实验六

一、相关知识点

1. JDBC基本概念
2. PreparedStatement的用法
3. JDBC数据增、删、改，事务控制等

二、实验目的：

理解Java连接数据库的基本概念。理解利用Statement对象、PreparedStatement对象进行增、删、改操作，理解事务的概念和JDBC编程方式。

三、实验内容：

1、利用PreparedStatement进行查询。

第一步：在PublisherManager类中增加方法`public List<BeanPublisher> searchPublisher(String keyword) throws BaseException`方法，要求根据关键字在出版社表中查询满足条件的出版社（出版社名称或地址中包含参数中的关键字），参考loadAllPublisher()方法，将查询结果封装为List

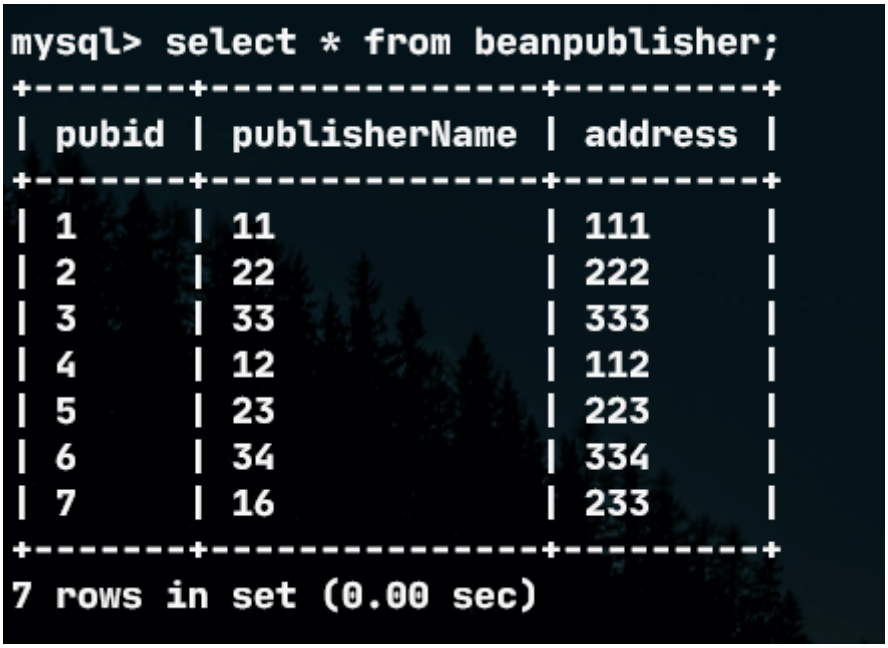
第二步：在main函数中编写测试代码进行该方法的调用测试。

```
public List<BeanPublisher> searchPublisher(String keyword) throws BaseException{
        List<BeanPublisher> result = new ArrayList<BeanPublisher>();
        Connection conn = null;
        try {
            conn = DBUtil.getConnection();
            String sql = "select pubid,publisherName,address" +
                    "  from beanpublisher";
            if (keyword != null && !"".equals(keyword)) {
                sql += " where publisherName like ? "
                        + "or address like ?";
```

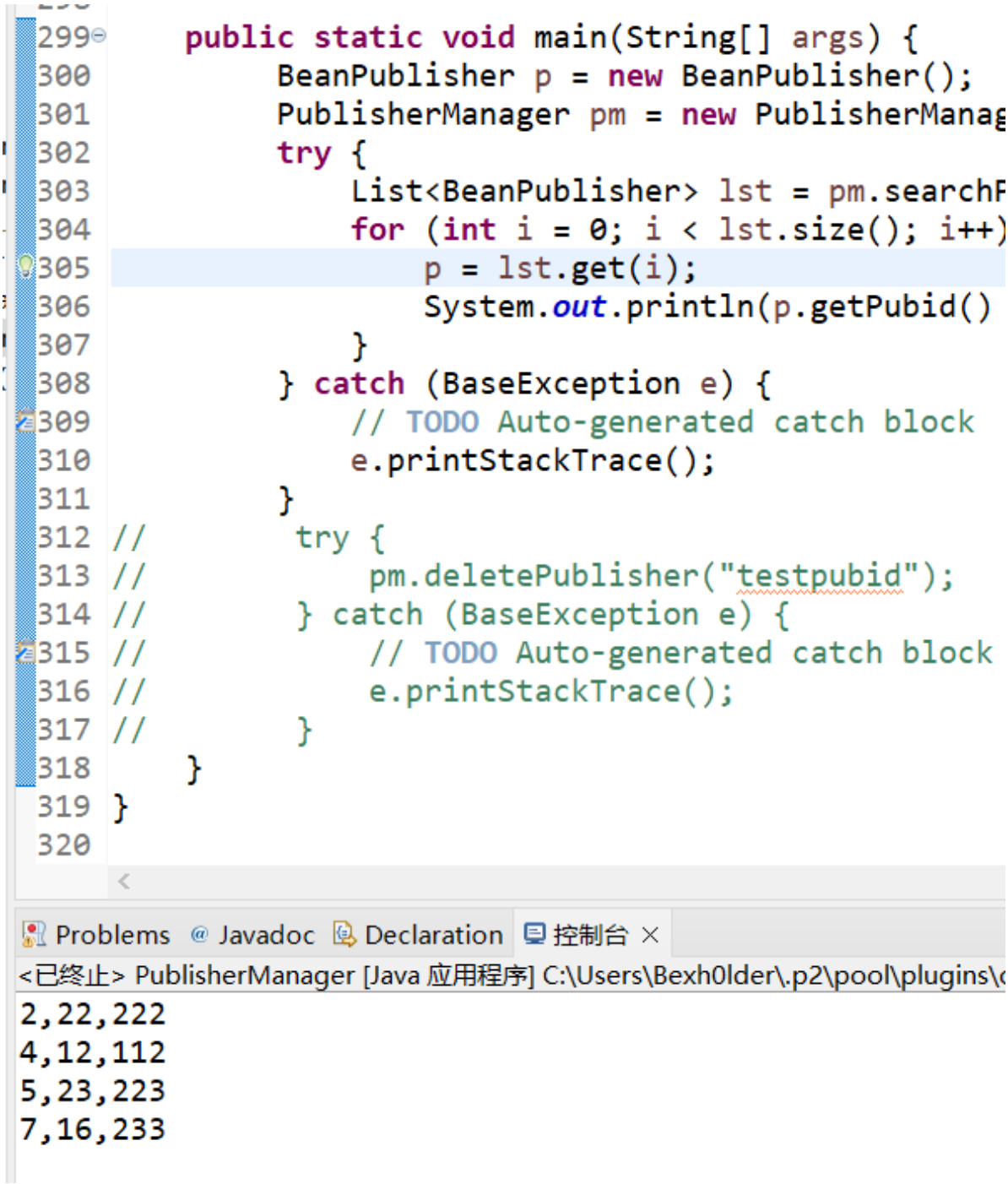
```
            }
            sql += " order by pubid";
            java.sql.PreparedStatement pst = conn.prepareStatement(sql);
            if (keyword != null && !"".equals(keyword)) {
                pst.setString(1, "%" + keyword + "%");
                pst.setString(2, "%" + keyword + "%");
            }
            java.sql.ResultSet rs = pst.executeQuery();
            while (rs.next()) {
                BeanPublisher r = new BeanPublisher();
                r.setPubid(rs.getString(1));
                r.setPublisherName(rs.getString(2));
                r.setAddress(rs.getString(3));
                result.add(r);
            }
        } catch (SQLException e) {
            e.printStackTrace();
            throw new DbException(e);
        } finally {
            if (conn != null)
                try {
                    conn.close();
                } catch (SQLException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
        }
        return result;
}
```

```
public static void main(String[] args) {
        BeanPublisher p = new BeanPublisher();
        PublisherManager pm = new PublisherManager();
        try {
            List<BeanPublisher> lst = pm.searchPublisher("2");
            for (int i = 0; i < lst.size(); i++) {
                p = lst.get(i);
                System.out.println(p.getPubid() + "," + p.getPublisherName() + "," +
p.getAddress());
            }
        } catch (BaseException e) {
            // TODO Auto-generated catch block
```

```
12              e.printStackTrace();
13          }
14  //      try {
15  //          pm.deletePublisher("testpubid");
16  //      } catch (BaseException e) {
17  //          // TODO Auto-generated catch block
18  //          e.printStackTrace();
19  //      }
20  }
```



```
mysql> select * from beanpublisher;
+-------+---------------+---------+
| pubid | publisherName | address |
+-------+---------------+---------+
| 1     | 11            | 111     |
| 2     | 22            | 222     |
| 3     | 33            | 333     |
| 4     | 12            | 112     |
| 5     | 23            | 223     |
| 6     | 34            | 334     |
| 7     | 16            | 233     |
+-------+---------------+---------+
7 rows in set (0.00 sec)
```



