|  |  |
| --- | --- |
| 文档编号 |  |
| 文档版本 |  |
| 拟制人 |  |
| 日期 |  |

Redis设计说明书

目录

[变更说明 5](#_Toc471822956)

[1. 简介 7](#_Toc471822957)

[1.1 目的 7](#_Toc471822958)

[2. 价格 7](#_Toc471822959)

[2.1 使用场景 7](#_Toc471822960)

[2.2 Key-Value设计和举例 7](#_Toc471822961)

[2.3 其他说明 8](#_Toc471822962)

[3. 序列号 8](#_Toc471822963)

[3.1 使用场景 8](#_Toc471822964)

[3.2 Key-Value设计和举例 8](#_Toc471822965)

[3.3 其他说明 8](#_Toc471822966)

[4. IP 8](#_Toc471822967)

[4.1 使用场景 8](#_Toc471822968)

[4.2 Key-Value设计和举例 9](#_Toc471822969)

[5. 服务开通状态 9](#_Toc471822970)

[5.1 使用场景 9](#_Toc471822971)

[5.2 Key-Value设计和举例 9](#_Toc471822972)

[5.3 其他说明 9](#_Toc471822973)

[6. ECSC Session 10](#_Toc471822974)

[6.1 使用场景 10](#_Toc471822975)

[6.2 Key-Value设计和举例 10](#_Toc471822976)

[6.3 其他说明 10](#_Toc471822977)

[7. 短信配额 11](#_Toc471822978)

[7.1 使用场景 11](#_Toc471822979)

[7.2 Key-Value设计和举例 11](#_Toc471822980)

[8. 对象存储 12](#_Toc471822981)

[8.1 使用场景 12](#_Toc471822982)

[8.2 Key-Value设计和举例 12](#_Toc471822983)

[8.3 其他说明 13](#_Toc471822984)

[9. 枚举字典 13](#_Toc471822985)

[9.1 使用场景 13](#_Toc471822986)

[9.2 Key-Value设计和举例 14](#_Toc471822987)

[10. 密钥 14](#_Toc471822988)

[10.1 使用场景 14](#_Toc471822989)

[10.2 Key-Value设计和举例 14](#_Toc471822990)

[11. 云资源同步 15](#_Toc471822991)

[11.1 使用场景 15](#_Toc471822992)

[11.2 Key-Value设计和举例 16](#_Toc471822993)

[12. 指标监控 17](#_Toc471822994)

[12.1 使用场景 17](#_Toc471822995)

[12.2 Key-Value设计和举例 17](#_Toc471822996)

[13. 监控报警 18](#_Toc471822997)

[13.1 使用场景 18](#_Toc471822998)

[13.2 Key-Value设计和举例 18](#_Toc471822999)

[14. 消息 20](#_Toc471823000)

[14.1 使用场景 20](#_Toc471823001)

[14.2 Key-Value设计和举例 20](#_Toc471823002)

[14.3 说明 20](#_Toc471823003)

[15. 忘记密码 21](#_Toc471823004)

[15.1 使用场景 21](#_Toc471823005)

[15.2 Key-Value设计和举例 21](#_Toc471823006)

[15.3 其他说明 21](#_Toc471823007)

[16. 标签 21](#_Toc471823008)

[16.1 使用场景 21](#_Toc471823009)

[16.2 Key-Value设计和举例 21](#_Toc471823010)

[17. 客户冻结 23](#_Toc471823011)

[17.1 使用场景 23](#_Toc471823012)

[17.2 Key-Value设计和举例 23](#_Toc471823013)

[18. 发送短信邮件提醒 23](#_Toc471823014)

[18.1 使用场景 23](#_Toc471823015)

[18.2 Key-Value设计和举例 23](#_Toc471823016)

[19. 管理系统计划任务漏跑信息 24](#_Toc471823017)

[19.1 使用场景 24](#_Toc471823018)

[19.2 Key-Value设计和举例 24](#_Toc471823019)

[20. OBS计费相关标识Key 24](#_Toc471823020)

[20.1 使用场景 24](#_Toc471823021)

[20.2 Key-Value设计和举例 25](#_Toc471823022)

[21. Api访问次数相关key 25](#_Toc471823023)

[21.1 使用场景 25](#_Toc471823024)

[21.2 Key-Value设计和举例 25](#_Toc471823025)

[22.按分钟统计API访问指标相关Key 27](#_Toc471823026)

[22.1使用场景 27](#_Toc471823027)

[22.2Key-Value设计和举例 27](#_Toc471823028)

[23.存放上一个时刻对应API访问指标信息相关Key 28](#_Toc471823029)

[23.1使用场景 28](#_Toc471823030)

[23.2Key-Value设计和举例 28](#_Toc471823031)

[24.存放报警条件分析数据相关Key 29](#_Toc471823032)

[24.1使用场景 29](#_Toc471823033)

[24.2Key-Value设计和举例 29](#_Toc471823034)

[25.是否触发报警提醒以及产生报警信息相关Key 30](#_Toc471823035)

[25.1使用场景 30](#_Toc471823036)

[25.2Key-Value设计和举例 31](#_Toc471823037)

[26.客户冻结黑名单同步 31](#_Toc471823038)

[26.1使用场景 31](#_Toc471823039)

[26.2Key-Value设计和举例 31](#_Toc471823040)

变更说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **日期** | **版本** | **变更位置** | **变更说明** | **作者** |
| 2016-09-01 | 1.0.0 |  | **新建，根据目前已使用的情况整理** | 陈浩 |
| 2016-09-26 |  | 18 | **新增记录是否已发送过资源停用消息** | 曹翔宇 |
| 2016-11-10 |  | 10.2 | **新增密钥key-value值** | 曹翔宇 |
| 2016-11-18 |  | 18.2 | **新增记录是否发送过首次达到信用额度消息** | 曹翔宇 |
| 2016-12-01 |  | 21 | **新增api访问次数相关key** | 曹翔宇 |
| 2017-01-10 |  | 22-25 | **新增api资源监控以及报警管理相关Key** | 王建凯 |
| 2017-01-10 |  | 26 | **新增黑名单key-value值** | 李岩超 |
| 2017-02-27 |  | 12.2 | **新增云数据库实例状态同步key-value值** | 刘壮壮 |
| 2017-03-09 |  | 31 | **新增负载均衡/vip状态** | 曹翔宇 |
| 2017-03-23 |  | 34 | **新增负载均衡乐观锁** | 曹翔宇 |
| 2017-04-12 |  | 34 | **增加删除负载均衡时成员状态rediskey** | 曹翔宇 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# 简介

## 目的

在Redis的使用中，key的设计一般是固定的前缀+可变的后缀，key如果有多段，用冒号（：）分隔。为便于Redis的管理，所有使用Redis的场景在本文档中维护，包括key的格式，value的内容和示例。

# 价格

## 使用场景

1、ECMC配置资源价格

包括添加、编辑、删除一种资源的价格，需要对redis进行相应更新操作。

2、ECSC计算业务价格

根据数据中心id、付费类型、计费单位英文名称，从redis中查询一种计费单位的所有设置的价格，之后找出符合条件的进行价格计算。

## Key-Value设计和举例

价格Key的设计为："price:"+数据中心id+":"+付费类型（预付费：1，后付费：2）+":"+计费单位英文名称。

Value的设计为PriceRedis对应集合的json字符串。

如：price:1509211142230:1:connCount

[

{

"id": "4028909956c082d70156c099059b0008", //记录id

"createTime": 1472110069146, //创建时间

"price": 100, //价格

"start": 0, //区间开始

"end": 5000 //区间结束

},

{

"id": "4028909956c082d70156c099394b0009",

"createTime": 1472110082378,

"price": 80,

"start": 5001,

"end": 10000

},

]

