

# “银行业务管理系统”

## 系统设计与实现报告

姓名：章东泉

学号：PB17121706

计算机科学与技术学院

中国科学技术大学

2020 年 6 月

## 目 录

<b>1</b>	<b>概述.....</b>	<b>1</b>
1.1	系统目标.....	1
1.2	需求说明.....	1
1.3	本报告的主要贡献.....	1
<b>2</b>	<b>总体设计.....</b>	<b>2</b>
2.1	系统模块结构.....	2
2.2	系统工作流程.....	2
2.3	数据库设计.....	2
<b>3</b>	<b>详细设计.....</b>	<b>3</b>
3.1	登录模块.....	3
3.2	主菜单模块.....	3
3.3	二级菜单模块.....	3
3.4	功能模块.....	3
<b>4</b>	<b>实现与测试.....</b>	<b>7</b>
4.1	实现结果.....	7
4.2	测试结果.....	10
<b>5</b>	<b>总结与讨论.....</b>	<b>14</b>

# 1 概述

## 1.1 系统目标

开发功能完备的银行业务管理系统。

## 1.2 需求说明

### (1) 客户管理

提供客户所有信息的增删改查功能；如果客户存在着关联账户或贷款记录，则不允许删除。

### (2) 账户管理

提供账户开户、销户、修改、查询功能；账户号不允许修改。

### (3) 贷款管理

提供贷款信息的增删查功能，提供贷款发放功能；贷款信息一旦发放成功后不允许修改；要求能查询每笔贷款的当前状态；处于发放中状态的贷款记录不允许删除。

### (4) 业务统计

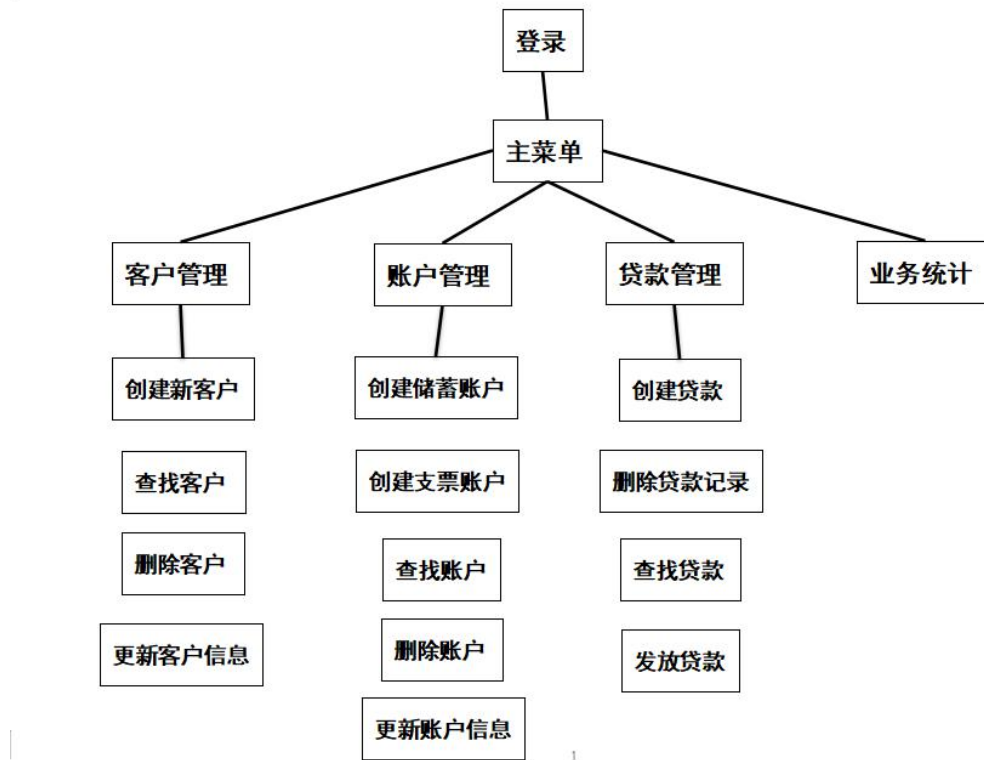
按业务分类和时间统计各个支行业务总金额和客户数，用表格和图两种可视化方式。

## 1.3 本报告的主要贡献

描述了系统的实现结果和实现细节。

## 2 总体设计

### 2.1 系统模块结构

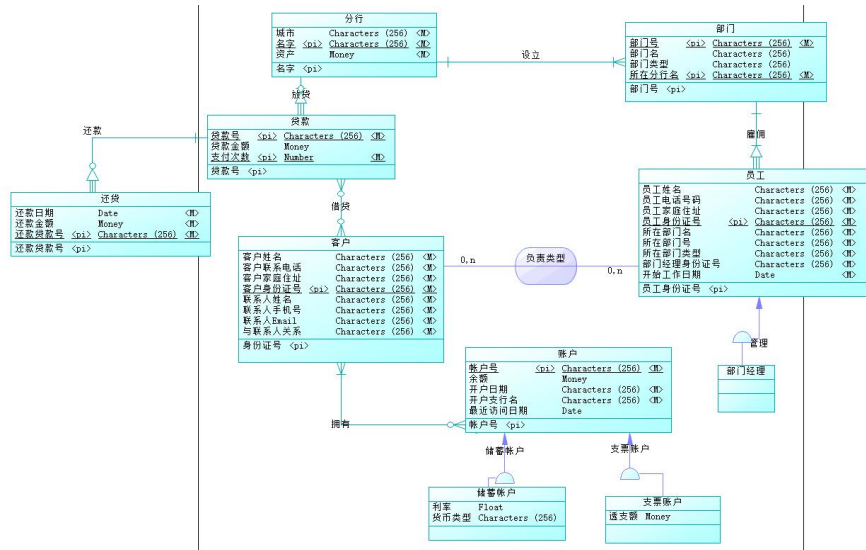


### 2.2 系统工作流程

登录→选择需要使用的服务功能→退出系统

每次进入新的网页都会重连 MySQL。

## 2.3 数据库设计



详细数据库逻辑设计见附件的 demo.sql，主要参照助教提供的 demo.sql 完成。

## 3 详细设计

### 3.1 登录模块

**输入：**数据库所在主机地址、访问的数据库名称、用户名、用户密码

**输出：**重定向至主菜单

### 3.2 主菜单模块

**输入：**用户选择二级菜单按钮

**输出：**重定向至用户选择的二级菜单

### 3.3 二级菜单模块

**输入：**用户选择具体功能按钮

**输出：**重定向至用户选择的功能

### 3.3.1 客户管理菜单

输入：用户选择具体功能按钮

输出：重定向至用户选择的功能

### 3.3.2 账户管理菜单

输入：用户选择具体功能按钮

输出：重定向至用户选择的功能

### 3.3.3 贷款管理菜单

输入：用户选择具体功能按钮

输出：重定向至用户选择的功能

### 3.3.4 业务统计菜单

输入：用户选择具体功能按钮

输出：重定向至用户选择的功能

## 3.4 功能模块

输入：用户输入表单数据

输出：数据库操作结果，并重定向至上级菜单

### 3.4.1 创建新客户

输入：客户名、客户号、客户电话、住址、联系人电话、联系人姓名、联系人邮箱、

与联系人关系

输出：MySQL 插入操作，重定向至上级菜单

