**服务端设计文档**

# 项目名称：微爱情侣点滴

|  |  |  |  |
| --- | --- | --- | --- |
| **版本** | **创建作者** | **修改作者** | **内容** |
| V1.0 | 陈文鸿  2014-08-17 |  | 数据库设计、ER图、表结构 |
|  |  |  |  |

项目名称：微爱情侣点滴 1

一、系统分析 2

二、数据库设计 2

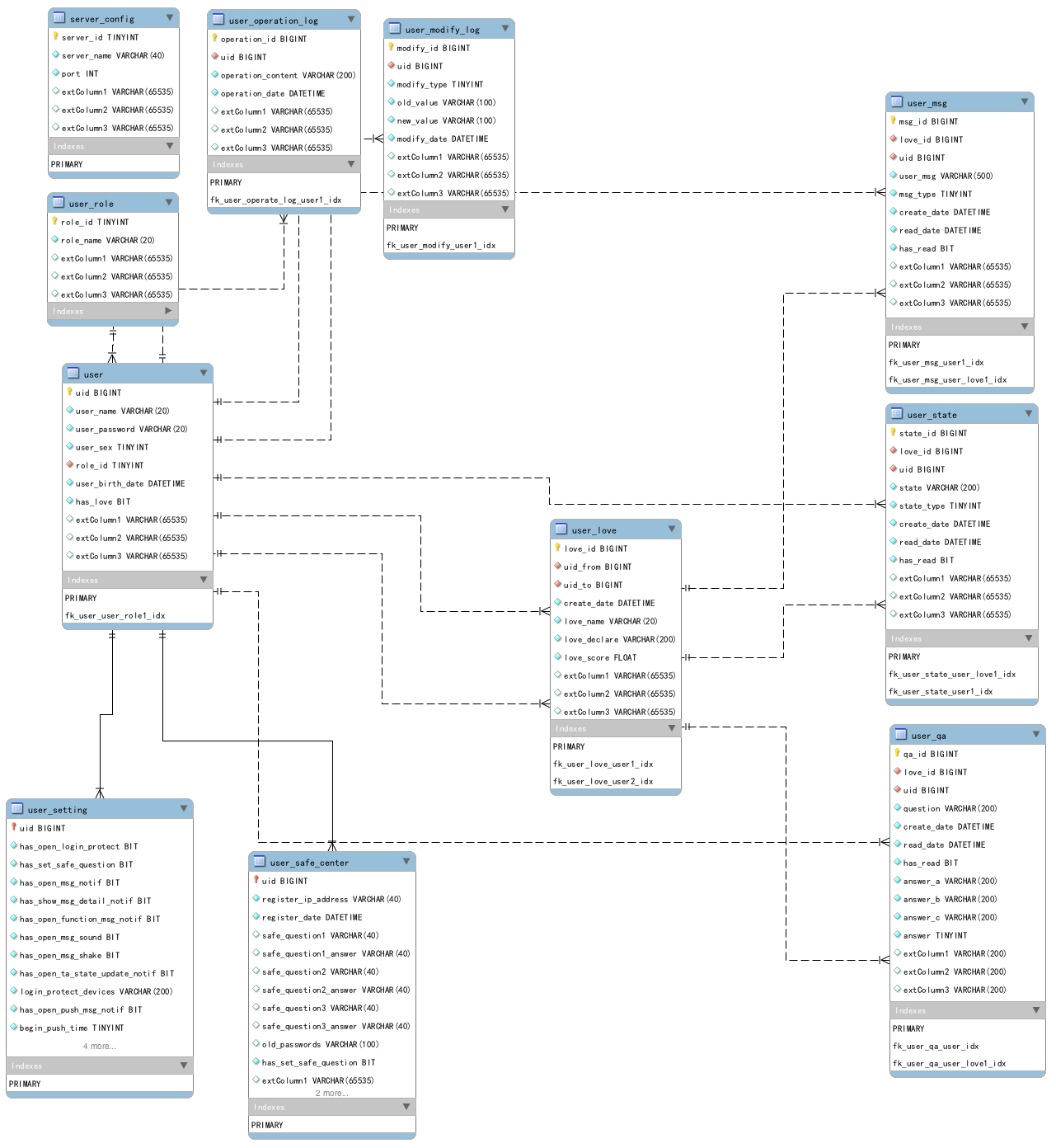
三、接口设计： 4

四、通信协议 5

## 一、系统分析

## 二、数据库设计

1、ER图



2.表结构

说明：每个表都有25个扩展字段，其中20个TEXT类型，5个BLOB类型。本版本尚未使用。

共有字段：

create\_time：创建时间；

modify\_time：修改时间；

create\_ip：创建这条记录的ip地址，通常是指服务端的地址；

remote\_ip：远程地址，通常是指用户的ip地址；

state\_id：当前记录的状态，（0是新增、1是修改、4是删除）；

（1）server

服务器表：将来可能因为用户量大，导致并发不稳定或者宕机情况。因此，这个表的设计是用来解决负载均衡。

server\_id：自增，唯一标识；

server\_name：服务器名称（地址）；

port：服务器端口；

（2）user\_role

用户角色表：存储用户角色信息。

user\_role\_id：自增，唯一标识；

role\_name：用户角色；

（3）user

用户表：存储系统用户信息；

uid：自增，唯一标识；

user\_name：用户名；

user\_password：用户密码；

user\_sex：用户性别；

user\_state\_id：用户状态id；

user\_role\_id：用户角色id；

user\_birth\_date\_time：用户出生日期；

has\_love：是否恋爱；

（4）user\_setting

用户设置表；

uid：用户id，唯一标识；

has\_open\_login\_protect：是否开启登录保护；

has\_open\_msg\_notif：是否开启消息提醒；

has\_msg\_detail\_notif：是否开启显示消息详情；

has\_open\_function\_msg\_notif：是否开启功能消息提醒；

has\_open\_msg\_sound：是否开启消息提醒声音；

has\_open\_msg\_shake：是否开启消息提醒震动；

has\_open\_ta\_state\_update\_notif：是否开启Ta的状态更新提醒；

login\_protect\_devices：登录保护设备列表；

has\_open\_push\_msg\_notif：是否开启push消息；

begin\_push\_time：push允许开始时间，单位小时；

end\_push\_time：push允许结束事件，单位小时；

（5）user\_safe\_center

用户安全中心表：提供用户安全信息验证，密码找回、申诉的依据。

uid：用户id，唯一标识；

register\_ip：用户注册的ip地址，用于验证用户注册的地址；

register\_time：用户注册日期；

has\_set\_safe\_question：是否设置了安全问题；

safe\_question1：安全问题1；

safe\_question1\_answer：安全问题1的答案；

safe\_question2：安全问题2；

safe\_question2\_answer：安全问题2的答案；

safe\_question3：安全问题3；

safe\_question3\_answer：安全问题3的答案；

old\_passwords：旧密码列表；

（6）user\_operation\_log

用户操作记录表：记录用户重要的操作记录。

operation\_id：自增，唯一标识；

uid：用户id；

operation\_content：操作内容；

operation\_date：操作日期；

（7）user\_operation\_log

用户操作记录表：记录用户的重要操作；

user\_operation\_log\_id：自增，唯一标识；