## 其他说明

价格信息同时存在于mysql和redis中，ECSC做计算业务时查询redis中的数据，ECMC查询价格时，查询mysql中的数据，做添加、编辑、删除时，同时更新mysql和redis。

# 序列号

## 使用场景

产生交易流水号、订单编号、支付流水号

## Key-Value设计和举例

序列号Key的设计为"serialnum:"+prefix

prefix分三类：

交易流水号："03"+dateStr

订单编号："01" + dateStr

支付流水号："02" + dateStr

dateStr形式为"yyyyMMdd "

Value设计为一个正整数：表达意思为当前时间这一天内该类型编号总条数。

如：serialnum:0220160823{

71

}

## 其他说明

使用时，首先更新redis里的记录值，然后返回一个该类型的编号

# IP

## 使用场景

ECMC在线人数统计客户的登录地址

## Key-Value设计和举例

IP的Key设计为"ip:"+ip地址

Value设计为：淘宝库返回的ip地址的json字符串信息

如：ip:192.168.16.25{

"region": "",

"area\_id": "",

"country\_id": "IANA",

"isp": "",

"region\_id": "",

"ip": "192.168.16.25",

"country": "未分配或者内网IP",

"city": "",

"isp\_id": "",

"city\_id": "",

"area": "",

"county": "",

"county\_id": ""

}

# 服务开通状态

## 使用场景

开通OBS服务

## Key-Value设计和举例

服务开通状态Key的设计为:"cusservicestate:cusid:"+客户id。

Value设计为:BaseCusServiceState对应的json字符串信息。

如：cusservicestate:cusid:402890b75358dc190153596761ea001f{

"csId": "402890b7535e1b7201535e6132db0002", //记录ID

"cusId": "402890b75358dc190153596761ea001f", //客户ID

"obsState": "1" //开通状态，1表示开通

}

## 其他说明

目前此状态只能针对OBS服务，若要适用其他服务，需要对redis和代码做相应调整。

# ECSC Session

## 使用场景

ECMC在线人数统计

## Key-Value设计和举例

session共享Key的设计为："ECSCSessionID:"+系统生成的序列号

Value设计为：redis将session信息序列化后的二进制串

如：ECSCSessionID:2C8E343165DD492D03F02BA16D5F1994{

\xAC\xED\x00\x05sr\x00Dcom.orangefunction.tomcat.redissessions.SessionSerializationMetadataB\xD9\xD9\xF7v\xA2\xDBL\x03\x00\x01[\x00\x15sessionAttributesHasht\x00\x02[Bxpw\x14\x00\x00\x00\x10\xD3\xD5\x92\_\xD7\x18J\xBD:Y\xFA\x00\xCC\x9C\xD9\xAExsr\x00\x0Ejava.lang.Long;\x8B\xE4\x90\xCC\x8F#\xDF\x02\x00\x01J\x00\x05valuexr\x00\x10java.lang.Number\x86\xAC\x95\x1D\x0B\x94\xE0\x8B\x02\x00\x00xp\x00\x00\x01V\xE9\xF14\x09sq\x00~\x00\x03\x00\x00\x01V\xE9\xF14\x09sr\x00\x11java.lang.Integer\x12\xE2\xA0\xA4\xF7\x81\x878\x02\x00\x01I\x00\x05valuexq\x00~\x00\x04\x00\x00\x07\x08sr\x00\x11java.lang.Boolean\xCD r\x80\xD5\x9C\xFA\xEE\x02\x00\x01Z\x00\x05valuexp\x00sq\x00~\x00\x09\x01sq\x00~\x00\x03\x00\x00\x01V\xE9\xF1\xBAHt\x00 2C8E343165DD492D03F02BA16D5F1994sq\x00~\x00\x07\x00\x00\x00\x03t\x00\x07PASSKEYt\x00\x06547358t\x00\x08USERINFOsr\x00'com.eayun.common.filter.SessionUserInfo\x18\x1D\xE50\xB5\x0F\x1E\x19\x02\x00\x0FL\x00\x02IPt\x00\x12Ljava/lang/String;L\x00\x09cusCpnameq\x00~\x00\x13L\x00\x05cusIdq\x00~\x00\x13L\x00\x07cusNameq\x00~\x00\x13L\x00\x06cusOrgq\x00~\x00\x13L\x00\x05emailq\x00~\x00\x13L\x00\x05errorq\x00~\x00\x13L\x00\x07isAdmint\x00\x13Ljava/lang/Boolean;L\x00\x08lastTimet\x00\x10Ljava/util/Date;L\x00\x11lastVerifySmsTimeq\x00~\x00\x15L\x00\x05phoneq\x00~\x00\x13L\x00\x06roleIdq\x00~\x00\x13L\x00\x08roleNameq\x00~\x00\x13L\x00\x06userIdq\x00~\x00\x13L\x00\x08userNameq\x00~\x00\x13xpt\x00\x0D192.168.16.25pt\x00 402890905635d078015635e10e890004pt\x00\x06\xE7\xBF\x94\xE5\xAE\x87ppq\x00~\x00\x0Asr\x00\x0Ejava.util.Datehj\x81\x01KYt\x19\x03\x00\x00xpw\x08\x00\x00\x01V\xE9\xF1\xBAQxppt\x00 402890905635d078015635e1133c0072t\x00\x09\xE7\xAE\xA1\xE7\x90\x86\xE5\x91\x98t\x00 ff80808156ba43f00156ba80e1a50002t\x00\x0Aguanliyuant\x00\x05logint\x00){"startTime":1472803744830,"code":"6101"}w\x08\x00\x00\x01V\xE9\xF14\x09

}

## 其他说明

redis数据由tomcat配置文件配置完成后，客户访问时由系统自动更新。有30分钟的时间期限，到期自动注销。

使用时首先将取出的数据反序列化，然后获取相应的信息

# 短信配额

## 使用场景

公有云平台给客户发送短信的相关业务

## Key-Value设计和举例

与此业务相关的redis有四种：

①、提醒客户报警短信配额不够的提醒短信消息队列

Key设计为："EAYUN\_QUOTA\_MSGQUEUE"

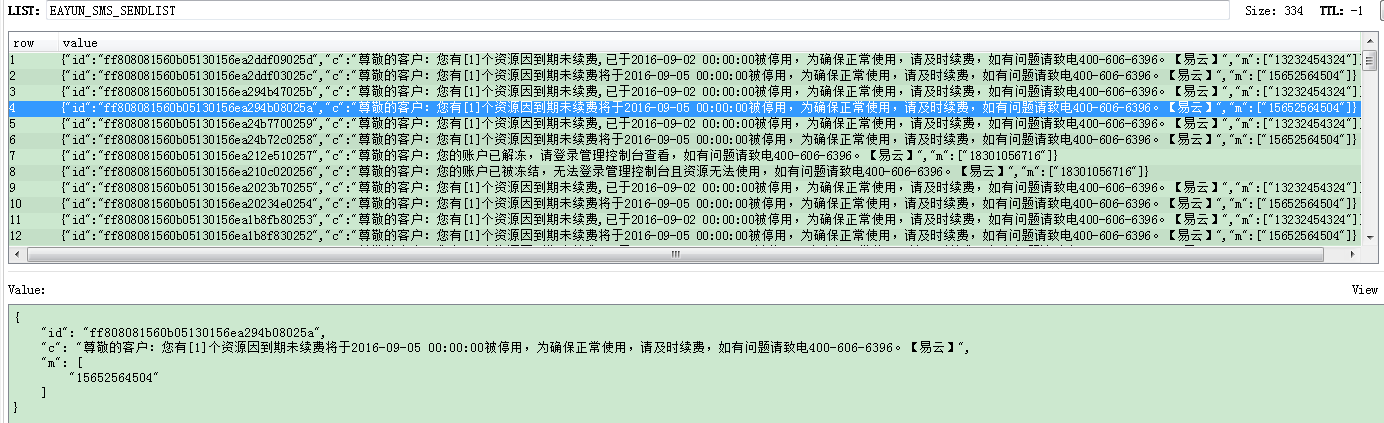
Value设计为信息相关json的集合

②、除①情况外的发送短信消息队列

Key设计为："EAYUN\_SMS\_SENDLIST"

