# Redis与关系型数据库的结合使用

## Redis 各种数据类型的封装

|  |  |
| --- | --- |
| 数据类型 | 对应类名 |
| 基类 | RedisHelperBase |
| String 类型 | RedisHelperString |
| Hash 哈希类型 | RedisHelperHash |
| List 列表类型 | RedisHelperList |
| SortedSet(有序集合) | RedisHelperSortedSet |

## Redis 各种数据类型的存储方式及获取方式

### String 类型

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 参数列表 | 返回值类型 | 描述 |
| StringSet | string key,  String value,  TimeSpan? expire | Bool | 保存单个key/value |
| StringSet | List<KeyValuePair<string,string>>keyValues | Bool | 保存多个key/value键值对 |
| StringSet | string key,T obj,TimeSpan? expire | Bool | 保存一个对象 |
| StringGet | string key | String | 获取单个key的值 |
| StringGet | List<string> listKey | RedisValue[] | 获取多个key的值 |
| StringGet | string key | T | 获取一个key的对象 |
| StringIncrement | string key, double val | Double | 为数字增长val |
| StringDecrement | string key, double val | Double | 为数字减少val |

说明：以上为同步方法，一部方法均在方法名称后增加Async后缀即可 例如 StringSetAsync

### Hash 哈希类型 （key:hash id，dataKey:字段key）

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 参数列表 | 返回值类型 | 描述 |
| HashExists | string key, string dataKey | Bool | 判断某个数据是否已经被缓存 |
| HashSet | string key,string dataKey,T t | Bool | 存储数据到hash表 |
| HashDelete | string key,string dataKey | Bool | 移除hash中的某值 |
| HashDelete | string key,List<string>dataKeys | Long | 移除hash中的多个值 |
| HashGet | string key,string dataKey | T | 从hash表获取数据 |
| HashIncrement | string key,string dataKey,double val | Double | 为数字增长val |
| HashDecrement | string key,string dataKey,double val | Double | 为数字减少val |
| HashKeys | string key | List<T> | 获取hash key中所有的对象 |

### List 列表类型

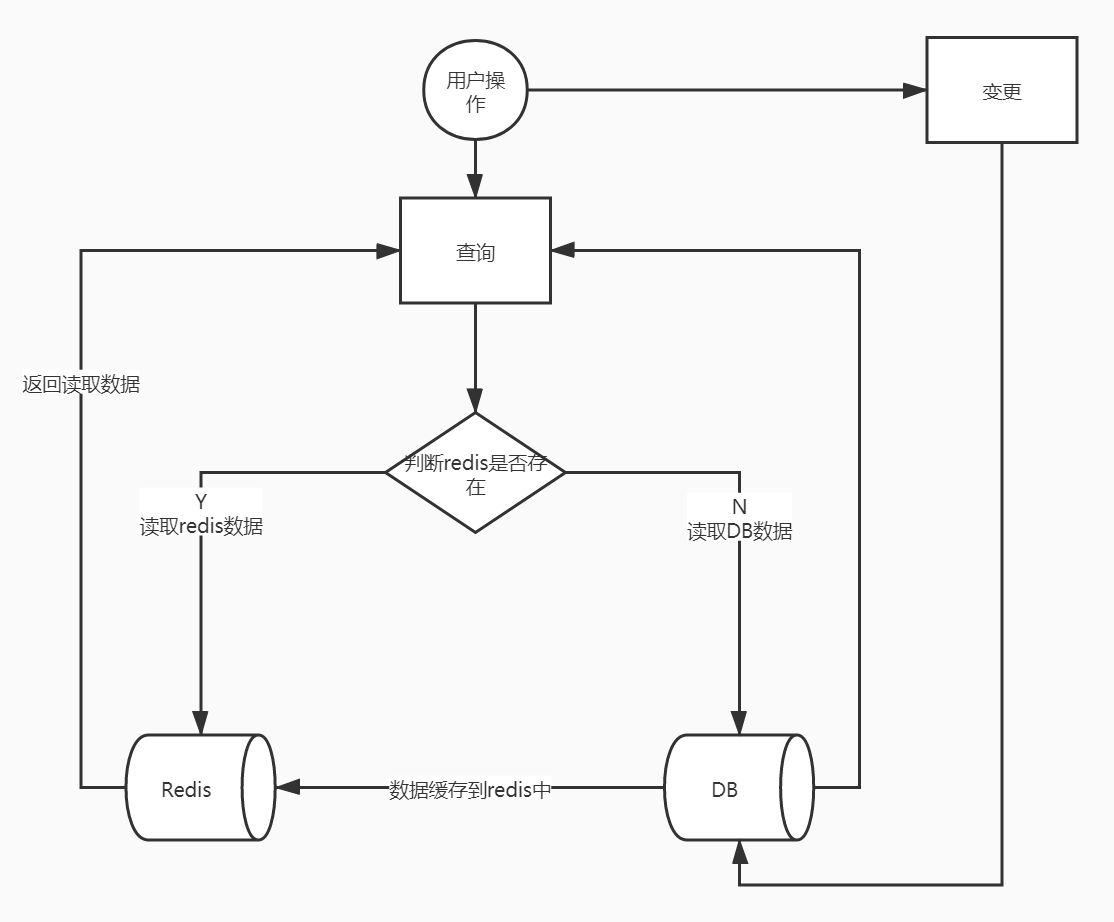
|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 参数列表 | 返回值类型 | 描述 |
| ListRemove | string key,T value | 无 | 移除指定List中的value |
| ListRange | string key | List<T> | 获取指定key的List |
| ListRightPush | string key,T value | 无 | 入队列 桶底 |
| ListRightPop | string key | T | 出队列 桶底 |
| ListLeftPush | string key,T value | 无 | 入栈 桶顶 |
| ListLeftPop | string key | T | 出栈 桶顶 |
| ListLength | string key | Long | 获取集合中的数量 |

