系统设计

C01研发部-统计平台项目组

2013年11月

目录

[系统设计 1](#_Toc371588055)

[C01研发部-统计平台项目组 1](#_Toc371588056)

[2013年11月 1](#_Toc371588057)

[1、日志格式 2](#_Toc371588058)

[1. 1、人数 人次(该示例已插入数据库配置表) 3](#_Toc371588059)

[1.2、人数 人次 数量 4](#_Toc371588060)

[1.3、人数 人次 数量1 数量2 4](#_Toc371588061)

[1.4、人数 人次 数量 等级分布(该示例已插入数据库配置表) 4](#_Toc371588062)

[1.5、人数人次各子项的人数人次 5](#_Toc371588063)

[1.6、人数 人次 各子项的人数人次数量 等级分布 6](#_Toc371588064)

[1.7、最大值 累加值分布的人数人次 6](#_Toc371588065)

[2、数据库设计 6](#_Toc371588066)

[2.1、游戏基本信息表t\_game\_info 6](#_Toc371588067)

[2.2、游戏平台区服映射表t\_gpzs\_info 7](#_Toc371588068)

[2.3、树结构表t\_web\_tree 7](#_Toc371588069)

[2.4、reportId映射表t\_report\_info 8](#_Toc371588070)

[2.5、数据项映射表t\_data\_info 9](#_Toc371588071)

[2.6、导航表t\_web\_navi 10](#_Toc371588072)

[3、模块设计 11](#_Toc371588073)

[3.1、Stat-logger 12](#_Toc371588074)

[3.2、Stat-client 12](#_Toc371588075)

[3.3、Stat-proxy 12](#_Toc371588076)

[3.4、Stat-server 12](#_Toc371588077)

[3.5、stat-center 12](#_Toc371588078)

[3.6、Db-server 13](#_Toc371588079)

[3.6.1、分钟数据的计算 13](#_Toc371588080)

[3.6.2、配置项的入库 13](#_Toc371588081)

[3.6.2、数据计算结果的入库 13](#_Toc371588082)

[3.7、config-server 13](#_Toc371588083)

[3.8、mapreduce 13](#_Toc371588084)

[3.9、mapreduce计算结果入库 13](#_Toc371588085)

# 1、日志格式

字段名 含义

hip 发送数据的客户端IP

stid 统计项名称，比如游戏输出

sstid 子统计项名称，比如撞球对抗赛、黄金矿工

gid 游戏ID

zid 区ID

sid 服ID

pid 平台ID -1表示全平台，默认淘米平台ID为1

ts 时间戳

acid 帐号(米米号)

plid 角色标识

key1=value1, key2=value2,… 自定义key-value对

op 对自定义key进行的操作

op取值

计算类型 说明

sum(key) 对指定key的value求和

max(key) 对指定key的value求最大值

set(key) 对指定key的value做set做操作

item(key) 对指定key的value计算出对应的人数人次

item\_sum(key1, key2) 按照key1分类，对key2的value字段求和

item\_max(key1, key2) 按照key1分类，对key2的value字段求最大值

item\_set(key1, key2) 按照key1分类，对key2的value字段做set操作，即取每天最后一个值

sum\_distr(key) 需要依赖于区间分布，然后计算出对key的value字段求和后属于哪个区间的分布

max\_distr(key) 需要依赖于区间分布，然后计算出对key的value字段求最大值后属于哪个区间的分布

set\_distr(key) 对key的 value字段做set操作后做分布

ip\_distr ip分布，需要根据IP地址查出对应的地区

对每一条日志，都会默认计算出人数人次 ；

每条日志在后台计算的时候都会扩充成至少三条日志进行计算，主要是对gid、zid、sid、pid进行扩充

比如

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Gid(游戏ID) | Zid(区ID) | Sid(服ID) | Pid(平台ID) | 备注 |
| 15 | 2 | 4 | 1 | 原始日志，游戏ID为15的游戏在淘米平台上的2区4服 |
| 15 | -1 | -1 | 1 | 扩充出来的日志，游戏ID为15的游戏在淘米平台上的全区全服数据 |
| 15 | -1 | -1 | -1 | 扩充出来的日志,游戏ID为15的游戏在全平台全区全服的数据 |

下面是具体的日志示例：

## 1. 1、人数 人次(该示例已插入数据库配置表)

**原始的日志格式**：

\_stid\_=领地争夺战 \_sstid\_=点击传送舱装置 acid=具体的米米号 plid=角色标识(如果需要按角色查看，不需要按角色查看时，为空即可)——可计算出在领地争夺战中点击传送舱装置的人数以及人次