2、 利用Statement对象进行数据添加。

第一步：修改PublisherManager类的createPublisher方法，将其中的insert语言改成用Statement对象执行；

第二步：运行图书管理系统，进行添加出版社测试。

【实验结果与分析】

A、 写出替换的代码部分。

```
public void createPublisher(BeanPublisher p) throws BaseException {
    if (p.getPubid() == null || "".equals(p.getPubid()) || p.getPubid().length() > 20) {
        throw new BusinessException("出版社编号必须是1-20个字");
    }
```

```
        if (p.getPublisherName() == null || "".equals(p.getPublisherName()) || p.getPublisherName().length() > 50) {
            throw new BusinessException("出版社名称必须是1-50个字");
        }
        if (p.getAddress() == null || "".equals(p.getAddress()) || p.getAddress().length() > 100) {
            throw new BusinessException("出版地址必须是1-100个字");
        }


        Connection conn = null;
        try {
            conn = DBUtil.getConnection();
            String sql = "select * from BeanPublisher where pubid=?";
            java.sql.PreparedStatement pst = conn.prepareStatement(sql);
            pst.setString(1, p.getPubid());
            java.sql.ResultSet rs = pst.executeQuery();
            if (rs.next()) throw new BusinessException("出版社编号已经被占用");
            rs.close();
            pst.close();
            sql = "select * from BeanPublisher where publisherName=?";
            pst = conn.prepareStatement(sql);
            pst.setString(1, p.getPublisherName());
            rs = pst.executeQuery();
            if (rs.next()) throw new BusinessException("出版社名称已经存在");
            rs.close();
            pst.close();
            //通过PreparedStatement实现
//            sql = "insert into BeanPublisher(pubid,publisherName,address) values(?,?,?)";
//            pst = conn.prepareStatement(sql);
//            pst.setString(1, p.getPubid());
//            pst.setString(2, p.getPublisherName());
//            pst.setString(3, p.getAddress());
//            pst.execute();
//            pst.close();
            //----------------------------修改部分----------------------------------
            //通过Statement实现
            sql = "insert into BeanPublisher(pubid,publisherName,address) "
                    + "values("+ p.getPubid() + "," + p.getPublisherName() + "," + p.getAddress() + ")";
            java.sql.Statement st = conn.createStatement();
```

```
            int i = st.executeUpdate(sql);
            if(i <= 0) throw new BusinessException("插入失败");
            //-------------------------------------------------------------------------
        } catch (SQLException e) {
            e.printStackTrace();
            throw new DbException(e);
        } finally {
            if (conn != null)
                try {
                    conn.close();
                } catch (SQLException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
        }
    }
```

```
    public static void main(String[] args) {
        try {
            BeanPublisher p = new BeanPublisher();
            p.setAddress("666");
            p.setPubid("8");
            p.setPublisherName("88");
            PublisherManager pm = new PublisherManager();
            pm.createPublisher(p);
            List<BeanPublisher> lst = pm.loadAllPublisher();
            for (int i = 0; i < lst.size(); i++) {
                p = lst.get(i);
                System.out.println(p.getPubid() + "," + p.getPublisherName() + "," + p.getAddress());
            }
        } catch (BaseException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
//        try {
//            pm.deletePublisher("testpubid");
//        } catch (BaseException e) {
//            // TODO Auto-generated catch block
//            e.printStackTrace();
//        }
    }
}
```

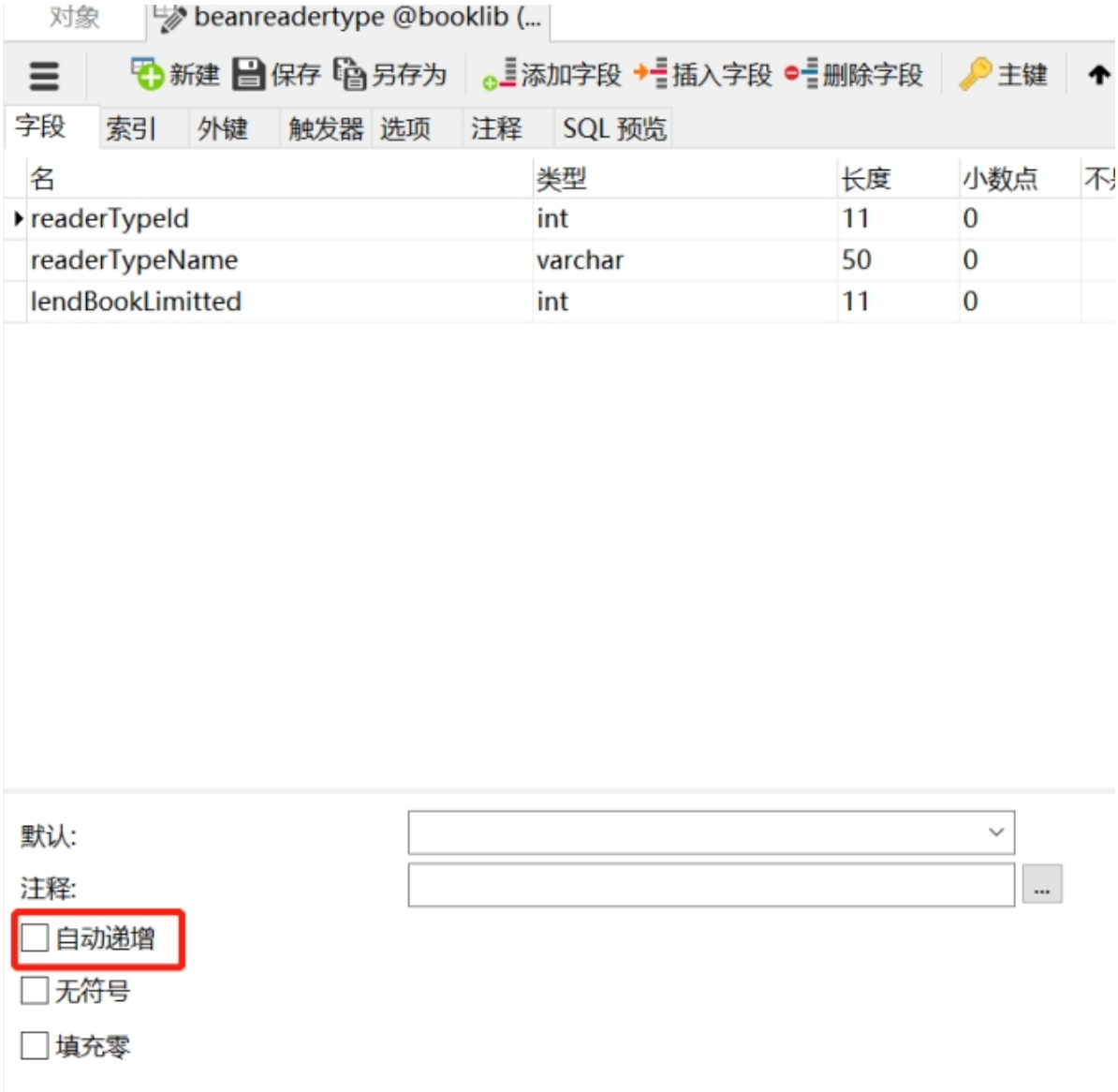
Problems Javadoc Declaration 控制台 ×

<已终止> PublisherManager [Java 应用程序] C:\Users\Bexh0lder\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208\jre\bi