Value设计为信息相关json的集合

如：



③、客户项目（新版为数据中心）下的短信配额

Key设计为："smsQuota:total:"+客户ID+":"+项目ID

Vaule设计为非负整数，表示短信配额数

如：smsQuota:total:40288ee654cbc2650154cbc43a6d0000:

05fba3165b6048069678f5814bfd9802{

5

}

④、客户项目（现在为数据中心）下已发送的报警短信条数

Key设计为："smsQuota:sent:"+客户ID+":"+项目ID

Value设计为非负整数，表示已发送短信条数

如：smsQuota:sent:402890b75358dc190153596761ea001f:

a1018b74b65040219fb6967b02ea177d{

0

}

# 对象存储

## 使用场景

①、根据客户Id得到所有属于他的bucket；统计当天bucket容量信息和客户存储容量，时间条件

②、统计当天bucket容量信息和客户存储容量，时间条件

③、统计数据获取所有开通OBS的客户id

④、OBS服务下载流量和请求次数的扣费计算

⑤、ECMC统计上周所有bucket的排序情况

## Key-Value设计和举例

①、每个客户的每个bucket的创建时间（精确到小时）

Key设计："bucket\_start:"+客户id+":"+bucket名称

Value设计：bucket的创建时间的字符串，"yyyy-MM-ddHH:mm:ss"（精确到小时）

如：bucket\_start:402890905635d078015635e10e890004:cxy222{

2016-07-11 13:00:00

}

②、每个客户的每个bucket的最后存在时间，即最后一次统计到数据的时间（精确到小时）

Key设计："bucket\_end:"+客户id+":"+bucket名称

Value设计：bucket的删除时间的字符串，"yyyy-MM-ddHH:mm:ss"（精确到小时）

如：bucket\_end:402890905635d078015635e10e890004:cxy222{

2016-09-06 10:00:00

}

③、每个客户开通OBS服务相关信息

Key设计："obsuser:cusid:"+客户id

Value设计：对象存储用户实体类BaseObsUser的json字符串

如：obsuser:cusid:40289092523f1cd301523faab5910068{

"cusId": "40289092523f1cd301523faab5910068",

"displayName": "999",

"maxBuckets": 0,

"userId": "40289092523f1cd301523faab5910068"

}

④、记录每个客户计费周期开始（即当月1号零点）到本次计费截止时间的下载流量总量，设置超时时间为1h多一点。

Key设计："charge:obs:download:"+客户id

Value设计：下载流量的值

⑤、记录每个客户计费周期开始（即当月1号零点）到本次计费截止时间的请求数总量，设置超时时间为1h多一点。

Key设计："charge:obs:ops:"

Value设计：请求数的值

⑥、上周系统内所有bucket按照统计下载流量数据排序集合信息（正序）

Key设计："obs:obsSortByDownload"

Value设计：请求次数、bucket名称、客户id、下载流量数据的json形式字符串的集合

如：obs:obsSortByDownload

{"countRequest": 6,"name": "haitaotest",

"owner": "402890905635d078015635e10e890004","download": 1820}

{"countRequest": 6,"name": "333",

"owner": "4028909551a8e5d30151a8e851e30009","download": 2742}

⑦、上周系统内所有bucket按照统计请求数数据排序集合信息（正序）

Key设计："obs:obsSortByCountRequest"

Value设计：下请求次数、bucket名称、客户id、下载流量数据的json形式字符串的集合

如：obs:obsSortByCountRequest

{"countRequest": 6,"name": "222",

"owner": "4028909551a8e5d30151a8e851e30009","download": 5052}

{"countRequest": 6,"name": "333",

"owner": "4028909551a8e5d30151a8e851e30009","download": 2742}

⑧、上周系统内所有bucket按照统计存储容量数据排序集合信息（正序）

Key设计："obs:obsSortByStorageUsed"

Value设计：存储容量、bucket名称、客户id数据的json形式字符串的集合

如：obs:obsSortByStorageUsed

{"name": "a22","owner": "40289092523f1cd301523faab5910068","storageUsed": 0}

{"name": "xinbanceshi","owner": "cusid1","storageUsed": 0}

## 其他说明

OBS服务下载流量和请求次数的扣费时，需要两个数据，即从月初一号零点到本次计费开始时间的统计数据，和从月初一号零点到本次计费截止时间的统计数据，正常运行情况下本次计费的截止时间即为下次计费的开始时间，将从月初一号零点到本次计费截止时间的统计数据存入到redis中，作为下次计费的开始时间数据。

# 枚举字典

## 使用场景

用于存储ECMC和ECSC所需要用到的一些系统参数和设置等。

底层返回英文信息需转译成中文

## Key-Value设计和举例

①、记录所有需要的系统参数信息（树形结构）

Key设计："sys\_data\_tree:"+系统nodeId

Value设计：数据字典实体类BaseSysDataTree的json字符串，记录数据字典属性信息。

如：sys\_data\_tree:0007001001{

"flag": "1",

"isRoot": "0",

"nodeId": "0007001001",

"nodeName": "工单级别",

"parentId": "0007001",

"sort": 1

}

②、记录数据字典相关中英文转义信息

Key设计："sys\_data\_tree:statues:"+上级nodeId（如无父类，则为本身的nodeId）+":"+英文名称

Value设计：数据字典中文名称

如：sys\_data\_tree:statues:0007002004REVERT\_RESIZE

取消调整中

③、记录每个数据字典的所有下级字典的nodeId信息

Key设计："sys\_data\_tree:parent:node\_id:"+数据字典nodeId

Value设计：所有下级数据字典nodeId的集合

如：sys\_data\_tree:parent:node\_id:0007002001003

0007002001003003

0007002001003002

0007002001003004

0007002001003001

# 密钥

## 使用场景

记录、管理用户的对象存储服务生成的accessKey和secretKey

## Key-Value设计和举例

①、每个客户下所拥有的所有accessKey

Key设计："ak:cusid:"+客户id

Value设计：客户下的accessKey的集合（最多4个）

如：ak:cusid:4028909551a8e5d30151a8e851e30009

402890a8539bd9f001539bde002b000c

4028909954e5ba110154e7111f5b0006

402890a8566f993c01566fb73dfe0004

402890a8554325c5015543282a8e0003

②、每个accessKey的属性信息

Key设计："ak:akid:"+accessKey的值

Value设计：BaseAccessKey的json字符串

如：ak:akid:402890a85249121101524988292f0011{

"accessKey": "TGS0042679923O512EIQ",

"acckIsShow": "0",

"acckState": "0",

"akId": "4028909056353367015635387f3f0003",

"createDate": 1469771710000,

"isDefault": "1",

"isStopService": true,

"secretKey": "1r5CWi40il17tmF5ELXN9HcdDs14rF96u5p2VAa7",

"userId": "402890a856116c250156117050d50001"

