**银行业务管理系统数据库设计**

PB20000215 丁程

1. **概念模型设计**
   1. 实体设计

总共设计了以下14个实体，其中标黑的为主键:

1. 支行(Branch): **唯一名称(Name)**, 城市(Location), 资产(Assets)
2. 客户(Customer): **身份证号(ID)**, 姓名(Name), 联系电话(Phone), 家庭住址(Address)
3. 联系人(Contact): 姓名(Name), **手机号(Phone)**, Email, 与客户的关系(Relationship)
4. 员工(Employee): **身份证号(ID)**, 姓名(Name), 电话号码(Phone), 家庭地址(Address), 开始工作的日期(StartDate)
5. 经理(Manager): **身份证号(ID)**, 姓名(Name), 电话号码(Phone), 家庭地址(Address), 开始工作的日期(Start\_Date) 备注：经理不在员工表中，这样避免了循环引用
6. 部门(Department): **部门号(ID)**, 部门名称(Name), 部门类型(Type)
7. 账户(Account): **账户号(ID)**, 余额(Balance), 开户日期(Open\_Date)
8. 储蓄账户(SavingsAccount), Account的子类, 继承Account属性: 利率(Interest), 货币类型(CurrencyType)
9. 支票账户(CheckingAccount), Account的子类, 继承Account属性: 透支额(Overdraft)
10. 贷款(Loan): **贷款号(ID)**, 贷款金额(Amount), 付款情况(Payment\_Details)
11. 付款(Payment): 日期(Date), 付款金额(Amount), **付款号(ID)**
12. RegisterCheckingAccount, 事件实体:Account\_ID, Last\_Access
13. RegisterSavingsAccount, 事件实体:Account\_ID, Last\_Access
14. Responsible, 事件实体: ServiceType
    1. 联系设计

设计了以下联系，联系含义均为英文字面意思

1. BranchLendLoan: 1:N联系，一个贷款必须对应且只能由一个支行发放，但一个支行可以发放多个贷款。
2. BranchRegisterAccount: 1:N联系，一个账户必须对应且只能由一个支行注册，但一个支行可以注册多个账户。
3. ContactBelongsToCustomer: 1:1联系，每个客户必须有一个联系人，每个联系人有且只能对应一个客户。
4. CustomerBorrowLoan: M:N联系，每个贷款能被一个或多个客户所共有，每个客户可以对应多个贷款（银行将贷款分几次付给客户）
5. DepartmentBelongsToBranch: 1:N联系，每个支行必须有且可以有多个部门，每个部门只能对应一个支行
6. EmployeeBelongsToDepartment: 1:N联系，每个员工必须对应一个部门，每个部门可以有多个员工
7. ManagerBelongsToDepartment: 1:1联系，每个经理对应一个部门，每个部门有一个经理
8. PaymentPayForLoan: 1:N联系，每个还款只能对应一个贷款，每个贷款可以对应多个还款
9. RegisterCheckingAccount: M:N联系，将对应的Account\_ID和Last\_Access提取为事件实体。
10. RegisterSavingsAccount: M:N联系，将对应的Account\_ID和Last\_Access提取为事件实体。
11. Responsible: M:N联系,客户可能和某个银行员工发生联系，该员工是此客户的贷款负责人或银行帐户负责人
    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。

