

**数据库系统实验一**

**社会网络数据库系统开发**

学 院 计算机科学与技术

学　　 号

班　　 级

学 生

日 期 2022年4月30日

目录

[1. ER图 3](#_Toc6912)

[2. 关系表、约束、视图和事务 3](#_Toc15336)

[2.1实验环境 3](#_Toc4893)

[2.2关系表 4](#_Toc4065)

[2.3约束 7](#_Toc16854)

[2.4视图 7](#_Toc25829)

[2.5事务 8](#_Toc31711)

[2.5.1用户注册 8](#_Toc30128)

[2.5.2新建日志 9](#_Toc20082)

[3. 关键代码 9](#_Toc10808)

[3.1界面 9](#_Toc31408)

[3.2数据库 10](#_Toc23579)

[3.3事务 10](#_Toc1235)

[4. 运行界面 11](#_Toc4978)

[4.1登录 11](#_Toc15919)

[4.2注册 12](#_Toc10076)

[4.3菜单 12](#_Toc10268)

[4.3.1修改个人签名 13](#_Toc18362)

[4.3.2查看个人信息 13](#_Toc16181)

[4.3.3修改个人信息 14](#_Toc25076)

[4.3.4修改个人经历 15](#_Toc31126)

[4.3.5好友 17](#_Toc8385)

[4.3.6日志 22](#_Toc22525)

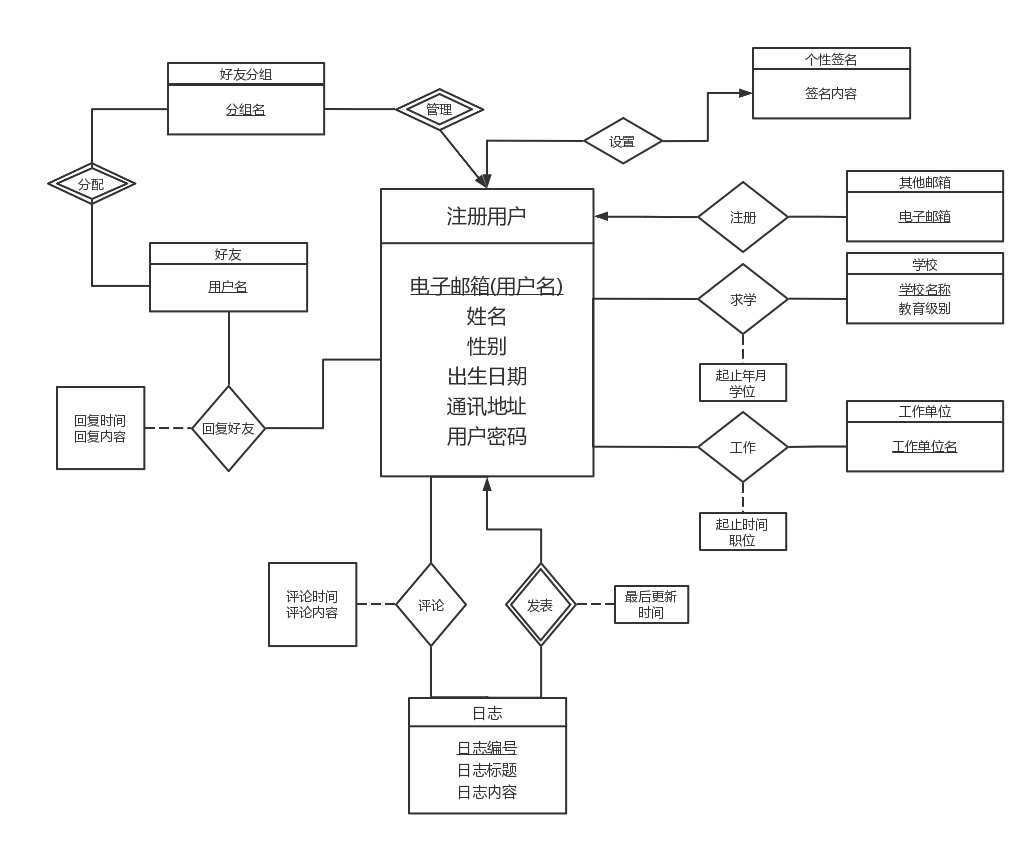
[4.3.7回复 24](#_Toc1858)