}

③、每个accessKey的属性信息,以accesskey作为key

如: ak:accesskey:WXF99IT8A1L7GL2F1488

Value: {

"accessKey": "WXF99IT8A1L7GL2F1488",

"acckIsShow": "1",

"acckState": "0",

"akId": "402890a8584be56501584c36000a0007",

"createDate": 1478747357182,

"isDefault": "1",

"isStopService": true,

"secretKey": "nv799745p9H1zF0yx9486BA2DZ655JcMNB3Tgmv3",

"userId": "402890a8549384fc015493fbecbc000d"

}

# 云资源同步

## 使用场景

用于同步底层的云资源状态信息，同步底层状态最多100次

## Key-Value设计和举例

①、云主机的状态信息消息队列

Key设计："CLOUD\_RESOURCE:SYNCVM"

Value设计：云主机状态信息的json字符串集合

②、云硬盘状态信息消息队列

Key设计："CLOUD\_RESOURCE:SYNCVOL"

Value设计：云硬盘状态信息的json字符串集合

③、云硬盘快照状态信息消息队列

Key设计："CLOUD\_RESOURCE:SYNCVOLSPH"

Value设计：云硬盘快照状态信息的json字符串集合

④、自定义镜像状态信息消息队列

Key设计："CLOUD\_RESOURCE:SYNCIMAGE"

Value设计：自定义镜像状态信息的json字符串集合

⑤、防火墙状态信息消息队列

Key设计："CLOUD\_RESOURCE:SYNCFIREWALL"

Value设计：防火墙状态信息的json字符串集合

⑥、负载均衡状态信息消息队列

Key设计："CLOUD\_RESOURCE:SYNCLDPOOL"

Value设计：负载均衡状态信息的json字符串集合

⑦、负载均衡成员状态信息的消息队列

Key设计："CLOUD\_RESOURCE:SYNCLDMEMBER"

Value设计：负载均衡成员状态信息的json字符串集合

⑧、VIP的状态信息消息队列

Key设计："CLOUD\_RESOURCE:SYNCLDVIP"

Value设计：VIP状态信息的json字符串集合

⑨、创建云主机绑定云硬盘消息队列（**已不用**）

Key设计："CLOUD\_RESOURCE:SYNCVMATTVOL"

⑩、VPN的状态信息消息队列

Key设计："CLOUD\_RESOURCE:SYNCVPN"

Value设计：VPN状态信息的json字符串集合

如：CLOUD\_RESOURCE:SYNCVPN

{"dcId": "1604271035270","prjId": "b2f039d97aa8434b82b14c3e25f17aa6",

"count": "0","status": "PENDING\_CREATE","vpnId": "2fb23fd2-7f2b-4ebd-88ca-0bebfed23772"}

{"dcId": "1604271035270","prjId": "b2f039d97aa8434b82b14c3e25f17aa6",

"count": "0","status": "PENDING\_CREATE","vpnId": "0e8f2c5f-fd85-4e64-a068-049699132ebd"}

⑪、云数据库的状态信息消息队列

Key设计："CLOUD\_RESOURCE:SYNCRDS"

