**银行业务管理系统数据库设计**

PB20111686 黄瑞轩

1. **概念模型设计**
   1. 实体设计

根据需求调研，设计的实体及其相关属性如下，画下划线的属性为主属性：

* 支行：支行名字、所在城市、管理的资产
* 部门：部门号、部门名称、部门类型
* 客户：客户身份证号、客户姓名、客户电话、客户住址、联系人姓名、联系人电话、联系人Email、与联系人关系
* 员工：员工身份证号、员工姓名、员工电话、员工住址、开始工作日期、是否部门经理
* 帐户：帐户号、余额、开户日期、最近访问日期、是否储蓄账户、货币类型、利率、透支额度
* 贷款：贷款号、贷款金额
* 付款：付款日期、付款金额

设计理由：

1. 由于客户的联系人不一定是客户，并且联系人与其他实体之间没有关系，所以将联系人作为客户的属性。
2. 由于部门经理和普通员工都是员工，且部门经理与其他实体之间没有联系，则不必将二者区分为两个实体，如果某员工的“是否部门领导”为真，那么他就是所属部门的部门经理。同理，帐户类型用“是否储蓄账户”来表示，若是，则“透支额度”属性为NULL；若否，则“货币类型”和“利率”属性为NULL。
3. 需求中对唯一标识“付款”没有要求，所以对“付款”实体不要求主属性。
   1. 联系设计

根据需求调研，设计的联系及理由如下：

管理部门：一对多关系，一个支行可能拥有多个部门，一个部门只能被一个支行拥有

管理员工：一对多关系，一个部门可能管理多个员工，一个员工只能被一个部门所管理

管理账户：一对多关系，一个支行可能管理多个账户，一个账户只能为一个支行所管理

发放贷款：一对多关系，一个支行可能发放多笔贷款，一笔贷款只能由一个支行发放

拨付贷款：一对多关系，一笔贷款可能分多次付款，一笔付款只能付给一笔贷款

开通账户：多对多关系，一个客户可能拥有多个账户，一个账户也可能为多个客户共有

联系客户：多对多关系，一个客户可能联系多个员工，一个员工也可能联系多个客户

服务客户：多对多关系，一个客户可能享受多个支行的服务，一个支行也可能服务多个客户

客户借贷：多对多关系，一个客户可能拥有多笔贷款，一笔贷款也可能为多个客户共有

* 1. Power Designer的ER图

基于前述分析，利用Power Designer设计了银行业务管理系统的数据库概念模型，结果如图1所示。



图1. 银行业务管理系统的数据库概念模型

1. **概念模型到逻辑模型的转换**
   1. 实体转换

实体转换的结果如下，画下划线的为主码：

* 支行（支行名字、所在城市、管理的资产）
* 部门（部门号、部门名称、部门类型）
* 客户（客户身份证号、客户姓名、客户电话、客户住址、联系人姓名、联系人电话、联系人Email、与联系人关系）
* 员工（员工身份证号、员工姓名、员工电话、员工住址、开始工作日期、是否部门经理）
* 帐户（帐户号、余额、开户日期、最近访问日期、是否储蓄账户、货币类型、利率、透支额度）
* 贷款（贷款号、贷款金额）
* 付款（付款流水号、付款日期、付款金额）

