**生活应用交易监控设计**

|  |  |  |  |
| --- | --- | --- | --- |
| **联系人** | **邮箱** | **手机** | **座机** |
| 李双 | lishuang@shhxzq.com | 13817533020 | 021-61260751 |
| 关向辉 | guanxianghui@shhxzq.com | 13764603603 | 021-61260751 |

文档历史

|  |  |  |  |
| --- | --- | --- | --- |
| 作者 | 日期 | 版本 | 说明 |
| 关向辉 | 2017-06-14 | 1.0 | 新建文档  搞定：设计 |
| 关向辉 | 2017-06-19 | 1.1 | 新增：  备注字段表达式设计  备注字段手动设置设计 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

## 库表设计

## la\_trans\_monitor交易监控表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **字段** | **类型** | **是否可空** | **约束** | **默认值** | **备注** |
| **id** | **int(11)** | **否** | **主键自增** |  | **主键ID** |
| **sys\_date** | **Char(8)** | **否** |  |  | **系统日期** |
| **sys\_time** | **Char(6)** | **否** |  |  | **系统时间** |
| **method\_type** | **varchar(10)** | **否** |  |  | **方法类型**  **PO:提供对外服务方法**  **IN:内部服务方法**  **IO:调用外部方法**  **SQL:内部sql** |
| **outer\_system** | **varchar(20)** |  |  |  | **外部系统代码** |
| **method\_route** | **varchar(200)** | **否** |  |  | **方法路径** |
| **method\_name** | **varchar(100)** | **否** |  |  | **方法名称** |
| **request** | **longtext** |  |  |  | **入参** |
| **response** | **longtext** |  |  |  | **出参** |
| **is\_exception** | **int(1)** | **否** |  | **0** | **是否异常发生**  **0:没发生异常**  **1:发生异常** |
| **begin\_time** | **timestamp(3)** | **否** |  |  | **开始时间** |
| **end\_time** | **timestamp(3)** | **否** |  |  | **结束时间** |
| **used\_time** | **bigint** | **否** |  |  | **耗时(毫秒)** |
| **ip** | **varchar(50)** |  |  |  | **机器ip** |
| **resv1** | **varchar(100)** |  |  |  | **备注字段1** |
| **resv2** | **varchar(100)** |  |  |  | **备注字段2** |
| **resv3** | **varchar(100)** |  |  |  | **备注字段3** |
| **is\_delete** | **int(1)** | **否** |  |  | **是否删除**  **0:未删除**  **1:已删除** |
| **created\_at** | **timestamp** | **否** |  |  | **创建时间** |
| **updated\_at** | **timestamp** | **否** |  |  | **修改时间** |

## 切面设计

execution(\* com.shhxzq.fin.lifeapp..\*.\*(..)) 监控 @Service @Controller

around方法前后操作

## 注解设计

注解@LogMethodTime

{

isMonitor: 是否监控 [true/false]

monitorMethodType: 监控方法类型枚举 [PO/ IN/ IO/ SQL]

outerSystem: 外部系统代码 [be/spw/beidou/cts…]

methodName: 方法名称 [查询是否显示信用卡管理菜单]

resv1: 备注字段1

resv2: 备注字段2

resv3: 备注字段3

}

比如：

@LogMethodTime(isMonitor = true, monitorMethodType =

MonitorMethodTypeEnum.PROVIDE\_OUT, methodName = "查询是否显示信用卡管理菜单",

resv1="{request[0].custNo}", resv2="{response}", resv3="NOTHING\_HERE!")

public boolean queryShowCreditMenu(QueryShowCreditMenuRequest request) {

...

