# 处理申诉

|  |  |  |
| --- | --- | --- |
| **编号** | **用例名称** | **状态** |
| **1** | 限制航班同步缓存 | 创建 |

# 简要说明

此功能为提高查询速度设计，将限制航班更新入缓存。

# 前置条件

无

# 后置条件

屏蔽虚假舱位定时任务

工单基础数据维护-新增限制航班

工单基础数据维护-修改限制航班

工单基础数据维护-删除限制航班

工单基础数据维护-限制航班信息一键入缓存

# 界面原型

待定

# 活动图



屏蔽虚假舱位job 及从工单录入限制舱位



从工单删除限制航班



工单修改限制航班信息

# 业务规则

R1占位失败 并且av查询还有余座，判定此条航班信息无效 入数据库 并同步缓存。

R2基础数据维护 对限制航班进行添加 修改 删除 进行数据库更新 并同步缓存。

R3 设计redis缓存key，业务码固定值\_出发机场三字码\_到达机场三字码。举例 LIMITF\_PEK\_SHE。

R4 设计redis缓存value，Map<String key , List<CcsLimitFlightM> list>。Map Key值：航班号\_是否虚假仓位（Y是N否）\_仓位。举例 CA1871\_Y\_M , CA1871\_N。Map Value值：List<CcsLimitFlightM> list。

R5 单条数据是否过期 计算规则为 isExpired =（销售有效截至日期<当前日期） ||（限制有效截至日期<当前日期）

R6 计算redis过期时间。

单条数据的过期时间=（限制有效截至时间 – 当前时间）与（销售有效结束时间 – 当前时间）的较小值。

Redis过期时间为 list中所有 单条策略过期时间较大者。

# 主流程

B1屏蔽虚假舱位job 及从工单录入限制舱位。

B2 验证此条数据是否过期，过期则不处理。移除当前redis key 里Map里 过期的限制航班信息。此条数据入缓存。重新计算当前redis 失效时间。

B3 工单基础数据维护 删除限制航班。

B4 遍历当前 redis里map 的list ，找到此条数据 并删除 同时重新计算redis 过期时间。

B5 工单基础数据维护 修改限制航班。

B6 从redis删除此条数据，更新此条数据并入缓存。

# 分支流

遍历map里的list 移除过期数据； 重新计算redis 过期时间。

# 异常流

# 输入输出

**输入**

Ccs\_limit\_flight\_m对象

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **参数名** | **中文含义** | **数据类型** | **必填** | **备注** |
| **id** | 主键 | Long | 是 |  |
| **air\_ways** | 航空公司 | String | 否 |  |
| **dep\_code** | 出发城市三字码 | String | 否 |  |
| **arr\_code** | 到达城市三字码 | String | 否 |  |
| **flight\_no** | 航班编号 | String | 否 |  |
| **seat\_no** | 限制舱位 | String | 否 |  |
| **limit\_time\_begin** | 限制开始时间 | Date | 是 |  |
| **limit\_time\_end** | 限制结束时间 | Date | 是 |  |
| **limit\_sale\_time\_begin** | 销售开始时间 | Date | 是 |  |
| **limit\_sale\_time\_end** | 销售结束时间 | Date | 是 |  |
| **is\_false\_seat** | 是否为虚拟舱位 0-否 1-是 | Boolean | 否 |  |
| **create\_time** | 创建日期 | Date | 是 |  |
| **update\_time** | 更新日期 | Date | 是 |  |
| **operator** | 操作人 | String | 是 |  |

**输出**

Boolean 入缓存是否成功。