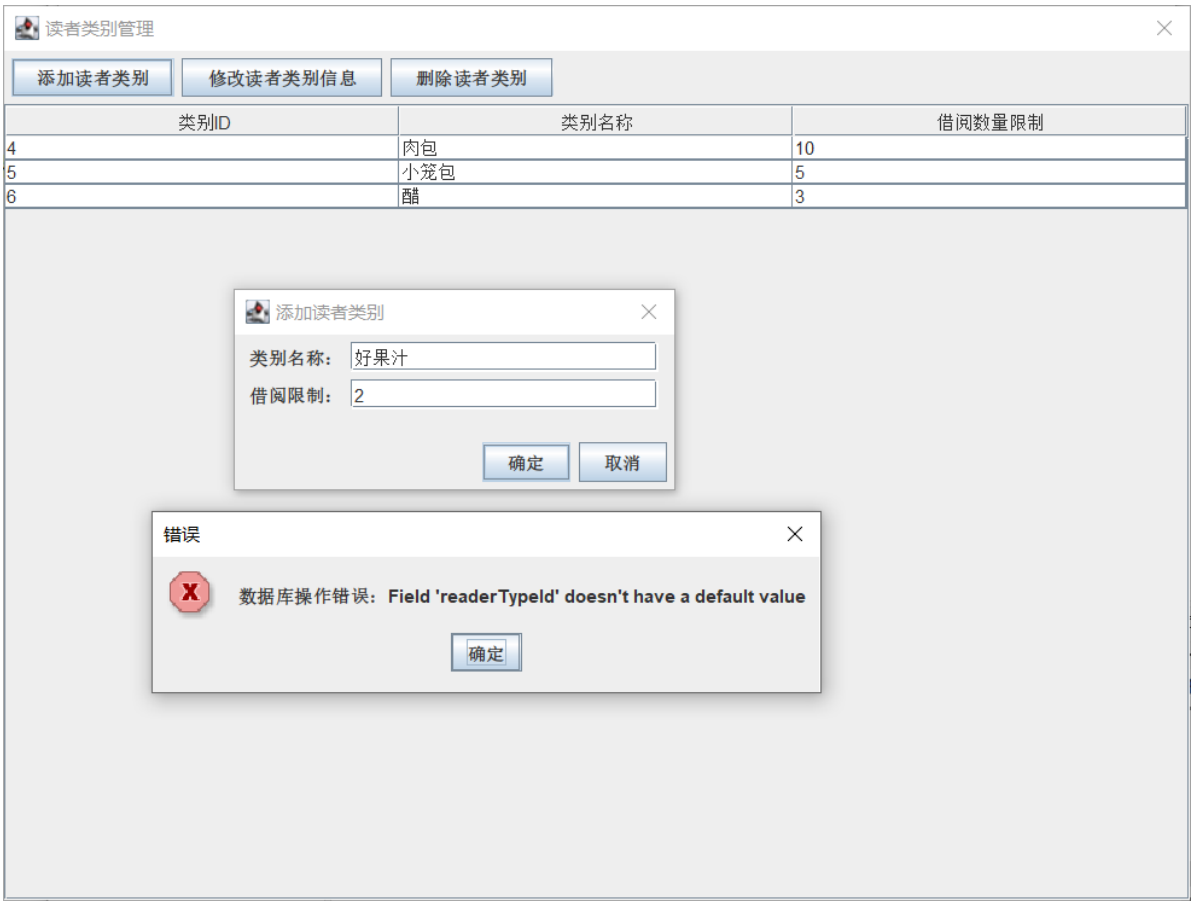
```
1,11,111
2,22,222
3,33,333
4,12,112
5,23,223
6,34,334
7,16,233
8,88,666
```

3、 利用**insert**语句添加数据时，未指定字段值处理。

第一步：将数据库表beanreadertype的readerTypeId的自动递增属性去掉。



第二步：运行图书管理系统，打开读者类别管理界面，并尝试添加一个读者类别；系统将会报一个错误，请分析说明错误原因。

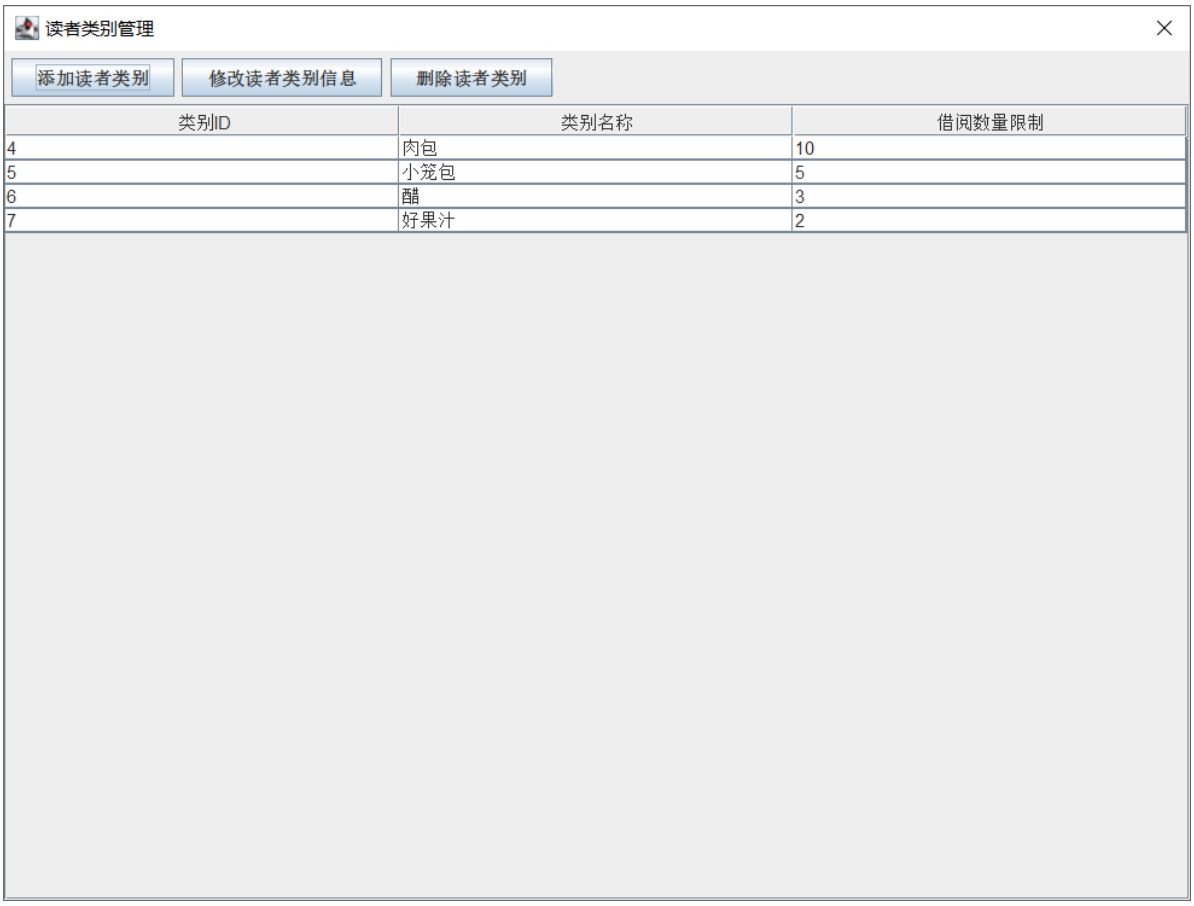


错误原因：报错内容信息是readerTypeId字段没有默认值，在将字段的自增属性取消后，插入新记录时,不会再自动地创建readerTypeId字段的值，所以报错