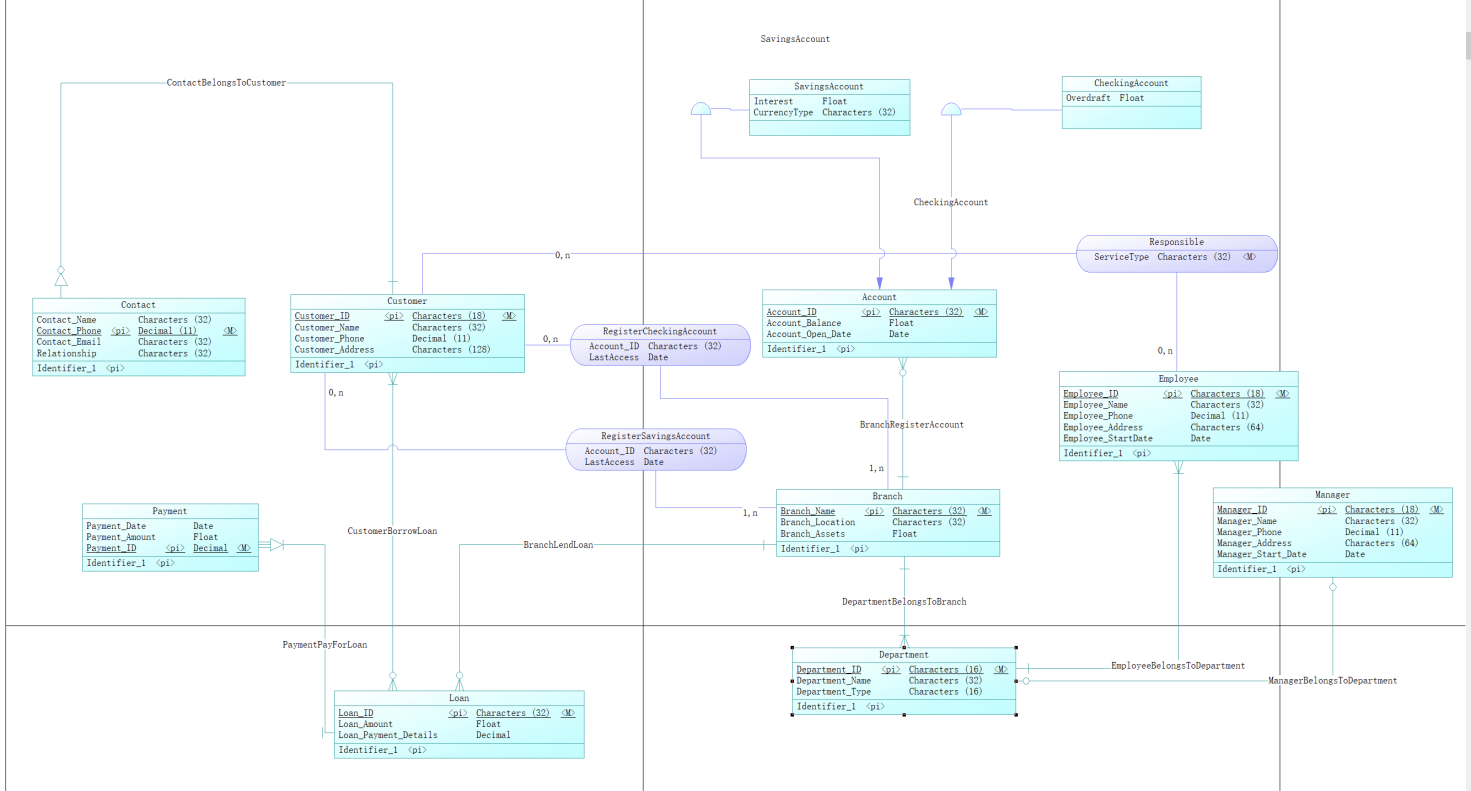


图1. 银行业务管理系统的数据库概念模型

1. **概念模型到逻辑模型的转换**
   1. 实体转换

把实体转换为关系模式：

1. Branch: **Branch\_Name**, Branch\_Location, Branch\_Assets
2. Customer: **Customer\_ID**, Customer\_Name, Customer\_Phone, Customer\_Address
3. Contact: Contact\_Name, **Contact\_Phone**, Contact\_Email, Relationship
4. Employee: **Employee\_ID**, Employee\_Name, Employee\_Phone, Employee\_Address, Employee\_Start\_Date
5. Manager: **Manager\_ID**, Manager\_Name, Manager\_Phone, Manager\_Address, Manager\_Start\_Date
6. Department: **Department\_ID**, Department\_Name, Department\_Type
7. Account: **Account\_ID**, Account\_Balance, Account\_Open\_Date
8. Loan: **Loan\_ID**, Loan\_Amount, Loan\_Payment\_Details
9. RegisterCheckingAccount: Account\_ID, Last\_Access
10. RegisterSavingsAccount: Account\_ID, Last\_Access
11. Responsible: ServiceType

