****

**G06小组事务管理系统**

**程序清单**

****

小组成员：江泓 曾雨晴 于欣汝

|  |  |  |
| --- | --- | --- |
| 文件状态：  [ ] 草稿  [ ] 正式发布  [ ] 正在修改 | 文件标识： | G06-project |
| 当前版本： | 1.1.0 |
| 作者： | 于欣汝 |
| 完成日期： | 2017-06-13 |

[1、标题 2](#_Toc3614)

[2、 PDL描述 2](#_Toc4876)

[3、模块开发表 5](#_Toc31123)

[4、 源代码 6](#_Toc8385)

# 程序清单

1、标题

用户注册

用户登录

事务增加

事务修改

事务删除（进回收站）

事务彻底删除

事务统计

事务状态更改

提醒

## PDL描述

（1）提醒和用户登录注册模块（负责人：江泓）

登录模块：

Procedure 登录 is

Begin

输入用户名和密码

通过SQL语句查找数据库中对应的用户名和密码

If（找到了）

then登录

Else

抛出异常（用户名不存在、密码错误）

End 登录

注册模块：

Procedure 注册 is

Begin

输入用户名和密码

检查用户名和密码是否符合格式

If（不符合）

Then 抛出异常（用户名应为XXX类型）

检查数据库中是否用相同用户名

If（有）

then抛出异常，已存在相同用户名

Else

对数据库进行增添

返回注册成功的提示

End 注册

（2）事务统计模块（负责人：于欣汝）

Begin 状态更改

对数据库查询的sql语句

通过preparedstatement传递参数

If<查询不到>

抛出异常

未能查询到

End if

对数据库的修改的sql语句

通过preparedstatement和execute传递参数

End

Begin 统计

对数据库查询的sql语句

通过preparedstatement传递参数

Loop while<查询结果不为空>

将查询到结果存入事务对象

将每个事物对象加入到list中

End loop

返回list类型的结果

End

（3）事务增删改模块（负责人：曾雨晴）

Begin 事务增加

输入时间，内容

If（时间不合理）

报错退出模块

If（事务内容过长）

报错退出模块

查询数据库

If（时间冲突）

提示用户

加入数据库

End

Begin 事务删除

选择事务

提示用户

If（true）

加入回收站中

End

Begin 回收站事务删除

选择事务

提示用户

If（true）

永久删除

End

Begin 事务修改

选择事务

输入时间，内容

If（时间不合理）

报错退出模块

If（事务内容过长）

报错退出模块

查询数据库

If（时间冲突）

提示用户

修改数据库内容

End

## 3、模块开发表

|  |  |
| --- | --- |
| 模块名称 | 用户注册 |
| 方法名 | register |
| 编写人员 | 曾雨晴 |
| 完成日期 | 2017-05-30 |

|  |  |
| --- | --- |
| 模块名称 | 用户登录 |
| 方法名 | Login |
| 编写人员 | 曾雨晴 |
| 完成日期 | 2017-05-30 |

|  |  |
| --- | --- |
| 模块名称 | 事务增加 |
| 方法名 | addEvent |
| 编写人员 | 于欣汝 |
| 完成日期 | 2017-05-30 |

|  |  |
| --- | --- |
| 模块名称 | 事务删除（进回收站） |
| 方法名 | intoRecycle |
| 编写人员 | 于欣汝 |
| 完成日期 | 2017-05-30 |

|  |  |
| --- | --- |
| 模块名称 | 事务彻底删除 |
| 方法名 | deleteEvent |
| 编写人员 | 于欣汝 |
| 完成日期 | 2017-05-30 |

|  |  |
| --- | --- |
| 模块名称 | 事务修改 |
| 方法名 | modifyEvent |
| 编写人员 | 于欣汝 |
| 完成日期 | 2017-05-30 |
| 模块名称 | 事务统计 |
| 方法名 | listAll |
| 编写人员 | 江泓 |
| 完成日期 | 2017-05-30 |

|  |  |
| --- | --- |
| 模块名称 | 事务状态更改 |
| 方法名 | changeState |
| 编写人员 | 于欣汝 |
| 完成日期 | 2017-05-30 |

|  |  |
| --- | --- |
| 模块名称 | 提醒 |
| 方法名 | remind |
| 编写人员 | 曾雨晴 |
| 完成日期 | 2017-05-30 |

## 源代码

1、用户模块

**public** **static** BeanUser *currentLoginUser*=**null**;

**public** **void** createUser(String userId,String pwd,String pwd1) **throws** Exception{

Connection conn=**null**;

**try**

{

conn=DBUtil.*getConnection*();

**if**(!pwd.equals(pwd1))

{

**throw** **new** BusinessException("两次输入密码不一致");

}

String sql="select userid from beanuser where userid=?";

java.sql.PreparedStatement pst=conn.prepareStatement(sql);

pst.setString(1, userId);

java.sql.ResultSet rs=pst.executeQuery();

