redis
Clients

Java SpringBoot New
- Introduction
- Strings
- Lists
- Sets
- ZSets
- Hashes
- Streams
- Common Keys
- Pipelining
- Pub/Sub
- Master/Replica
- Sentinel
- Cluster
- ---
- Auto Config
- Manual Config
- Load Balancing (readFrom)
- RedisTemplate
- Connection Pool & Thread
- Async Spring & Lettuce
- DB select
- Spring Multi Data Source
- Lettuce Multi Data Source
- Spring Project Create
- Spring Project Eclipse
- Spring Project IntelliJ
- ---
- Spring Session Standalone
- Spring Session MasterRepli
- Spring Session Sentinel
- Spring Session Cluster

Java Lettuce(Spring)New

Java Lettuce(Plain)

Java Jedis

Java Redisson

C Hiredis

C# StackExchange

PHP PhpRedis

PHP Predis

Redis Admin & Monitoring Tool

Spring Data Redis ZSets

Spring Data Redis ZSets

Java Spring Framework를 사용한 레디스 정렬셋(ZSets) 명령 사용법입니다.

ZSets 소스

Redis4_ZSet.java

```
package com.redisgate.redis;

import lombok.extern.slf4j.Slf4j;
import org.springframework.data.redis.core.*;
import org.springframework.data.redis.core.ZSetOperations.TypedTuple;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.data.domain.Range;
//import io.lettuce.core.Range;

import java.util.HashSet;
import java.util.Set;
import java.util.concurrent.TimeUnit;

@RestController
@Slf4j
public class Redis04_ZSet {

   private final StringRedisTemplate stringRedisTemplate;
   private final ZSetOperations<String,String> zsetOperations;

   public Redis04_ZSet(StringRedisTemplate stringRedisTemplate) {
      this.stringRedisTemplate = stringRedisTemplate;
      this.zsetOperations = stringRedisTemplate.opsForZSet();
   }
   // 여기에 각 명령(메서드) 별 소스가 들어갑니다.
}
```

각 명령(메서드) 별 표시

ZADD

```
// 예제 1) ZADD: 집합에 데이터를 스코어와 함께 추가
// ZADD key [NX|XX] [GT|LT] [CH] [INCR] score member [score member ...]
// http://localhost:8080/zadd/zset1
@GetMapping("/zadd/{key}")
public String zadd(@PathVariable("key") String key) {
    String msg = "예제 1) ZADD(add) -> ";
    // Boolean add(K key, V value, double score); Lettuce와 value, score 위치가 다름.
    Boolean result = zsetOperations.add(key,"user01",1d);
    System.out.println(msg+result);

    // Long add(K key, Set<TypedTuple<V>> tuples)
    Set<TypedTuple<String>> members = new HashSet<>();
    members.add(new DefaultTypedTuple<>("user02",2d));
    members.add(new DefaultTypedTuple<>("user03",3d));
    members.add(new DefaultTypedTuple<>("user04",4d));
    members.add(new DefaultTypedTuple<>("user05",5d));
    members.add(new DefaultTypedTuple<>("user06",6d));
    Long result2 = zsetOperations.add(key,members);
    System.out.println(msg+result2);

    msg = "예제 1) ZADD(addIfAbsent) -> ";
    // addIfAbsent: 해당 맴버가 기존에 없을 경우에만 추가한다.
    // 대기열에 이미 있는지 확인
    // Boolean addIfAbsent(K key, V value, double score)
    Boolean result3 = zsetOperations.addIfAbsent(key,"user04",4d);
    System.out.println("ZADD NX(addIfAbsent) user4 -> "+result3);
    Boolean result4 = zsetOperations.addIfAbsent(key,"user07",7d);
    System.out.println("ZADD NX(addIfAbsent) user7 -> "+result4);
    // (대기열) 순서 조회 -> 0부터 시작.
    Long result5 = zsetOperations.rank(key,"user04");
    System.out.println("user04 rank -> "+result5);

    // 스코어를 현재 시작(millisecond)로 입력
    result = zsetOperations.add(key, "user08", System.currentTimeMillis());
    System.out.println("ZADD user08 -> "+result);
    result = zsetOperations.add(key, "user09", System.currentTimeMillis());
    System.out.println("ZADD user09 -> "+result);

    // 테스트 데이터 입력
    for (int i=10; i<=30; i++) {
        zsetOperations.add(key,"user"+String.format("%02d",i), i);
    }
    msg += "OK";
    System.out.println(msg);
    return msg;
}
```