转换前后的主要区别在于，“付款”实体缺少主码，这里给其加入“付款流水号”作为主码。

* 1. 联系转换

考虑每个联系，联系转换的结果如下，画波浪线的为外码，画双横线表示既是主码也是外码：

管理部门：一对多关系，一个支行可能拥有多个部门，一个部门只能被一个支行拥有

* 给部门增加“支行名字”外码：部门（部门号、支行名字、部门名称、部门类型）

管理员工：一对多关系，一个部门可能管理多个员工，一个员工只能被一个部门所管理

* 给员工增加“部门号”外码：员工（员工身份证号、部门号、员工姓名、……）

管理账户：一对多关系，一个支行可能管理多个账户，一个账户只能为一个支行所管理

* 给账户增加“支行名字”外码：帐户（帐户号、支行名字、余额、……）

发放贷款：一对多关系，一个支行可能发放多笔贷款，一笔贷款只能由一个支行发放

* 给贷款增加“支行名字”外码：贷款（贷款号、客户身份证号、贷款金额）

拨付贷款：一对多关系，一笔贷款可能分多次付款，一笔付款只能付给一笔贷款

* 给付款增加“贷款号”外码：付款（付款流水号、贷款号、付款日期、付款金额）

开通账户：多对多关系，一个客户可能拥有多个账户，一个账户也可能为多个客户共有

* 增加“开通账户”模式：开通账户（客户身份证号、账户号）

联系客户：多对多关系，一个客户可能联系多个员工，一个员工也可能联系多个客户

* 增加“联系客户”模式：联系客户（客户身份证号、员工身份证号）

服务客户：多对多关系，一个客户可能享受多个支行的服务，一个支行也可能服务多个客户

* 增加“服务客户”模式：服务客户（客户身份证号、支行名字）

客户借贷：多对多关系，一个客户可能拥有多笔贷款，一笔贷款也可能为多个客户共有

* 增加“客户借贷”模式：客户借贷（客户身份证号、贷款号）
  1. 最终的关系模式

经过实体转换和联系转换，最终的关系模式如下：

* 支行（支行名字、所在城市、管理的资产）
* 部门（部门号、支行名字、部门名称、部门类型）
* 客户（客户身份证号、客户姓名、客户电话、客户家庭住址、联系人姓名、联系人电话、联系人Email、与联系人关系）
* 员工（员工身份证号、部门号、员工姓名、员工电话、员工家庭地址、开始工作日期、是否部门经理）
* 帐户（帐户号、支行名字、余额、开户日期、最近访问日期、是否储蓄账户、货币类型、利率、透支额度）
* 贷款（贷款号、客户身份证号、贷款金额）
* 付款（付款流水号、贷款号、付款日期、付款金额）
* 客户借贷（客户身份证号、贷款号）
* 服务客户（客户身份证号、支行名字）
* 开通账户（客户身份证号、账户号）
* 联系客户（客户身份证号、员工身份证号）

1. **MySQL数据库结构实现**
   1. Power Designer的PDM设计

使用Power Designer把CDM转换成面向MySQL 5.0的PDM，得到的PDM如图2所示。



图2 银行业务管理系统的PDM设计结果

* 1. 数据库表定义

Power Designer的PDM可以直接转换为MySQL中的基本表。下面给出了基于PDM构建的MySQL基本表设计结果，见表1~表11。

表1. 支行表(Bank)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空 | 是否主键 | 是否外键 |
| Name | 支行名字 | Varchar(1024) | 否 | 是 | 否 |
| City | 所属城市 | Varchar(1024) | 否 | 否 | 否 |
| Property | 资产 | Float(100) | 否 | 否 | 否 |

表2. 部门表(Depart)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空 | 是否主键 | 是否外键 |
| Departno | 部门号 | Varchar(1024) | 否 | 是 | 否 |
| Bankname | 支行名字 | Varchar(1024) | 否 | 否 | 是，引用Bank.Name |
| Name | 部门名称 | Varchar(1024) | 否 | 否 | 否 |
| Type | 部门类型 | Varchar(1024) | 否 | 否 | 否 |

表3. 员工表(Employee)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空 | 是否主键 | 是否外键 |
| Eid | 员工身份证号 | Varchar(1024) | 否 | 是 | 否 |
| Departno | 部门号 | Varchar(1024) | 否 | 否 | 是，引用Depart.Departno |
| Name | 员工姓名 | Varchar(1024) | 否 | 否 | 否 |
| Tel | 员工电话 | Varchar(1024) | 否 | 否 | 否 |
| Address | 员工住址 | Varchar(1024) | 否 | 否 | 否 |
| Startdate | 开始工作日期 | Date | 否 | 否 | 否 |
| Manager | 是否部门经理 | Boolean | 否 | 否 | 否 |

表4. 账户表(Account)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空 | 是否主键 | 是否外键 |
| Aid | 账户号 | Varchar(1024) | 否 | 是 | 否 |
| Bankname | 支行名字 | Varchar(1024) | 否 | 否 | 是，引用Bank.Name |
| Balance | 账户余额 | Varchar(1024) | 否 | 否 | 否 |
| Startdate | 开户日期 | Date | 否 | 否 | 否 |
| Recentdate | 最近访问日期 | Date | 否 | 否 | 否 |
| Deposit | 是否储蓄账户 | Boolean | 否 | 否 | 否 |
| Rate | 利率 | Float(8) | 是 | 否 | 否 |
| Type | 货币类型 | Varchar(1024) | 是 | 否 | 否 |
| Overdraft | 透支额 | Float(100) | 是 | 否 | 否 |