第三步：应该如何修改程序，使新增读者类别的ID为表中现有数据的最大ID值+1。

```
1  public void createReaderType(BeanReaderType rt) throws BaseException {
2      if (rt.getReaderTypeName() == null || "".equals(rt.getReaderTypeName()) ||
   rt.getReaderTypeName().length() > 20) {
3          throw new BusinessException("读者类别名称必须是1-20个字");
4      }
5      if (rt.getLendBookLimitted() < 0 || rt.getLendBookLimitted() > 100) {
6          throw new BusinessException("借阅图书数量必须在0-100之间");
7      }
8      Connection conn = null;
9      int maxReaderTypeId = 0;//修改处
10     try {
11         conn = DBUtil.getConnection();
12         String sql = "select * from BeanReaderType where readerTypeName=?";
13         java.sql.PreparedStatement pst = conn.prepareStatement(sql);
14         pst.setString(1, rt.getReaderTypeName());
15         java.sql.ResultSet rs = pst.executeQuery();
16         if (rs.next()) throw new BusinessException("读者类别名称已经被占用");
17         rs.close();
18         pst.close();
```

```
19              //-------------------------修改处-----------------------------
20              sql = "select max(readerTypeId) from beanreadertype";
21              pst = conn.prepareStatement(sql);
22              rs = pst.executeQuery();
23              if (rs.next()) maxReaderTypeId = rs.getInt(1);
24              //-----------------------------------------------------------------
25              sql = "insert into
    BeanReaderType(readerTypeId,readerTypeName,lendBookLimitted)
    values(?,?,?)";//修改处
26              pst = conn.prepareStatement(sql);
27              //-------------------------修改处-----------------------------
28              pst.setInt(1, maxReaderTypeId + 1);
29              pst.setString(2, rt.getReaderTypeName());
30              pst.setInt(3, rt.getLendBookLimitted());
31               //-----------------------------------------------------------------
32              pst.execute();
33              pst.close();
34          } catch (SQLException e) {
35              e.printStackTrace();
36              throw new DbException(e);
37          } finally {
38              if (conn != null)
39                  try {
40                      conn.close();
41                  } catch (SQLException e) {
42                      // TODO Auto-generated catch block
43                      e.printStackTrace();
44                  }
45          }
46      }
```



4、 利用PreparedStatement对象进行数据修改。

在SystemUserManager类中，新建一个modifyUserName方法，实现用户名称（username字段）的修改功能。并修改其main函数，将admin用户的名称改为：超级管理员。

【实验结果与分析】

A、 请提供方法代码和main函数代码。

```
1  public void modifyUserName(String userid, String newName) throws BaseException
   {
2      Connection conn = null;
3      try {
4          conn = DBUtil.getConnection();
5          String sql = "select * from BeanSystemUser where userid=?";
6          java.sql.PreparedStatement pst = conn.prepareStatement(sql);
7          pst.setString(1, userid);
8          java.sql.ResultSet rs = pst.executeQuery();
9          if (!rs.next()) throw new BusinessException("账号不存在");
10         rs.close();
11         pst.close();
12         sql = "update BeanSystemUser set username=? where userid=?";
13         pst = conn.prepareStatement(sql);
14         pst.setString(1, newName);
```

```
            pst.setString(2, userid);
            pst.execute();
            pst.close();
        } catch (SQLException e) {
            e.printStackTrace();
            throw new DbException(e);
        } finally {
            if (conn != null)
                try {
                    conn.close();
                } catch (SQLException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
        }
}
```

```
public static void main(String[] args) {
        BeanSystemUser test = new BeanSystemUser();
//        user.setUserid("admin");
//        user.setUsername("系统管理员");
//        user.setUsertype("管理员");
        try {
            new SystemUserManager().modifyUserName("admin", "超级管理员");
            test = new SystemUserManager().loadUser("admin");
            System.out.println("用户id: " + test.getUserid()
            + " 用户姓名: " + test.getUsername()
            + " 用户类型: " + test.getUsertype());
        } catch (BaseException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
}
```

```
259     public static void main(String[] args) {
260         BeanSystemUser test = new BeanSystemUser();
261 //        user.setUserid("admin");
262 //        user.setUsername("系统管理员");
263 //        user.setUsertype("管理员");
264         try {
265             new SystemUserManager().modifyUserName("admin", "超级管理员");
266             test = new SystemUserManager().loadUser("admin");
267             System.out.println("用户id: " + test.getUserid()
268             + " 用户姓名: " + test.getUsername()
269             + " 用户类型: " + test.getUsertype());
270         } catch (BaseException e) {
271             // TODO Auto-generated catch block
272             e.printStackTrace();
273         }
274     }
275 }
276
```

Problems Javadoc Declaration 控制台 ×

<已终止> SystemUserManager [Java 应用程序] C:\Users\Bexh0lder\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.f

用户id: admin 用户姓名: 超级管理员 用户类型: 管理员

```
mysql> select * from beansystemuser;
+--------+----------+-------+----------+---------------------+------------+
| userid | username | pwd   | usertype | createDate          | removeDate |
+--------+----------+-------+----------+---------------------+------------+
| admin  | 管理员   | admin | 管理员   | 2013-01-01 00:00:00 | NULL       |
+--------+----------+-------+----------+---------------------+------------+
1 row in set (0.00 sec)

mysql> select * from beansystemuser;
+--------+------------+-------+----------+---------------------+------------+
| userid | username   | pwd   | usertype | createDate          | removeDate |
+--------+------------+-------+----------+---------------------+------------+
| admin  | 超级管理员 | admin | 管理员   | 2013-01-01 00:00:00 | NULL       |
+--------+------------+-------+----------+---------------------+------------+
1 row in set (0.00 sec)
```