[5. 遇到并解决的问题 25](#_Toc10487)

[5.1MySQL 中Date类型与JAVA相关类型转换操作 25](#_Toc27128)

[5.2JAVA Swing JTextArea组件不显示拖动条 26](#_Toc22152)

# ER图



实体（8个）

联系（9个）包含多对多、一对一、多对一联系

# 关系表、约束、视图和事务

除事务外，关系表、约束和视图定义详见脚本文件social.sql

## 2.1实验环境

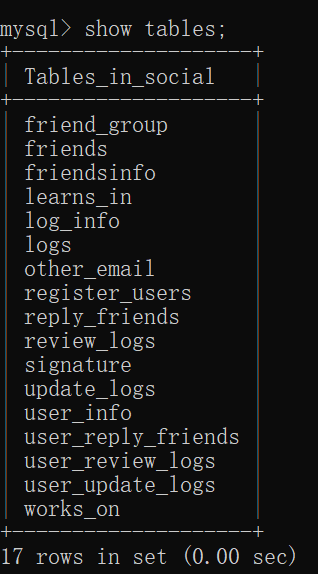
操作系统：win10

IDE：IDEA 2021.2

语言： Java 1.8 (version1.8.0\_291)

数据库：mysql Ver 8.0.26 for Win64 on x86\_64 (MySQL Community Server - GPL)

## 2.2关系表



CREATE TABLE REGISTER\_USERS -- 注册的用户表

(

USERNAME varchar(18) not null, -- 姓名

SEX char(2) not null, -- 性别

BIRTHDAY date not null, -- 生日

EMAIL varchar(20) not null, -- 邮箱（登录用户名）

PHONE varchar(11) not null, -- 联系方式

PASSWD varchar(18) not null, -- 密码

Primary Key(EMAIL),

Check (SEX in ('f','m'))

);

CREATE TABLE OTHER\_EMAIL -- 其他邮箱表（用户注册的其他邮箱，但不能作为登录用户名）

(

EMAIL varchar(20) not null, -- 用户名（登陆用邮箱）

OTHER\_EMAIL varchar(20) not null, -- 其他邮箱

Primary Key(OTHER\_EMAIL),

Foreign Key(EMAIL) references REGISTER\_USERS(EMAIL)

);

CREATE TABLE LEARNS\_IN -- 学习经历表

(

EMAIL varchar(20) not null, -- 用户名

SCHOOL\_NAME varchar(36) not null, -- 学校名

EDUCATE\_LEVEL varchar(20) not null, -- 教育级别

DEGREE varchar(18) not null, -- 学位

START\_DATE DATE not null, -- 开始日期

END\_DATE DATE not null, -- 结束日期

Primary Key(EMAIL,SCHOOL\_NAME,DEGREE),

Foreign Key(EMAIL) references REGISTER\_USERS(EMAIL)

);

CREATE TABLE WORKS\_ON -- 工作经历

(

EMAIL varchar(20) not null, -- 用户名

WORKPLACE\_NAME varchar(36) not null, -- 工作单位名

JOB varchar(18) not null, -- 工作名

START\_DATE DATE not null, -- 开始日期

END\_DATE DATE not null, -- 结束日期

Primary Key(EMAIL,WORKPLACE\_NAME,JOB),

Foreign Key(EMAIL) references REGISTER\_USERS(EMAIL)

);

CREATE TABLE LOGS -- 日志表

(

LNO bigint(20) not null auto\_increment, -- 唯一标志日志号

LOG\_NAME varchar(36) not null, -- 日志标题

LOG\_DATA TEXT not null, -- 日志内容

Primary Key(LNO)

);

CREATE TABLE UPDATE\_LOGS -- 更新日志记录表

(

EMAIL varchar(20) not null, -- 更新日志用户名

LNO bigint(20) not null unique, -- 更新日志的日志号

LAST\_UPDATE DATE not null, -- 最后更新日期

Primary Key(EMAIL,LNO),

Foreign Key(EMAIL) references REGISTER\_USERS(EMAIL),

Foreign Key(LNO) references LOGS(LNO)

);

