版权

创作中心

RedisTemplate



### 热门文章

Redis实现排行榜功能(实战) ① 23583 springboot aop 自定义注解方式实现一套完 善的日志记录(完整源码) ① 7282 正则表达式-linux路径匹配 ① 5969 解决springcloud Eureka启动报错:Unable to

npm设置和查看仓库源 ① 3845

# 分类专栏

```
11篇
                          1篇
   spring
                          3篇
                          2篇
(inux
```

## 最新评论

Redis实现排行榜功能(实战) 俊俊的小熊饼干: redis是缓存, 存的是热点 数据哈, 具体看自己的情况 Redis实现排行榜功能(实战) Luhuaiwei: 那岂不是相当于就是替代了一个 OrderBy的作用而已

Hai - W: 如果排行榜需要显示各学员的头 像,头像地址存数据库吗 每更新一次排 .... Redis实现排行榜功能(实战) 俊俊的小熊饼干: 历史数据可以批量加到red is,和新数据一样就行

Redis实现排行榜功能(实战)

Redis实现排行榜功能(实战) 俊俊的小熊饼干: 维护还是用数据库吧, 用 这个功能本身就是为了排行,并不是为 : ...

### 最新文章

```
让线程按顺序执行8种方式
Elasticsearch与Solr 选型
BlockingQueue 阻塞队列实现异步事件
```

2020年 1篇 2019年 3篇 2018年 16篇

# 目录

#### 需求 分析

结语

实现

一.redis sorts sets简介 二.springboot 中使用RedisTemplate

三.代码实现 四.归纳

```
俊俊的小熊饼干 2018-10-08 18:04:01 ◎ 23586 ☆ 收藏 90
分类专栏: java redis 文章标签: redis RedisTemplate 排行榜
```

转载请注明出处: https://blog.csdn.net/m0\_37459380/article/details/82971525

# 需求

前段时间,做了一个世界杯竞猜积分排行榜。对世界杯64场球赛胜负平进行猜测,猜对+1分,错误+0分,一 人一场只能猜一次。 1.展示前一百名列表。

2.展示个人排名(如:张三,您当前的排名106579)。

# 分析

一开始打算直接使用mysql数据库来做,遇到一个问题,每个人的分数都会变化,如何能够获取到个人的排名 呢?数据库可以通过分数进行row\_num排序,但是这个方法需要进行全表扫描,当参与的人数达到10000的时 候查询就非常慢了。

redis的排行榜功能就完美锲合了这个需求。来看看我是怎么实现的吧。

## 实现

## 一.redis sorts sets简介

Sorted Sets数据类型就像是set和hash的混合。与sets一样,Sorted Sets是唯一的,不重复的字符串组成。可 以说Sorted Sets也是Sets的一种。

Sorted Sets是通过Skip List(跳跃表)和hash Table(哈希表)的双端口数据结构实现的,因此每次添加元素时, Redis都会执行O(log(N))操作。所以当我们要求排序的时候,Redis根本不需要做任何工作了,早已经全部排 好序了。元素的分数可以随时更新。

# 二.springboot 中使用RedisTemplate

本文主要通过redisTemplate来操作redis,当然也可以使用redis-client,看个人喜好

我在本机开启了一个单点的redis, 配置文件如下

```
1 | server:
      port: 9001
 3 | spring:
      redis:
        database: 0
        url: redis://user:123@127.0.0.1:6379
        host: 127.0.0.1
        password: 123
        port: 6379
10
        ssl: false
11
        timeout: 5000
```

#### Maven依赖引入如下

```
1 | <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.0.4.RELEASE
    </parent>
    <dependencies>
            <dependency>
                <groupId>org.springframework.boot</groupId>
10
                <artifactId>spring-boot-starter-web</artifactId>
11
            </dependency>
12
            <dependency>
13
                <groupId>org.springframework.boot</groupId>
14
                <artifactId>spring-boot-starter-data-redis</artifactId>
15
            </dependency>
16
            <dependency>
17
               <groupId>org.springframework.boot</groupId>
18
                <artifactId>spring-boot-starter-test</artifactId>
19
            </dependency>
20 </dependencies>
```

# 三.代码实现

1.注入redis,将key声明为常量SCORE\_RANK

```
@Autowired
private StringRedisTemplate redisTemplate;
public static final String SCORE_RANK = "score_rank";
```

# 2.新增默认排行数据

```
/**
        * 批量新增
        */
       @Test
       public void batchAdd() {
           Set<ZSetOperations.TypedTuple<String>> tuples = new HashSet<>();
           long start = System.currentTimeMillis();
           for (int i = 0; i < 100000; i++) {
               DefaultTypedTuple<String> tuple = new DefaultTypedTuple<>("张三" + i, 1D
10
               tuples.add(tuple);
11
12
           System.out.println("循环时间:" +( System.currentTimeMillis() - start));
13
           Long num = redisTemplate.opsForZSet().add(SCORE_RANK, tuples);
14
           System.out.println("批量新增时间:" +(System.currentTimeMillis() - start));
15
           System.out.println("受影响行数: " + num);
16
```

```
1 //输出
2 循环时间:56
3 批量新增时间:1015
4 受影响行数: 100000
```