ZREM

```
// 예제 9) ZREM: 멤버 삭제
// ZREM key member [member]
// http://localhost:8080/zrem/zset1
@GetMapping("/zrem/{key}")
public String zrem(@PathVariable("key") String key) {
    String msg = "예제 9) ZREM(remove) -> ";
    Long result = zsetOperations.remove(key, "user04");
    msg += result;
    System.out.println(msg);
    return msg;
}
```

ZRANGE

```
// 예제 3) ZRANGE: 멤버 리스트 조회(스코어 작은것 부터)
// ZRANGE key start stop [REV] [[BYSCORE|BYLEX] [LIMIT offset count]] [withscores]
// http://localhost:8080/zrange/zset1
@GetMapping("/zrange/{key}")
public String zrange(@PathVariable("key") String key) {
    String msg = "예제 3) ZRANGE(range) -> ";
    // 멤버 조회
    // Set<V> range(K key, long start, long end)
    Set<String> result = zsetOperations.range(key,0,-1);
    System.out.println(msg+result);
    // 멤버와 스코어 조회
    // Set<TypedTuple<V>> rangeWithScores(K key, long start, long end)
    Set<TypedTuple<String>> result2 = zsetOperations.rangeWithScores(key,0,-1);
    if (result2 == null) return null;
    for (TypedTuple<String> member: result2) {
        System.out.println(member.getValue()+" "+member.getScore());
    }
    msg += result2.size();
    System.out.println(msg);
    return msg;
}
```

ZCARD

```
// 예제 2) ZCARD: 멤버 개수 조회
// ZCARD key
// http://localhost:8080/zcard/zset1
@GetMapping("/zcard/{key}")
public String zcard(@PathVariable("key") String key) {
    String msg = "예제 2) ZCARD(size) -> ";
    Long result = zsetOperations.size(key);
    System.out.println(msg+result);
    return msg+result;
}
```

ZCOUNT

```
// 예제 7) ZCOUNT: 스코어로 범위를 지정해서 개수 조회
// ZCOUNT key min max
// http://localhost:8080/zcount/zset1
@GetMapping("/zcount/{key}")
public String zcount(@PathVariable("key") String key) {
    String msg = "예제 7) ZCOUNT(count) -> ";
    // 스코어로 조회
    // Long count(K key, double min, double max)
    Long result = zsetOperations.count(key, 3d, 5d);
    System.out.println(msg+result);

    // 멤버로 조회: 스코어가 모두 0일 때
    // Long lexCount(K key, Range<String> range)
    Range<String> range = Range.closed("member03","member05");
    result = zsetOperations.lexCount("zset2", range);
    msg = "예제 7) ZCOUNT(lexCount) -> ";
    System.out.println(msg+result);
    return msg+result;
}
```

ZRANGEBYSCORE

```
// 예제 5) ZRANGEBYSCORE: 스코어로 범위를 지정해서 조회
// ZRANGEBYSCORE key min max [withscores] [limit offset count]
// Double.MIN_VALUE, Double.MAX_VALUE
// http://localhost:8080/zrangebyscore/zset1
@GetMapping("/zrangebyscore/{key}")
public String zrangebyscore(@PathVariable("key") String key) {
    String msg = "예제 5) ZRANGEBYSCORE(reverseRange) -> ";
    System.out.println(msg);
    // 멤버 전체 조회
    // Set<V> rangeByScore(K key, double min, double max)
    Set<String> result = zsetOperations.rangeByScore(key, Double.MIN_VALUE, Double.MAX_VALUE);
    System.out.println("모두 -> "+result);

    // 범위 지정 조회 (스코어 2, 5 포함)
    Set<String> result2 = zsetOperations.rangeByScore(key, 2d, 5d);
    System.out.println("2,5 포함 -> "+result2);

    // 범위 지정 조회 (스코어 2, 5 제외) -> 기능 없음.

    // 멤버와 스코어 조회
    // Set<TypedTuple<V>> rangeByScoreWithScores(K key, double min, double max)
    Set<TypedTuple<String>> result4 = zsetOperations.rangeByScoreWithScores(key, 2d, 5d);
    if (result4 == null) return null;
    System.out.println("2,5 포함(with score)");
    for (TypedTuple<String> member: result4) {
        System.out.println(member.getValue()+" "+member.getScore());
    }
    msg += result4.size();
    System.out.println(msg);
    return msg;
}
```

