**后端存储设计**

后端存储的内容有任务详情，任务进度，用户信息，第三方绑定信息表，任务派发的金额。

任务详情和任务进度存储在leveldbserver中，用户信息以及任务派发的金额存储在mysql中。

leveldbserver是nosql，存储kv，key也有固定格式。

## 任务详情

每条任务由任务id唯一区分。

key:taskinfo\_${taskid}

value:${taskinfo}

其中由${}表示的是变量，${taskid}是64位无符号整数，由时间戳+服务id+自增id组成，${taskinfo}是个对象序列化后的值。如下是一个例子

taskinfo\_1442277780699022608 =>xxxxxxxxxxxxxxxxxxxxx

taskinfo\_1442277790294022321 =>xxxxxxxxxxxxxxxxxxxxx

由于leveldb中默认按key排序，排序按照key的字典序进行，所以${taskid}不足位数的，前面填0补齐：

taskinfo\_0922277780639022601 =>xxxxxxxxxxxxxxxxxxxxx

taskinfo\_0082277795294022367 =>xxxxxxxxxxxxxxxxxxxxx

## ~~任务进度（改存mysql）~~

~~任务进度与用户挂钩，key会包含uid。~~

~~key:taskprogress\_${uid}\_${taskid}~~

~~value:${total\_step}\_${current\_step}~~

~~如下例子说明：~~

~~taskprogress\_601100\_1442277780699022608 => 3\_1~~

~~taskprogress\_601100\_1442277790294022321 => 2\_2~~

~~uid为601100的用户，任务1442277780699022608总共需要3步完成，当前完成1步；~~

~~uid为601100的用户，任务1442277790294022321总共需要2步完成，当前完成2步；~~

~~上述两个组成了基本的任务体系，用户拉取任务列表是从任务进度的项目里面获取taskid，而后根据taskid从详情获取具体信息。~~

## 用户id列表

存储用户的设备id对应的uid。根据deviceid检测用户是否存在，不存在则分配新id插入，不做分库分表操作，uid从500000自增。Fdeviceid和Fuid都是唯一键，Fuid是自增主键，表结构：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **type** | **null** | **key** | **comments** |
| Fdeviceid | Varchar(64) | NO | YES | 设备id |
| Fuid | Int(11) | NO | YES | 用户id，自增，步长为1，unique |
| Fupdate\_time | timestamp | NO | NO | 更新时间 |

表名： db\_user.t\_reg\_list

用户登录会来此表查uid，没有就会新写入uid到此表，并写账户信息到用户表。

## 用户信息表

用户信息比较重要，且维度较多，存储选用mysql。uid为key。表结构如下：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **type** | **null** | **key** | **comments** |
| Fuid | Int(11) | NO | YES | 用户id |
| Fdeviceid | Varchar(64) | NO | YES | 设备id |
| Fver | Int(11) | YES | NO | 标识用户数据的版本号 |
| Faccount | Varchar(256) | YES | NO | 用户账户名(唯一) |
| Fmobile | Varchar(20) | YES | NO | 手机号(只能绑定一个，用于密码找回) |
| Fmail | Varchar(20) | YES | NO | 邮箱(只能绑定一个，用于密码找回) |
| Fsign | Varchar(256) | YES | NO | 个性签名 |
| Faccounttype | Int(11) | NO | NO | 当前账户绑定的账户类型 |
| Fpassword | Varchar(64) | NO | NO | 密码 |
| Fnick\_name | Varchar(256) | YES | NO | 用户昵称，本系统中的昵称 |
| Fpre\_cash | decimal(12,2) | NO | NO | 任务预计分配的的总金额 |
| Fuseable\_cash | decimal(12,2) | NO | NO | 任务可提取的金额 |
| Ffetched\_cash | decimal(12,2) | NO | NO | 任务已被提取的金额 |
| Fcreate\_time | timestamp | NO | NO | 创建时间 |
| Fupdate\_time | timestamp | NO | NO | 更新时间 |

用户表为大表，按照uid分库分表。总共16个库，16\*256张表。

分库方式：

string md5str = md5\_str(uid);

stringdbname = db\_user\_md5str[0];

分表方式：

stringtablename = t\_user\_md5str[1]md5str[2];