**if**(rs.next())

{

**throw** **new** BusinessException("该用户已存在");

}

sql="insert into beanuser(userid,pwd) values(?,?)";

pst=conn.prepareStatement(sql);

pst.setString(1, userId);

pst.setString(2, pwd);

pst.execute();

pst.close();

rs.close();

}**catch**(SQLException e)

{

e.printStackTrace();

}**finally**

{

**if**(conn!=**null**)

**try** {

conn.close();

} **catch** (SQLException e) {

e.printStackTrace();

}

//DriverManager.getConnection("jdbc:derby:H:\\db\\event;shutdown=true");

}

}

**public** **void** login(String userId,String pwd) **throws** Exception{

Connection conn=**null**;

**try**

{

conn=DBUtil.*getConnection*();

String sql="select \* from beanuser where userid=? and pwd=?";

java.sql.PreparedStatement pst=conn.prepareStatement(sql);

pst.setString(1, userId);

pst.setString(2, pwd);

java.sql.ResultSet rs=pst.executeQuery();

**if**(!rs.next())

{

**throw** **new** BusinessException("密码输入错误");

}

}**catch**(SQLException e)

{

e.printStackTrace();

}**finally**

{

**if**(conn!=**null**)

**try** {

conn.close();

} **catch** (SQLException e) {

e.printStackTrace();

}

//DriverManager.getConnection("jdbc:derby:H:\\db\\event;shutdown=true");

}

}

**public** **void** modifyPwd(String userId,String pwd,String newpwd,String newpwd1) **throws** Exception{

Connection conn=**null**;

**if**(!newpwd.equals(newpwd1))

{

**throw** **new** BusinessException("两次输入密码不一致");

}

**try**

{

conn=DBUtil.*getConnection*();

String sql="select userid,pwd from beanuser where userid=? and pwd=?";

java.sql.PreparedStatement pst=conn.prepareStatement(sql);

pst.setString(1, userId);

pst.setString(2, pwd);

java.sql.ResultSet rs=pst.executeQuery();

**if**(!rs.next())

{

**throw** **new** BusinessException("密码输入错误");

}

sql="update beanuser set pwd=? where userid=?";

pst=conn.prepareStatement(sql);

pst.setString(1, userId);

pst.setString(2, newpwd);

pst.execute();

pst.close();

rs.close();

}**catch**(SQLException e)

{

e.printStackTrace();

}**finally**

{

**if**(conn!=**null**)

**try** {

conn.close();

} **catch** (SQLException e) {

e.printStackTrace();

}

//DriverManager.getConnection("jdbc:derby:H:\\db\\event;shutdown=true");

}

}

**public** BeanUser loaduser (String userid) **throws** BaseException, SQLException

{

Connection conn=**null**;

**try**

{

conn=DBUtil.*getConnection*();

String sql="select userid,pwd from beanuser where userid=?";

java.sql.PreparedStatement pst=conn.prepareStatement(sql);

pst.setString(1,userid);

java.sql.ResultSet rs=pst.executeQuery();

**if**(!rs.next())

{

**throw** **new** BusinessException("登陆账号不存在");

}

BeanUser u=**new** BeanUser();

u.setUserid(rs.getString(1));

u.setPwd(rs.getString(2));

pst.close();

rs.close();

**return** u;

}**catch** (SQLException e) {

e.printStackTrace();

**throw** **new** DbException(e);

}**finally**

{

**if**(conn!=**null**)

**try** {

conn.close();

} **catch** (SQLException e) {

e.printStackTrace();

}

//DriverManager.getConnection("jdbc:derby:H:\\db\\event;shutdown=true");

}

}

2、增删改模块

**public** **void** addEvent(String contect,String reminderTime,String state,String reminderMode,String classification )**throws** Exception{

Time t = **new** Time();

**if**(t.*timeCompare*(t.*getCurrentTime*(), reminderTime)>=0){

**throw** **new** Exception("时间设置不合理，该时间已过");

}

**if**(contect.length()>200 ){

**throw** **new** Exception("事务内容长度不能超过200个字");

}

Connection conn=**null**;

**try** {

conn=DBUtil.*getConnection*();

String sql="select count(\*) from beanevent where reminderTime = ? ";

java.sql.PreparedStatement pst = conn.prepareStatement(sql);

pst.setString(1,reminderTime );

java.sql.ResultSet rs = pst.executeQuery();

**if**(rs.getInt(1)>0){

**throw** **new** Exception("该时间已有"+rs.getInt(1)+"个事务");

}

}**catch**(Exception e){

e.printStackTrace();

conn=DBUtil.*getConnection*();