uid：用户id；

operation\_type\_id：修改类型id；

operation\_content：操作内容；

old\_value：旧记录值；

new\_value：新记录值；

table\_name：操作的数据表名；

table\_column：操作的数据表列名；

（8）operation\_type

操作类型表：存储操作类型；

operation\_type\_id：自增，唯一标识；

operation\_type：操作类型名；

（9）user\_love

用户爱情信息表：记录用户爱情相关信息；

user\_love\_id：自增，唯一标识；

uid\_from：用户id，邀请者；

uid\_to：用户id，被邀请者；

love\_name：爱情标题；

love\_declare：爱情宣言；

love\_score：爱情分值；

（10）state

数据库表记录状态表：记录每条记录的状态信息，0指新增，1指修改，4指删除。

state\_id：自增，唯一标识；

state：状态名；

（11）user\_status

用户状态表：记录用户状态信息（由用户自己填写的状态）。

user\_status\_id：自增，唯一标识；

uid：用户id；

user\_status：用户的状态；

user\_status\_type\_id：用户状态信息类型；

read\_time：状态被读取的时间；

has\_read：是否已读；

（12）user\_login\_log

用户登录记录表：记录用户登录信息。

user\_login\_log\_id：自增，唯一标识；

uid：用户id；

login\_type\_id：登录类型id；

login\_time：登录时间；

login\_device：登录设备；

is\_safety：是否安全；

unsafe\_reason：不安全的原因；

（13）user\_login\_type

用户登录类型表：存储登录类型信息。

user\_login\_type\_id：自增，唯一标识；

login\_type：登录类型；

（14）user\_qa

用户问答表：存储用户问答信息。

user\_qa\_id：自增，唯一标识；

user\_love\_id：用户情侣id；

uid：用户id；

question：问题；

## 三、接口设计：

errorCode:

200：成功

201：成功，但不返回data数据

400：请求参数错误

401：uid不存在

<http://myphp/RemoteDBMngr/Business/Interface/GetUserState>

请求方式：GET

请求参数：uid,isWifi,sid(如果带上去，接口会判断，假如没有新的状态了，它不返回，否则，取最后一条已读消息)

返回：

{"data":

{"sid":"1","uid":"10000","user\_state":"我在写代码！","create\_date":"2014-07-21 22:44:00","read\_date":"2014-07-22 07:02:02","has\_read":"0","extColumn1":null,"extColumn2":null,"extColumn3":null},"errorCode":"200",

"errorMsg":""}

示例：

<http://myphp/RemoteDBMngr/Business/Interface/GetUserState?uid=10000>

## 四、通信协议

消息格式：

发送消息格式{operation:XX, key:value,…}

operation:

op\_login登陆{uid, user\_password}

op\_logout登出{uid}

op\_send\_msg发送消息{to\_uid, msg,msg\_type}

op\_unread\_msg\_count未读消息数{uid}

op\_unread\_msg未读消息{from\_uid, msg, msg\_type}

op\_user\_login\_state用户登陆状态{to\_uid}

接收消息格式{data:{ operation:XX, key:value,…} ,errorCode:xx, errorMsg:xx}

消息类型：

text、image、url