B、 思考：如果上述方法的返回值为布尔类型，即如果成功修改了用户名称，则返回true，如果用户不存在或修改失败返回false。应该如何完善代码。提示：主要statement或PreparedStatement对象的execute方法和executeUpdate方法的区别。

```
1  public boolean modifyUserName(String userid, String newName) throws
   BaseException {
2      Connection conn = null;
3      try {
4          conn = DBUtil.getConnection();
5  //         String sql = "select * from BeanSystemUser where userid=?";
6  //         java.sql.PreparedStatement pst = conn.prepareStatement(sql);
7  //         pst.setString(1, userid);
8  //         java.sql.ResultSet rs = pst.executeQuery();
9  //         if (!rs.next()) throw new BusinessException("账号不存在");
```

```
//            rs.close();
//            pst.close();
//            sql = "update BeanSystemUser set username=? where userid=?";
//            pst = conn.prepareStatement(sql);
//            pst.setString(1, newName);
//            pst.setString(2, userid);
//            pst.execute();
//            pst.close();
            String sql = "update BeanSystemUser set username= \""+ newName + "\" where userid=\"" + userid + "\"";
            java.sql.Statement st = conn.createStatement();
            int i = st.executeUpdate(sql);
            if(i <= 0) return false;
        } catch (SQLException e) {
            e.printStackTrace();
            throw new DbException(e);
        } finally {
            if (conn != null)
                try {
                    conn.close();
                } catch (SQLException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
        }
        return true;
    }
```

```
public static void main(String[] args) {
        BeanSystemUser test = new BeanSystemUser();
        try {
            List<BeanSystemUser> testList = new ArrayList<BeanSystemUser>();
            testList = new SystemUserManager().loadAllUsers(true);
            if(new SystemUserManager().modifyUserName("111", "超级管理员"))
                System.out.println("修改成功");
            else
                System.out.println("修改失败");
            for(int i = 0; i < testList.size(); ++i)
            {
                test = testList.get(i);
                System.out.println("用户id: " + test.getUserid()
                + " 用户姓名: " + test.getUsername()
```

```
            + " 用户类型: " + test.getUsertype());
        }
        if(new SystemUserManager().modifyUserName("admin", "超级管理员"))
            System.out.println("修改成功");
        else
            System.out.println("修改失败");
        testList = new SystemUserManager().loadAllUsers(true);
        for(int i = 0; i < testList.size(); ++i)
        {
            test = testList.get(i);
            System.out.println("用户id: " + test.getUserid()
            + " 用户姓名: " + test.getUsername()
            + " 用户类型: " + test.getUsertype());
        }
    } catch (BaseException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}
```

executeUpdate 的返回值是一个整数（int），指示受影响的行数（即更新计数），如果没有修改成功则返回值不会大于0，所以可以通过返回值来确定是否修改成功

```
    public static void main(String[] args) {
        BeanSystemUser test = new BeanSystemUser();
        try {
            List<BeanSystemUser> testList = new ArrayList<BeanSystemUser>();
            testList = new SystemUserManager().loadAllUsers(true);
            if(new SystemUserManager().modifyUserName("111", "超级管理员"))
                System.out.println("修改成功");
            else
                System.out.println("修改失败");
            for(int i = 0; i < testList.size(); ++i)
            {
                test = testList.get(i);
                System.out.println("用户id: " + test.getUserid()
                + " 用户姓名: " + test.getUsername()
                + " 用户类型: " + test.getUsertype());
            }
            if(new SystemUserManager().modifyUserName("admin", "超级管理员"))
                System.out.println("修改成功");
            else
                System.out.println("修改失败");
            testList = new SystemUserManager().loadAllUsers(true);
            for(int i = 0; i < testList.size(); ++i)
            {
                test = testList.get(i);
                System.out.println("用户id: " + test.getUserid()
                + " 用户姓名: " + test.getUsername()
                + " 用户类型: " + test.getUsertype());
            }
        } catch (BaseException e) {
            // TODO Auto-generated catch block
```

```
Problems  Javadoc  Declaration  控制台 ×
<已终止> SystemUserManager [Java 应用程序] C:\Users\Bexh0lder\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v2022020
修改失败
用户id: admin 用户姓名: 管理员 用户类型: 管理员
修改成功
用户id: admin 用户姓名: 超级管理员 用户类型: 管理员
```

5、 Delete语句的执行。修改用户管理类中的用户删除方法，用删除数据库表中数据的形式代替现有软删除模式。

【实验结果与分析】

A、修改后的sql语句部分是。

```
public void deleteUser(String userid) throws BaseException {
        Connection conn = null;
        try {
            conn = DBUtil.getConnection();
            String sql = "select removeDate from BeanSystemUser 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("登陆账号不存在或已被删除");
            rs.close();
            pst.close();
            //----------------修改后的sql语句--------------------
            sql = "delete from beansystemuser where userid=?";
            //---------------------------------------------------------
            pst = conn.prepareStatement(sql);
            pst.setString(1, userid);
            pst.execute();
            pst.close();
        } catch (SQLException e) {
            e.printStackTrace();
            throw new DbException(e);
        } finally {
            if (conn != null)
                try {
                    conn.close();
                } catch (SQLException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
        }
}
```

```
293    public static void main(String[] args) {
294        try {
295            List<BeanSystemUser> testList = new ArrayList<BeanSystemUser>();
296            BeanSystemUser newUser = new BeanSystemUser();
297            BeanSystemUser test = new BeanSystemUser();
298            newUser.setUsername("bex");
299            newUser.setUserid("bex");
300            newUser.setUsertype("管理员");
301            new SystemUserManager().createUser(newUser);
302            testList = new SystemUserManager().loadAllUsers(true);
303            for(int i = 0; i < testList.size(); ++i)
304            {
305                test = testList.get(i);
306                System.out.println("用户id: " + test.getUserid()
307                + " 用户姓名: " + test.getUsername()
308                + " 用户类型: " + test.getUsertype());
309            }
310            new SystemUserManager().deleteUser("bex");
311            testList = new SystemUserManager().loadAllUsers(true);
312            for(int i = 0; i < testList.size(); ++i)
313            {
314                test = testList.get(i);
315                System.out.println("用户id: " + test.getUserid()
316                + " 用户姓名: " + test.getUsername()
317                + " 用户类型: " + test.getUsertype());
318            }
319        } catch (BaseException e) {
320            // TODO Auto-generated catch block
321            e.printStackTrace();
322        }
```

Problems @ Javadoc Declaration 控制台 ×

```
<已终止> SystemUserManager [Java 应用程序] C:\Users\Bexh0lder\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win
用户id: admin 用户姓名: 超级管理员 用户类型: 管理员
用户id: bex 用户姓名: bex 用户类型: 管理员
用户id: admin 用户姓名: 超级管理员 用户类型: 管理员
```

B、如果对删除函数进行限制，要求不能删除已经有过借阅操作的用户。应如何修改代码。提示：可参考读者管理类中的读者类别删除方法。

```
1  public void deleteUser(String userid) throws BaseException {
2      Connection conn = null;
3      try {
4          conn = DBUtil.getConnection();
5          String sql = "select username from BeanSystemUser where userid=?";
6          java.sql.PreparedStatement pst = conn.prepareStatement(sql);
7          pst.setString(1, userid);
8          java.sql.ResultSet rs = pst.executeQuery();
9          if (!rs.next()) throw new BusinessException("账号不存在或已被删除");
10         String deleteName = rs.getString(1);
11         System.out.println(deleteName);
12         rs.close();
13         pst.close();
14         sql = "select count(*) from beanbooklendrecord where lendOperUserid=?";
15         pst = conn.prepareStatement(sql);
16         pst.setString(1, userid);
17         rs = pst.executeQuery();
```

```
            rs.next();
            int n = rs.getInt(1);
            if (n > 0) throw new BusinessException(deleteName + "已经执行过" + n + "个借阅操作，不能删除");
            sql = "delete from beansystemuser where userid=?";
            pst = conn.prepareStatement(sql);
            pst.setString(1, userid);
            pst.execute();
            pst.close();
        } catch (SQLException e) {
            e.printStackTrace();
            throw new DbException(e);
        } finally {
            if (conn != null)
                try {
                    conn.close();
                } catch (SQLException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
        }
}
```

```
public static void main(String[] args) {
        try {
            List<BeanSystemUser> testList = new ArrayList<BeanSystemUser>();
            BeanSystemUser newUser = new BeanSystemUser();
            BeanSystemUser test = new BeanSystemUser();
            newUser.setUsername("bex");
            newUser.setUserid("bex");
            newUser.setUsertype("管理员");
            new SystemUserManager().createUser(newUser);
            testList = new SystemUserManager().loadAllUsers(true);
            for(int i = 0; i < testList.size(); ++i)
            {
                test = testList.get(i);
                System.out.println("用户id: " + test.getUserid()
                + " 用户姓名: " + test.getUsername()
                + " 用户类型: " + test.getUsertype());
            }
            new SystemUserManager().deleteUser("bex");
            testList = new SystemUserManager().loadAllUsers(true);
```

```
            for(int i = 0; i < testList.size(); ++i)
            {
                test = testList.get(i);
                System.out.println("用户id: " + test.getUserid()
                + " 用户姓名: " + test.getUsername()
                + " 用户类型: " + test.getUsertype());
            }
            new SystemUserManager().deleteUser("admin");
        } catch (BaseException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
```