**转发给实时计算模块的日志格式**：

\_stid\_=领地争夺战 \_sstid\_=点击传送舱装置 acid=米米号 plid=角色标识 op=count(1)

**原始的日志格式：**

\_stid\_=领地争夺战 \_sstid\_=成功报名 acid=具体的米米号 plid=角色标识(如果需要按角色查看，不需要按角色查看时，为空即可)——可计算出在领地争夺战中成功报名的人数以及人次

**转发给实时计算模块的日志格式**：

\_stid\_=领地争夺战 \_sstid\_=成功报名 acid=米米号 plid=角色表示 op=count(1)

## 1.2、人数 人次 数量

**原始的日志格式**:

\_stid\_=游戏输出 sstid=撞球对抗赛 acid=米米号1 plid=角色标识1 赛尔豆=17 op=sum:赛尔豆

\_stid\_=游戏输出 sstid=撞球对抗赛 acid=米米号2 plid=角色标识2 赛尔豆=27 op=sum:赛尔豆

——可计算出撞球对抗赛的人数人次以及输出的赛尔豆总和

**转发给实时计算模块的日志格式**:

\_stid\_=游戏输出 \_sstid\_=撞球对抗赛 acid=米米号1 plid=角色标识1 op=count:1

\_stid\_=游戏输出 \_sstid\_=撞球对抗赛 acid=米米号1 plid=角色标识1 赛尔豆=17 op=sum:赛尔豆

\_stid\_=游戏输出 \_sstid\_=撞球对抗赛 acid=米米号2 plid=角色标识2 op=count:1

\_stid\_=游戏输出 \_sstid\_=撞球对抗赛 acid=米米号2 plid=角色标识2 塞尔豆=27 op=sum:赛尔豆

## 1.3、人数 人次 数量1 数量2

**原始的日志格式：**

\_stid\_=购买家具类商品 \_sstid\_=皇冠猪猪屋 acid=具体的米米号 plid=角色标识 购买数量=10 花费金额=180 op=sum:购买数量|sum:花费金额

——可计算出购买皇冠猪猪屋的人数人次、购买总数量、金额总和

如果要分别统计VIP和非VIP用户购买的数量，\_stid\_分别填成vip用户购买家具类商品和非VIP用户购买家具类商品

**转发给实时计算模块的日志格式:**

\_stid\_=购买家具类商品 \_sstid\_=皇冠猪猪屋 acid=具体的米米号 plid=角色标识 op=count:1

\_stid\_=购买家具类商品 \_sstid\_=皇冠猪猪屋 acid=具体的米米号 plid=角色标识 购买数量=10 op=sum:购买数量

\_stid\_=购买家具类商品 \_sstid\_=皇冠猪猪屋 acid=具体的米米号 plid=角色标识 花费金额=180 op= sum:花费金额

## 1.4、人数 人次 数量 等级分布(该示例已插入数据库配置表)

**原始的日志格式:**

\_stid\_=运营活动 \_sstid\_=宇宙大漫游 acid=具体的米米号 plid=角色标识 经验值=120 等级=’11-20’ 18 op=sum:经验值|item:等级

——可计算出参加宇宙大漫游活动的人数人次以及输出的经验值,还有参加该活动的各等级用户数

**转发给实时计算模块的日志格式**：

\_stid\_=运营活动 \_sstid\_=宇宙大漫游 acid=具体的米米号 plid=角色标识 op=count:1

\_stid\_=运营活动 \_sstid\_=宇宙大漫游 acid=具体的米米号 plid=角色标识 经验值=120 op=sum:经验值

\_stid\_=运营活动 \_sstid\_=宇宙大漫游 acid=具体的米米号 plid=角色标识 等级= 18 op= count:1

(可以求出item的人次，遇到op=count的情况，前面所有的都是key,如果op!=count,那么只有最后一个字段不是key)