String sql="insert into Beanevent(contect,reminderTime,state,reminderMode,classification) values(?,?,?,?,?)";

java.sql.PreparedStatement pst = conn.prepareStatement(sql);

pst.setString(1, contect);

pst.setString(2,reminderTime );

pst.setString(3, state);

pst.setString(4,reminderMode );

pst.setString(5, classification);

pst.execute();

pst.close();

}

}

**public** **void** deleteEvent(BeanEvent beanevent){

Connection conn=**null**;

**try**{

conn=DBUtil.*getConnection*();

String sql="DELETE FROM beanevent WHERE eventid = ?";

java.sql.PreparedStatement pst = conn.prepareStatement(sql);

pst.setInt(1, beanevent.getEventId());

pst.execute();

pst.close();

}**catch**(Exception e){

e.printStackTrace();

}

}

**public** **void** modifyEvent(BeanEvent beanevent,String contect,String reminderTime,String state,String reminderMode,String classification )**throws** Exception{

Time t = **new** Time();

**if**(t.*timeCompare*(t.*getCurrentTime*(), reminderTime)>=0){

**throw** **new** Exception("时间设置不合理，该时间已过");

}

**if**(contect.length()>200 ){

**throw** **new** Exception("事务内容长度不能超过200个字");

}

Connection conn=**null**;

**try** {

conn=DBUtil.*getConnection*();

String sql="select count(\*) from beanevent where reminderTime = ? ";

java.sql.PreparedStatement pst = conn.prepareStatement(sql);

pst.setString(1,reminderTime );

java.sql.ResultSet rs = pst.executeQuery();

**if**(rs.getInt(1)>0){

**throw** **new** Exception("该时间已有"+rs.getInt(1)+"个事务");

}

}**catch**(Exception e){

e.printStackTrace();

conn=DBUtil.*getConnection*();

String sql="update Beanevent set contect=?,reminderTime=?,state=?,reminderMode=?,classification = ? where eventid = ?";

java.sql.PreparedStatement pst = conn.prepareStatement(sql);

pst.setString(1, contect);

pst.setString(2,reminderTime );

pst.setString(3, state);

pst.setString(4,reminderMode );

pst.setString(5, classification);

pst.setInt(6, beanevent.getEventId());

pst.execute();

pst.close();

}

}

**public** **void** intoRecycle(BeanEvent beanevent){

Connection conn=**null**;

**try**{

conn=DBUtil.*getConnection*();

String sql="UPDATE beanevent SET state = '回收站' WHERE eventid = ？";

java.sql.PreparedStatement pst = conn.prepareStatement(sql);

pst.setInt(1, beanevent.getEventId());

pst.execute();

pst.close();

}**catch**(Exception e){

e.printStackTrace();

}

}

**public** List<BeanEvent> listState(String userid){

List<BeanEvent> result=**new** ArrayList<BeanEvent>();

Connection conn=**null**;

**try** {

conn=DBUtil.*getConnection*();

String sql ="select eventId,contect,reminderTime,state,classification from BeanEvent where userId=? order by reminderTime desc";

java.sql.PreparedStatement pst=conn.prepareStatement(sql);

pst.setString(1, userid);

java.sql.ResultSet rs=pst.executeQuery();

**while**(rs.next()){

BeanEvent r=**new** BeanEvent();

r.setEventId(rs.getString(1));

r.setContect(rs.getString(2));

r.setReminderTime(rs.getString(3));

r.setState(rs.getString(4));

r.setClassification(rs.getString(5));

result.add(r);

}

} **catch** (SQLException e) {

e.printStackTrace();

}

**return** result;

}

3、统计模块

**public** List<BeanEvent> listClassification(String classification){

List<BeanEvent> result = **new** ArrayList<BeanEvent>();

Connection conn = **null**;

**try**{

conn = DBUtil.*getConnection*();

String sql= "select eventId,contect,reminderTime,state,classification from BeanEvent where classification=? order by reminderTime desc";

java.sql.PreparedStatement pst=conn.prepareStatement(sql);

pst.setString(1, classification);

java.sql.ResultSet rs=pst.executeQuery();

**while**(rs.next()){

BeanEvent r=**new** BeanEvent();

r.setEventId(rs.getString(1));

r.setContect(rs.getString(2));

r.setReminderTime(rs.getString(3));

r.setState(rs.getString(4));

result.add(r);

}

} **catch** (SQLException e) {

e.printStackTrace();

}

**return** result;

}

**public** **void** changeState(String eventid,String state) {

Connection conn=**null**;

**try** {

conn=DBUtil.*getConnection*();

String sql="update BeanEvent set state=? where eventid=?";

java.sql.PreparedStatement pst=conn.prepareStatement(sql);

pst=conn.prepareStatement(sql);

pst.setString(1, eventid);

pst.setString(2, state);

pst.execute();

pst.close();

} **catch** (SQLException e) {

e.printStackTrace();

}

}