CREATE TABLE REVIEW\_LOGS -- 评论日志记录表

(

RNO bigint(20) not null auto\_increment, -- 评论日志记录唯一标志号

EMAIL varchar(20) not null, -- 发表评论用户名

LNO bigint(20) not null, -- 所评论的日志的日志号

REVIEW\_DATE DATE not null, -- 评论日期

REVIEW\_DATA TEXT not null, -- 评论内容

Primary Key(RNO),

Foreign Key(EMAIL) references UPDATE\_LOGS(EMAIL),

Foreign Key(LNO) references LOGS(LNO)

);

CREATE TABLE FRIEND\_GROUP -- 好友分组表

(

EMAIL varchar(20) not null, -- 拥有分组的用户名

GROUP\_NAME varchar(20) default 'ALL', -- 拥有的分组名，每个用户注册时默认“ALL”分组

Primary Key(EMAIL,GROUP\_NAME),

Foreign Key(EMAIL) references REGISTER\_USERS(EMAIL)

);

CREATE TABLE FRIENDS -- 好友表

(

EMAIL varchar(20) not null, -- 拥有好友的用户名

FRIEND varchar(20) not null, -- 好友的用户名

GROUP\_NAME varchar(20) default 'ALL', -- 好友所在的分组，新加好友时默认加入“ALL”分组

Primary Key(EMAIL,FRIEND,GROUP\_NAME),

Foreign Key(FRIEND) references REGISTER\_USERS(EMAIL),

Foreign Key(EMAIL,GROUP\_NAME) references FRIEND\_GROUP(EMAIL,GROUP\_NAME)

);

CREATE TABLE REPLY\_FRIENDS -- 回复好友记录表

(

RNO bigint(20) not null auto\_increment, -- 回复好友记录唯一标识号

EMAIL varchar(20) not null, -- 回复者用户名

FRIEND varchar(20) not null, -- 被回复者用户名

REPLY\_DATE DATE not null, -- 回复日期

REPLY\_DATA TEXT not null, -- 回复内容

Primary Key(RNO),

Foreign Key(EMAIL,FRIEND) references FRIENDS(EMAIL,FRIEND)

);

CREATE TABLE Signature -- 个人签名表

(

EMAIL varchar(20) not null, -- 用户名

DATA TEXT, -- 个人签名内容

Foreign Key(EMAIL) references REGISTER\_USERS(EMAIL)

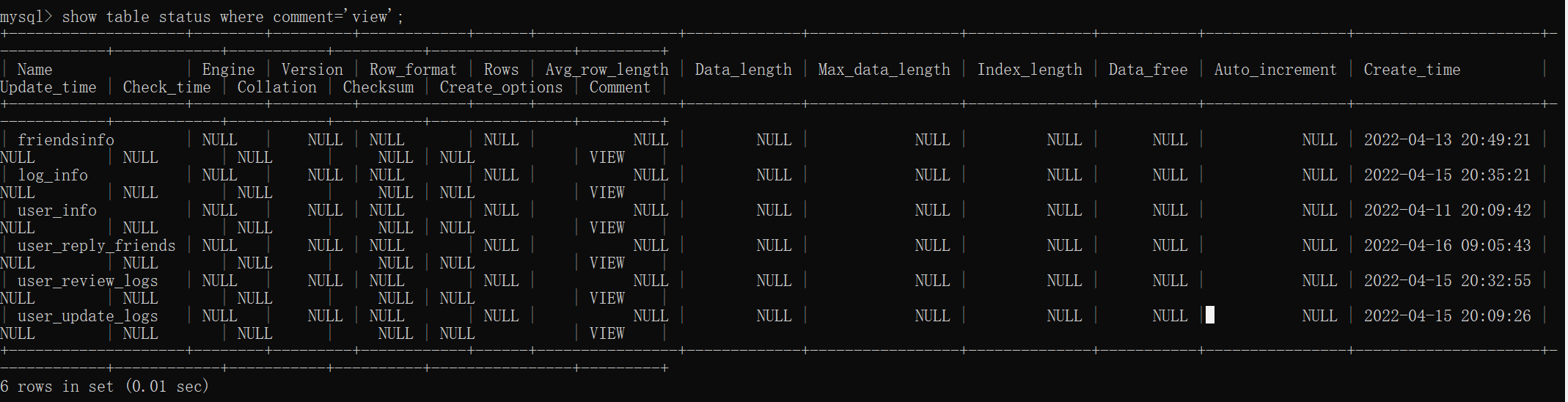
);