**原始的日志格式**：

\_stid\_=游戏输出 sstid=保护导航仪 acid=米米号1 plid=角色标识1 赛尔豆=17 op=sum:赛尔豆

\_stid\_=游戏输出 sstid=保护导航仪 acid=米米号2 plid=角色标识2 赛尔豆=27 op=sum:赛尔豆

——可计算出保护导航仪的人数人次以及输出的赛尔豆总和

**转发给实时计算模块的日志格式**:

\_stid\_=游戏输出 sstid=保护导航仪 acid=米米号1 plid=角色标识1 op=count:1

\_stid\_=游戏输出 sstid=保护导航仪 acid=米米号1 plid=角色标识1 赛尔豆=17 op=sum:赛尔豆

\_stid\_=游戏输出 sstid=保护导航仪 acid=米米号2 plid=角色标识2 op=count:1

\_stid\_=游戏输出 sstid=保护导航仪 acid=米米号2 plid=角色标识2 赛尔豆=27 op=sum:赛尔豆

## 1.5、人数人次各子项的人数人次

**原始的日志格式**:

\_stid\_=新手任务 \_sstid\_=制作物品 acid=米米号 plid=角色标识 关卡=第1关 op=item:关卡

\_stid\_=新手任务 \_sstid\_=制作物品 acid=米米号 plid=角色标识 关卡=第2关 op=item:关卡

\_stid\_=新手任务 \_sstid\_=制作物品 acid=米米号 plid=角色标识 关卡=第3关 op=item:关卡

\_stid\_=新手任务 \_sstid\_=制作物品 acid=米米号 plid=角色标识 关卡=第4关 op=item:关卡

\_stid\_=新手任务 \_sstid\_=制作物品 acid=米米号 plid=角色标识 关卡=第5关 op=item:关卡

——可计算出各关卡的人数人次

也可以通过将\_stid\_=新手任务-制作物品 \_sstid\_=第1关的方式，按照第1种人数人次的方式做

**原始的日志格式:**

\_stid\_=战豆分布 \_sstid\_=战豆产出 acid=米米号 plid=角色标识 产出数量=’1-10’ op=item:产出数量

\_stid\_=战豆分布 \_sstid\_=战豆产出 acid=米米号 plid=角色标识 产出数量=’11-20’ op=item:产出数量

\_stid\_=战豆分布 \_sstid\_=战豆产出 acid=米米号 plid=角色标识 产出数量=’21-50’ op=item:产出数量

\_stid\_=战豆分布 \_sstid\_=战豆产出 acid=米米号 plid=角色标识 产出数量=’51以上’ op=item:产出数量

——可统计出战豆产出的区间分布人数

## 1.6、人数 人次 各子项的人数人次数量 等级分布

**原始的日志格式**:

\_stid\_=购买道具 \_sstid\_=白甲机动战士 acid=米米号 plid=角色标识 职业=熊猫 数量=15 价格=150 等级=15 op=sum:数量|item:职业|item\_sum:职业，数量|item\_sum:职业,价格|item\_set\_distr:职业,等级

\_stid\_=购买道具 \_sstid\_=白甲机动战士 acid=米米号 plid=角色标识 职业=兔子 数量=15 价格=150 等级=18 op= sum:数量|item:职业|item\_sum:职业，数量|item\_sum:职业,价格 |item\_set\_distr:职业,等级

\_stid\_=购买道具 \_sstid\_=白甲机动战士 acid=米米号 plid=角色标识 职业=猴子 数量=15 价格=150 等级=24 op= sum:数量|item:职业|item\_sum:职业，数量|item\_sum:职业,价格 |item\_set\_distr:职业,等级

——可计算出购买各个道具的人数人次 以及各个职业购买的人数人次，数量总和，价格总和以及购买该道具的各职业用户等级分布

## 1.7、最大值 累加值分布的人数人次

**原始的日志格式**

\_stid\_=战豆分布 \_sstid\_=战斗消耗 acid=米米号 plid=角色标识 消耗数量=3 op=dsitr\_sum:消耗数量|distr\_max:消耗数量

\_stid\_=战豆分布 \_sstid\_=战斗消耗 acid=米米号 plid=角色标识 消耗数量=16 op=dsitr\_sum:消耗数量,区间|distr\_max:消耗数量，区间

——可计算出消耗的战豆分布 分最大和累积

# 2、数据库设计

## 2.1、游戏基本信息表t\_game\_info

|  |  |  |
| --- | --- | --- |
| 字段名 | 类型 | 备注 |
| game\_id | int | 主键字段 游戏ID 与account平台保持一致 |
| game\_name | char(64) | 唯一键字段 游戏名称 |
| game\_type | enum | 游戏类型页游(webgame) 手游(mobilegame) 端游(clientgame) 测试(test) |
| auth\_id | varchar(32) | 游戏权限ID，由权限系统分配 |
| status | tinyint | 状态字段 0:未使用 1：已使用 2：删除 |

