用户系统，主要分为账号体系和用户信息两大类。

账号体系包括：登录验证、注册、第三方授权、以及权限管理。

用户信息包括：用户属性、用户设备信息、用户地理位置、还有用户日志信息。

**登录验证**

在一般项目账号体系中，一般会要求支持手机、邮箱、账号、QQ、微信、微博实现登录。后面三种方式都是基于第三方授权后，完成的身份验证。手机、邮箱、账号则是相对传统的登录方式。

用户身份与登录的授权方式是独立开的，即用户uid和登录方式是一对多的关系。举例来说，用户A在使用微博授权登录后，服务端鉴别身份信息为uid=123。用户A下次使用微信登录，服务端鉴别身份同样为uid=123。不存在同一用户A拥有多个账号信息的现象。登录授权表设计如下。

|  |
| --- |
| //用户授权表  CREATE TABLE user\_auth (  id bigint(20) NOT NULL AUTO\_INCREMENT,  uid bigint(20) unsigned NOT NULL DEFAULT '0' COMMENT '用户id',  identity\_type tinyint(4) unsigned NOT NULL DEFAULT '1' COMMENT '1手机号 2邮箱 3用户名 4qq 5微信 6腾讯微博 7新浪微博',  identifier varchar(50) NOT NULL DEFAULT '' COMMENT '手机号 邮箱 用户名或第三方应用的唯一标识',  certificate varchar(20) NOT NULL DEFAULT '' COMMENT '密码凭证(站内的保存密码，站外的不保存或保存token)',  create\_time datetime COMMENT '绑定时间',  update\_time datetime COMMENT '更新绑定时间',  PRIMARY KEY (id),  UNIQUE KEY only (uid,identity\_type),  KEY idx\_uid (uid) USING BTREE  ); |

**用户信息**

用户信息，为便于扩展，分成两类。用户基础信息和用户拓展信息。基本信息用来保存用户的基本属性，年龄、性别、生日、头像、手机号码等。扩展信息，用来保存用户的设备信息或其他可扩展的内容。另外还有位置信息，这个可独立出来，也可合并到扩展信息中，根据自己的使用场景来定。

|  |
| --- |
| **//用户基础信息**  CREATE TABLE user\_base (  uid bigint(20) NOT NULL COMMENT '用户ID',  user\_role tinyint(2) unsigned NOT NULL DEFAULT '2' COMMENT '2正常用户 3禁言用户 4虚拟用户 5运营',  register\_source tinyint(4) unsigned NOT NULL DEFAULT '0' COMMENT '注册来源：1手机号 2邮箱 3用户名 4qq 5微信 6腾讯微博 7新浪微博',  user\_name varchar(32) NOT NULL DEFAULT '' COMMENT '用户账号，必须唯一',  nick\_name varchar(32) NOT NULL DEFAULT '' COMMENT '用户昵称',  gender tinyint(1) unsigned NOT NULL DEFAULT '0' COMMENT '用户性别 0-female 1-male',  birthday bigint(20) unsigned NOT NULL DEFAULT '0' COMMENT '用户生日',  signature varchar(255) NOT NULL DEFAULT '' COMMENT '用户个人签名',  mobile varchar(16) NOT NULL DEFAULT '' COMMENT '手机号码(唯一)',  mobile\_bind\_time datetime COMMENT '手机号码绑定时间',  email varchar(100) NOT NULL DEFAULT '' COMMENT '邮箱(唯一)',  email\_bind\_time datetime COMMENT '邮箱绑定时间',  face varchar(255) NOT NULL DEFAULT '' COMMENT '头像',  face200 varchar(255) NOT NULL DEFAULT '' COMMENT '头像 200x200x80',  srcface varchar(255) NOT NULL DEFAULT '' COMMENT '原图头像',  create\_time datetime COMMENT '创建时间',  update\_time datetime COMMENT '修改时间',  push\_token varchar(50) NOT NULL COMMENT '用户设备push\_token',  PRIMARY KEY (uid)  ); |

|  |
| --- |
| **//用户扩展信息**  CREATE TABLE user\_extra (  uid bigint(20) NOT NULL COMMENT '用户 ID',  vendor varchar(64) NOT NULL DEFAULT '' COMMENT '手机厂商：apple|htc|samsung，很少用',  client\_name varchar(50) NOT NULL DEFAULT '' COMMENT '客户端名称，如hjskang',  client\_version varchar(50) NOT NULL DEFAULT '' COMMENT '客户端版本号，如7.0.1',  os\_name varchar(16) NOT NULL DEFAULT '' COMMENT '设备号:android|ios',  os\_version varchar(16) NOT NULL DEFAULT '' COMMENT '系统版本号:2.2|2.3|4.0|5.1',  device\_name varchar(32) NOT NULL DEFAULT '' COMMENT '设备型号，如:iphone6s、u880、u8800',  device\_id varchar(128) NOT NULL DEFAULT '' COMMENT '设备ID',  idfa varchar(50) NOT NULL DEFAULT '' COMMENT '苹果设备的IDFA',  idfv varchar(50) NOT NULL DEFAULT '' COMMENT '苹果设备的IDFV',  market varchar(20) NOT NULL DEFAULT '' COMMENT '来源',  create\_time datetime COMMENT '添加时间',  update\_time datetime COMMENT '更新时间',  extend1 varchar(100) NOT NULL DEFAULT '' COMMENT '扩展字段1',  extend2 varchar(100) NOT NULL DEFAULT '' COMMENT '扩展字段2',  extend3 varchar(100) NOT NULL DEFAULT '' COMMENT '扩展字段3',  PRIMARY KEY (uid)  ); |