考虑子类，加入父类标识作为主码：

1. SavingsAccount: Interest, CurrencyType, **Account\_ID**
2. CheckingAccount: Overdraft, **Account\_ID**

考虑弱实体Payment，加入所依赖强实体标识，修改主码为Loan\_ID + Payment\_ID

1. Payment: Payment\_Date, Payment\_Amount, **Payment\_ID, Loan\_ID**
   1. 联系转换
2. BranchLendLoan: 1:N联系，在Loan中加入Branch\_Name属性
3. BranchRegisterAccount: 1:N联系，在Account中加入Branch\_Name属性
4. ContactBelongsToCustomer: 1:1联系，在Customer中加入Contact\_Phone属性
5. CustomerBorrowLoan: M:N联系，新增一个CustomerBorrowLoan关系模式，加入Customer\_ID和Loan\_ID属性作为主码
6. DepartmentBelongsToBranch: 1:N联系，在Department中加入Branch\_Name属性
7. EmployeeBelongsToDepartment: 1:N联系，在Employee中加入Department\_ID属性
8. ManagerBelongsToDepartment: 1:1联系，在Department中加入Manager\_ID属性
9. PaymentPayForLoan: 1:N联系，在Payment中加入Loan\_ID属性
10. RegisterCheckingAccount: M:N联系，在已提取出的实体中加入Customer\_ID和Branch\_Name作为主码
11. RegisterSavingsAccount: M:N联系，在已提取出的实体中加入Customer\_ID和Branch\_Name作为主码
12. Responsible: M:N联系，在已提取出的实体中加入Customer\_ID和Employee\_ID作为主码
    1. 最终的关系模式
13. Branch: **Branch\_Name**, Branch\_Location, Branch\_Assets
14. Customer: **Customer\_ID**, Customer\_Name, Customer\_Phone, Customer\_Address, Contact\_Phone
15. Contact: Contact\_Name, **Contact\_Phone**, Contact\_Email, Relationship
16. Employee: **Employee\_ID**, Employee\_Name, Employee\_Phone, Employee\_Address, Employee\_StartDate, Department\_ID
17. Department: **Department\_ID**, Department\_Name, Department\_Type, Manager\_ID, Branch\_Name
18. Manager: **Manager\_ID**, Manager\_Name, Manager\_Phone, Manager\_Address, Manager\_Start\_Date
19. Account: **Account\_ID**, Account\_Balance, Account\_Open\_Date, Branch\_Name
20. Loan: **Loan\_ID**, Loan\_Amount, Loan\_Payment\_Details, Branch\_Name
21. RegisterCheckingAccount: Account\_ID, Last\_Access, **Customer\_ID, Branch\_Name**
22. RegisterSavingsAccount: Account\_ID, Last\_Access, **Customer\_ID, Branch\_Name**
23. Responsible: ServiceType, **Customer\_ID, Employee\_ID**
24. SavingsAccount: Interest, CurrencyType, **Account\_ID**
25. CheckingAccount: Overdraft, **Account\_ID**
26. Payment: Payment\_Date, Payment\_Amount, **Payment\_ID, Loan\_ID**
27. CustomerBorrowLoan: **Customer\_ID, Loan\_ID**
28. **MySQL数据库结构实现**
    1. Power Designer的PDM设计