Value设计：云数据库实例状态信息的json字符串集合

# 指标监控

## 使用场景

用于监控主机各项指标信息，提供本次指标信息与上一次指标信息

## Key-Value设计和举例

①、主机CPU使用率最新一次监控采集数据

Key设计："monitor:cpu\_util:"+主机id

Value设计：CPU使用率指标数据（Double类型）

如：monitor:cpu\_util:006b5ddb-7d72-46cd-82fa-25ae48c0b41b

7.4858569051580695

②、主机CPU使用率上一次监控采集数据

Key设计："monitor:cpu\_util:last"+主机id

Value设计：CPU使用率指标数据（Double类型）

如：monitor:cpu\_util:last2af1c27a-37bc-441b-886e-53495d2bfc64

4.5

③、主机内存使用率最新一次监控采集数据

Key设计："monitor:memory.usage:"+主机id

Value设计：内存使用率指标数据（Double类型）

如：monitor:memory.usage:00812594-d676-4769-9731-99d9d72a0424

16.38965606689453

④、主机内存使用上一次监控采集数据

Key设计："monitor:memory.usage:last"+主机id

Value设计：内存使用率指标数据（Double类型）

如：monitor:memory.usage:last8556d9c2-4bc2-48fd-9286-5353123bb6ac

17.53997802734375

⑤、主机磁盘读速度最新一次监控采集数据

Key设计："monitor:disk.read.bytes.rate:"+主机id

Value设计：磁盘读速度指标数据（ Double类型）

如：monitor:disk.read.bytes.rate:00812594-d676-4769-9731-99d9d72a0424

0

⑥、主机磁盘读速度上一次监控采集数据

Key设计："monitor:disk.read.bytes.rate:last"+主机id

Value设计：磁盘读速度指标数据（ Double类型）

如：monitor:disk.read.bytes.rate:last0daa70cf-f845-44e0-b773-f9bbae8df409

0

⑦、主机磁盘写速度最新一次监控采集数据

Key设计："monitor:disk.write.bytes.rate:"+主机id

Value设计：磁盘写速度指标数据（ Double类型）

如：monitor:disk.write.bytes.rate:006b5ddb-7d72-46cd-82fa-25ae48c0b41b

0

⑧、主机磁盘写速度上一次监控采集数数据

Key设计："monitor:disk.write.bytes.rate:last"+主机id

Value设计：磁盘写速度指标数据（ Double类型）

如：monitor:disk.write.bytes.rate:last3ece9ea6-1e1a-467c-ae84-c13240cd3123

0.000008138020833333333

⑨、主机网卡上行速率最新一次监控采集数据

Key设计："monitor:network.incoming.bytes.rate:"+主机id

Value设计：网卡上行速率指标数据（Double类型）

如：monitor:network.incoming.bytes.rate:006b5ddb-7d72-46cd-82fa-25ae48c0b41b

0

⑩、主机网卡上行速率上一次监控采集数据

Key设计："monitor:network.incoming.bytes.rate:last"+主机id

Value设计：网卡上行速率指标数据（Double类型）

如：monitor:network.incoming.bytes.rate:last5c36b4f5-3200-407c-b7fd-45b9d9e92c41

0.0000957489013671875

11、主机网卡下行速率最新一次监控采集数据

Key设计："monitor:network.outgoing.bytes.rate:"+主机id

Value设计：网卡下行速率指标数据（Double类型）

如：monitor:network.outgoing.bytes.rate:006b5ddb-7d72-46cd-82fa-25ae48c0b41b

0

12、主机网卡下行速率上一次监控采集数据

Key设计："monitor:network.outgoing.bytes.rate:last"+主机id

Value设计：网卡下行速率指标数据（Double类型）

如：monitor:network.outgoing.bytes.rate:lastc2459757-fc17-4963-8fa1-cd713fe605f3

0.0003083546956380208

13、数据库实例磁盘使用率最新一次监控采集数据

Key设计：" **monitor:volume\_used:**"+实例的虚拟机id

Value设计：网卡下行速率指标数据（Double类型）

如：**monitor:volume\_used:**006b5ddb-7d72-46cd-82fa-25ae48c0b41b

0

14、数据库实例磁盘使用率上一次监控采集数据

Key设计：" **monitor:volume\_used:last:**"+实例的虚拟机id

Value设计：网卡下行速率指标数据（Double类型）

如：**monitor:volume\_used:last:**c2459757-fc17-4963-8fa1-cd713fe605f3

0.0003083546956380208

15、负载均衡最新一次监控采集数据

Key设计：" **monitor:exp:ldpool:**"+负载均衡id

Value设计：网卡下行速率指标数据（Double类型）

如：**monitor:exp:ldpool:**3a83d0a6-ba61-460f-aac7-c3542a1139fd

{

"mode" : "", //负载均衡模式（0：普通模式；1：主备模式）

"member" : "", //成员数量

"master\_member" : "", //主节点个数

"slave\_member " : "", //从节点个数

"exp\_member " : "", //不活跃成员个数

"exp\_master" : "", //不活跃主节点个数

"exp\_salve" : "", //不活跃从节点个数

"exp\_member\_ratio" : "", //不活跃成员百分比

"exp\_master\_ratio" : "", //不活跃主节点百分比

"exp\_salve\_ratio" : "", //不活跃从节点百分比

}

16、负载均衡上一次监控采集数据

Key设计：" **monitor:exp:ldpool:last:**"+ 负载均衡id

Value设计：网卡下行速率指标数据（Double类型）

如：**monitor:exp:ldpool:last:**3a83d0a6-ba61-460f-aac7-c3542a1139fd

{

"mode" : "", //负载均衡模式（0：普通模式；1：主备模式）

"member" : "", //成员数量

"master\_member" : "", //主节点个数

"slave\_member " : "", //从节点个数

"exp\_member " : "", //不活跃成员个数

"exp\_master" : "", //不活跃主节点个数

"exp\_salve" : "", //不活跃从节点个数

"exp\_member\_ratio" : "", //不活跃成员百分比

"exp\_master\_ratio" : "", //不活跃主节点百分比

"exp\_salve\_ratio" : "", //不活跃从节点百分比

}

# 监控报警

## 使用场景

记录监控报警项信息

## Key-Value设计和举例

①、记录ECSC系统的报警对象信息

Key设计："alarmObject:"+ECSC报警对象记录id

Value设计：ECSC报警对象BaseAlarmObject的json字符串

如：alarmObject:4028b881540fe6e101540fea17e40002{

"alarmRuleId": "4028b881540d880001540da824530010",

"monitorType": "008001001" //监控项类型（网络1.3新增）

"id": "4028b881540fe6e101540fea17e40002",

"vmId": "90363984-c548-479b-9c64-e1102de03a5f"

}

②、记录ECMC系统的报警对象信息

Key设计："ecmcAlarmObject:"+ECMC报警对象记录id

Value设计：ECMC报警对象BaseEcmcAlarmObject的json字符串

如：ecmcAlarmObject:4028909954096e0f015409dd1902000f{

"alarmRuleId": "4028909954096e0f0154098e21d80002",

"aoObjectId": "6f9eac58-5119-4c93-a856-a96ea28ad784",

"aoType": "0010003001",

"cusId": "1509101341180",

"dcId": "096ef11347754b31a926518cdf103e2b",

"id": "4028909954096e0f015409dd1902000f",

"objName": "6f9eac58-5119-4c93-a856-a96ea28ad784",

"prjId": "aaa"

}

③、记录ECSC系统的报警触发条件

Key设计："alarmTrigger:"+ECSC系统触发条件记录id

Value设计：ECSC系统触发条件的json字符串

如：alarmTrigger:40288ee955dd30250155ddd9ecb60011{

"alarmRuleId": "40288ee955dd30250155ddd9ec170010",

"id": "40288ee955dd30250155ddd9ecb60011",

"isTriggered": "0",

"lastTime": 60,

"operator": "=",

"threshold": 80,

"unit": "%",

"zb": "0008001002"

}

④、记录ECMC系统的报警触发条件

Key设计："ecmcAlarmTrigger:"+ECMC系统触发条件记录id

Value设计：ECMC系统触发条件的json字符串

如：ecmcAlarmTrigger:402890a8545fcb51015460a3d642000e{

"alarmRuleId": "402890a8545fcb51015460a3d641000d",

"id": "402890a8545fcb51015460a3d642000e",

"isTriggered": "0",

"lastTime": 60,

"operator": ">",

"threshold": 1,

"unit": "MB/s",

"zb": "0010003001004"

}

# 消息

## 使用场景

## Key-Value设计和举例

①、每条消息的收藏次数

Key设计："message:collect:count:MSG\_"+消息id

Value设计：消息收藏次数

如：message:collect:count:MSG\_402890ab54b9e28b0154b9ee712a003b

2

②、每条消息的已读次数

Key设计："message:statu:count:MSG\_"+消息id

Value设计：消息已读次数

如：message:statu:count:MSG\_40288eeb54b87ec40154b89560150001

3

③、每条消息的取消收藏次数

Key设计："message:uncollect:count:MSG\_"+消息id

Value设计：消息已读次数

如：message:uncollect:count:MSG\_40288eeb54949238015494a5df320003

3

④、消息读取次数计数器使用

Key设计："message:statu:MSG\_STATU\_QUEUE"

Value设计：消息id

⑤、消息取消收藏计数器使用

Key设计："message:uncollect:MSG\_UNCOLLECT\_QUEUE"

Value设计：消息id

⑥、消息收藏次数计数器使用

Key设计："message:collect:MSG\_COLLECT\_QUEUE"

Value设计：消息id

## 说明

"message:collect:count:MSG\_"是一条消息的收藏次数，而不是当前的收藏数

"message:uncollect:count:MSG\_"是一条消息的取消收藏次数

如一条数据若被三个客户都收藏了，然后有两个客户取消了收藏，则两个数据分别为3和2

④、⑤、⑥具体作用暂不清楚

# 忘记密码

## 使用场景

用户找回密码

## Key-Value设计和举例

①、用户经过找回密码第一步时记录的消息队列

Key设计："forgotcode:codeCheck:"+账户名

Value设计："code\_pass"

如：forgotcode:codeCheck:zhouhaitaoadmin

code\_pass

②、用户经过找回密码第二步时记录的消息队列

Key设计："forgotcode:phoneCheck:"+账户名

Value设计："phone\_pass"

如：forgotcode:phoneCheck:zhouhaitaoadmin

phone \_pass

## 其他说明

此方法目前还存在问题，不能完全阻止用户跳过前两步

# 标签

## 使用场景

用于记录云资源标记标签信息

## Key-Value设计和举例

①、记录一个客户所建立的标签类别的id信息

Key设计为："cus:tgrp:"+客户id

Value设计为：标签类别id的集合

如：cus:tgrp:4028909551a8e5d30151a8e851e30009{

4028909956de493e0156df1df13a0000

4028b881539de1a101539de380170001

}

②、记录每个标签类别详细信息

Key设计："taggroup:"+标签类别id

Value设计：标签类别BaseTagGroup的json字符串

如：taggroup:40288eed559595510155959982460005{

"abbreviation": "test002",

"createDate": 1467093713000,

"creatorId": "40288ee4549415970154942a55d300f1",

"cusId": "40288ee4549415970154942a4da60003",

"description": "002",

"enabled": "1",

"id": "40288eed559595510155959982460005",

"name": "test002",

"resType": "securityGroup",

"unique": "0"

}

③、记录一个标签类别下属的标签的id信息

Key设计："tgrp:tg:"+标签类别id

Value设计：标签id的集合

如：tgrp:tg:402890e75629f03401562c18d8e40025{

4028909956de493e0156df1ebea10001

4028909156e3ca8f0156e3db2fa00000

}

④、记录每个标签的详细信息

Key设计："tag:"+标签id

Value设计：标签model BaseTag的json字符串

如：tag:40288eed559595510155959ba1270008{

"createDate": 1467093852314,

"creatorId": "40288ee4549415970154942a55d300f1",

"description": "0002",

"groupId": "40288eed559595510155959982460005",

"id": "40288eed559595510155959ba1270008",

"name": "0002"

}

⑤、每个标签标记资源及资源类别的集合信息

Key设计："tg:res:"+标签id

Value设计：资源类型+":"+资源id字符串的集合

如：tg:res:40288eed559595510155959dc8d8000d{

network:71ca926b-6ac0-45c9-80fd-598f09683dcd

firewall:06396ede-14af-4c9a-b781-d3fa2a939652

}

⑥、一个资源所标记的标签的id集合信息

Key设计："res:tg:"+资源类型+":"+资源id

Value设计：标签id的集合

如：res:tg:firewall:06396ede-14af-4c9a-b781-d3fa2a939652{

40288eed559595510155959ba1270008

40288eed559595510155959dc8d8000d

}

# 客户冻结

## 使用场景

冻结客户和登录验证，状态为true表示冻结，false表示未冻结

## Key-Value设计和举例

记录曾经冻结过的客户状态

Key设计："cus:block:"+客户id

Value设计："true"或"false"

如：cus:block:402890a856116c250156117050d50001

False

# 发送短信邮件提醒

## 使用场景

保证按需付费超过保留时长后,只向客户发送一次短信邮件提醒.