## 2.2、游戏平台区服映射表t\_gpzs\_info

|  |  |  |
| --- | --- | --- |
| 字段名 | 类型 | 备注 |
| gpzs\_id | bigint | 自增主键 |
| game\_id | int | 唯一键字段 游戏ID与t\_game\_info表的game\_id一致 |
| platform\_id | int | 唯一键字段 平台ID，如淘米平台规定为1 |
| zone\_id | int | 唯一键字段 区ID，多区多服游戏需要使用,默认全区全服为-1 |
| server\_id | int | 唯一键字段 服ID，多区多服游戏需要使用,默认全区全服为-1 |
| gpzs\_name | char(64) | 平台区服名字 |
| status | tinyint | 状态字段 0：正常 1：下架 |

## 2.3、树结构表t\_web\_tree

|  |  |  |
| --- | --- | --- |
| 字段名 | 类型 | 备注 |
| node\_id | int | 自增主键 |
| node\_name | char(128) | 节点名字 ，默认取值stid或sstid 支持修改 |
| game\_id | int | 唯一键字段 游戏id |
| parent\_id | int | 父节点ID |
| is\_leaf | tinyint | 是否叶子节点 1:叶子节点 0：非叶子节点 |
| is\_basic | tinyint | 是否基础统计项 1:基础统计项 0：非基础统计项 |
| hide | tinyint | 是否隐藏 0:显示 1：隐藏 |
| stid | char(64) | 唯一键字段 统计项名称 |
| sstid | char(64) | 唯一键字段 子统计项名称 |

>>比如一条在线日志和自定义的赛尔豆产出日志

>>\_hip\_=10.1.1.184 \_stid\_=\_olcnt\_ \_sstid\_=\_olcnt\_ \_gid\_=13 \_zid\_=-1 \_sid\_=-1 \_pid\_=1 \_ts\_=1383705667 \_acid\_=-1 \_plid\_=-1 \_olcnt\_=0 \_op\_=max:\_olcnt\_

>>\_hip\_=10.1.1.60 \_stid\_=游戏输出 \_sstid\_=撞球对抗赛 \_gid\_=2 \_zid\_=-1 \_sid\_=-1 \_pid\_=1 \_ts\_=1383705667 \_acid\_=47159775 \_plid\_=-1 赛尔豆=100 \_op\_=sum:赛尔豆

>>\_hip\_=10.1.1.60 \_stid\_=游戏输出 \_sstid\_=划船比赛 \_gid\_=2 \_zid\_=-1 \_sid\_=-1 \_pid\_=1 \_ts\_=1383705667 \_acid\_=47159775 \_plid\_=-1 赛尔豆=100 \_op\_=sum:赛尔豆

>>\_hip\_=10.1.1.182 \_stid\_=每周任务(接收) \_sstid\_=幸福的使命 \_gid\_=5 \_zid\_=-1 \_sid\_=-1 \_pid\_=1 \_ts\_=13837896 \_acid\_=34545221 \_plid\_=-1 lv='1-5' \_op\_=item:lv

>>\_hip\_=10.1.1.182 \_stid\_=每周任务(接收) \_sstid\_=幸福的使命 \_gid\_=5 \_zid\_=-1 \_sid\_=-1 \_pid\_=1 \_ts\_=13837896 \_acid\_=13545221 \_plid\_=-1 lv='6-10' \_op\_=item:lv

>>\_hip\_=10.1.1.182 \_stid\_=每周任务(接收) \_sstid\_=幸福的使命 \_gid\_=5 \_zid\_=-1 \_sid\_=-1 \_pid\_=1 \_ts\_=13837896 \_acid\_=78545221 \_plid\_=-1 lv='11-20' \_op\_=item:lv

>>\_hip\_=10.1.1.182 \_stid\_=每周任务(接收) \_sstid\_=幸福的使命 \_gid\_=5 \_zid\_=-1 \_sid\_=-1 \_pid\_=1 \_ts\_=13837896 \_acid\_=5654345221 \_plid\_=-1 lv='21以上' \_op\_=item:lv

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| node\_id | node\_name | game\_id | parent\_id | is\_leaf | is\_basic | hide | stid | sstid |
| 1 | \_olcnt\_ | 13 | 0 | 0 | 1 | 0 | \_olcnt\_ | '空字符串' |
| 2 | \_olcnt | 13 | 1 | 1 | 1 | 0 | \_olcnt\_ | \_olcnt\_ |
| 3 | 游戏输出 | 2 | 0 | 0 | 0 | 0 | 游戏输出 | '空字符串' |
| 4 | 撞球对抗赛 | 2 | 3 | 1 | 0 | 0 | 游戏输出 | 撞球对抗赛 |
| 5 | 划船比赛 | 2 | 3 | 1 | 0 | 0 | 游戏输出 | 划船比赛 |
| 6 | 每周任务(接收) | 5 | 0 | 0 | 0 | 0 | 每周任务(接收) |  |
| 7 | 幸福的使命 | 5 | 6 | 1 | 0 | 0 | 每周任务(接收) | 幸福的使命 |