大表取md5字符串的两位，小表只取一位。

此表是以uid做为维度的表，一个uid可以绑定多个第三方账户，同种类型的第三方只能绑定一个。因此这张表只存与本uid联系的信息，用户的信息存在第三方绑定关系表中

## 第三方账号绑定关系表

thirdid和uid组成联合key。一个账号可以绑定多个第三方账号，一个第三方账号也可以绑定多个本账号。

注：目前一个账号只能绑定一个第三方账号！

第三方关系表为小表，按照uid分库分表。共分16个库，16\*16张表。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **type** | **null** | **key** | **comments** |
| Fuid | Int(11) | NO | YES | 用户Id |
| Fthird\_id | Varchar(256) | NO | YES | 第三方用户账号 |
| Fthird\_type | Int(11) | NO | NO | 微信，QQ，支付宝等 |
| Fnick\_name | Varchar(256) | YES | NO | 用户昵称 |
| ~~Fmobile~~ | ~~Varchar(20)~~ | ~~YES~~ | ~~NO~~ | ~~手机号(挪到用户表中)~~ |
| Fgender | tinyint(3) | YES | NO | 性别 |
| Favatar | Varchar(256) | YES | NO | 头像链接地址 |
| Finfo | Varchar(256) | YES | NO | 用户简介 |
| ~~Fsign~~ | ~~Varchar(256)~~ | ~~YES~~ | ~~NO~~ | ~~个性签名(挪到用户表中)~~ |
| Flocation | Varchar(64) | YES | NO | 地域 |
| Fcity | Varchar(64) | YES | NO | 所在城市 |
| Fstatus | Int(11) | NO | NO | 状态(未激活，已激活，已删除) |
| Fcreate\_time | timestamp | NO | NO | 创建时间 |
| Fupdate\_time | timestamp | NO | NO | 更新时间 |

分库方式：

string md5str = md5\_str(uid);

stringdbname = db\_third\_md5str[0];

分表方式：

stringtablename = t\_user\_md5str[1];

Fstatus为1表示已激活当前使用的账号

Fstatus为0表示处于未激活状态的账号

## 任务信息表

以任务为维度，包含任务信息，统计每个任务派发的金额。该表应该再添加任务的**奖励办法**以及**任务状态**字段。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **type** | **null** | **key** | **comments** |
| Ftaskid | Int(21) | NO | YES | 任务id |
| Ftask\_status | Int(11) | NO | NO | 任务当前的状态 |
| Ftask\_algorithm | Varchar(512) | NO | NO | 任务金额分配算法，json |
| Fpre\_cash | decimal(12,2) | NO | NO | 任务预计分配的的总金额 |
| Faward\_cash | decimal(12,2) | NO | NO | 任务已下发奖励的总金额 |
| Ffetched\_cash | decimal(12,2) | NO | NO | 任务已被提取的总金额 |
| Flimit\_num | Int(11) | NO | NO | 限制任务份数( Fstart\_num<= Flimit\_num) |
| Fstart\_num | Int(11) | NO | NO | 已开始任务份数(上报任务+1) |
| Fstart\_time | timestamp | NO | NO | 任务启动时间 |
| Fend\_time | timestamp | NO | NO | 任务结束时间 |
| Fcreate\_time | timestamp | NO | NO | 创建时间 |
| Fupdate\_time | timestamp | NO | NO | 更新时间 |

Ftask\_status为0表示关闭，为1表示打开

Fpre\_cash表示预计奖励的总金额；

Faward\_cash表示已经给用户下发的总金额；

Ffetched\_cash表示已经被提取的总金额（不方便做统计，可以忽略此字段）

Ftask\_algorithm示例：

{

“step”:3,

“alg”:

{

“1”:1, // 完成第一步给奖励1元

“2”:1, // 完成第二步给奖励1元

“3”:1 // 完成第三部给奖励1元

}

}

任务派发的金额表为小表，按照uid分库分表。共分16个库，16\*16张表。

分库方式：

string md5str = md5\_str(uid);

string dbname = db\_task\_md5str[0];

分表方式：

String tablename = t\_task\_info\_md5str[1];

## 用户的任务列表

Fuid和Ftaskid组成唯一key，代表用户的任务。这里存的是简要信息，详细信息存储在leveldb，一份。任务可以先写进db，只是拉取的时候，没到时间拉不到。如此减轻服务压力，写任务可以晚间进行。***但是有个问题，何时给用户推push告知新任务？***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field** | **type** | **null** | **key** | **comments** |
| Fuid | Int(11) | NO | YES | 用户Id |
| Ftaskid | Int(21) | NO | YES | 任务id |
| Fflag | Int(4) | NO | NO | 任务标志(未完成/已完成) |
| Ftask\_progress | Int(4) | NO | NO | 任务当前进度 |
| Faward\_cash | decimal(12,2) | NO | NO | 任务已下发奖励的金额 |
| Fcreate\_time | timestamp | NO | NO | 创建时间 |
| Fupdate\_time | timestamp | NO | NO | 更新时间 |

Fflag,为3表示已过期的任务，为2表示已完成的任务，为1表示进行中的任务，为0表示未开始的任务。此处的Faward\_cash表示已给此用户的奖励金额，并不是任务已经奖励的总金额。

用户任务列表为小表，按照uid分库分表。共分16个库，16\*16张表。

分库方式：

string md5str = md5\_str(uid);

stringdbname = db\_user\_md5str[0];

分表方式：

stringtablename = t\_task\_list\_md5str[1];

**~~附加一个全局id生成库，生成用户id。~~**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **~~Field~~** | **~~type~~** | **~~null~~** | **~~key~~** | **~~comments~~** |
| ~~Fid~~ | ~~Int(11)~~ | ~~NO~~ | ~~YES~~ | ~~Id~~ |
| ~~Fupdate\_time~~ | ~~Int(11)~~ | ~~NO~~ | ~~NO~~ | ~~更新时间~~ |

~~id库只有一个。16张表，但是只使用第一张表。~~

~~分库方式：~~

~~string md5str = md5\_str(devid);~~

~~stringdbname = “db\_global\_id\_0”;~~

~~分表方式：~~

~~stringtablename = “t\_id\_0”;~~