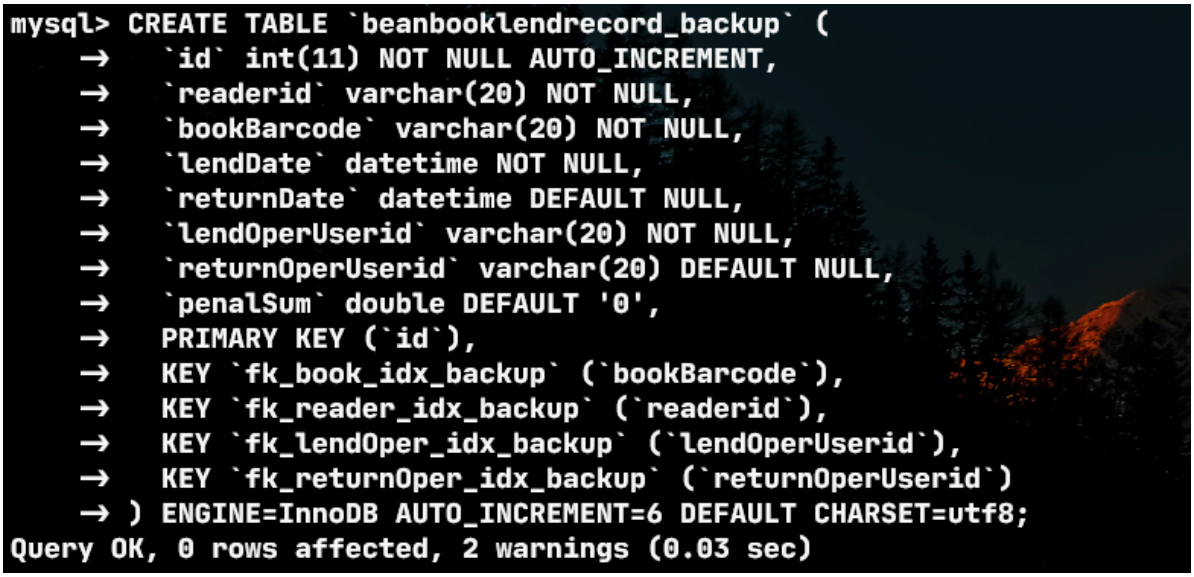
```
    public static void main(String[] args) {
        try {
            List<BeanSystemUser> testList = new ArrayList<BeanSystemUser>();
            BeanSystemUser newUser = new BeanSystemUser();
            BeanSystemUser test = new BeanSystemUser();
            newUser.setUsername("bex");
            newUser.setUserid("bex");
            newUser.setUsertype("管理员");
            new SystemUserManager().createUser(newUser);
            testList = new SystemUserManager().loadAllUsers(true);
            for(int i = 0; i < testList.size(); ++i)
            {
                test = testList.get(i);
                System.out.println("用户id: " + test.getUserid()
                + " 用户姓名: " + test.getUsername()
                + " 用户类型: " + test.getUsertype());
            }
            new SystemUserManager().deleteUser("bex");
            testList = new SystemUserManager().loadAllUsers(true);
            for(int i = 0; i < testList.size(); ++i)
            {
                test = testList.get(i);
                System.out.println("用户id: " + test.getUserid()
                + " 用户姓名: " + test.getUsername()
                + " 用户类型: " + test.getUsertype());
            }
            new SystemUserManager().deleteUser("admin");
        } catch (BaseException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
}

Problems  Javadoc  Declaration  控制台 ×
<已终止> SystemUserManager [Java 应用程序] C:\Users\Bexh0lder\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.2.v20220201-1208
用户id: admin 用户姓名: 超级管理员 用户类型: 管理员
用户id: bex 用户姓名: bex 用户类型: 管理员
bex
用户id: admin 用户姓名: 超级管理员 用户类型: 管理员
超级管理员
cn.edu.zucc.booklib.util.BusinessException: 超级管理员已经执行过18个借阅操作，不能删除
        at cn.edu.zucc.booklib.control.SystemUserManager.deleteUser(SystemUserManager.java:248)
        at cn.edu.zucc.booklib.control.SystemUserManager.main(SystemUserManager.java:328)
```

> **6、 (修改)在数据库中建立一张BeanBookLendRecord_backup表，用于保存已经归还图书的借阅记录。其表结构与BeanBookLendRecord表完全一致。要求在借阅管理类中，增加方法，实现已经归还数据的备份功能（备份完成后，在原表中删除备份成功的数据）。提示：注意事务控制。**

【实验结果与分析】

A 请提供备份表的建表语句

```
CREATE TABLE `beanbooklendrecord_backup` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `readerid` varchar(20) NOT NULL,
  `bookBarcode` varchar(20) NOT NULL,
  `lendDate` datetime NOT NULL,
  `returnDate` datetime DEFAULT NULL,
  `lendOperUserid` varchar(20) NOT NULL,
  `returnOperUserid` varchar(20) DEFAULT NULL,
  `penalSum` double DEFAULT '0',
  PRIMARY KEY (`id`),
  KEY `fk_book_idx_backup` (`bookBarcode`),
  KEY `fk_reader_idx_backup` (`readerid`),
  KEY `fk_lendOper_idx_backup` (`lendOperUserid`),
  KEY `fk_returnOper_idx_backup` (`returnOperUserid`)
) ENGINE=InnoDB AUTO_INCREMENT=6 DEFAULT CHARSET=utf8;
```



B 请提供备份函数代码

```
//三金指定very优雅版
```

```
    public void BackupReturnedRecord()throws DbException, SQLException{
        Connection conn = null;
        String sql = null;
        try {
            conn = DBUtil.getConnection();
            //关闭事务自动提交
            conn.setAutoCommit(false);

            sql = "insert beanbooklendrecord_backup select * from beanbooklendrecord where returnDate is not null";
            java.sql.PreparedStatement pst = conn.prepareStatement(sql);
            pst.execute();
            pst.close();
            sql = "delete from beanbooklendrecord where id in (select id from beanbooklendrecord_backup)";
            pst = conn.prepareStatement(sql);
            pst.execute();
            pst.close();
            //提交存档，如果第一步成功而第二步失败时方便回到第一步执行之前
            conn.commit();
        }catch(SQLException e) {
            //出错就回滚到第一步之前
            conn.rollback();
            e.printStackTrace();
        }finally {
            if (conn != null)
                try {
                    conn.close();
                } catch (SQLException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
    }
}
```