### 3.4.2 删除客户

输入：客户号

输出：MySQL 删除操作，重定向至上级菜单。如果客户号有相连的账户或贷款，由于外键存在不允许修改

### 3.4.3 查找客户

输入：无

输出：返回所有客户的客户号、客户名、联系电话结果，同时提供详细搜索页面

### 3.4.4 修改客户信息

输入：客户号

输出：返回该客户号对应客户的除客户号以外属性，并提供新信息的输入口

### 3.4.5 创建储蓄账户

输入：客户号、账户号、金额、创建时间、利率、存储类型

输出：MySQL 插入操作，同时向 accounts, saveacc, cusforacc 三个表项插入数据

### 3.4.6 创建支票账户

输入：客户号、账户号、金额、创建时间、透支额

输出：MySQL 插入操作，同时向 accounts, checkacc, cusforacc 三个表项插入数

据

### 3.4.7 查找账户

输入：无

输出：返回所有账户的账户号、客户号、账户类型

### 3.4.8 删除账户

输入：账户号

输出：MySQL 删除操作，重定向至上级菜单

### 3.4.9 修改账户信息

输入：账户号

输出：返回该账户号对应账户的除账户号以外属性，并提供新信息的输入口

### 3.4.10 发放贷款

输入：贷款号、客户号、金额

输出：MySQL 插入操作，同时向 loan, cusforloan 两个表项插入数据

### 3.4.11 删除贷款

输入：贷款号

输出：MySQL 删除操作，重定向至上级菜单

### 3.4.12 查找贷款



**输入：**无

**输出：**返回所有贷款信息

### 3.4.13 还款

**输入：**贷款号、客户号、发放金额、发放日期

**输出：**向 payinfo 表插入表项，使用触发器自动修改贷款状态

### 3.4.13 业务统计

**输入：**储蓄/贷款，时间限度

**输出：**可视化表格和图表

## 4 实现与测试

### 4.1 实现结果

由于实现方法和效果类似，增删查改操作各给出一个样例，插入数据后的结果在“测试结果”部分给出：

#### 4.1.1 插入客户信息

图 4.1.1-1 插入客户信息界面

**Insert**

cusID

cusname

cusphone

address

contact\_phone

contact\_name

contact\_Email

relation

Insert

代码 4.1.1-1 插入客户信息代码实现

```
def db_InsertCustomer(db, cusID, cusname, cusphone, address, contact_phone, contact_name, contact_Email, relation):
    cursor = db.cursor()
    try:
        sql = "INSERT INTO customer VALUES('%s', '%s', '%s', '%s', '%s', '%s', '%s', '%s', null, null)" % \
            (cusID, cusname, cusphone, address, contact_phone, contact_name, contact_Email, relation)
        cursor.execute(sql)
        db.commit()
    except:
        db.rollback()
    cursor.close()
```