ZRANGEBYLEX

```
// 예제 6) ZRANGEBYLEX: 멤버로 범위를 지정해서 조회(스코어가 모두 0인 경우)
// ZRANGEBYLEX key min max [limit offset count]
// http://localhost:8080/zrangebylex/zset2
@GetMapping("/zrangebylex/{key}")
public String zrangebylex(@PathVariable("key") String key) {
    String msg = "예제 6) ZRANGEBYLEX(rangeByLex) -> ";
    System.out.println(msg);
    // 스코어가 0인 테스트 데이터 필요: 데이터가 없을 경우만 입력
    Long size = zsetOperations.size(key);
    if (size != null && size == 0) {
        for (int i=0; i<10; i++) {
            zsetOperations.add(key,"member"+String.format("%02d",i), 0d);
        }
    }
    // 멤버 전체 조회
    // import io.lettuce.core.Range -> 아니고
    // import org.springframework.data.domain.Range -> 이것을 사용하세요.
    Range<String> range = Range.unbounded();
    // default Set<V> rangeByLex(K key, Range<String> range)
    Set<String> result = zsetOperations.rangeByLex(key, range);
    System.out.println("모두(unbounded) -> "+result);
    // 멤버 포함: including
    Range<String> range2 = Range.closed("member03","member05");
    Set<String> result2 = zsetOperations.rangeByLex(key, range2);
    System.out.println("포함(closed) -> "+result2);
    // 멤버 제외: excluding
    Range<String> range3 = Range.open("member03","member05");
    Set<String> result3 = zsetOperations.rangeByLex(key, range3);
    System.out.println("제외(open) -> "+result3);
    msg += result3;
    return msg;
}
```

ZREMRANGEBYRANK

```
// 예제 12) ZREMRANGEBYRANK: 순서(index) 범위로 멤버 삭제
// ZREMRANGEBYRANK key start stop
// http://localhost:8080/zremrangebyrank/zset1
@GetMapping("/zremrangebyrank/{key}")
public String zremrangebyrank(@PathVariable("key") String key) {
    String msg = "예제 12) ZREMRANGEBYRANK(removeRange) -> ";
    Long result = zsetOperations.removeRange(key, 2,5);
    msg += result;
    System.out.println(msg);
    return msg;
}
```

ZREVRANGE

```
// 예제 4) ZREVRANGE: 멤버 리스트 조회(스코어 큰것 부터)
// ZREVRANGE key start stop [withscores]
// http://localhost:8080/zrevrange/zset1
@GetMapping("/zrevrange/{key}")
public String zrevrange(@PathVariable("key") String key) {
    String msg = "예제 4) ZREVRANGE(reverseRange) -> ";
    // 멤버 조회
    // Set<V> reverseRange(K key, long start, long end)
    Set<String> result = zsetOperations.reverseRange(key,0,-1);
    System.out.println(result.toString());
    // 멤버와 스코어 조회
    //     Set<TypedTuple<V>> reverseRangeWithScores(K key, long start, long end)
    Set<TypedTuple<String>> result2 = zsetOperations.reverseRangeWithScores(key,0,-1);
    if (result2 == null) return null;
    for (TypedTuple<String> member: result2) {
        System.out.println(member.getValue()+" "+member.getScore());
    }
    msg += result2.size();
    System.out.println(msg);
    return msg;
}
```

ZRANK

```
// 예제 8) ZRANK: 순서(index) 조회
// ZRANK key member
// http://localhost:8080/zrank/zset1
@GetMapping("/zrank/{key}")
public String zrank(@PathVariable("key") String key) {
    String msg = "예제 8) ZRANK(rank) -> ";
    Long result = zsetOperations.rank(key,"user04");
    msg += result;
    System.out.println(msg);
    return msg;
}
```

ZINCRBY

```
// 예제 13) ZINCRBY:        조회수 증가, 좋아요 수 증가/감소, 상품A가 장바구니에 담긴 회수
// ZINCRBY key increment member
// http://localhost:8080/zincrby/zset1
@GetMapping("/zincrby/{key}")
public String zincrby(@PathVariable("key") String key) {
    String msg = "예제 13) ZINCRBY(incrementScore) -> ";
    System.out.println(msg);
    Double result;
    // Double incrementScore(K key, V value, double delta)
    result = zsetOperations.incrementScore(key, "user90",1);
    System.out.println(result);
    result = zsetOperations.incrementScore(key, "user90", 1);
    System.out.println(result);
    result = zsetOperations.incrementScore(key, "user90", 1);
    System.out.println(result);
    return msg+result;
}
```