备份前

```
mysql> select * from beanbooklendrecord;
+----+----------+-------------+---------------------+---------------------+---------------+-----------------+----------+
| id | readerid | bookBarcode | lendDate            | returnDate          | lendOperUserid | returnOperUserid | penalSum |
+----+----------+-------------+---------------------+---------------------+---------------+-----------------+----------+
| 26 | 32001272 | barcode1    | 2022-05-15 14:15:22 | 2022-05-15 14:30:05 | admin         | admin           |        0 |
| 27 | 32001272 | barcode4    | 2022-05-15 14:15:26 | 2022-05-15 14:37:49 | admin         | admin           |        0 |
| 28 | 32001002 | barcode2    | 2022-05-15 14:18:02 | 2022-05-15 14:37:53 | admin         | admin           |        0 |
| 29 | 32001003 | barcode6    | 2022-05-15 14:18:10 | NULL                | admin         | NULL            |        0 |
| 30 | 32001001 | barcode9    | 2022-05-15 14:18:16 | NULL                | admin         | NULL            |        0 |
| 31 | 32001003 | barcode5    | 2022-05-15 14:18:24 | NULL                | admin         | NULL            |        0 |
+----+----------+-------------+---------------------+---------------------+---------------+-----------------+----------+
6 rows in set (0.00 sec)

mysql> select * from beanbooklendrecord_backup;
+----+----------+-------------+---------------------+---------------------+---------------+-----------------+----------+
| id | readerid | bookBarcode | lendDate            | returnDate          | lendOperUserid | returnOperUserid | penalSum |
+----+----------+-------------+---------------------+---------------------+---------------+-----------------+----------+
| 23 | 32001272 | barcode1    | 2022-05-09 22:55:23 | 2022-05-15 14:15:05 | admin         | admin           |        0 |
| 25 | 32001272 | barcode2    | 2022-05-15 14:14:54 | 2022-05-15 14:15:13 | admin         | admin           |        0 |
+----+----------+-------------+---------------------+---------------------+---------------+-----------------+----------+
2 rows in set (0.00 sec)
```

备份后

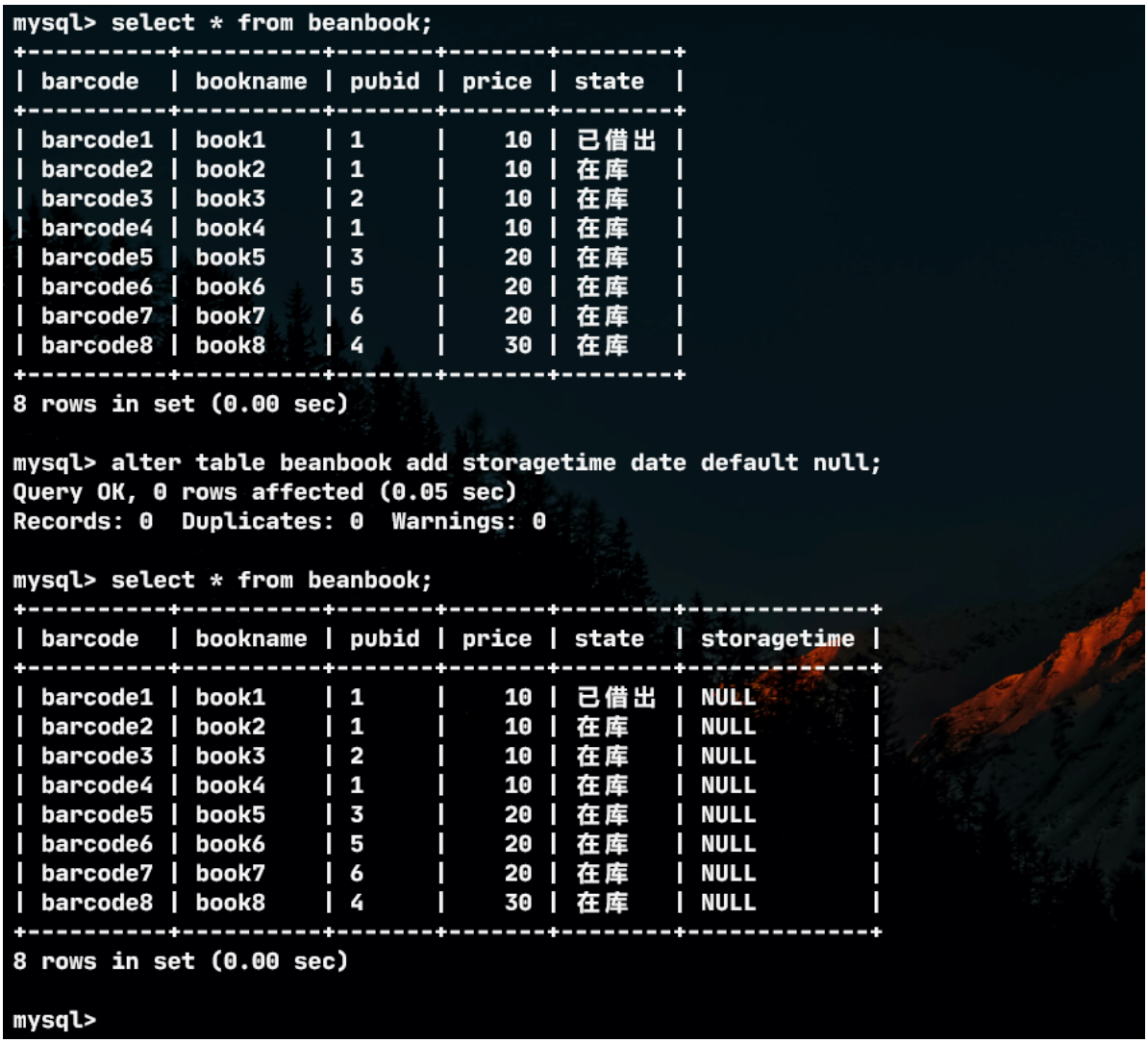
```
mysql> select * from beanbooklendrecord;
+----+----------+-------------+---------------------+------------+---------------+-----------------+----------+
| id | readerid | bookBarcode | lendDate            | returnDate | lendOperUserid | returnOperUserid | penalSum |
+----+----------+-------------+---------------------+------------+---------------+-----------------+----------+
| 29 | 32001003 | barcode6    | 2022-05-15 14:18:10 | NULL       | admin         | NULL            |        0 |
| 30 | 32001001 | barcode9    | 2022-05-15 14:18:16 | NULL       | admin         | NULL            |        0 |
| 31 | 32001003 | barcode5    | 2022-05-15 14:18:24 | NULL       | admin         | NULL            |        0 |
+----+----------+-------------+---------------------+------------+---------------+-----------------+----------+
3 rows in set (0.00 sec)

mysql> select * from beanbooklendrecord_backup;
+----+----------+-------------+---------------------+---------------------+---------------+-----------------+----------+
| id | readerid | bookBarcode | lendDate            | returnDate          | lendOperUserid | returnOperUserid | penalSum |
+----+----------+-------------+---------------------+---------------------+---------------+-----------------+----------+
| 23 | 32001272 | barcode1    | 2022-05-09 22:55:23 | 2022-05-15 14:15:05 | admin         | admin           |        0 |
| 25 | 32001272 | barcode2    | 2022-05-15 14:14:54 | 2022-05-15 14:15:13 | admin         | admin           |        0 |
| 26 | 32001272 | barcode1    | 2022-05-15 14:15:22 | 2022-05-15 14:30:05 | admin         | admin           |        0 |
| 27 | 32001272 | barcode4    | 2022-05-15 14:15:26 | 2022-05-15 14:37:49 | admin         | admin           |        0 |
| 28 | 32001002 | barcode2    | 2022-05-15 14:18:02 | 2022-05-15 14:37:53 | admin         | admin           |        0 |
+----+----------+-------------+---------------------+---------------------+---------------+-----------------+----------+
5 rows in set (0.00 sec)

mysql>
```

