服务端 数据库设计

mysql 数据库设计

- **用户信息表格设计**

> 用户信息表格，保存用户的 账号 id，用户名，密码，邮箱等重要信息
>
> 以及辅助信息
>
> 用途：根据指定用户 id 获取信息，并展示
>
> 重要属性：邮箱，邮箱是用户的安全保证，通过邮箱可以注册，重置密码，进行重要操作

字段名	字段类型	键属性	说明
`uid`	`int`	`primary key` `auto_increment`	用户 id
`uname`	`varchar`	`not null`	用户名
`password`	`varchar`	`not null`	密码
`email`	`varchar`	`not null` & `unique key`	用户认证凭证
`sex`	`tinyint`	`default null`	性别
`age`	`tinyint`	`defalut null`	年龄
`create_time`	`int`	`not null`	注册时间 unix 时间戳
`more_info`	`text`	`default null`	额外信息 json

建表语句

```
create table `user` (
    `uid` int primary key auto_increment COMMENT '用户 id 唯一 自增',
    `uname` varchar(64) not null COMMENT '用户名',
    `password` varchar(32) not null COMMENT '密码',
    `email` varchar(32) not null COMMENT '绑定邮箱',
    `sex` tinyint default null COMMENT '性别',
    `age` tinyint default null COMMENT '年龄',
    `create_time` int not null COMMENT '注册时间',
    `more_info` text COMMENT '额外信息 json 格式保存',
    unique key (`email`) COMMENT '邮箱 唯一'
) COMMENT = '用户表';
```

- **用户注册表格设计** `效验作用`

> 注册信息表：用于用户注册验证真实性，是成为真正用户的过渡信息
>
> 这里使用 邮箱服务对用户进行验证

字段名	字段类型	键属性	说明
email	varchar	primary	注册邮箱
auth	varchar	not null	验证码
expire_time	int	not null	失效时间

> 验证码设计 6位
>
> 失效时间用途：定时任务清除失效的验证码，确保数据表不会有过多垃圾数据

建表语句

```
create table `register` (
    `email` varchar(32) primary key COMMENT '验证邮箱',
    `auth` varchar(10) not null COMMENT '验证码',
    `expire_time` int not null COMMENT '失效时间'
) COMMENT = '用户注册验证表';
```

- **重置密码信息表格**（用于忘记密码情景）

> 数据表同上面的注册过渡表

表设计相同，是可以合并使用一个表格，但是分开设计的原因在于：

考虑一种不太合理的情形，假如 A 如果使用 a 邮箱注册账号，服务端已发送验证码，这样在表格 `register` 中会有一个注册记录，其中一个字段保存着验证码。

假设在 用户 A 在使用 验证码进行认证之前，B用户在请求找回密码的服务，B 理应输入自己的邮箱，但是错误的输成了 A 的邮箱 a（这种情形下对 B 是没有副作用的，对 A 的影响就是 A 会再次接收到一个包含用于重置密码的验证码的邮件），如果将注册和重置密码使用的是同一个表格，会导致注册验证码被 重置验证码覆盖。

分开设计可以避免验证码被覆盖，两个表格为不同的 api 提供服务（但是不足的是 A 用户依旧会收到重置密码的邮件）

> 在重置密码时主要用到邮箱验证，我们无法确认用户输入的邮箱就是其本人的邮箱，即使要求用户重置密码时输入 自己的账号 id 与 邮箱 ，然后在服务端进行验证后决定是否发送消息。但是依旧避免不了利用服务端进行恶意操作的行为。这里我们不考虑这种恶意行为。

- **好友关系表**

> 表格设计：
>
> 由于关系是相互的，这样设计可以减少数据冗余，每一对关系在数据表中只会存在一份.