# 3.获取前10名(根据分数倒序)

```
* 获取排行列表
        */
       @Test
       public void list() {
           Set<String> range = redisTemplate.opsForZSet().reverseRange(SCORE_RANK, 0,
           System.out.println("获取到的排行列表:" + JSON.toJSONString(range));
           Set<ZSetOperations.TypedTuple<String>> rangeWithScores = redisTemplate.opsF
10
           System.out.println("获取到的排行和分数列表:" + JSON.toJSONString(rangeWithScore
11
```

1 //输出

2 | 获取到的排行列表:["张三99999","张三99998","张三99997","张三99996","张三99995","张三99994", 3 | 获取到的排行和分数列表:[{"score":100000.0,"value":"张三99999"},{"score":99999.0,"value":



```
2021/5/5
```

```
* 单个新增
   3
         */
        @Test
        public void add() {
           redisTemplate.opsForZSet().add(SCORE_RANK, "李四", 8899);
5.获取李四单人的排行
        /**
         * 获取单个的排行
  3
         */
  4
        @Test
        public void find(){
           Long rankNum = redisTemplate.opsForZSet().reverseRank(SCORE_RANK, "李四");
           System.out.println("李四的个人排名: " + rankNum);
           Double score = redisTemplate.opsForZSet().score(SCORE_RANK, "李四");
  10
           System.out.println("李四的分数:" + score);
  11
  1 //输出
  2 李四的个人排名: 91101
  3 李四的分数:8899.0
6.统计分数之间有多少人
        /**
         * 统计两个分数之间的人数
         */
        @Test
        public void count(){
           Long count = redisTemplate.opsForZSet().count(SCORE_RANK, 8001, 9000);
           System.out.println("统计8001-9000之间的人数:" + count);
   8
  1 //输出
  2 统计8001-9000之间的人数:1001
7.获取集合的基数(数量大小)
        /**
         * 获取整个集合的基数(数量大小)
  3
         */
        @Test
        public void zCard(){
          Long aLong = redisTemplate.opsForZSet().zCard(SCORE_RANK);
           System.out.println("集合的基数为: " + aLong);
  8 }
  1 //输出
  2 集合的基数为: 100001
8.使用加法操作分数
        /**
         * 使用加法操作分数
  3
         */
  4
        @Test
        public void incrementScore(){
           Double score = redisTemplate.opsForZSet().incrementScore(SCORE_RANK, "李四",
           System.out.println("李四分数+1000后: " + score);
  8
  1 //输出
  2 李四分数+1000后: 9899.0
四.归纳
在以上测试类中我们使用了redis的那些功能呢?在以上的例子中我们使用了单个新增,批量新增,获取前十,
获取单人排名这些操作,但是redisTemplate还提供了更多的方法。
新增or更新
有三种方式,一种是单个,一种是批量,对分数使用加法(如果不存在,则从0开始加)。
  1 //单个新增or更新
  2 | Boolean add(K key, V value, double score);
   3 //批量新增or更新
  4 Long add(K key, Set<TypedTuple<V>>> tuples);
  5 //使用加法操作分数
  6 | Double incrementScore(K key, V value, double delta);
删除
删除提供了三种方式:通过key/values删除,通过排名区间删除,通过分数区间删除。
   1 //通过key/value删除
   2 Long remove(K key, Object... values);
  4 //通过排名区间删除
   5 Long removeRange(K key, long start, long end);
  7 //通过分数区间删除
   8 Long removeRangeByScore(K key, double min, double max);
查
1.列表查询:分为两大类,正序和逆序。以下只列表正序的,逆序的只需在方法前加上reverse即可:
//通过排名区间获取列表值集合
   1 | Set<V> range(K key, long start, long end);
   3 //通过排名区间获取列表值和分数集合
   4 | Set<TypedTuple<V>> rangeWithScores(K key, long start, long end);
   6 //通过分数区间获取列表值集合
   7 | Set<V> rangeByScore(K key, double min, double max);
  9 //通过分数区间获取列表值和分数集合
  10 | Set<TypedTuple<V>> rangeByScoreWithScores(K key, double min, double max);
  11
  12 //通过Range对象删选再获取集合排行
 13 | Set<V> rangeByLex(K key, Range range);
  14
 15 //通过Range对象删选再获取limit数量的集合排行
  16 | Set<V> rangeByLex(K key, Range range, Limit limit);
2.单人查询
可获取单人排行,和通过key/value获取分数。以下只列表正序的,逆序的只需在方法前加上reverse即可:
   1 //获取个人排行
   2 Long rank(K key, Object o);
```

(1条消息) Redis实现排行榜功能(实战)\_俊俊的小熊饼干的博客-CSDN博客\_redis实现排行榜

★ 点赞25 □ 评论11 ▼ 分享 ★ 收藏90 ¥ 打赏 关注 —键三连
https://blog.csdn.net/m0\_37459380/article/details/82971525