7、如果需要记录图书的入库时间（需要包含时分秒），应如何修改数据库表结构和相关代码？

【实验结果与分析】

添加字段

```
1 alter table beanbook add storagetime datetime default null;
```



```
mysql> select * from beanbook;
+----------+----------+-------+-------+--------+
| barcode  | bookname | pubid | price | state  |
+----------+----------+-------+-------+--------+
| barcode1 | book1    | 1     |    10 | 已借出 |
| barcode2 | book2    | 1     |    10 | 在库   |
| barcode3 | book3    | 2     |    10 | 在库   |
| barcode4 | book4    | 1     |    10 | 在库   |
| barcode5 | book5    | 3     |    20 | 在库   |
| barcode6 | book6    | 5     |    20 | 在库   |
| barcode7 | book7    | 6     |    20 | 在库   |
| barcode8 | book8    | 4     |    30 | 在库   |
+----------+----------+-------+-------+--------+
8 rows in set (0.00 sec)

mysql> alter table beanbook add storagetime date default null;
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> select * from beanbook;
+----------+----------+-------+-------+--------+-------------+
| barcode  | bookname | pubid | price | state  | storagetime |
+----------+----------+-------+-------+--------+-------------+
| barcode1 | book1    | 1     |    10 | 已借出 | NULL        |
| barcode2 | book2    | 1     |    10 | 在库   | NULL        |
| barcode3 | book3    | 2     |    10 | 在库   | NULL        |
| barcode4 | book4    | 1     |    10 | 在库   | NULL        |
| barcode5 | book5    | 3     |    20 | 在库   | NULL        |
| barcode6 | book6    | 5     |    20 | 在库   | NULL        |
| barcode7 | book7    | 6     |    20 | 在库   | NULL        |
| barcode8 | book8    | 4     |    30 | 在库   | NULL        |
+----------+----------+-------+-------+--------+-------------+
8 rows in set (0.00 sec)

mysql>
```

修改代码

- BookManager.createBook(BeanBook b)

```
1  public void createBook(BeanBook b) throws BaseException {
2
3
4      if (b.getBarcode() == null || "".equals(b.getBarcode()) ||
   b.getBarcode().length() > 20) {
5          throw new BusinessException("条码必须是1-20个字");
6      }
7      if (b.getBookname() == null || "".equals(b.getBookname()) ||
   b.getBookname().length() > 50) {
8          throw new BusinessException("图书名称必须是1-50个字");
9      }
10     Connection conn = null;
11     try {
12         conn = DBUtil.getConnection();
13         String sql = "select * from BeanBook where barcode=?";
14         java.sql.PreparedStatement pst = conn.prepareStatement(sql);
```

```
            pst.setString(1, b.getBarcode());
            java.sql.ResultSet rs = pst.executeQuery();
            if (rs.next()) throw new BusinessException("条码已经被占用");
            rs.close();
            pst.close();
            sql = "insert into BeanBook(barcode,bookname,pubid,price,state,storagetime) values(?,?,?,?,'在库',?)";//修改处
            pst = conn.prepareStatement(sql);
            pst.setString(1, b.getBarcode());
            pst.setString(2, b.getBookname());
            pst.setString(3, b.getPubid());
            pst.setDouble(4, b.getPrice());
            pst.setTimestamp(5, new java.sql.Timestamp(System.currentTimeMillis()));//修改处
            pst.execute();
            pst.close();
        } catch (SQLException e) {
            e.printStackTrace();
            throw new DbException(e);
        } finally {
            if (conn != null)
                try {
                    conn.close();
                } catch (SQLException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
        }
    }
```

- BookManager.loadBook(String barcode)

```
public BeanBook loadBook(String barcode) throws DbException {
        Connection conn = null;
        try {
            conn = DBUtil.getConnection();
            String sql = "select b.barcode,b.bookname,b.pubid,b.price,b.state,b.storagetime,p.publishername " + //修改处
                    " from beanbook b left outer join beanpublisher p on (b.pubid=p.pubid)" +
```

```
                " where  b.barcode=? ";
            java.sql.PreparedStatement pst = conn.prepareStatement(sql);
            pst.setString(1, barcode);
            java.sql.ResultSet rs = pst.executeQuery();
            if (rs.next()) {
                BeanBook b = new BeanBook();
                b.setBarcode(rs.getString(1));
                b.setBookname(rs.getString(2));
                b.setPubid(rs.getString(3));
                b.setPrice(rs.getDouble(4));
                b.setState(rs.getString(5));
                b.setStorageTime(rs.getDate(6));//修改处
                b.setPubName(rs.getString(7));
                return b;
            }
        } catch (SQLException e) {
            e.printStackTrace();
            throw new DbException(e);
        } finally {
            if (conn != null)
                try {
                    conn.close();
                } catch (SQLException e) {
                    // TODO Auto-generated catch block
                    e.printStackTrace();
                }
        }
        return null;
    }
```

- BeanBook

```
package cn.edu.zucc.booklib.model;

import java.util.Date;

public class BeanBook {
    private String barcode;
    private String bookname;
    private String pubid;
    private double price;
    private String state;//状态：已借出,在库,已删除
```

```
11      private Date storageTime;
12
13      private String pubName;//出版社名称，在图书表中不存储名称，只存储出版社
        ID
14
15      public String getBarcode() {
16          return barcode;
17      }
18
19      public void setBarcode(String barcode) {
20          this.barcode = barcode;
21      }
22
23      public String getBookname() {
24          return bookname;
25      }
26
27      public void setBookname(String bookname) {
28          this.bookname = bookname;
29      }
30
31      public String getPubid() {
32          return pubid;
33      }
34
35      public void setPubid(String pubid) {
36          this.pubid = pubid;
37      }
38
39      public double getPrice() {
40          return price;
41      }
42
43      public void setPrice(double price) {
44          this.price = price;
45      }
46
47      public String getState() {
48          return state;
49      }
50
51      public void setState(String state) {
```

```
52          this.state = state;
53      }
54
55      //------------修改处----------------
56      public void setStorageTime(Date storageTime) {
57          this.storageTime = storageTime;
58      }
59
60      public Date getStroageTime() {
61          return storageTime;
62      }
63      //----------------------------------
64
65      public String getPubName() {
66          return pubName;
67      }
68
69      public void setPubName(String pubName) {
70          this.pubName = pubName;
71      }
72  }
```

```
mysql> select * from beanbook;
+-----------+----------+-------+-------+--------+---------------------+
| barcode   | bookname | pubid | price | state  | storagetime         |
+-----------+----------+-------+-------+--------+---------------------+
| barcode1  | book1    | 1     |    10 | 已借出 | NULL                |
| barcode10 | book10   | 4     |    90 | 在库   | 2022-05-13 21:30:08 |
| barcode2  | book2    | 1     |    10 | 在库   | NULL                |
| barcode3  | book3    | 2     |    10 | 在库   | NULL                |
| barcode4  | book4    | 1     |    10 | 在库   | NULL                |
| barcode5  | book5    | 3     |    20 | 在库   | NULL                |
| barcode6  | book6    | 5     |    20 | 在库   | NULL                |
| barcode7  | book7    | 6     |    20 | 在库   | NULL                |
| barcode8  | book8    | 4     |    30 | 在库   | NULL                |
| barcode9  | book9    | 1     |    10 | 在库   | 2022-05-13 21:24:28 |
+-----------+----------+-------+-------+--------+---------------------+
10 rows in set (0.00 sec)

mysql>
```