//把CDM转换成面向MySQL的物理数据模型（PDM）

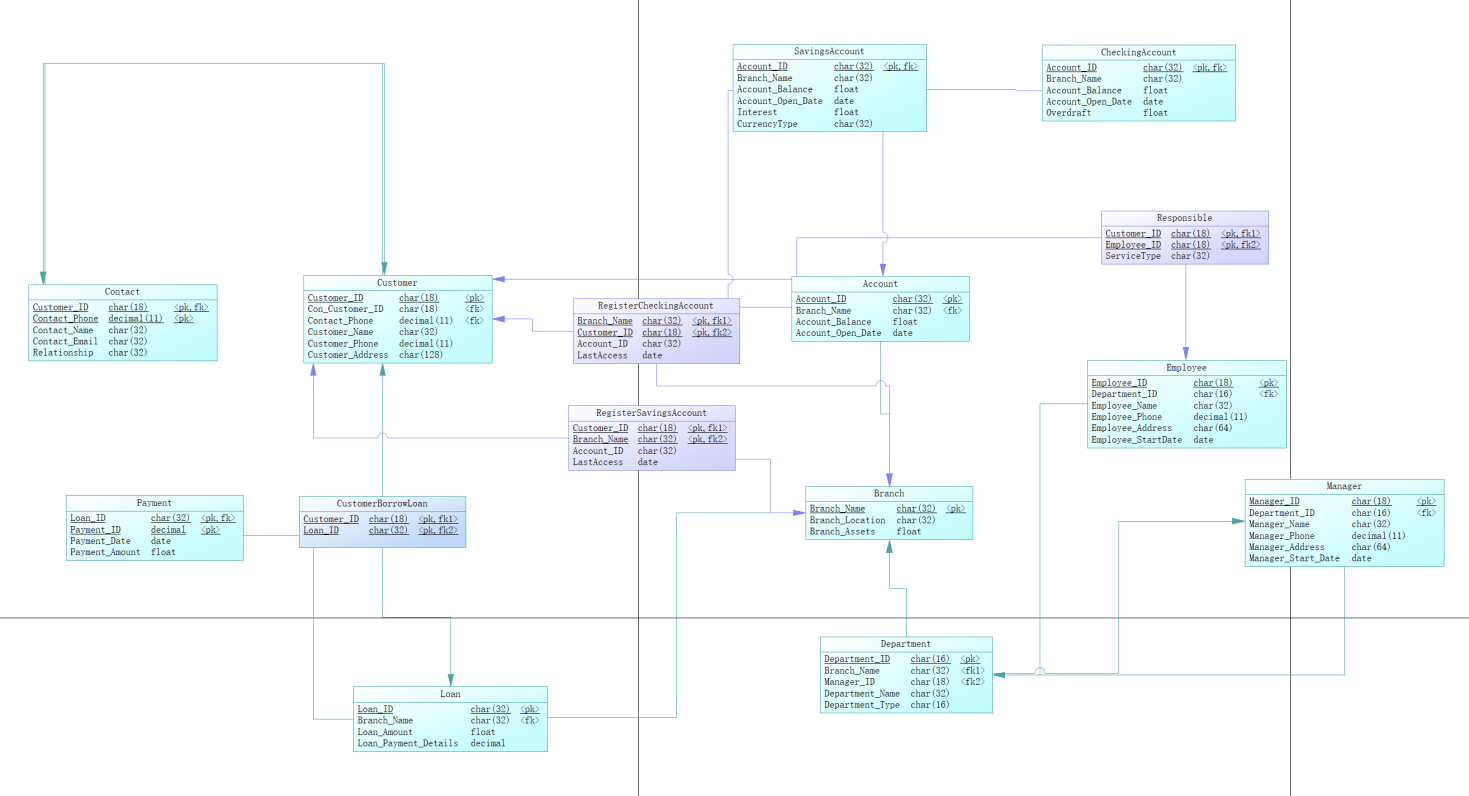


图2 银行业务管理系统的PDM设计结果

* 1. 数据库表定义

//给出各个Table的定义

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X。

表1. Account(账户)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空  (Null) | 是否主键  (Primary Key) | 是否外键  (若是则给出引用的表和列名) |
| Account\_ID | 账户编号 | char(32) | 否 | 是 | 否 |
| Branch\_Name | 支行名 | char(32) | 否 | 否 | 是, Branch表Branch\_Name列 |
| Account\_Balance | 账户余额 | float | 是 | 否 | 否 |
| Account\_Open\_Date | 开户日期 | date | 是 | 否 | 否 |

表2. Branch(支行)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空  (Null) | 是否主键  (Primary Key) | 是否外键  (若是则给出引用的表和列名) |
| Branch\_Name | 支行名 | char(32) | 否 | 是 | 否 |
| Branch\_Location | 支行位置 | char(32) | 是 | 否 | 否 |
| Branch\_Assets | 支行资产 | float | 是 | 否 | 否 |

表3. CheckingAccount(支票账户)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空  (Null) | 是否主键  (Primary Key) | 是否外键  (若是则给出引用的表和列名) |
| Account\_ID | 账户编号 | char(32) | 否 | 是 | 是, Account表Account\_ID列 |
| Branch\_Name | 支行名 | char(32) | 是 | 否 | 否 |
| Account\_Balance | 账户余额 | float | 是 | 否 | 否 |
| Account\_Open\_Date | 开户日期 | date | 是 | 否 | 否 |
| Overdraft | 透支额 | float | 是 | 否 | 否 |

表4. Contact(联系人)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空  (Null) | 是否主键  (Primary Key) | 是否外键  (若是则给出引用的表和列名) |
| Customer\_ID | 联系人对应客户ID | char(18) | 否 | 是 | 是, Customer表Customer\_ID列 |
| Contact\_Phone | 联系人电话 | decimal(11) | 否 | 是 | 否 |
| Contact\_Name | 联系人姓名 | char(32) | 是 | 否 | 否 |
| Contact\_Email | 联系人邮箱 | char(32) | 是 | 否 | 否 |
| Relationship | 联系人和客户关系 | char(32) | 是 | 否 | 否 |

表5. Customer(客户)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空  (Null) | 是否主键  (Primary Key) | 是否外键  (若是则给出引用的表和列名) |
| Customer\_ID | 客户ID | char(18) | 否 | 是 | 否 |
| Contact\_Phone | 联系人电话 | decimal(11) | 是 | 否 | 是, Contact表Contact\_Phone列 |
| Customer\_Name | 客户姓名 | char(32) | 是 | 否 | 否 |
| Customer\_Phone | 客户电话 | decimal(11) | 是 | 否 | 否 |
| Customer\_Address | 客户地址 | char(128) | 是 | 否 | 否 |
| Con\_Customer\_ID | 联系人对应客户ID | char(18) | 是 | 否 | 是，Customer表Customer\_ID列 |