## 2.4、reportId映射表t\_report\_info

|  |  |  |
| --- | --- | --- |
| 字段名 | 类型 | 备注 |
| report\_id | bigint | 自增 主键 |
| report\_name | char(64) | 统计项名称 默认取stid-sstid |
| game\_id | int | 唯一键字段 游戏id |
| stid | char(64) | 唯一键字段 统计项名称 |
| sstid | char(64) | 唯一键字段 子统计项名称 |
| op\_fields | char(64) | 唯一键字段 操作的字段 |
| op\_type | enum | 唯一键字段 操作类型 人次(count) 人数(ucount) 求和(sum) 最大(max) 覆盖(set) IP分布(ip\_distr) 等级分布(distr\_max,distr\_sum,distr\_set) |
| is\_multi | tinyint | 0- 非multi类型，1- multi类型 |
|  |  |  |

比如对于上面的三条日志会产生如下的记录

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| report\_id | report\_name | game\_id | stid | sstid | op\_fields | op\_type |  |
| 1 | \_olcnt\_-\_olcnt\_ | 13 | \_olcnt\_ | \_olcnt\_ | \_olcnt\_ | max |  |
| 2 | 游戏输出-撞球对抗赛 | 2 | 游戏输出 | 撞球对抗赛 | 空字段串 | Count |  |
| 3 | 游戏输出-撞球对抗赛 | 2 | 游戏输出 | 撞球对抗赛 | 空字段串 | Ucount |  |
| 4 | 游戏输出-撞球对抗赛 | 2 | 游戏输出 | 撞球对抗赛 | 赛尔豆 | Sum |  |
| 5 | 游戏输出-划船比赛 | 2 | 游戏输出 | 划船比赛 | 空字段串 | Count |  |
| 6 | 游戏输出-划船比赛 | 2 | 游戏输出 | 划船比赛 | 空字段串 | Ucount |  |
| 7 | 游戏输出-划船比赛 | 2 | 游戏输出 | 划船比赛 | 赛尔豆 | Sum |  |
| 8 | 每周任务(接收)-幸福的使命 | 5 | 每周任务(接收) | 幸福的使命 | 空字符串 | Count |  |
| 9 | 每周任务(接收)-幸福的使命 | 5 | 每周任务(接收) | 幸福的使命 | 空字符串 | Ucount |  |
| 10 | 每周任务(接收)-幸福的使命 | 5 | 每周任务(接收) | 幸福的使命 | lv | Count |  |
| 11 | 每周任务(接收)-幸福的使命 | 5 | 每周任务(接收) | 幸福的使命 | lv | ucount |  |

## 2.5、数据项映射表t\_data\_info

|  |  |  |
| --- | --- | --- |
| 字段名 | 类型 | 备注 |
| data\_id | bigint | 自增主键 |
| data\_name | char(64) | 数据项名称 默认取值为range - datatype |
| r\_id | Bigint | 唯一键字段 代表是统计项report\_id还是加工项result\_id |
| type | Enum | 唯一键字段 统计项(report) 加工项(result) |
| range | Char(64) | 唯一键字段 范围字段 |
| display\_order | Int | 排序字段,显示时用 |
| hide | tinyint | 是否隐藏0:显示 1:隐藏 |
| shtash | int | stid的hash值，用于分库分表 |