## 2.3约束

详见关系表定义中定义的主键约束、外键约束和空约束；

以及REGISTER\_USERS表中的自定义检查约束 ：Check (SEX in ('f','m'))

## 2.4视图



-- 连接用户表、日志更新表和日志表，解决根据用户名查找日志号、日志名等信息的连接查询

CREATE VIEW user\_update\_logs(USERNAME,EMAIL,LNO,LOG\_NAME,LAST\_UPDATE) AS

(

SELECT USERNAME,REGISTER\_USERS.EMAIL,LOGS.LNO,LOG\_NAME,LAST\_UPDATE

FROM REGISTER\_USERS,UPDATE\_LOGS,LOGS

WHERE REGISTER\_USERS.EMAIL = UPDATE\_LOGS.EMAIL AND UPDATE\_LOGS.LNO = LOGS.LNO

);

-- 连接用户表、日志更新表和日志表，解决根据日志号查找日志拥有者用户名、日志内容等信息的连接查询

CREATE VIEW log\_info(USERNAME,EMAIL,LNO,LOG\_NAME,LAST\_UPDATE,LOG\_DATA) AS

(

SELECT USERNAME,REGISTER\_USERS.EMAIL,LOGS.LNO,LOG\_NAME,LAST\_UPDATE,LOG\_DATA

FROM REGISTER\_USERS,UPDATE\_LOGS,LOGS

WHERE REGISTER\_USERS.EMAIL = UPDATE\_LOGS.EMAIL AND UPDATE\_LOGS.LNO = LOGS.LNO

);

-- 连接用户表、评论记录表和日志表，解决根据日志号查找评论者、评论内容等信息的连接查询

CREATE VIEW user\_review\_logs(LNO,REVIEW\_DATE,REVIEW\_DATA,REVIEW\_USER,REVIEW\_EMAIL) AS

(

SELECT LOGS.LNO,REVIEW\_DATE,REVIEW\_DATA,REVIEWER.USERNAME,REVIEWER.EMAIL

FROM REGISTER\_USERS AS REVIEWER,REVIEW\_LOGS,LOGS

WHERE REVIEWER.EMAIL = REVIEW\_LOGS.EMAIL AND REVIEW\_LOGS.LNO = LOGS.LNO

);

-- 连接用户表、回复记录表，解决根据用户名查找回复者、对应回复内容等信息的连接查询

CREATE VIEW user\_reply\_friends(RESPONDER\_NAME,RESPONDER\_EMAIL,REPLY\_DATE,REPLY\_DATA,RECIEVER\_EMAIL) AS

(

SELECT RESPONDER.USERNAME,RESPONDER.EMAIL,REPLY\_DATE,REPLY\_DATA,RECIEVER.EMAIL

FROM REGISTER\_USERS AS RECIEVER,REGISTER\_USERS AS RESPONDER,REPLY\_FRIENDS

WHERE REPLY\_FRIENDS.EMAIL = RESPONDER.EMAIL AND REPLY\_FRIENDS.FRIEND = RECIEVER.EMAIL

);

-- 连接用户表和个人签名表，解决查找签名的连接查询

CREATE VIEW user\_info(USERNAME,SEX,BIRTHDAY,SIGNUP\_EMAIL,PHONE,DATA) AS

(

SELECT REGISTER\_USERS.USERNAME,REGISTER\_USERS.SEX,REGISTER\_USERS.BIRTHDAY,REGISTER\_USERS.EMAIL,REGISTER\_USERS.PHONE,Signature.DATA

FROM REGISTER\_USERS,Signature

WHERE REGISTER\_USERS.EMAIL = Signature.EMAIL

);

-- 连接用户表和好友列表，解决根据用户名查找其好友的连接查询

CREATE VIEW friendsInfo(email,friend\_name,friend\_email,group\_name) AS

(

SELECT FRIENDS.EMAIL,REGISTER\_USERS.USERNAME,FRIENDS.FRIEND ,FRIENDS.GROUP\_NAME

FROM REGISTER\_USERS,FRIENDS

WHERE FRIENDS.FRIEND = REGISTER\_USERS.EMAIL

);