}

获取[类.方法()]路径，入参，出参，请求时间，结束时间，消耗时间，是否异常发生，本地IP，**备注字段1，2，3**

## 备注字段表达式设计

注意：在切面中通过反射解析获取值

(1).普通字符串:直接保存字符串内容

(2).支持表达式:引入commons-jexl3实现类似el/velocity表达式获取对象属性

1.监控方法[获取请求]表达式:{request}

1.1.{request} : 取第一个request参数

1.2.{request[0]} : 取第一个request参数

1.3.{request[1]} : 取第二个request参数

1.4.{request.custNo} : 取第一个request参数的属性

1.5.{request.createdAt.time} : 取第一个request参数的属性的属性

1.6.{request[0].custNo} : 取第一个request参数的属性

1.7.{request[0].createdAt.time} : 取第一个request参数的属性的属性

1.8.{request[1].createdAt.time} : 取第二个request参数的属性的属性

2.监控方法[获取返回]表达式:{response}

2.1.{response} : 取第一个request参数

2.2.{response.custNo} : 取第一个request参数的属性

2.3.{response.createdAt.time} : 取第一个request参数的属性的属性

3. 表达式新增分隔符{|}支持，让一个备注字段可以存多个信息"|"拼接

比如：{request[0].custNo}{|}普通字符串{|}{response.userCardDto.bankNo}

## 备注字段手动设置设计

注意：通过在ThreadLocal中设置值，在切面中获取

1.新增手动设置备注字段方法，不在注解中使用表达式反射(相对提高性能)(但是这种方式有代码侵入到业务中)

2.手动设置备注字段优先级高于(覆盖)注解中的设置(普通字符串/支持表达式)

比如：TransMonitorUtils.setResv1(this.getClass().getName(),

"queryShowCreditMenu", new Object[]{request}, "我的老娘亲!");

## mq异步推送交易监控信息

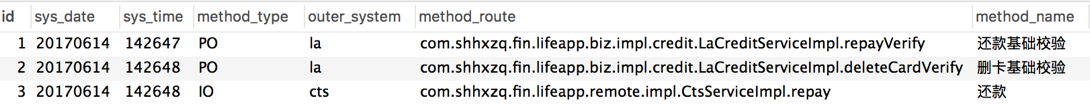
异步操作，mq推送交易监控信息，不拖慢主流程

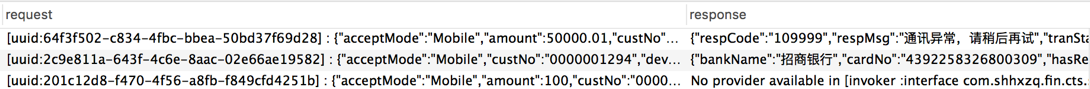


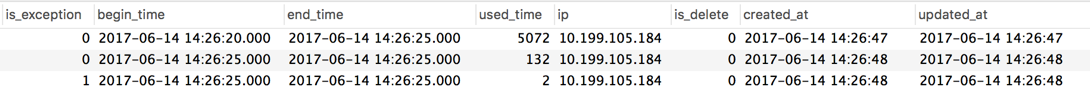
## mq异步消费

insert到库中：la\_trans\_monitor交易监控表

## 数据库中数据







## 查询明细SQL

SELECT \* FROM la\_trans\_monitor WHERE 对应条件

## 统计SQL

统计：并发数，平均耗时，TPS

TPS=并发数/平均耗时(s)

######查询结果######

select

#\*

#并发数，平均耗时，TPS

count(1), avg(used\_time), count(1)\*1000.0/avg(used\_time)

from la\_trans\_monitor

######条件######

where 1=1

#and method\_type='PO'

#and method\_type='IO' and outer\_system='be'

#and method\_route='com.shhxzq.fin.lifeapp.biz.impl.credit.LaCreditServiceImpl.getUserCardById'

and is\_exception=0

#and begin\_time between '2017-06-14 11:15:00.000' and '2017-06-15 11:30:00.000'

#and end\_time between '2017-06-14 11:15:00.000' and '2017-06-15 11:30:00.000'

#and used\_time > 1000

;

## 实时监控界面

CMS中实现折线图，SQL参考上面统计SQL，可能要实时去计算[每分钟/每10分钟/每小时/每天]的统计数据

如果实时监控查询统计SQL比较慢，后面考虑使用缓存或者异步或者定时去算