比如上面的日志入库记录如下

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Data\_id | Data\_name | R\_id | type | range | Display\_order | hide | sthash |
| 1 | 在线人数 | 1 | Report | 空字符串 | 0 | 0 | 13 |
| 2 | 人次 | 2 | Report | 空字符串 | 0 | 0 | 15 |
| 3 | 人数 | 3 | Report | 空字符串 | 0 | 0 | 15 |
| 4 | 求和 | 4 | Report | 空字符串 | 0 | 0 | 15 |
| 5 | 人次 | 5 | Report | 空字符串 | 0 | 0 | 15 |
| 6 | 人数 | 6 | Report | 空字符串 | 0 | 0 | 15 |
| 7 | 求和 | 7 | Report | 空字符串 | 0 | 0 | 15 |
| 8 | 人次 | 8 | Report | 空字符串 | 0 | 0 | 19 |
| 9 | 人数 | 9 | Report | 空字符串 | 0 | 0 | 19 |
| 10 | 人次 | 10 | Report | 1-5 | 0 | 0 | 19 |
| 11 | 人次 | 10 | Report | 6-10 | 0 | 0 | 19 |
| 12 | 人次 | 10 | Report | 11-20 | 0 | 0 | 19 |
| 13 | 人次 | 10 | Report | 20以上 | 0 | 0 | 19 |
| 14 | 人数 | 11 | Report | 1-5 | 0 | 0 | 19 |
| 15 | 人数 | 11 | Report | 6-10 | 0 | 0 | 19 |
| 16 | 人数 | 11 | Report | 11-20 | 0 | 0 | 19 |
| 17 | 人数 | 11 | report | 20以上 | 0 | 0 | 19 |
|  |  |  |  |  |  |  |  |

## 2.6、导航表t\_web\_navi(新统计页面导航表)

|  |  |  |
| --- | --- | --- |
| 字段名 | 类型 | 备注 |
| navi\_id | int | 自增主键 |
| navi\_name | varchar(128) | 导航名字 |
| navi\_key | varchar(128) | 导航键，用于加载静态资源 |
| navi\_url | varchar(128) | 导航地址 |
| parent\_id | int | 父导航ID |
| auth\_id | varchar(32) | 导航权限ID，由权限系统分配 |
| level | tinyint | 导航层级 |
| display\_order | int | 导航显示顺序，值越小越靠前，同层级有效 |
| status | tinyint | 显示状态：1-显示，0-不显示 |

# 3、模块设计



## 3.1、Stat-logger

提供给项目部使用，用于收集统计日志,落到本地

## 3.2、Stat-client

将项目部落在本地的统计日志发送给统计中转

具体实现细节待补充

## 3.3、Stat-proxy

各个IDC部署一个中转，负责转发统计日志给服务端

具体实现细节待补充

## 3.4、Stat-server

统计服务端，主要功能分为两块

1. 将接收到的每条日志写本地文件，由专门的脚本定时上传到hadoop集群，上传脚本参考文档《上传服务部署文档》
2. 根据每条日志的OP字段对日志进行拆分，默认都会拆分出一条人次的日志发给dbserver模块

比如下面一条日志：

\_hip\_=10.1.1.60 \_stid\_=游戏输出 \_sstid\_=撞球对抗赛 \_gid\_=2 \_zid\_=-1 \_sid\_=-1 \_pid\_=1 \_ts\_=1383705667 \_acid\_=47159775 \_plid\_=-1 赛尔豆=100 lv=23 \_op\_=sum:赛尔豆|set\_distr:lv

会被拆成下面三条日志发给db-server

\_hip\_=10.1.1.60 \_stid\_=游戏输出 \_sstid\_=撞球对抗赛 \_gid\_=2 \_zid\_=-1 \_sid\_=-1 \_pid\_=1 \_ts\_=1383705667 \_acid\_=47159775 \_plid\_=-1 \_op\_=count //统计人次

\_hip\_=10.1.1.60 \_stid\_=游戏输出 \_sstid\_=撞球对抗赛 \_gid\_=2 \_zid\_=-1 \_sid\_=-1 \_pid\_=1 \_ts\_=1383705667 \_acid\_=47159775 \_plid\_=-1 赛尔豆=100 \_op\_=sum:赛尔豆

\_hip\_=10.1.1.60 \_stid\_=游戏输出 \_sstid\_=撞球对抗赛 \_gid\_=2 \_zid\_=-1 \_sid\_=-1 \_pid\_=1 \_ts\_=1383705667 \_acid\_=47159775 \_plid\_=-1 lv=23 \_op\_=set\_distr:lv

具体实现细节待补充

## 3.5、stat-center

暂时未实现，将来用于控制统计系统中各个模块的

## 3.6、Db-server

Dbserver模块负责实时数据也即分钟数据的计算、配置项的入库和数据计算结果的入库

### 3.6.1、分钟数据的计算

1.取gpzs\_id/data\_id:在hash中则直接取出,否则去redis中查找,若没有则连接config\_server(处理参见3.7).若存在则取出该值并插入hash,否则不继续下面步骤.

2.缓存实时入库的数据:

1)本地缓存map中未找到则插入map,并执行3).

2)已存在于本地缓存,根据操作符更新本地数据后执行3).