ZPOPMIN

```
// 예제 10) ZPOPMIN: 스코어가 작은 멤버부터 꺼내온다.
// ZPOPMIN key [count], ZPOPMAX key [count]
// http://localhost:8080/zpopmin/zset1
@GetMapping("/zpopmin/{key}")
public String zpopmin(@PathVariable("key") String key) {
    String msg = "예제 10) ZPOPMIN(popMin) -> ";
    // TypedTuple<V> popMin(K key);
    TypedTuple<String> result = zsetOperations.popMin(key);
    if (result == null) return null;
    msg += result.getValue()+" "+result.getScore();
    System.out.println(msg);
    return msg;
}
```

BZPOPMIN

```
// 예제 11) BZPOPMIN: 데이터가 들어오면 스코어가 작은 멤버부터 꺼내온다.
// ** Spring에서는 BZPOPMIN 명령 실행마다 새로 연결(connection)을 맺어서 처리하고 끊는다. **
// 이것은 LettuceConnectionFactory 의 shareNativeConnection 설정과 관계없다.
// BZPOPMIN key [key ...] timeout, BZPOPMAX key [key ...] timeout
// http://localhost:8080/bzpopmin/zset1
@GetMapping("/bzpopmin/{key}")
public String bzpopmin(@PathVariable("key") String key) {
    String msg = "예제 11) BZPOPMIN(popMin) -> ";
    // TypedTuple<V> popMin(K key, long timeout, TimeUnit unit)
    // key를 여러 개 지정할 수 없음.
    TypedTuple<String> result = zsetOperations.popMin(key,0, TimeUnit.SECONDS);
    if (result == null) return null;
    msg += result.getValue()+" "+result.getScore();
    System.out.println(msg);
    return msg;
}
```

ZSCAN

```
// 예제 14) ZSCAN: 조회
// ZSCAN key cursor [MATCH pattern] [COUNT count]
// 내부에서 ZSCAN 명령을 반복 실행함.
// http://localhost:8080/zscan/zset5
// http://localhost:8080/zscan/zset5:user*
// http://localhost:8080/zscan/zset5:member*
@GetMapping("/zscan/{key}")
public String zscan(@PathVariable("key") String key) {
    String msg = "예제 14) ZSCAN(scan) -> ";
    // key와 pattern을 나눈다.
    String[] keyPattern = key.split(":");
    String key1 = null;
    String pattern = null;
    // ScanOptions -> COUNT(limit)와 MATCH(pattern) 설정
    ScanOptions scanOptions;
    if (keyPattern.length == 1) {   // pattern을 입력하지 않았을 경우를 고려
        key1 = keyPattern[0];
        scanOptions = ScanOptions.scanOptions().count(20).build();
    } else {
        key1 = keyPattern[0];
        pattern = keyPattern[1];
        scanOptions = ScanOptions.scanOptions().count(20).match(pattern).build();
    }

    // 테스트 데이터 입력 (키가 없을 경우에만 데이터를 입력)
    Long size = zsetOperations.size(key1);
    if (size != null && size == 0) {
        for (int i = 1; i <= 100; i++) {
            zsetOperations.add(key1, "member"+i, i);
            zsetOperations.add(key1, "value"+i, i);
        }
    }
    // ZSCAN 시작
    System.out.println(msg+"0");
    // Cursor<TypedTuple<V>> scan(K key, ScanOptions options)
    Cursor<TypedTuple<String>> result = zsetOperations.scan(key1, scanOptions);
    long cursor = result.getCursorId();
    while (result.hasNext()) {
        if (cursor != result.getCursorId()) {
            System.out.println("Next cursor: "+result.getCursorId());
            cursor = result.getCursorId();
        }
        TypedTuple<String> member = result.next();
        System.out.println("  "+ result.getPosition() +") "+member.getValue()+" "+member.getScore());
    }
    return msg+result.getCursorId();
}
```

<< Sets ZSets Hashes >>

redisgate@gmail.com 02.503.2235
서울시 강남구 강남대로 342 역삼빌딩 5층 (역삼동) 우 06242