二

34 实战：Redis 慢查询

Redis 慢查询作用和 MySQL 慢查询作用类似，都是为我们查询出不合理的执行命令，然后让开发人员和运维人员一起来规避这些耗时的命令，从而让服务器更加高效和健康的运行。对于单线程的 Redis 来说，不合理的使用更是致命的，因此掌握 Redis 慢查询技能对我们来说非常的关键。

如何进行慢查询？

在开始之前，我们先要了解一下 Redis 中和慢查询相关的配置项，Redis 慢查询重要的配置项有以下两个：

- slowlog-log-slower-than：用于设置慢查询的评定时间，也就是说超过此配置项的命令，将会被当成慢操作记录在慢查询日志中，它执行单位是微秒（1 秒等于 1000000 微秒）；
- slowlog-max-len：用来配置慢查询日志的最大记录数。

我们先来看它们的默认配置值：

```
127.0.0.1:6379> config get slowlog-log-slower-than #慢查询判断时间
1) "slowlog-log-slower-than"
2) "10000"
127.0.0.1:6379> config get slowlog-max-len #慢查询最大记录条数
1) "slowlog-max-len"
2) "128"
```

可以看出慢查询的临界值是 10000 微秒，默认保存 128 条慢查询记录。

修改配置项

slowlog-log-slower-than 和 slowlog-max-len 可以通过 `config set xxx` 的模式来修改，例如 `config set slowlog-max-len 200` 设置慢查询最大记录数为 200 条。

慢查询演示

我们先来设置慢查询的判断时间为 0 微秒，这样所有的执行命令都会被记录，设置命令如下：

```
127.0.0.1:6379> config set slowlog-log-slower-than 0
OK
```

接下来我们执行两条插入命令：

```
127.0.0.1:6379> set msg xiaoming
OK
127.0.0.1:6379> set lang java
OK
```

最后我们使用 `slowlog show` 来查询慢日志，结果如下：

```
127.0.0.1:6379> slowlog get #慢日志查询
1) 1) (integer) 2 #慢日志下标
   2) (integer) 1581994139 #执行时间
   3) (integer) 5 #花费时间 (单位微秒)
   4) 1) "set" #执行的具体命令
      2) "lang"
      3) "java"
   5) "127.0.0.1:47068"
   6) ""
2) 1) (integer) 1
   2) (integer) 1581994131
   3) (integer) 6
   4) 1) "set"
      2) "msg"
      3) "xiaoming"
   5) "127.0.0.1:47068"
   6) ""
3) 1) (integer) 0
   2) (integer) 1581994093
   3) (integer) 5
   4) 1) "config"
      2) "set"
      3) "slowlog-log-slower-than"
      4) "0"
   5) "127.0.0.1:47068"
   6) ""
```

加上本身的设置命令一共有三条“慢操作”记录，按照插入的顺序倒序存入慢查询日志中。

> 小贴士：当慢查询日志超过设定的最大存储条数之后，会把最早的执行命令依次舍弃。

慢查询其他相关命令

查询指定条数慢日志

语法： `slowlog get n` 。

```
127.0.0.1:6379> slowlog get 2 #查询两条
1) 1) (integer) 20
   2) (integer) 1581997567
   3) (integer) 14
   4) 1) "slowlog"
      2) "get"
      3) "4"
   5) "127.0.0.1:47068"
   6) ""
2) 1) (integer) 19
   2) (integer) 1581997544
   3) (integer) 11
   4) 1) "slowlog"
      2) "get"
      3) "3"
   5) "127.0.0.1:47068"
   6) ""
127.0.0.1:6379> slowlog get 3 #查询三条
1) 1) (integer) 22
   2) (integer) 1581997649
   3) (integer) 25
   4) 1) "set"
      2) "msg"
      3) "hi"
   5) "127.0.0.1:47068"
   6) ""
2) 1) (integer) 21
   2) (integer) 1581997613
   3) (integer) 9
   4) 1) "slowlog"
      2) "get"
      3) "2"
   5) "127.0.0.1:47068"
   6) ""
3) 1) (integer) 20
   2) (integer) 1581997567
   3) (integer) 14
   4) 1) "slowlog"
      2) "get"
      3) "4"
   5) "127.0.0.1:47068"
   6) ""
```

获取慢查询队列长度

语法： `slowlog len` 。

```
127.0.0.1:6379> slowlog len
(integer) 16
```

清空慢查询日志

使用 `slowlog reset` 来清空所有的慢查询日志，执行命令如下：

```
127.0.0.1:6379> slowlog reset
OK
```

代码实战

本文我们使用 Java 来实现慢查询日志的操作，代码如下：

```
import redis.clients.jedis.Jedis;
import redis.clients.jedis.util.Slowlog;
import utils.JedisUtils;

import java.util.List;

/**
 * 慢查询
 */
public class SlowExample {
    public static void main(String[] args) {
        Jedis jedis = JedisUtils.getJedis();
        // 插入慢查询（因为 slowlog-log-slower-than 设置为 0，所有命令都符合慢操作）
        jedis.set("db", "java");
        jedis.set("lang", "java");
        // 慢查询记录的条数
        long logLen = jedis.slowlogLen();
        // 所有慢查询
        List<Slowlog> list = jedis.slowlogGet();
        // 循环打印
        for (Slowlog item : list) {
            System.out.println("慢查询命令："+ item.getArgs()+
                    " 执行了："+item.getExecutionTime()+" 微秒");
        }
```

```
            // 清空慢查询日志
            jedis.slowlogReset();
        }
    }
```

以上代码执行结果如下：

```
慢查询命令：[SLOWLOG, len] 执行了：1 微秒
慢查询命令：[SET, lang, java] 执行了：2 微秒
慢查询命令：[SET, db, java] 执行了：4 微秒

慢查询命令：[SLOWLOG, reset] 执行了：155 微秒
```

小结

本文我们介绍了慢查询相关的两个重要参数 slowlog-log-slower-than（用于设置慢查询的评定时间）和 slowlog-max-len 用来配置慢查询日志的最大记录数，然后通过修改 `config set slowlog-log-slower-than 0` 把所有操作都记录在慢日志进行相关测试。我们可以使用 `slowlog get [n]` 查询慢操作日志，使用 `slowlog reset` 清空慢查询日志。最后给大家一个建议，可以定期检查慢查询日志，及时发现和改进 Redis 运行中不合理的操作。

上一页 下一页