



您的位置： 首页 → 网络编程 → PHP编程 → php实例 → php+redis消息队列实现抢购

请输入关键词

php+redis消息队列实现抢购功能

更新时间：2018年02月08日 08:32:37 作者：linlin_xia 我要评论

这篇文章主要为大家详细介绍了php+redis消息队列抢购实现代码，具有一定的参考价值，感兴趣的小伙伴们可以参考一下

本文实例为大家分享了php+redis消息队列实现抢购的具体代码，供大家参考，具体内容如下

实现功能：

1. 基于redis队列，防止高并发的超卖
2. 基于mysql的事务加排它锁，防止高并发的超卖

基于redis队列工作流程：

1. 管理员根据goods表中的库存，创建redis商品库存队列
2. 客户端访问秒杀API
3. web服务器先从redis的商品库存队列中查询剩余库存重点内容
4. redis队列中有剩余，则在mysql中创建订单，去库存，抢购成功
5. redis队列中没有剩余，则提示库存不足，抢购失败重点内容

基于mysql事务和排它锁工作流程：

1. 开启事务
2. 查询库存，并显示的设置写锁（排他锁）：SELECT * FROM goods WHERE id = 1 FOR UPDATE
3. 生成订单
4. 去库存，隐示的设置写锁（排他锁）：UPDATE goods SET counts = counts – 1 WHERE id = 1
5. commit，释放锁

注意：第二步步可以设置共享锁，不然有可能会造成死锁。

代码：

```
1 <?php
2 /*****
3  * 抢购模块
4  *
5  * @author liubin
6  * @date 2016-02-10
7  *
8  * ab -n 1000 -c 100 http://192.168.16.73/Seckill/buy.php
9  *
10 */
11 class seckill extends common
12 {
13
14     private $_orderModel = null;
15     private $_goodsModel = null;
16     private $_redis = null;
17     /*
18     * 错误信息
19     */
20     protected $_error = '';
21     /**
22     * 构造器
23     *
24     */
25     public function __construct()
26     {
27         if($this->_orderModel === null){
28             $this->_orderModel = new OrderModel();
29         }
30         if($this->_goodsModel === null){
31             $this->_goodsModel = new GoodsModel();
32         }
33         if($this->_redis === null){
34             $this->_redis = new QRedis();
35         }
36     }
37     /*
38     * 秒杀API
39     *
40     * @author liubin
41     * @date 2017-02-10
42     */
```

站长推荐

正版 Windows 10
正版Windows 10 家庭/专
作系统限时抢购[¥4998→

站长推荐

正版 Office 软件
Microsoft Office 2016/20
正版最低价仅需[¥148元

大家感兴趣的内容

- 1 php获取数组长度的方法（有实
- 2 微信公众平台实现获取用户Op
- 3 教你如何使用php session
- 4 使用PHP生成二维码的两种方
- 5 php发送get、post请求的6种方
- 6 PHP中把stdClass Object转ar
- 7 微信公众平台网页授权获取用
- 8 Laravel框架数据库CURD操作
- 9 PHP字符串的连接简单实例
- 10 PHP删除数组中空值的方法介绍