|  |
| --- |
| **// 用户位置信息**  CREATE TABLE user\_location (  uid bigint(20) unsigned NOT NULL COMMENT '用户ID',  curr\_nation varchar(10) NOT NULL DEFAULT '' COMMENT '所在地国',  curr\_province varchar(10) NOT NULL DEFAULT '' COMMENT '所在地省',  curr\_city varchar(10) NOT NULL DEFAULT '' COMMENT '所在地市',  curr\_district varchar(20) NOT NULL DEFAULT '' COMMENT '所在地地区',  location varchar(255) NOT NULL DEFAULT '' COMMENT '具体地址',  longitude decimal(10,6) DEFAULT NULL COMMENT '经度',  latitude decimal(10,6) DEFAULT NULL COMMENT '纬度',  update\_time datetime COMMENT '修改时间',  PRIMARY KEY (uid)  ) ; |

**用户日志信息**

日志信息，用来保存用户注册或者登录行为的。另外会有一些修改密码或者修改重要信息的日志记录。

|  |
| --- |
| **//用户登录日志**  CREATE TABLE user\_login\_log (  id bigint(20) NOT NULL AUTO\_INCREMENT,  uid bigint(20) unsigned NOT NULL DEFAULT '0' COMMENT '用户uid',  type tinyint(3) unsigned NOT NULL DEFAULT '1' COMMENT '登录方式 第三方/邮箱/手机等',  command tinyint(3) unsigned NOT NULL DEFAULT '1' COMMENT '操作类型 1登录成功 2登出成功 3登录失败 4登出失败',  version varchar(32) NOT NULL DEFAULT '1.0' COMMENT '客户端版本号',  client varchar(20) NOT NULL DEFAULT 'dabaozha' COMMENT '客户端',  device\_id varchar(64) NOT NULL DEFAULT '' COMMENT '登录时设备号',  lastip varchar(32) NOT NULL DEFAULT '' COMMENT '登录ip',  os varchar(16) NOT NULL DEFAULT '' COMMENT '手机系统',  osver varchar(32) NOT NULL DEFAULT '' COMMENT '系统版本',  text varchar(200) NOT NULL DEFAULT '',  create\_time datetime COMMENT '操作时间',  PRIMARY KEY (id),  KEY idx\_uid\_type\_time (uid,type,create\_time) USING BTREE,  KEY idx\_create\_time (create\_time)  ); |

|  |
| --- |
| **//用户注册日志**  CREATE TABLE user\_register\_log (  id bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '自增ID',  uid bigint(20) unsigned NOT NULL COMMENT '用户ID',  register\_method tinyint(2) unsigned NOT NULL COMMENT '注册方式1手机号 2邮箱 3用户名 4qq 5微信 6腾讯微博 7新浪微博',  register\_time int(11) NOT NULL COMMENT '注册时间',  register\_ip varchar(16) NOT NULL DEFAULT '' COMMENT '注册IP',  register\_client varchar(16) NOT NULL DEFAULT '' COMMENT '注册客户端',  PRIMARY KEY (id)  ) ; |

|  |
| --- |
| **//修改信息日志**  CREATE TABLE user\_info\_update (  id bigint(20) NOT NULL AUTO\_INCREMENT COMMENT '自增ID',  uid bigint(20) unsigned NOT NULL COMMENT '用户ID',  attribute\_name varchar(30) NOT NULL COMMENT '属性名',  attribute\_old\_val varchar(30) NOT NULL DEFAULT '' COMMENT '属性对应旧的值',  attribute\_new\_val varchar(30) NOT NULL DEFAULT '' COMMENT '属性对应新的值',  update\_time datetime COMMENT '修改时间',  PRIMARY KEY (id)  ) ; |

**全局uid**

建议不要使用表的主键作为用户ID，而是使用ID生成器(发号器)生成用户的唯一标示guid。当用户量急剧上升时，往往会采取分库分表的方法，然后通过将uid取余写到不同的表中。如果单纯的以某个表主键作为ID。会限制插入性能和增加业务复杂度,其次在分布式数据库中也无法保证ID唯一性。

全局ID生成，是有很多方案的。简单一点，可以采用redis自增属性，因为其具有原子性，在分布式坏境中，能保证ID的唯一性。另外还有其他的一些开源方案，可自行Google。

**Access Token**

与传统的Session相比，Access Token比较适合做RESTful Api开发。传统Web应用中，用户登录后会写用户信息到cookie中，服务端通过Session就能得到用户的身份。

Access Token的是OAuth2.0中用户经过授权后，返回调用API的凭证。对于自己的应用来讲，用户在登录后，即返回access*token。在token有效期内可凭借此凭证，调用其他接口。对于access*token的刷新有两种方案，第一种每次用户重启app时，重新refresh。第二种，在调用周期内服务端发现access token可能过期时，返回新的token给客户端。

至于Access Token的生成，这个并没有规定，只要保证其唯一性即可。简单点，对用户uid和当前时间哈希得到新的Access Token，并设置过期时间。另外也可以采用[JWT](https://link.zhihu.com/?target=https%3A//jwt.io/)实现。