | ZCOUNT key score1 score2 | Đếm số member có score tương ứng trong đoạn score1 đến score2 |

ZSET là 1 cấu trúc dữ liệu đặc biệt của riêng Redis, nó chuyên dùng cho các bài toán dạng tìm “top”. Top người dùng theo score, top webpage theo số lượng view, “top whatever”. Cách Redis lưu trữ dữ liệu zset cũng rất thú vị, chúng ta sẽ tìm hiểu ở phần tiếp theo của bài viết.

1. **Redis Transaction**

Một điểm khá thú vị trong Redis là transaction. Redis transaction cho phép một nhóm các lệnh thực hiện theo thứ tự cho đến khi lệnh cuối cùng được thực hiện xong. Khi này Redis mới cập nhật đồng thời dữ liệu thay đổi bởi nhóm lệnh này. Redis transaction bắt đầu bằng lệnh MULTI và kết thúc bằng lệnh EXEC.

Các lệnh thường dùng:

|  |  |  |
| --- | --- | --- |
| **STT** | **Command** | **Ý nghĩa** |
| 1 | MULTI | Đánh dấu bắt đầu khối lệnh transaction |
| 2 | EXEC | Thực hiện khối lệnh |