字段名	字段类型	键属性	说明
`uid_1`	`int`	`not null`	用户 1
`uid_2`	`int`	`not null`	用户 2
`remark_1_2`	`varchar`	`default null`	1 对 2 的备注
`remark_2_1`	`varchar`	`default null`	2 对 1 的备注
`group_1_2`	`varchar`	`default null`	1 对 2 的分组
`group_2_1`	`varchar`	`default null`	2 对 1 的分组

> 暂不考虑 黑名单，单向删除就会删除好友关系
>
> `uid_1` 和 `uid_2` 组合作为主键

建表语句

```
create table `friend` (
    `uid_1` int not null COMMENT '用户 1',
    `uid_2` int not null COMMENT '用户 2',
    `remark_1_2` varchar(64) COMMENT '用户1 对 用户2 的备注',
    `remark_2_1` varchar(64) COMMENT '用户2 对 用户1 的备注',
    `group_1_2`  varchar(64) COMMENT '用户1 对 用户2 的分组',
    `group_2_1`  varchar(64) COMMENT '用户2 对 用户1 的分组',
    primary key(`uid_1`, `uid_2`) COMMENT '好友关系 唯一性'
) COMMENT = '好友关系表';
```

- **好友离线消息表**

字段名	字段类型	键属性	说明
`to_uid`	`int`	`index` & `not null`	接收者 id 索引
`from_uid`	`int`	`not null`	发送者 id
`msg`	`text`	`not null`	消息 json 格式

> 好友离线表格只有在 接收方离线时才会使用。接收方在线时 不经过 mysql ， 会直接推送给目标用户
>
> 离线消息表 以 接收方为主体，设计索引，当用户上线时直接拉取所有 `to_uid` 等于 `用户 id` 的消息即可

> 离线消息不仅仅保存聊天的 消息，同时包含 好友申请消息，群聊邀请消息

建表语句

```
create table `pri_msg` (
    `to_uid` int not null COMMENT '接受者 id',
    `from_uid` int not null COMMENT '发送者 id',
    `msg` text not null COMMENT '消息 json 格式',
    index (`to_uid`) COMMENT '建立索引'
) COMMENT '私聊离线消息表';
```

- **群聊信息表**

> 群聊信息表：
>
> 主要保存群聊的一些基本信息，便于用户查看群聊时显示

字段名	字段类型	键属性	说明
gid	int	primary key & auto_increment	群聊 id 主键 自增
owner	int	not null	群主 id
gname	varchar	not null	群聊名称
create_time	int	not null	群聊建立时间
person_number	int	not null	群人数

建表语句

```
create table `group_info` (
    `gid` int primary key auto_increment COMMENT '群聊 id 自增',
    `owner` int not null COMMENT '群主',
    `gname` varchar(64) not null COMMENT '群名称',
    `create_time` int not null COMMENT '建群时间',
    `person_number` int not null COMMENT '群人数'
) COMMENT = '群聊信息表';

/* index (`owner`) */ /* 暂不考虑为 owner 建立 索引 */
```

- **群聊用户关系表**

字段名	字段类型	键属性	说明
gid	int	not null	群聊 id
uid	int	not null	用户 id
join_time	int	not null	入群时间
remark	varchar	not null	群昵称
last_msg_id	int	no null	已读群聊消息的最大 id

> 群聊用户关系表 主键设计为 (gid, uid)，last_msg_id 的作用在于标记用户已读的消息，在用户上线时推送未读消息（配合下面的群聊消息储存表使用）

建表语句

```
create table `group_person` (
    `gid` int not null COMMENT '群聊 id',
    `uid` int not null COMMENT '用户 id',
    `join_time` int not null COMMENT '加群时间',
    `remark` varchar(64) COMMENT '群聊备注',
    `last_msg_id` int not null COMMENT '已读的当前群聊最后一条消息 id',
    primary key(`gid`, `uid`)
) COMMENT = '群聊 用户关系表';
```

- **群聊离线消息表**

> 针对不同的群聊，每个群聊设计一个表格。便于消息 id 的自增
>
> 群聊表名设计为 `group:$gid` `$gid` 表示群号

字段名	字段类型	键属性	说明
`mid`	`int`	`primary key` `auto_increment`	`消息 id`
`from_uid`	`int`	`not null`	`发送者 id`
`msg`	`text`	`not null`	`消息`

> 说明：群聊消息表格 消息 id 设置为自增，进行群聊离线消息推送时，只需要记录发送者 id 与发送的消息。

建表语句

```
create table `group:gid` (
    `mid` int primary key auto_increment COMMENT '消息 id, 自增',
    `from_uid` int not null COMMENT '发送者 id',
    `msg` text not null COMMENT '群聊消息'
) COMMENT = '群聊离线消息列表';
```