保证按需付费首次达到信用额度时,只向客户发送一次短信邮件提醒.

## Key-Value设计和举例

在超过保留时长处理中,获取key为outRentionTime:xxxx的值,若为null,则说明之前没有发送过邮件短信提醒,则本次可以向客户发送邮件短信提醒,若不为null,则说明之前已经发送过邮件短信提醒,则无需重复发送.key在成功恢复了所有后付费资源后清空.

Key:

outRentionTime: 402890a8549384fc015493fbecbc000d

value:

1

在首次达到信用额度处理中,获取key为reachCreditLimit:xxxx的值,若为null,则说明之前没有发送过邮件短信提醒,则本次可以向客户发送邮件短信提醒,若不为null,则说明之前已经发送过邮件短信提醒,则无需重复发送.key在成功恢复了所有后付费资源后清空.

Key:

reachCreditLimit: 402890a8549384fc015493fbecbc000d

value:

1

# 管理系统计划任务漏跑信息

## 使用场景

负责管理系统所布置好的一系列计划任务的运行状态，提取出符合条件的漏跑任务信息并且提醒相关负责人。

## Key-Value设计和举例

key :schedule\_task\_lost:checked value : 1

此key的作用在于控制集群环境布置监测的计划任务时，同一个运行时刻只允许有一台机器在进行程序的逻辑处理（increase该值，若值为1，表明可以执行），给此key设置一定的超时时间，保证该计划任务在下一个运行时刻可以正常执行。

key : schedule\_task\_lost:store\_schedule\_lost\_information(计划任务名称)

value : 数组类型

示例 : [

"sysLogJob"

]

此key的作用为存储每一次执行计划任务漏跑监测任务的时候将检测到的信息以数组的形式存放在缓存中；然后每次将本次监测到的信息与存放在缓存中的信息作差异判断，二者之间若有差异，则应该发送邮件和短信提醒管理员；否则不需要执行提醒操作。

# OBS计费相关标识Key

## 使用场景

几个标识OBS相关数据统计成功与否的Key，用于OBS计费时判断是否需要触发执行计费。

## Key-Value设计和举例

Key:charge:obs:storage:[cusId]:[currentChargeTime] Value:统计完成则为1

该Key的目的在于标识客户cusId在当前计费时间（整点）时是否统计了OBS的下载流量，统计完成则置value为1，过期时间为1800s。

Key:charge:obs:used:[cusId]:[currentChargeTime] Value:统计完成则为1

该Key的目的在于标识客户cusId在当前计费时间（整点）时是否统计了OBS的使用量（含下载流量和请求数），统计完成则置为1，过期时间为1800s。

Key:charge:obs:backsource:[cusId]:[currentChargeTime] Value:统计完成则为1

该key的目的在于标识客户cusId在当前计费时间（整点）时是否统计了OBS开通CDN导致的CDN回源流量，统计完成则置为1，过期时间为1800s。

Key:charge:obs:chargedone:[cusId]:[currentChargeTime] Value:云资源计费完成则为1

该Key的目的在于标识客户cudId在当前计费时间（整点）时完成了后付费云资源的计费工作——如果没有后付费云资源，消息还是会发送。

Key:charge:obs:gatherdone:[cusId]:[currentChargeTime] Value:CDN用量统计完成则为1

该Key的目的在于标识客户cudId在当前计费时间（整点）时完成了CDN用量的统计，含下载流量、动态请求数和HTTPS请求数.

上述5个Key配合使用，当5个计划任务都发出了结束的消息，obs计费消息监听监听到全部的消息，才会进入实际的本小时对客户[cusId]的OBS计费。

# Api访问次数相关key

## 使用场景

Api鉴权中所需的当前访问次数与设置的api’限制次数相关的key

## Key-Value设计和举例

1. 系统默认访问限制次数key

Key设计："apiRequestCountDefault:"+版本+action名称

Value设计：BaseApiDefaultCount的json串

如：apiRequestCountDefault:v1:createInstance

{

id:” 402890a8549384fc015493fbecbc000d”,

version:”v1”,

apiType:”Instance”,

action:” CreateInstance”,

count:”9000”,

apiTypeName:”云主机”,

actionName:”创建云主机”

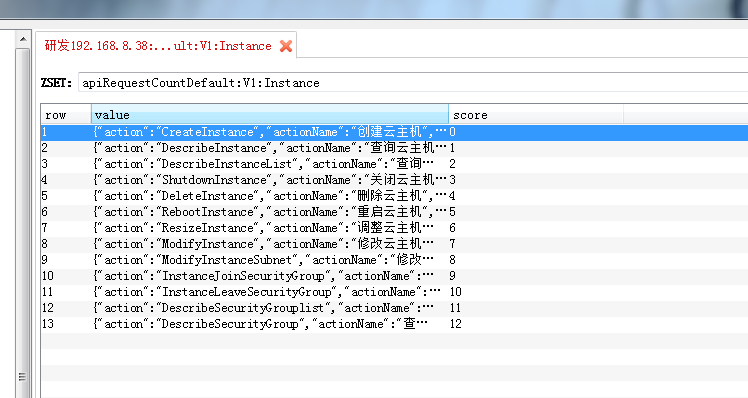
}

1. 系统默认访问限制次数(类型)key,有序,用于按顺序展示该类型下的actions

Key设计："apiRequestCountDefault:"+版本+资源类型

Value设计：BaseApiDefaultCount的json串

如：apiRequestCountDefault:v1: Instance



③、api请求次数限制key

Key设计：apiRequestCount:cusId:version:action

Value设计：BaseApiCountRestrict的json字符串

如：apiRequestCount:402890a8549384fc015493fbecbc000d:v1:createInstance

{

id:” 402890a8549384fc015493fbecbc000d”,

cusId:” 402890a8549384fc015493fbecbc000d”,

version:”v1”,

action:” CreateInstance”,

count:”9000”,

apiType:”Instance”,

apiTypeName:”云主机”,

actionName:”创建云主机”

}

④、客户当前小时访问api次数

Key设计：requestCount:cusId:version:action:formatTime

Value设计：从0自增

如: requestCount: 402890a8549384fc015493fbecbc000d:v1:createInstance: 2016111616

Value:100

# 按分钟统计API访问指标相关Key

## 22.1使用场景

该套Redis的设计目的在于为API的资源监控与数据管理提供最开始的数据来源，后续的数据处理以及程序逻辑都依赖于此。该套Key分别对应API访问过程中四个指标的数据统计信息。

## 22.2Key-Value设计和举例

1. 统计一分钟内API访问服务的正确次数累计值

Key设计：api:monitoringalarm:service:log:base:rightNumber:timestamp

Value设计：以Redis HashMap为数据结构，对应的维度值为键值，对应的正确次数为具体指标值的数据集合。

使用示例：

api:monitoringalarm:service:log:base:rightNumber:timestamp

{

cus1:ip1:dc1 = 1 ,

cus1:ip1:dc2 = 4 ,

...