最近更新的内容

PHP中使用GD库创建圆形饼图的

PHP 面向对象程序设计（oop）：

Zend Framework动作助手Flash

Zend Framework动作控制器用法

PHP常见漏洞攻击分析

ThinkPHP整合百度Ueditor图文

调试WordPress中定时任务的相

PHP获取音频文件的相关信息

php接口和抽象类使用示例详解

Laravel框架实现的批量删除功能

常用在线小工具

```
43 public function addQsec(){
44     $gid = intval($_GET['gid']);
45     $type = isset($_GET['type']) ? $_GET['type'] : 'mysql';
46     switch ($type) {
47         case 'mysql':
48             $this->order_check_mysql($gid);
49             echo $this->getError();
50             break;
51         case 'redis':
52             $this->order_check_redis($gid);
53             echo $this->getError();
54             break;
55         case 'transaction':
56             $this->order_check_transaction($gid);
57             echo $this->getError();
58             break;
59         default:
60             echo '类型错误';
61             break;
62     }
63 }
64 /*
65  * 获取错误信息
66  *
67  * @author liubin
68  * @date 2017-02-10
69 */
70 public function getError(){
71     return $this->_error;
72 }
73 /*
74  * 基于mysql验证库存信息
75  * @desc 高并发下会导致超卖
76  *
77  * @author liubin
78  * @date 2017-02-10
79 */
80 protected function order_check_mysql($gid){
81
82
83     $model = $this->_goodsModel;
84     $pdo = $model->getHandler();
85     $gid = intval($gid);
86
87     /*
88      * 1: $sql_forlock如果不加事务，不加写锁：
89      * 超卖非常严重，就不说了
90      *
91      * 2: $sql_forlock如果不加事务，只加写锁：
92      * 第一个会话读$sql_forlock时加写锁，第一个会话$sql_forlock查询结束会释放该行锁.
93      * 第二个会话在第一个会话释放后读$sql_forlock的写锁时，会再次$sql_forlock查库存
94      * 导致超卖现象产生
95      *
96      */
97     $sql_forlock = 'select * from goods where id = '.$gid.' limit 1 for updat
98     //$sql_forlock = 'select * from goods where id = '.$gid.' limit 1';
99     $result = $pdo->query($sql_forlock,PDO::FETCH_ASSOC);
100     $goodsInfo = $result->fetch();
101
102     if($goodsInfo['counts']>0){
103
104         //去库存
105         $gid = $goodsInfo['id'];
106         $sql_inventory = 'UPDATE goods SET counts = counts - 1 WHERE id = '.$gid;
107         $result = $this->_goodsModel->exect($sql_inventory);
108         if($result){
109             //创订单
110             $data = [];
111             $data['order_id'] = $this->_orderModel->buildOrderNo();
112             $data['goods_id'] = $goodsInfo['id'];
113             $data['addtime'] = time();
114             $data['uid'] = 1;
115             $order_rs = $this->_orderModel->create_order($data);
116             if($order_rs){
117                 $this->_error = '购买成功';
118                 return true;
119             }
120         }
121     }
122
123     $this->_error = '库存不足';
124     return false;
125 }
126
127 /*
128  * 基于redis队列验证库存信息
129  * @desc Redis是底层是单线程的, 命令执行是原子操作, 包括lpush,lpop等. 高并发下不会导致超
130  *
131  * @author liubin
132  * @date 2017-02-10
133 */
134 protected function order_check_redis($gid){
135     $goodsInfo = $this->_goodsModel->getGoods($gid);
136     if(!$goodsInfo){
137         $this->_error = '商品不存在';
138         return false;
139     }
140     $key = 'goods_list_'.$goodsInfo['id'];
141     $count = $this->_redis->getHandel()->lpop($key);
142     if(!$count){
143         $this->_error = '库存不足';
144         return false;
145     }
146     //生成订单
147     $data = [];
148     $data['order_id'] = $this->_orderModel->buildOrderNo();
149     $data['goods_id'] = $goodsInfo['id'];
150     $data['addtime'] = time();
151     $data['uid'] = 1;
152     $order_rs = $this->_orderModel->create_order($data);
153
154     //库存减少
```

```
155     $gid = $goodsInfo['id'];
156     $sql = 'UPDATE goods SET counts = counts - 1 WHERE id = '.$gid;