## 2.5事务

### 2.5.1用户注册

用户注册时，需要在各个表中初始化用户信息。过程如下：

1. 检查用户表、其他用户表中预注册邮箱是否已经注册
2. 在用户表中初始化新注册用户信息，将注册用户信息插入表中
3. 在签名表中初始化新注册用户信息，将个人签名初始化为NULL
4. 在好友列表中初始化新注册用户信息，将拥有好友列表初始化为ALL

在其中(2)(3)(4)语句需要保证操作过程中数据库的完整性

对应代码详见3.3

### 2.5.2新建日志

新建日志时，由于“用户表”中的用户和“日志表”中该用户的日志依靠“用户更新日志表”联系，但日志由一个自动增长的唯一的日志号标识，在一个用户更新一个日志时，我们需要以此完成以下三步：

1. 在日志表创建新日志
2. 得到刚刚创建的日志的日志号
3. 根据用户名和日志号更新日志更新表

(1)(2)(3)需要保证数据库完整性才能正确执行

对应代码详见3.3

# 关键代码

## 3.1界面

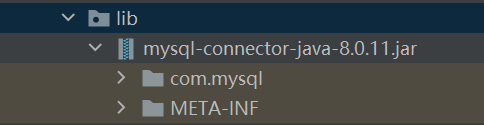
使用JAVA自带的GUI工具包Swing编写简单代码

使用的组件包括

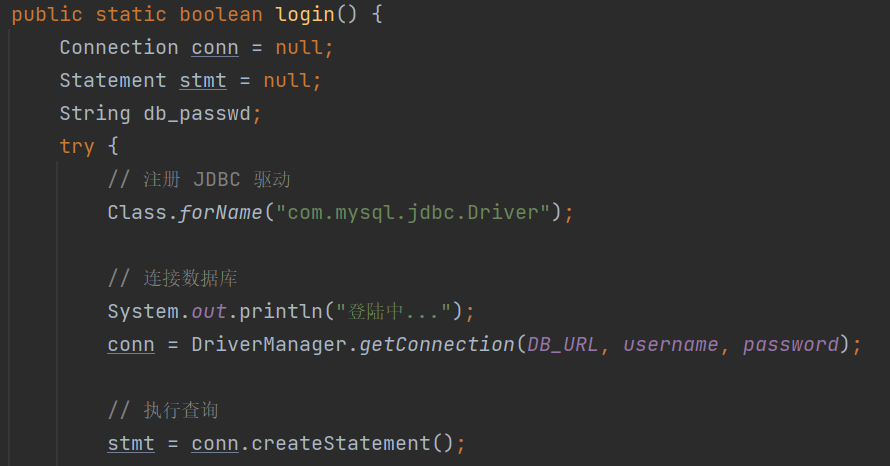
|  |  |
| --- | --- |
| [JLabel](http://blog.csdn.net/xietansheng/article/details/74362076" \t "https://blog.csdn.net/m0_37550986/article/details/_blank) | 标签 |
| [JButton](http://blog.csdn.net/xietansheng/article/details/74363221" \t "https://blog.csdn.net/m0_37550986/article/details/_blank) | 按钮 |
| [JRadioButton](http://blog.csdn.net/xietansheng/article/details/74363379" \t "https://blog.csdn.net/m0_37550986/article/details/_blank) | 单选按钮 |
| [JCheckBox](http://blog.csdn.net/xietansheng/article/details/74363460" \t "https://blog.csdn.net/m0_37550986/article/details/_blank) | 复选框 |
| [JTextField](http://blog.csdn.net/xietansheng/article/details/74363582" \t "https://blog.csdn.net/m0_37550986/article/details/_blank) | 文本框 |
| [JPasswordField](http://blog.csdn.net/xietansheng/article/details/74363621" \t "https://blog.csdn.net/m0_37550986/article/details/_blank) | 密码框 |
| [JTextArea](http://blog.csdn.net/xietansheng/article/details/74363663" \t "https://blog.csdn.net/m0_37550986/article/details/_blank) | 文本区域 |