3)根据是否达到更新时间或者是否需要强制或者是否达到缓存上限决定是否需要写入DB并清除该条缓存.具体写入规则参见3.6.3.

### 3.6.2、配置项的入库

先从redis查询配置项是否存在，

如果已经存在，忽略

如果不存在，将项目配置数据传给config-server, config-server会直接插入数据库相关配置表，然后更新到缓存redis中

细节待补充

### 3.6.3、数据计算结果的入库

MINUTE数据写入t\_db\_data\_minute\_\*库,DAY数据写入t\_db\_data\_day\_\*库,具体表根据hash值而定(db\_id=(hash/100)%100,table\_id=hash%100),不同操作符写入规则如下:

1. COUNT:/SUM:直接插入值,若主键已存在则将待写入的值与表中值求和后写入.
2. MAX:直接插入值,若主键已存在则取待写入值与表中值最大值.
3. SET:直接插入值,若主键已存在则用待写入值覆盖表中值.
4. 其他目前未定义.

## 3.7、config-server

将db-server的请求写入数据库(已存在则不写)并更新redis,对于添加统计项操作,分布的统计项id直接检索redis,并根据检索结果返回失败或成功.

## 3.8、mapreduce

1. 起始调度入口
2. [集群hadoop@192.168.11.129 的crontab](mailto:集群hadoop@192.168.11.129%20的crontab) -l，每日9时执行

0 9 \* \* \* cd /opt/taomee/hadoop/bigdata && /bin/sh calc\_yesterday.sh

1. 所有脚本路径：/opt/taomee/hadoop/bigdata
2. shell调度流程如图3.8.1，详见calc.sh：



图3.8.1

备注：

1. 调度从calc\_yesterday.sh开始，计算前一日的源数据
2. calc.sh包含了基础项、加工项等所有调度脚本
3. calc\_basic.sh将源数据按stid分出不同的文件，以供后续并行计算
   1. 基础项： COUNT、UCOUNT、SUM、DISTR\_SUM、MAX、DISTR\_MAX、SET、DISTR\_SET、IP\_DISTR等
   2. 加工项：从192.168.71.68:3306的db\_stat\_config库中获取stid信息select stid from t\_common\_stid，不同stid输出一个文件
   3. 示例如下图3.8.2

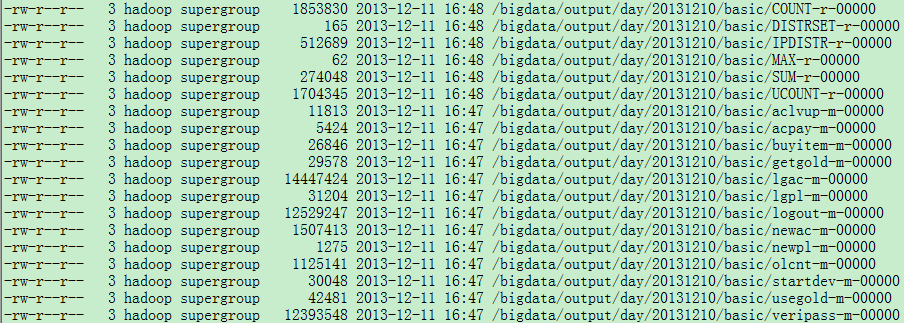
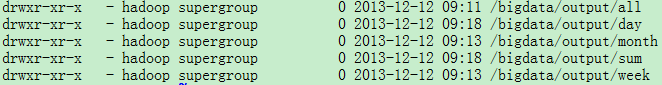


图3.8.2

1. calc\_basic.sh执行结束后，后续加工如图3.8.1可部分并行执行
2. 对新增用户的计算在calc\_account\_all.sh中，其他依赖新增的task需在calc\_account\_all.sh执行结束后再执行
3. Hdfs目录结构

/bigdata/input/：输入路径：源数据

/bigdata/output/：输出路径：基础项、加工项等，明细如下：

 all：累计数据；day：日数据；month：月数据；sum：求和数据；week：周数据

1. 输入输出，示例：

"\_hip=10.1.1.63\t\_stid\_=active\t\_sstid\_=active\t\_gid\_=seer\t\_pid\_=taomee\t\_zid\_=0\t\_sid\_=1\t\_ts\_=1383117270\t\_acid\_=185908545\tproduct=5\tcoins=10\t\_plid\_=1383117270\t\_op\_=sum:coins|item:product|item\_sum:product,coins"