157     $result = $this->_goodsModel->exect($sql);
158     $this->_error = '购买成功';
159     return true;
160 }
161 /*
162  * 基于mysql事务验证库存信息
163  * @desc 事务 和 行锁 模式,高并发下不会导致超卖, 但效率会慢点
164  * @author liubin
165  * @date 2017-02-10
166
167
168 说明:
169 如果$sql_forlock不加写锁, 并发时, $sql_forlock查询的记录存都大于0, 可以减库存操作.
170 如果$sql_forlock加了写锁, 并发时, $sql_forlock查询是等待第一次链接释放后查询. 所以库
171
172 */
173 protected function order_check_transaction($gid) {
174
175     $model = $this->_goodsModel;
176     $pdo = $model->getHandler();
177     $gid = intval($gid);
178
179     try{
180         $pdo->beginTransaction();//开启事务处理
181
182
183         /*
184          * 1: $sql_forlock如果只加事务, 不加写锁:
185          * 开启事务
186          * 因为没有加锁, 读$sql_forlock后, 并发时$sql_inventory之前还可以再读。
187          * $sql_inventory之后和commit之前才会锁定
188          * 出现超卖跟事务的一致性不冲突
189          *
190          *
191          * 2: $sql_forlock如果加了事务, 又加读锁:
192          * 开启事务
193          * 第一个会话读$sql_forlock时加读锁, 并发时, 第二个会话也允许获得$sql_forlock的读锁
194          * 但是在第一个会话执行去库存操作时 (写锁), 写锁便会等待第二个会话的读锁, 第二个会话执
195          * 出现死锁
196
197          * 3: $sql_forlock如果加了事务, 又加写锁:
198          * 开启事务
199          * 第一个会话读$sql_forlock时加写锁, 直到commit才会释放写锁, 并发查询不会出现超卖现
200          *
201          */
202
203         $sql_forlock = 'select * from goods where id = '.$gid .' limit 1 for upda
204         //$sql_forlock = 'select * from goods where id = '.$gid .' limit 1 LOCK I
205         //$sql_forlock = 'select * from goods where id = '.$gid .' limit 1';
206         $result = $pdo->query($sql_forlock,PDO::FETCH_ASSOC);
207         $goodsInfo = $result->fetch();
208
209         if($goodsInfo['counts']>0) {
210
211             //去库存
212             $gid = $goodsInfo['id'];
213             $sql_inventory = 'UPDATE goods SET counts = counts - 1 WHERE id = '.$gid;
214             $result = $this->_goodsModel->exect($sql_inventory);
215
216             if(!$result){
217                 $pdo->rollBack();
218                 $this->_error = '库存减少失败';
219                 return false;
220             }
221
222             //创订单
223             $data = [];
224             $data['id'] = 'null';
225             $data['order_id'] = $this->_orderModel->buildOrderNo();
226             $data['goods_id'] = $goodsInfo['id'];
227             $data['uid'] = 'abc';
228             $data['addtime'] = time();
229
230             $sql = 'insert into orders (id,order_id,goods_id,uid,addtime) values ('.
231             $result = $pdo->exec($sql);
232             if(!$result){
233                 $pdo->rollBack();
234                 $this->_error = '订单创建失败';
235                 return false;
236             }
237             $pdo->commit();//提交
238             $this->_error = '购买成功';
239             return true;
240
241         }else{
242             $this->_error = '库存不足';
243             return false;
244         }
245     }catch(PDOException $e){
246         echo $e->getMessage();
247         $pdo->rollBack();
248     }
249
250 }
251
252 /*
253  * 创建订单
254  * mysql 事物处理, 也可以用存储过程
255  *
256  */
257 private function create_order($goodsInfo) {
258     //生成订单
259     $data = [];
260     $data['order_id'] = $this->_orderModel->buildOrderNo();
261     $data['goods_id'] = $goodsInfo['id'];
262     $data['addtime'] = time();
263     $data['uid'] = 1;
264     $order_rs = $this->_orderModel->create_order($data);
265
266     //库存减少
```

```
267         $gid = $goodsInfo['id'];
268         $sql = 'UPDATE goods SET counts = counts - 1 WHERE id = '.$gid;
269         $result = $this->_goodsModel->exect($sql);
270         return true;
271     }
272 }
```