## 3.2数据库

使用MySQL数据库保存数据，使用JAVA中mysql-connector-java-8.0.11.jar提供的的java.sql.Driver连接数据库，进行SQL语句编写和处理。







## 3.3事务

将数据库提交方式改为手动提交，即开启事务，在事务中各操作没有错误时进行commit，否则进行rollback。





# 运行界面

## 4.1登录



## 4.2注册



## 4.3菜单



### 4.3.1修改个人签名



### 4.3.2查看个人信息



### 4.3.3修改个人信息



#### 4.3.3.1添加注册邮箱



### 4.3.4修改个人经历



#### 4.3.4.1添加求学经历



#### 4.3.4.2添加工作经历



#### 4.3.4.3查看个人经历



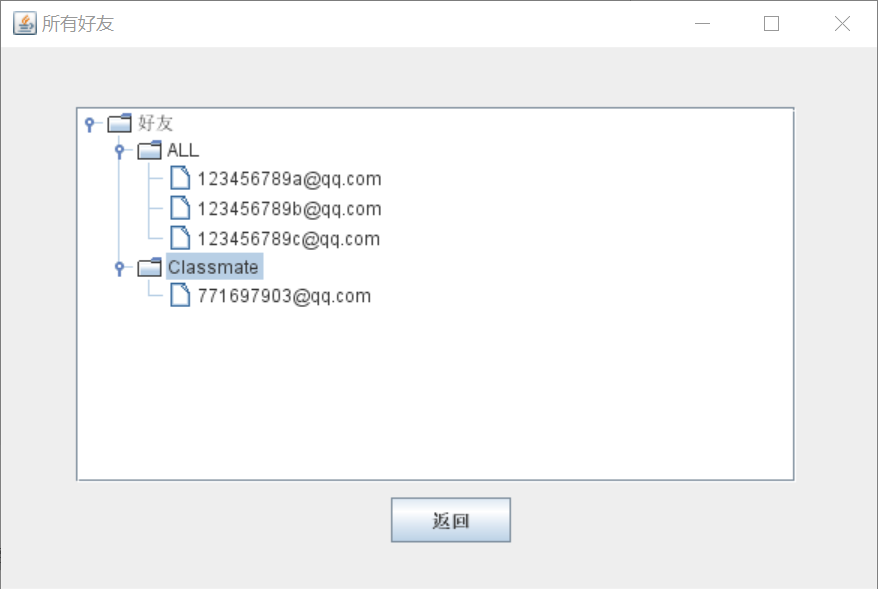
### 4.3.5好友



#### 4.3.5.1添加好友



#### 4.3.5.2所有好友



#### 4.3.5.3管理好友分组



**(1)添加分组**



**(2)移动好友**



**(3)所有分组**



#### 4.3.5.4搜索好友



**(1)按姓名查找**



**(2)按邮箱查找**



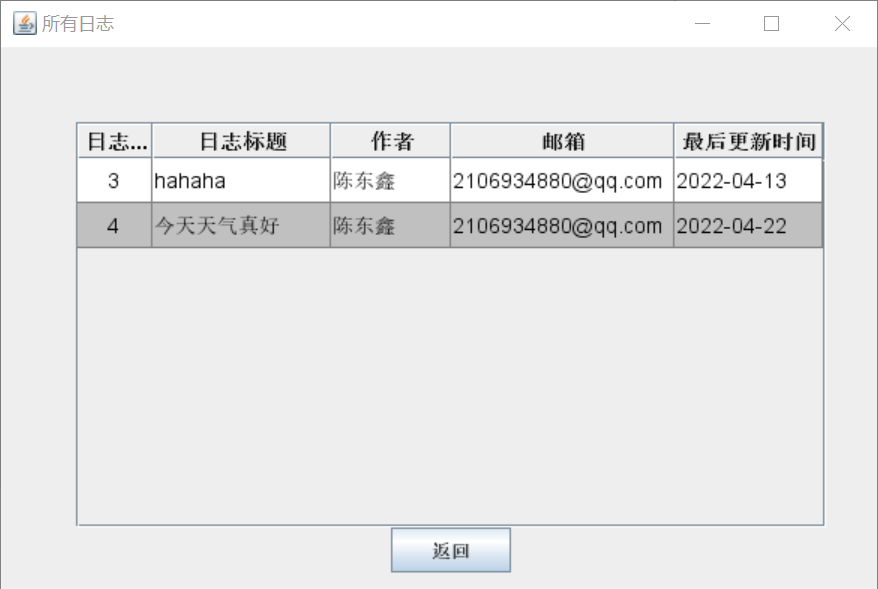
#### 4.3.5.5删除好友



### 4.3.6日志



#### 4.3.6.1所有日志



#### 4.3.6.2查看日志



#### 4.3.6.3写日志



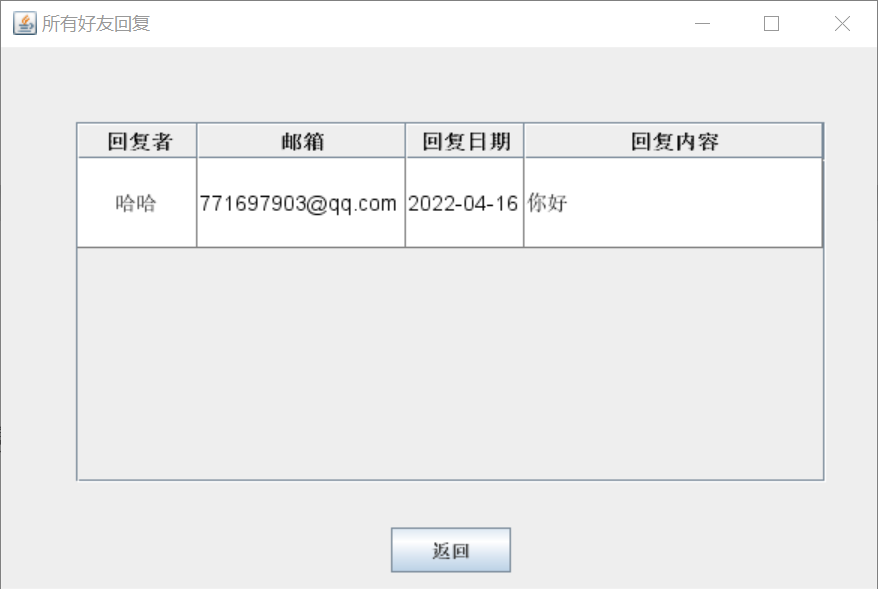
#### 4.3.6.4评论日志



### 4.3.7回复



#### 4.3.7.1查看好友日志



#### 4.3.7.2回复好友



# 遇到并解决的问题

## 5.1MySQL 中Date类型与JAVA相关类型转换操作

MySQL和Java的日期格式有很多种，Java的日期格式可以是String，Date。MySQL格式可以是varchar,datetime。