}

1. 统计一分钟内API访问服务的访问总次数累计值

Key设计：api:monitoringalarm:service:log:base:allNumber:timestamp

Value设计：以Redis HashMap为数据结构，对应的维度值为键值，对应的访问总次数为具体指标值的数据集合。

使用示例：

api:monitoringalarm:service:log:base:allNumber:timestamp

{

cus1:ip1:dc1 = 10 ,

cus1:ip1:dc2 = 10 ,

...

}

1. 统计一分钟内API访问服务的可用访问次数累计值

Key设计：api:monitoringalarm:service:log:base:availableNumber:timestamp

Value设计：以Redis HashMap为数据结构，对应的维度值为键值，对应的可用访问次数为具体指标值的数据集合。

使用示例：

api:monitoringalarm:service:log:base:availableNumber:timestamp

{

cus1:ip1:dc1 = 5 ,

cus1:ip1:dc2 = 5 ,

...

}

1. 统计一分钟内API访问服务的总访问时间累计值

Key设计：api:monitoringalarm:service:log:base:dateNumber:timestamp

Value设计：以Redis HashMap为数据结构，对应的维度值为键值，对应的总访问时间为具体指标值的数据集合。

使用示例：

api:monitoringalarm:service:log:base:dateNumber:timestamp

{

cus1:ip1:dc1 = 852,

cus1:ip1:dc2 = 1024 ,

...

}

# 存放上一个时刻对应API访问指标信息相关Key

## 23.1使用场景

该套Redis的设计目的：在API的实时监控中，每一次计算当前时刻所对应的数据指标的时候，需要同时计算得到相对于上次时刻指标的变化范围，该套Redis存储上一时刻的指标数据，本次数据读出其数据，比较两个值的大小得出相对变化趋势。

## 23.2Key-Value设计和举例

1. 存放上一时刻数据指标值所对应的具体时间

Key设计：api:monitoringalarm:prev:time

Value设计：字符串，存储时间值对应的毫秒数。

使用示例：

api:monitoringalarm:prev:time = 123456789

1. 存放上一时刻数据具体的四个指标值

Key设计：api:monitoringalarm:prev

Value设计：HashMap数据结构（以具体的维度：客户Id+IP+数据中心API别名为键值，以可用率、正确率、总次数以及平均处理时间四个指标值组成的JSON字符串作为对应的键值）。

使用示例：

api:monitoringalarm:prev

{

cus1:ip1:dc1 = {

"allNumberValue":1,

"availableValue":100,

"dateValue":116,

"rightValue":0

},

...

}

# 存放报警条件分析数据相关Key

## 24.1使用场景

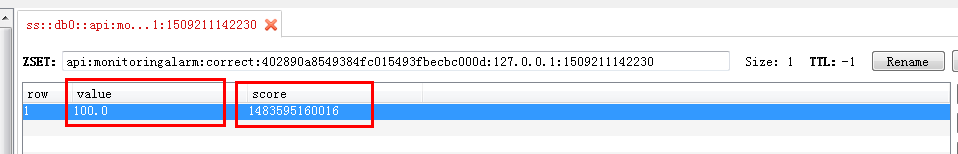
此套Redis的数据是依据实时监测得到的API访问指标数据所得，目的是为了方便的统计当前时刻是否触发满足系统配置的各个报警规则条件，从而以便于发送API报警提醒。该套Redis数据根据指标的种类一共分为四组，每一组对应的值实质上都为不同时刻对应的数据指标值。

## 24.2Key-Value设计和举例

1. 存放连续时间段内数据指标正确率数值

Key设计：api:monitoringalarm:correct:cus1:ip1:dc1

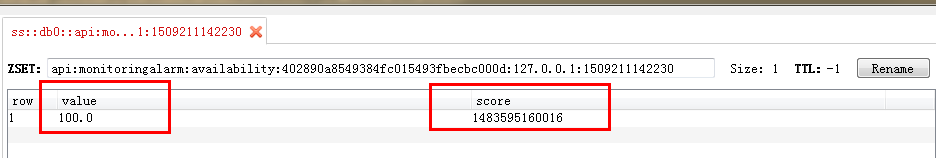
Value设计：本质上为Redis中ZSet数据结构，对应的值为不同时刻的计算指标值，该值被设置了得分，即当前时刻毫秒数。主要目的在于可以对其使用范围查询方式查找指定数据。



1. 存放连续时间段内数据指标可用率数值

Key设计：api:monitoringalarm:availability:cus1:ip1:dc1

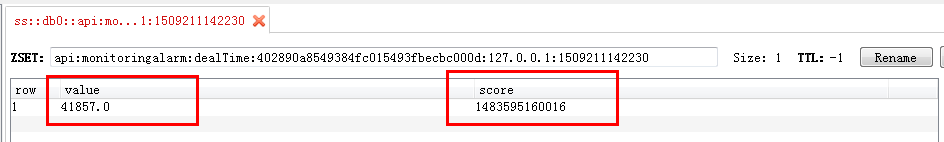
Value设计：本质上为Redis中ZSet数据结构，对应的值为不同时刻的计算指标值，该值被设置了得分，即当前时刻毫秒数。主要目的在于可以对其使用范围查询方式查找指定数据。



1. 存放连续时间段内数据指标平均处理时间值

Key设计：api:monitoringalarm:dealTime:cus1:ip1:dc1

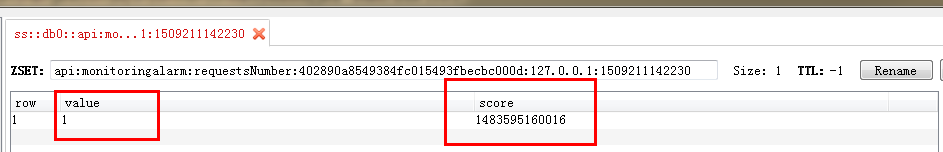
Value设计：本质上为Redis中ZSet数据结构，对应的值为不同时刻的计算指标值，该值被设置了得分，即当前时刻毫秒数。主要目的在于可以对其使用范围查询方式查找指定数据。



1. 存放连续时间段内数据指标总API请求次数值

ey设计：api:monitoringalarm:requestsNumber:cus1:ip1:dc1

Value设计：本质上为Redis中ZSet数据结构，对应的值为不同时刻的计算指标值，该值被设置了得分，即当前时刻毫秒数。主要目的在于可以对其使用范围查询方式查找指定数据。



# 是否触发报警提醒以及产生报警信息相关Key

## 25.1使用场景

报警提醒所依赖的规则如下，在连续的时间段之内连续触发报警的时候，只发送一次提醒，只在第一次触发的时候就发送相关信息。简而言之就是只在每一段连续时间的开头发送相关的提醒。该套Redis的作用便是解决此问题。

## 25.2Key-Value设计和举例

存储上一个时刻以（客户+IP+数据中心+触发条件+时间戳字符串）为键值，简单字符串为对应的Value的数据结构保存上一个时刻是否已经发送过报警信息提醒，若已经发送，则再Redis中存在该类型数据，反之则无。

Key设计：api:monitoringalarm:isNeedNotify:cus1:ip1:dc1:trigger1:timestamp

Value设计：简单字符串，例如“warning”

使用示例：api:monitoringalarm:isNeedNotify:cus1:ip1:dc1:trigger1:timestamp = “warning”