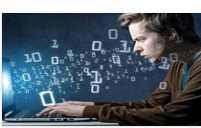



以上就是本文的全部内容，希望对大家的学习有所帮助，也希望大家多多支持脚本之家。

您可能感兴趣的文章：

- php-beanstalkd消息队列类实例分享
- PHP的Laravel框架中使用消息队列queue及异步队列的方法
- PHP+memcache实现消息队列案例分享
- PHP下操作Linux消息队列完成进程间通信的方法
- PHP使用php-resque库配合Redis实现MQ消息队列的教程
- PHP消息队列用法实例分析
- PHP+RabbitMQ实现消息队列的完整代码
- PHP+MySQL实现消息队列的方法分析
- php基于Redis消息队列实现的消息推送的方法
- PHP基于Redis消息队列实现发布微博的方法
- PHP Beanstalkd消息队列的安装与使用方法实例详解

php redis 消息队列 抢购

相关文章

	ThinkPHP多语言支持与多模板支持概述 <p>这篇文章主要介绍了ThinkPHP多语言支持与多模板支持,是ThinkPHP非常重要的技巧,需要的朋友可以参考下</p>	2014-08-08
	php过滤敏感词的示例 <p>这篇文章主要介绍了php过滤敏感词的示例,需要的朋友可以参考下</p>	2014-03-03
	Laravel如何创建服务器提供者实例代码 <p>这篇文章主要给大家介绍了关于Laravel如何创建服务器提供者的相关资料，文中通过示例代码介绍的非常详细，对大家学习或者使用Laravel具有一定的参考学习价值，需要的朋友...</p>	2019-04-04
	Yii 框架控制器创建使用及控制器响应操作示例 <p>这篇文章主要介绍了Yii 框架控制器创建使用及控制器响应操作,结合实例形式分析了Yii框架控制器调用、参数传递与响应相关操作技巧,需要的朋友可以参考下</p>	2019-10-10
	Laravel 修改默认日志文件名称和位置的例子 <p>今天小编就为大家分享一篇Laravel 修改默认日志文件名称和位置的教程，具有很好的参考价值，希望对大家有所帮助。一起跟随小编过来看看吧</p>	2019-10-10
	Laravel框架实现的记录SQL日志功能示例 <p>这篇文章主要介绍了Laravel框架实现的记录SQL日志功能,结合实例形式总结分析了Laravel 框架监听并记录SQL相关操作技巧与注意事项,需要的朋友可以参考下</p>	2018-06-06
	PHP 面向对象程序设计（oop）学习笔记(一) - 抽象类、对象接口、 <p>面向对象程序设计（OOP）是一种计算机编程架构。OOP的一条基本原则是计算机程序是由单个能够起到子程序作用的单元或对象组合而成。OOP达到了软件工程的三个主要目标： ...</p>	2014-06-06
	利用中国天气预报接口实现简单天气预报 <p>这篇文章主要介绍了利用中国天气预报接口实现简单天气预报的示例，大家参考使用吧</p>	2014-01-01
	Laravel 批量更新多条数据的示例 <p>本篇文章主要介绍了Laravel 批量更新多条数据的示例，小编觉得挺不错的，现在分享给大家，也给大家做个参考。一起跟随小编过来看看吧</p>	2017-11-11



ThinkPHP中的常用查询语言汇总

这篇文章主要介绍了ThinkPHP中的常用查询语言汇总,是ThinkPHP中常用的技巧,在项目开发中非常有实用价值,需要的朋友可以参考下

2014-08-08

最新评论

[关于我们](#) - [广告合作](#) - [联系我们](#) - [免责声明](#) - [网站地图](#) - [投诉建议](#) - [在线投稿](#)

©Copyright 2006-2020 JB51.Net Inc All Rights Reserved. 脚本之家 版权所有