1 java 和 mysql 都定义为字符串类型保存日期，此时不用转化

2 java 和 mysql 都定义为 Date 和 Datetime 类型，也不用相互转化。返回前端时可以使用 simpledateformat 进行格式化

3 java 保存为 date，mysql 保存为 varchar，java bean的getter和setter方法使用 SimpledateFormat 格式化。

另外java中resultSet.getString(Date)可自动将Data转为“YYYY-MM-DD格式”字符串；

MySQL中将“YYYY-MM-DD格式”String类型插入Date格式数据会自动转为Date类型。

## 5.2JAVA Swing JTextArea组件不显示拖动条

当直接将JTextArea、JList、JTable等组件加入JPanel时，组件没有滚动条，当组件显示内容超过限制时则无法显示。

滚动面板JScrollPane能实现这样的要求，JScrollPane是带有滚动条的面板。JScrollPane是[Container](https://so.csdn.net/so/search?q=Container&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/qq_40270751/article/details/_blank)类的子类，也是一种容器，但是只能添加一个组件。JScrollPane的一般用法是先将一些组件添加到一个JPanel中，然后再把这个JPanel添加到JScrollPane中。这样，从界面上看，在滚动面板上，好像也有多个组件。在Swing中，像JTextArea、JList、JTable等组件都没有自带滚动条，都需要将它们放置于滚动面板，利用滚动面板的滚动条，浏览组件中的内容。