表5. 客户表(Client)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空 | 是否主键 | 是否外键 |
| Cid | 客户身份证号 | Varchar(1024) | 否 | 是 | 否 |
| Cname | 客户姓名 | Varchar(1024) | 否 | 否 | 否 |
| Cphone | 客户电话 | Varchar(1024) | 否 | 否 | 否 |
| Caddress | 客户住址 | Varchar(1024) | 否 | 否 | 否 |
| Lname | 联系人姓名 | Varchar(1024) | 否 | 否 | 否 |
| Lphone | 联系人电话 | Varchar(1024) | 否 | 否 | 否 |
| Lemail | 联系人Email | Varchar(1024) | 否 | 否 | 否 |
| Relation | 联系人关系 | Varchar(1024) | 否 | 否 | 否 |

表6. 放贷表(Loan)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空 | 是否主键 | 是否外键 |
| Lid | 贷款号 | Varchar(1024) | 否 | 是 | 否 |
| Bankname | 支行名字 | Varchar(1024) | 否 | 否 | 是，引用Bank.Name |
| Lamount | 贷款金额 | Float(100) | 否 | 否 | 否 |

表7. 付款表(Appropriate)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空 | 是否主键 | 是否外键 |
| Pid | 付款流水号 | Varchar(1024) | 否 | 是 | 否 |
| Lid | 贷款号 | Varchar(1024) | 否 | 否 | 是，引用Loan.Lid |
| Pdate | 付款日期 | Date | 否 | 否 | 否 |
| Pamount | 付款金额 | Float(100) | 否 | 否 | 否 |

表8. 开通账户表(Affiliation)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空 | 是否主键 | 是否外键 |
| Cid | 客户身份证号 | Varchar(1024) | 否 | 是 | 是，引用Client.Cid |
| Aid | 账户号 | Varchar(1024) | 否 | 是 | 是，引用Account.Aid |

表9. 联系客户表(Contact)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空 | 是否主键 | 是否外键 |
| Cid | 客户身份证号 | Varchar(1024) | 否 | 是 | 是，引用Client.Cid |
| Eid | 员工身份证号 | Varchar(1024) | 否 | 是 | 是，引用Employee.Eid |

表10. 服务客户表(Service)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空 | 是否主键 | 是否外键 |
| Cid | 客户身份证号 | Varchar(1024) | 否 | 是 | 是，引用Client.Cid |
| Bankname | 支行名字 | Varchar(1024) | 否 | 是 | 是，引用Bank.Name |

表11. 客户借贷表(Borrow)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空 | 是否主键 | 是否外键 |
| Cid | 客户身份证号 | Varchar(1024) | 否 | 是 | 是，引用Client.Cid |
| Lid | 贷款号 | Varchar(1024) | 否 | 是 | 是，引用Loan.Lid |

1. **总结与体会**

本报告给出了利用Power Designer进行一个银行业务管理系统数据库的基本过程，包括概念模型设计、概念模型到逻辑模型的转换以及最终的MySQL数据库结构实现。

设计过程中的一些个人体会如下：

1. 通过这个实验，我巩固了数据库设计的流程知识：确定实体和它们之间的关系、根据实体和关系设计概念模型、将CDM转换为逻辑模型，并生成MySQL数据库结构、整理设计文档，在撰写设计文档的过程中，我不断发现问题、优化设计，如发现冗余属性并试图合并，最终设计出了比较满意的数据库模型。
2. 通过这个实验，我意识到调查并分析需求在设计数据库过程中非常重要。通过调查分析了解需求，才可以确定相关实体及其属性，有助于概念模型的设计。只有做好前期的调查与分析，才能保证设计的准确性与可用性。
3. 通过这个实验，我了解了如何使用工具来支持数据库设计工作，为数据库建模设计和后续知识学习学习打下良好的基础。Power Designer软件功能繁多，上手时有点困惑，但是其强大的功能为本次实验提供了许多方便，借助其绘制出的逻辑模型图，也能方便地发现一些冗余属性。