## 4.1.2 查找客户信息

图 4.1.2-1 查找客户信息界面

**All records in Table -**

cusID	cusname	cusphone
11	MoSalah	11
26	Robertson	26
7	J.Milner	7

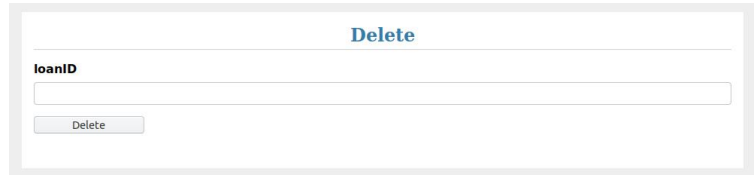
View Contact      Back to Menu

代码 4.1.2-1 查找客户信息代码实现

```
def db_SearchCustomer(db):
    cursor = db.cursor()
    cursor.execute("select * from customer")
    tabs = cursor.fetchall()
    res = list()
    for tab in tabs:
        cusID = tab[0]
        cusname = tab[1]
        cusphone = tab[2]
        res.append((cusID, cusname, cusphone))
    cursor.close()
    return res
```

### 4.1.3 删除贷款信息

图 4.1.3-1 删除贷款信息界面

A web form titled "Delete" in blue. It contains a label "loanID" above a text input field. Below the input field is a "Delete" button.

代码 4.1.3-1 删除贷款信息代码实现

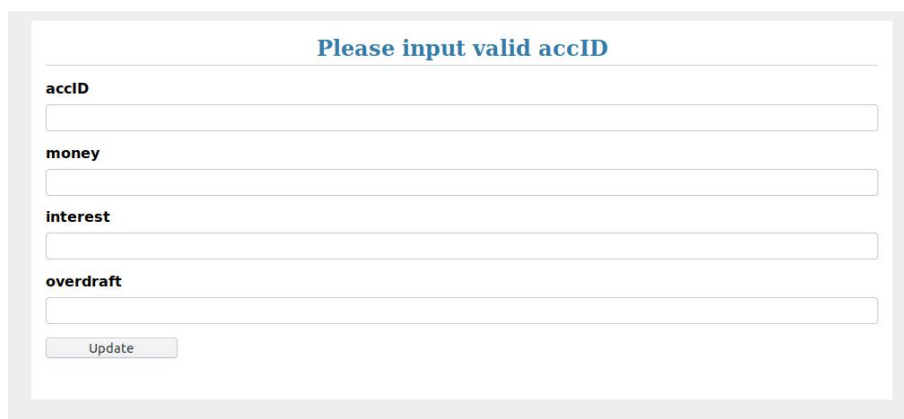
```
def db_DeleteLoan(db, loanID):  
    cursor = db.cursor()  
    try:  
        sql = "DELETE FROM cusforloan WHERE loanID = '%s'" %(loanID)  
        cursor.execute(sql)  
        sql = "DELETE FROM loan WHERE loanID = '%s'" %(loanID)  
        cursor.execute(sql)  
        db.commit()  
    except:  
        db.rollback()  
        cursor.close()
```

贷款删除约束使用触发器实现，如果贷款状态处于发放中则删除操作无效。

### 4.1.4 修改账户信息

提供了给账户修改金额，给储蓄账户修改利率，给支票账户修改额度功能，其中账户类型通过输入的账户号自动判断。

图 4.1.4-1 更新账户数据

A web form titled "Please input valid accID" in blue. It contains four labels: "accID", "money", "interest", and "overdraft", each followed by a text input field. At the bottom is an "Update" button.