### SortedSet有序集合

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 参数列表 | 返回值类型 | 描述 |
| SortedSetAdd | string key,T value,double score | Bool | 添加 |
| SortedSetRemove | string key,T value | Bool | 删除 |
| SortedSetRangeByRank | string key | List<T> | 获取全部 |
| SortedSetLength | string key | Long | 获取集合中的数量 |
| SortedSetRangeByRank | string key,long start, long stop,Order order | List<T> | 获取全部根据score的起止位 |

## Redis与关系型数据库的数据一致性方案

1. 基本流程图



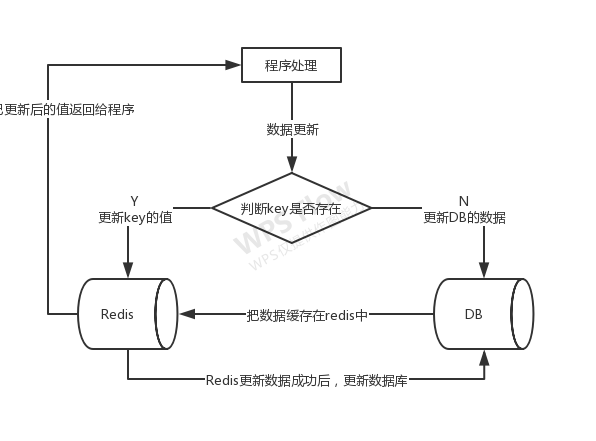
此流程图的方案解读

①查询的时候，先读缓存，如果没有，则读数据库，同时将数据放入缓存，并返回响应

②更新的时候，先更新数据库，......然后变更Redis缓存

适用场合，如果数据量比较大，但不是经常更新的情况

1. 第二种Redis的使用



①先去Redis中判断数据是否存在，如果存在，直接更新对应的数据，把更新后的数据返回给页面。如果不存在的话，就会先更新DB数据库中的内容，然后把数据缓存到Redis

②Redis中的数据更新成功后，把对应的数据库中的数据更新

优点：这个流程把Redis当做数据库使用，更新和查询数据都比DB数据库快

缺点：对Redis依赖很大，要做好宕机时的数据保存。

难点：在前期规划key的格式，存储类型很重要。