# 26.客户冻结黑名单同步

## 26.1使用场景

ECMC中，用来同步黑名单客户与黑名单IP数据

## 26.2Key-Value设计和举例

记录黑名单的客户信息

Key设计：" api:black:black\_cus:"+客户id

Value设计：

{

"apiId": "4028909058ce0c030158ce86dca301eb",

"apiType": "blackCus",

"apiValue": "40288ee554e6844a0154e68820b10003",

"createTime": 1480933694000,

"memo": "黑名单客户"

}

如：api:black:black\_cus:40288ee554e6844a0154e68820b10003

{

"apiId": "4028909058ce0c030158ce86dca301eb",

"apiType": "blackCus",

"apiValue": "40288ee554e6844a0154e68820b10003",

"createTime": 1480933694000,

"memo": "黑名单客户"

}

记录黑名单的IP信息

Key设计：" api:black:black\_ip: "+黑名单IP

Value设计：

{

"apiId": "402890905972757601597277e6cb0000",

"apiType": "blackIp",

"apiValue": "192.168.16.16",

"createTime": 1483684177604,

"memo": "黑名单IP"

}

如：api:black:black\_ip:192.168.16.16

{

"apiId": "402890905972757601597277e6cb0000",

"apiType": "blackIp",

"apiValue": "192.168.16.16",

"createTime": 1483684177604,

"memo": "黑名单IP"

}

# API开关状态

## 25.1使用场景

调用API接口时，首先需要查询当前数据中心的API开关状态。

## 25.2Key-Value设计和举例

Key设计：api:switch:status:[dcId]

Value设计：0或1（0代表关闭，1代表开启）

使用示例：api:switch:status:1604271035270

{

1

}

# 开启/关闭API开关所用验证码

## 25.1使用场景

操作API开关时，需要校验手机验证码，验证码保存在redis中，时效5分钟，试用后删除

## 25.2Key-Value设计和举例

Key设计：api:switch:code:[dcId]:[phone]

Value设计：验证码字符串

使用示例：api:switch:code:1604271035270:15266666666

{

695325

}

# 修改API权限手机所用验证码

## 25.1使用场景

绑定或修改具有API开关权限的手机号码时，需要给旧手机和新手机发送验证码，校验通过后方可生效。

## 25.2Key-Value设计和举例

Key设计：api:switch:phone:[new/old]:[phone]

(new表示此为给新手机号码发送的验证码，old表示此为给旧手机号码发送的验证码)

Value设计：验证码字符串

使用示例：api:switch:phone:new:15266666666

{

695325

}

api:switch:phone:old:15288888888

{

965842

}

# 以数据中心的Region标识查询数据中心的id信息

## 25.1使用场景

以数据中心的Region标识查询数据中心的id信息。

## 25.2Key-Value设计和举例

Key设计：api:dc:code:[ Region标识]

Value设计：value为一个json

使用示例：api:dc:code:wuli11-777

{

"dcId": "1604271035270",

"dcName": "物理11\_3"

}

# 以数据中心的id查询数据中心的Region

## 25.1使用场景

以数据中心的id查询数据中心的Region信息。

## 25.2Key-Value设计和举例

Key设计："api:dc:dcId:[dcId]

Value设计：value为一个json

使用示例：api:dc:dcId:1509211142230

{

"dcName": "研发32",

"apiDcCode": "yanfa-32"

}

# 负载均衡/vip资源状态同步

## 25.1使用场景

创建负载均衡时,保证负载均衡和VIP状态都为ACTIVE时,才将订单状态置为已完成.

## 25.2Key-Value设计和举例

Key设计：" cloudLdVipSync:[vipId]

Value设计：value为字符串ACTIVE

使用示例：cloudLdVipSync:5d34a4e3-9caa-439f-84e3-e126e58f5668

ACTIVE

Key设计：" cloudLdPoolSync:[poolId]

Value设计：value为json

使用示例：cloudLdPoolSync:5d34a4e3-9caa-439f-84e3-e126e58f5668

{

orderNo:123456789,

poolId: 5d34a4e3-9caa-439f-84e3-e126e58f566,

dcId:778899445566,

connectionLimit:5000,

poolStaus:ACTIVE,

cusId:3f13468f1648f9234f461fwef24

}

# 32.修改/绑定云硬盘分类限速权限手机所用验证码

## 32.1使用场景

绑定或修改具有云硬盘分类限速最高权限的手机号码时，需要给新手机发送验证码，校验通过后方可生效。

## 32.2Key-Value设计和举例

Key设计：volumetype:phone:new:[phone]

(new表示此为给新手机号码发送的验证码)

Value设计：验证码字符串

使用示例：volumetype:phone:new:15266666666

{

695325

}

# 33.获取云硬盘分类限速相关操作权限

## 33.1使用场景

操作云硬盘分类限速相关操作时，需要校验手机验证码，验证码保存在redis中，时效5分钟，试用后删除

## 33.2 Key-Value设计和举例

Key设计：volumetype:code:[phone]

Value设计：验证码字符串

使用示例：volumetype:code:15266666666

{

695325

}

# 34.成员状态(流量承担者)改变

## 34.1使用场景

在1min一次的成员状态刷新计划任务中,为了确保与人为操作成员(添加,删除,更新,解绑)不出现冲突,使用向redis中存状态的方式,加了一个乐观锁,保证在执行计划任务时,剔除了人为操作所带来的影响.

## 34.2 Key-Value设计和举例

Key设计：member:add:[memberId]

Value设计：”1”/”2”

使用示例：member:add: ce15d318-2564-4ec3-a456-49395175066c

{

“1”

}

Key设计：member:update:[memberId]

Value设计：”1”/”2”

使用示例：member: update: ce15d318-2564-4ec3-a456-49395175066c

{

“1”

}

Key设计：member:delete:[memberId]

Value设计：”1”/”2”

使用示例：member: delete: ce15d318-2564-4ec3-a456-49395175066c

{

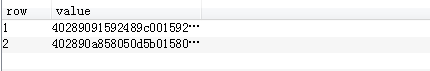
“1”

}

Key设计：member:delete:poolId:[poolId]

Value设计：”[memberId]”

使用示例：member: delete:poolId: ce15d318-2564-4ec3-a456-49395175066c



Key设计：member:sync24h:[memberId]

Value设计：”1”/”2”

使用示例：member: sync24h: ce15d318-2564-4ec3-a456-49395175066c

{

“1”

}

Key设计：memberstatus:[memberId]

Value设计：{

"dcId": "1604271035270",

"prjId": "452d8315ceca44969ebd9e0d83f4e58b",

"count": "0",

"memberId": "2062356d-7fbf-4fa1-9dd7-65c0e299ceaa",

"memberStatus": "PENDING\_UPDATE"

}

使用示例：member: sync24h: ce15d318-2564-4ec3-a456-49395175066c

{

"dcId": "1604271035270",

"prjId": "452d8315ceca44969ebd9e0d83f4e58b",

"count": "0",

"memberId": "2062356d-7fbf-4fa1-9dd7-65c0e299ceaa",

"memberStatus": "PENDING\_UPDATE"

}

Key设计：delete:pool:[poolId]

Value设计：”1”/”2”

使用示例：delete:pool: ce15d318-2564-4ec3-a456-49395175066c

{

“1”

}

# 35.RDS开通和限制服务

## 35.1 用于表示资源是否受限。0：服务处于开通状态；1：服务处于限制状态。

## 35.2 Key-Value设计和举例

Key设计：charge:rds:restricted:[rdsId]

Value设计：”0”/”1”

使用示例：charge:rds:restricted: 1549a691-fa3e-4252-86b5-b907c742d145

{

1

}

该实例（1549a691-fa3e-4252-86b5-b907c742d145）对应的服务已经受限制。