* calc\_basic.sh基础项：输入原始日志格式，分op解析

**（op [value] account[-player] game platform zone server stid sstid [op-field] [key] time）**

* 1. UCOUNT,COUNT

map outputkey：**opcode\t\_acid\_-\_plid\_\t\_gid\_\t\_zid\_\t\_sid\_\t\_pid\_\t\_stid\_\t\_sstid\_**

map outputvalue：**1**

reduce outputkey：**\_acid\_-\_plid\_[\tsum(v)]\t\_gid\_\t\_zid\_\t\_sid\_\t\_pid\_\t\_stid\_\t\_sstid\_**

reduce outputvalue：**NullWritable**

* 1. ITEM的UCOUNT,COUNT

map outputkey：**opcode\t\_acid\_-\_plid\_\t\_gid\_\t\_zid\_\t\_sid\_\t\_pid\_\t\_stid\_\t\_sstid\_\tproduct\t5**

map outputvalue：**1**

reduce outputkey：**\_acid\_-\_plid\_[\tsum(v)]\t\_gid\_\t\_zid\_\t\_sid\_\t\_pid\_\t\_stid\_\t\_sstid\_\tproduct\t5**

reduce outputvalue：**NullWritable**

* 1. SUM,MAX,SET,IP\_DISTR,DISTR\*（SET、DISTR\_SET有\_ts\_）

map outputkey：**opcode\t\_acid\_-\_plid\_\t\_gid\_\t\_zid\_\t\_sid\_\t\_pid\_\t\_stid\_\t\_sstid\_\tcoins[\t\_ts\_]**

map outputvalue：**10**

reduce outputkey：**\_acid\_-\_plid\_[\tsum(v)]\t\_gid\_\t\_zid\_\t\_sid\_\t\_pid\_\t\_stid\_\t\_sstid\_\tcoins**

reduce outputvalue：**NullWritable**

* 1. ITEM\_SUM,ITEM\_MAX,ITEM\_SET,ITEM\_DISTR（ITEM\_DISTR有\_ts\_）

map outputkey：

**opcode\t\_acid\_-\_plid\_\t\_gid\_\t\_zid\_\t\_sid\_\t\_pid\_\t\_stid\_\t\_sstid\_\tproduct\tcoins\t5\t10[\t\_ts\_]**

map outputvalue：**1**

reduce outputkey：

**\_acid\_-\_plid\_[\tsum(v)]\t\_gid\_\t\_zid\_\t\_sid\_\t\_pid\_\t\_stid\_\t\_sstid\_\ tproduct\tcoins\t5\t10[\t\_ts\_]**

reduce outputvalue：**NullWritable**

* calc\_basic.sh分事件：输入原始日志，按stid分文件输出（如图3.8.2）
* calc\_account\_all.sh：计算首次登陆用户，task12累计活跃用户

## 3.9、mapreduce计算结果入库

一段独立的客户端程序，负责将hdfs上的结果数据发送给dbserver进行入库操作

1. 客户端结构



1. 结构说明：
2. hdfs客户端包含：
   * 流程作业HdfsJob
   * 发送任务SendTask
   * 数据解析器DataParser
   * 发送协议TranProtocol
   * 初始化配置init.properties
3. 流程作业HdfsJob：
   1. 负责作业流程的控制，初始化加载hdfs result files到文件队列FileQueue里面，初始化系统预置任务线程；
   2. Job内部启动定时线程，分配文件队列里的文件给到发送任务；
4. 发送任务SendTask：
   1. 负责将分配到的文件，通过DataParser解析生成相应的TranProtocol；
   2. 从配置的Socket服务列表中随机获取一个服务地址，将TranProtocol发送至DBServer
5. 初始化配置init.properties：
   1. 可配置核心（task.thread.corenum）、最大（task.thread.maxnum）线程数量
   2. 可配置多个（逗号分隔）DBServer服务地址（server.socket.host= 192.168.71.76）、端口（server.socket.port=19902）
6. 调用说明
   1. Shell脚本相关定义（config.sh）：

DB\_UPLOAD="java –cp ${HADOOP\_JAR\_PATH}:`${HADOOP\_PATH}hadoop classpath` com.taomee.bigdata.client.HdfsDataFetchJob -uri hdfs://192.168.11.128:8020/ "

$DB\_UPLOAD [-path <input path> //加工结果数据在hadoop上的输出地址

-type <data\_type> //数据类型 0-分钟，1-小时，2-天

-date <time> //要处理的日期 20131210

-task <taskid>] //加工项id

* 1. 调用示例：

$DB\_UPLOAD -type 2 -date $date -path ${DAY\_DIR}/$date/sum/part\*