活动前的缓存预热

(难度: ★★★★★)

1、前置知识

为什么要做预热:

当活动真正开始时,需要超高的并发访问活动相关信息

必须把必要的数据提前加载进redis

预热的策略:

在msg中写一个定时任务

每分钟扫描一遍card_game表

把(开始时间 > 当前时间)&& (开始时间 <= 当前时间+1分钟)的活动及相关信息放入redis

缓存预热流程:

- 1、查询1分钟内的活动
- 2、循环遍历活动列表,挨个处理,假设当前取出的是A
- 3、查询A相关的奖品列表及数量
- 4、根据总数量生成奖品相关的令牌桶,详细设计参考《缓存预热设计》视频
- 5、查询A相关的活动策略:抽奖次数、中奖次数等,放入Redis

注意事项:

代码在msg项目下的GameTask里,已集成Spring调度

commons模块下有个RedisKeys,已经定义了可用的Redis key前缀,可以直接使用

更多内容参考:

《设计文档》第2.2.3章节

《缓存预热设计》相关视频:

链接: https://pan.baidu.com/s/1Mu5yLP5jZH75r8ZBh6Z8WQ?pwd=mw55

2、任务清单

完成缓存预热相关代码

完成缓存查询接口,返回生成的缓存信息

3、完成标准

3.1 完成基础代码编写

缓存调度代码:

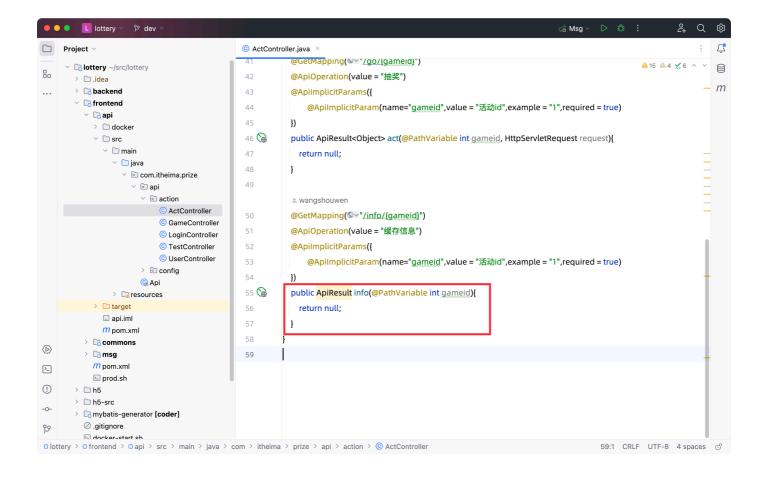
```
lottery – GameTask.java [msg]
Notifications

✓ In lottery ~/src/lottery

                                                   package com.itheima.prize.msg;
                                                                                                                           A10 A5 ★4 ^ ∨
    > lackend
                                                 import ...
Commit
    > 🖿 deploy
                                            19
    Y 📭 frontend
                                            20
      > 📭 api
                                            21
                                                   * 活动信息预热,每隔1分钟执行一次
      > 📭 commons
      🗸 📭 msg
                                            22
                                                   * 查找未来1分钟内(含),要开始的活动
       > III docker
                                            23
        ∨ ■ src
                                            24
                                            25 ♦ public class GameTask {
           🗡 🖿 java
                                            26
                                                       private final static Logger log = LoggerFactory.getLogger(GameTask.class);
             ✓ 🖿 com.itheima.prize
                                            27
                                                       @Autowired
               ∨ 🖿 msg
                                            28 😂
                                                       private CardGameService gameService;
               © GameTask
                                            29
                  PrizeGameReceiver
                                                       @Autowired
                  © PrizeHitReceiver
                                            30 😭
                                                       private CardGameProductService gameProductService;
                © Msg
                                            31
                                                       @Autowired
           > resources
                                            32 😭
                                                       private CardGameRulesService gameRulesService;
      > larget
                                            33
                                                       @Autowired
         msg.iml
                                            34 😭
                                                       private GameLoadService gameLoadService;
                                            35
                                                       @Autowired
       m pom.xml
       prod.sh
                                            36 😭
                                                       private RedisUtil redisUtil;
    → l h5
                                            37
    > h5-vue
                                            38
                                                       @Scheduled(cron = "0 * * * * ?")
    > 🖿 pic
                                            39 🕞 🖪
                                                       public void execute() {...}
     aitianore 🖟
                                            131
      nginx.conf
      aprize_2023-11-19.sql
      README.md
  > III External Libraries
  > Scratches and Consoles
  P Git Q Find ▶ Run ♣ Debug ≡ TODO ● Problems ■ Terminal Q Profiler ● Services 	 Build 등 Database Changes
// Build completed successfully in 4 sec, 503 ms (6 minutes ago)
                                                                                                          132:1 CRLF UTF-8 4 spaces 12 mybatis-plus 1
```

缓存查询接口:

提示:可以将所有本活动相关的信息放在一个Map中集中返回,方便swagger中查看



3.2 完成代码自测

【需要提交预热成功的缓存内容截图】

- 1、在管理后台创建相关活动
- 2、验证活动到期1分钟前,能够正常预热写入数据到Redis
- 3、用swagger访问缓存接口,返回缓存信息,以下结构仅供参考,具体可以自行定义

应尽量涵盖:活动基本信息、令牌桶及令牌对应的奖品信息、活动的策略设置

```
"code": 200,
"msg": "缓存信息",
"data": {
    "game_info_8": null,
    "game_tokens_8": {},
    "game_maxgoal_8": {},
    "game_maxenter_8": {}
},
    "now": "2023/11/27 15:18:29"
}
```

一个参考样例:

```
"code": 200,
"msg": "缓存信息",
"data": {
 "game_info_7": {
   "id": 7,
   "title": "超级女神",
   "pic": "/upload/images/girl.png",
   "info": "女神来抽奖",
   "starttime": "2023/11/20 18:50:23",
   "endtime": "2023/12/10 18:50:23",
   "type": 1,
   "status": 1
 },
 "game tokens 7": {
   // 注意! 实际Redis里的令牌是时间戳,返回前可以将其转成日期类型,方便查看//
   "2023-11-22 05:50:50.866": {
     "id": 5,
     "name": "美食大礼盒",
     "pic": "/upload/images/food.png",
     "info": "来自好利来的轻奢月饼,是家人食用,馈赠亲友的不二选择",
     "price": 1000,
     "amount": 200
   },
   "2023-11-22 06:15:04.434": {
     "id": 6,
     "name": "5积分",
     "pic": "/upload/images/555555.png",
     "info": "积分可以在积分商城兑换相应商品,例如5个积分可以兑换50元话费",
     "price": 500,
     "amount": 500
   },
   "2023-12-10 18:46:15.510": {
     "id": 10,
     "name": "青铜",
     "pic": "/upload/images/money.png",
     "info": "看上去和金子一样, 但是不是",
     "price": 1,
     "amount": 100
   }
 },
 "game_maxgoal_7": {
```

```
"0": 5,
    "1": 10
},
    "game_maxenter_7": {
        "0": 10,
        "1": 20
}
},
    "now": "2023/11/27 15:15:38"
}
```