### 4.1.5 发放贷款

在每次还款后，会有触发器自动更新贷款的 state，在 4.2 节中演示：

图 4.1.5-1 发放贷款



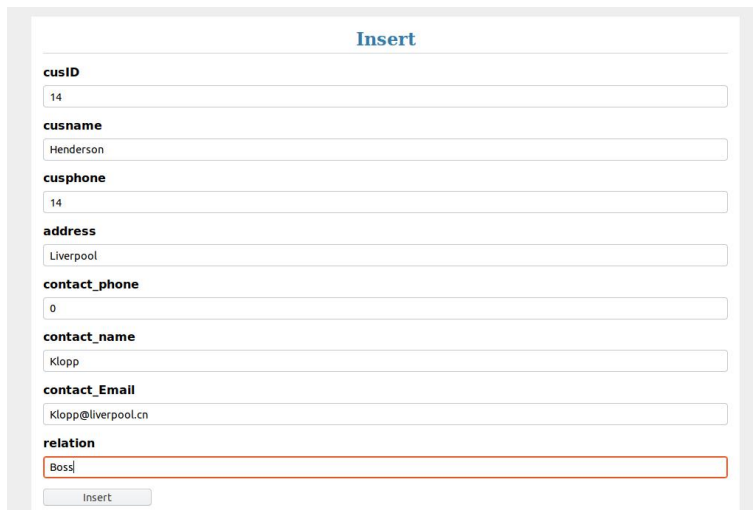
The screenshot shows a web form titled "Pay Loan". It has four input fields labeled "loanID", "cusID", "money", and "paytime". Below these fields is a button labeled "Pay".

## 4.2 测试结果

由于实现方法和效果类似，增删查改操作各给出一个样例：

### 4.2.1 插入客户信息

图 4.2.1-1 插入客户信息输入



The screenshot shows a web form titled "Insert". It has eight input fields labeled "cusID", "cusname", "cusphone", "address", "contact\_phone", "contact\_name", "contact\_Email", and "relation". Below these fields is a button labeled "Insert".

图 4.2.1-1 插入客户信息结果

All records in Table -

cusID	cusname	cusphone
11	MoSalah	11
14	Henderson	14
26	Robertson	26
7	J.Milner	7

[View Contact](#) [Back to Menu](#)

4.2.2 查找账户信息

图 4.2.2-1 查找客户信息结果

All records in Table -

cusID	accID	AccType
0	7	储蓄账户
1	14	储蓄账户
2	14	支票账户

[Back to Menu](#)

4.2.3 删除账户信息

在上面的结果中删除 0 号记录：

图 4.2.3-1 删除账户信息结果

All records in Table -

accID	cusID	AccType
1	14	储蓄账户
2	14	支票账户

[Back to Menu](#)

4.2.4 修改客户信息

比如，将 cusID=7 的客户名从 J.Milner 改为 Milner:

图 4.2.4-1 更新账户信息表单

### Please input valid cusID

**cusID**

**cusname**

**cusphone**

**address**

**contact\_phone**

**contact\_name**

**contact\_Email**

**relation**

图 4.2.4-2 更新账户信息结果

All records in Table -		
cusID	cusname	cusphone
11	MoSalah	11
14	Henderson	14
26	Robertson	26
7	Milner	7

#### 4.2.5 发放贷款

首先给客户号 14 号的 Henderson 发放一笔 500 元贷款：

图 4.2.5-1 发放贷款

发放后可以看到贷款处于未发放状态：

图 4.2.5-2 贷款未发放

**All records in Table - Loan**

loanID	cusID	money	state
0	7	100.0	2
3	14	500.0	0

[Back to Menu](#)

测试分一笔 200 一笔 300 发放贷款：

图 4.2.5-3 发放第一笔贷款

**Pay Loan**

**loanID**

**cusID**

**money**

**paytime**

[Pay](#)

此时 3 号贷款的状态变成 1 “发放中”

图 4.2.5-4 贷款发放中

**All records in Table - Loan**

loanID	cusID	money	state
0	7	100.0	2
3	14	500.0	1

[Back to Menu](#)

再发放第二笔贷款：

图 4.2.5-5 发放第二笔贷款

**Pay Loan**

**loanID**

**cusID**

**money**

**paytime**

[Pay](#)

此时 3 号贷款状态变为 2 “已全部发放”

图 4.2.5-6 贷款已全部发放

All records in Table - Loan

loanID	cusID	money	state
0	7	100.0	2
3	14	500.0	2

[Back to Menu](#)

## 5 总结与讨论

这次的实验是在 Ubuntu 环境下完成的，因为 Ubuntu 连接 MySQL 需要 sudo 的权限，所以一开始使用 pymysql 连接始终失败。这应该是整个实验过程中耗时最长的坑。

因为第一个实验是我直接在 Ubuntu 命令行完成的，所以通过实验 3 不仅学习了使用 Python 连接 MySQL 数据库的方法，也了解了十分好用的 Flask，能够让我在前段几乎没有花费时间。能够使用 Python 直接处理前端还是很方便的。

由于实验开始的比较迟，在一些功能和视图的完善方面做的还不到位，UI 也有点点丑。我会在提交实验报告以后继续进行代码的完善，在验收时展现一个更好的作品。

感谢助教在设计实验时的努力！