4 //获取个人分数

5 | Double score(K key, Object o);

举报

(1条消息) Redis实现排行榜功能(实战)\_俊俊的小熊饼干的博客-CSDN博客\_redis实现排行榜

统计

统计分数区间的人数,统计集合基数。

redis实现排行榜\_qq18938308968的博客

```
1 //统计分数区间的人数
2 Long count(K key, double min, double max);
3 
4 //统计集合基数
5 Long zCard(K key);
```

## 结语

以上就是redis中使用排行榜功能的一些例子,和对redis的操作方法了。redis不仅仅只是作为缓存,它更是数据库,提供了许多的功能,我们都可以好好的利用。

在这里我使用redis来实现了世界杯积分排行的展示,无论是在批量更新或是获取个人排行等方便,都有着很高效率,也降低了对数据库操作的压力,达到了很好的效果。

感谢您的阅读,如果您觉得阅读本文对您有帮助,请点一下"**推荐**"按钮。本文欢迎各位转载,但是转载文章之后必须在文章开头给出原文链接。



## 使用Redis实现实时排行榜\_igo9go\_zq的博客 使用Redis实现实时排行榜 微信扫码关注下方公众号,获取更多学习资源游戏中存在各种各样的排行榜,比如玩家的等级排名... redis日排行榜,周排行榜 排行榜功能是一个很普遍的需求。使用 Redis 中有序集合的特性来实现排行榜是又好又快的选择。 一般排行榜都是有实效... Redis 实现日排行榜,周排行榜算法 首先,来个"今日积分榜"吧,排序规则是今日用户新增积分从多到少。 那么用户增加积分时,都操作一下记录当天积分增加... redis实战之使用redis实现排行榜\_且行且吟 主要的实现思路是: 1、在一个新的玩家参与到游戏中时,在redis中的zset中新增一条记录(记录内容看具体的需求)score为0 2... redis实现简单排行榜 鸢的博客 redis实现简单排行榜 我们打开redis,使用 select 0 切换到0号数据库,使用 如下语句添加到有序集合中: zadd leaderboard 10... 基于redis的zset实现排行榜功能 lunwuciyu的专栏 ◎ 2万+ 基于redis的zset实现排行榜功能 使用redis实现排行榜功能 排行榜功能是一个很普遍的需求。 设想在一个游戏中,有上百万的玩家数据,如果现在需要你根据玩家的经验值整理一个... 使用redis实现实时排行榜\_曹品东 使用redis实现实时排行榜游戏中存在各种各样的排行榜,比如玩家的等级排名、分数排名等。玩家在排行榜中的名次是其实... redis mysql排行榜实现\_redis实现排行榜的简单方法\_六...

redis实现排行榜 1 前言 实现一个排版榜,我们通常想到的就是mysql的order by 简单粗暴就撸出来了。但是这样真的优雅吗?...

 redis实现排行榜

 redis实现简单排行榜,和消息处理。

 mysql实现分组查询每个班级的前三名

 weixin\_37820504的博客 ◎ 1万+

1、创建表drop table student; create table student( id varchar(20),-- 编号 class varchar(20),-- 年级 score int-- 分数 );2、创...

Redis排行榜的实现

古柏树下的博客 ◎ 128
根据上一篇的设计进一步结合实际场景进行实现,通过伪代码的形式展开。现假设有以下场景:游戏闯关排行榜以通关次...

使用 Redis 实现排行榜功能
weixin\_34101784的博客 ② 239 排行榜功能是一个很普遍的需求。使用 Redis 中有序集合的特性来实现排行榜是又好又快的选择。一般排行榜都是有实效...
Redis Template ZSet使用、实现排行榜
雪落南城的博客 ② 1623
1、@Bean public Redis Template < Object, Object> redis Template (Redis Connection Factory redis Connection Factory) thro...

limit mongodb 聚合\_MongoDB 聚合操作 weixin\_39951181的博客 ◎ 25 在MongoDB中,有两种方式计算聚合: Pipeline 和 MapReduce。Pipeline查询速度快于MapReduce,但是MapReduce的...

C# 浏览器CefSharp71 支持视频播放(Winform&WPF) 05-21

需求说明 水晶数量排行榜 英雄熟练度排版 只排前一万名,玩家只能看到前200名的数据和自己的名次 每个排行榜实时刷新...

支持mp3 mp4 flash 插件视频播放,包括Winform、WPF两个完整64位Demo,直接运行。

©2020 CSDN 皮肤主题: 创作都市 设计师:CSDN官方博客 返回首页

关于我们 招贤纳士 广告服务 开发助手 ☎ 400-660-0108 ☑ kefu@csdn.net ⑤ 在线客服 工作时间 8:30-22:00 公安备案号11010502030143 京ICP备19004658号 京网文〔2020〕1039-165号 经营性网站备案信息 北京互联网违法和不良信息举报中心 网络110报警服务 中国互联网举报中心 家长监护 Chrome商店下载 ⑥1999-2021北京创新乐知网络技术有限公司 版权与免责声明 版权申诉 出版物许可证 营业执照





举报