表6. CustomerBorrowLoan

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空  (Null) | 是否主键  (Primary Key) | 是否外键  (若是则给出引用的表和列名) |
| Customer\_ID | 客户ID | char(18) | 否 | 是 | 是，Customer表Customer\_ID列 |
| Loan\_ID | 贷款ID | char(32) | 否 | 是 | 是，Loan表Loan\_ID列 |

表7. Department(部门)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空  (Null) | 是否主键  (Primary Key) | 是否外键  (若是则给出引用的表和列名) |
| Department\_ID | 部门ID | char(16) | 否 | 是 | 否 |
| Branch\_Name | 支行名 | char(32) | 否 | 否 | 是, Branch表Branch\_Name列 |
| Manager\_ID | 经理ID | char(18) | 是 | 否 | 是, Manager表Manager\_ID列 |
| Department\_Name | 部门名 | char(32) | 是 | 否 | 否 |
| Department\_Type | 部门类型 | char(16) | 是 | 否 | 否 |

表8. Employee(员工)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空  (Null) | 是否主键  (Primary Key) | 是否外键  (若是则给出引用的表和列名) |
| Employee\_ID | 员工ID | char(18) | 否 | 是 | 否 |
| Department\_ID | 部门ID | char(16) | 否 | 否 | 是, Department表Department\_ID列 |
| Employee\_Name | 员工名 | char(32) | 是 | 否 | 否 |
| Employee\_Phone | 员工电话 | decimal(11) | 是 | 否 | 否 |
| Employee\_Address | 员工地址 | char(64) | 是 | 否 | 否 |
| Employee\_StartDate | 员工开始工作日期 | date | 是 | 否 | 否 |

表9. Loan(贷款)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空  (Null) | 是否主键  (Primary Key) | 是否外键  (若是则给出引用的表和列名) |
| Loan\_ID | 贷款ID | char(32) | 否 | 是 | 否 |
| Branch\_Name | 支行名 | char(32) | 否 | 否 | 是, Branch表Branch\_Name列 |
| Loan\_Amount | 贷款金额 | float | 是 | 否 | 否 |
| Loan\_Payment\_Details | 贷款还款详情 | decimal | 是 | 否 | 否 |

表10. Manager(经理)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空  (Null) | 是否主键  (Primary Key) | 是否外键  (若是则给出引用的表和列名) |
| Manager\_ID | 经理ID | char(18) | 否 | 是 | 否 |
| Department\_ID | 部门ID | char(16) | 是 | 否 | 是, Department表Department\_ID列 |
| Manager\_Name | 经理名 | char(32) | 是 | 否 | 否 |
| Manager\_Phone | 经理电话 | decimal(11) | 是 | 否 | 否 |
| Manager\_Address | 经理地址 | char(64) | 是 | 否 | 否 |
| Manager\_Start\_Date | 经理开始工作日期 | date | 是 | 否 | 否 |

表11. Payment(还款)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空  (Null) | 是否主键  (Primary Key) | 是否外键  (若是则给出引用的表和列名) |
| Loan\_ID | 贷款ID | char(32) | 否 | 是 | 是，Loan表Loan\_ID列 |
| Payment\_ID | 还款ID | decimal | 否 | 是 | 否 |
| Payment\_Date | 还款日期 | date | 是 | 否 | 否 |
| Payment\_Amount | 还款金额 | float | 是 | 否 | 否 |

表12. RegisterCheckingAccount

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空  (Null) | 是否主键  (Primary Key) | 是否外键  (若是则给出引用的表和列名) |
| Branch\_Name | 支行名 | char(32) | 否 | 是 | 是, Branch表Branch\_Name列 |
| Customer\_ID | 客户ID | char(18) | 否 | 是 | 是，Customer表Customer\_ID列 |
| Account\_ID | 账户ID | char(32) | 是 | 否 | 否 |
| LastAccess | 最后访问时间 | date | 是 | 否 | 否 |

表13. RegisterSavingsAccount

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空  (Null) | 是否主键  (Primary Key) | 是否外键  (若是则给出引用的表和列名) |
| Branch\_Name | 支行名 | char(32) | 否 | 是 | 是, Branch表Branch\_Name列 |
| Customer\_ID | 客户ID | char(18) | 否 | 是 | 是，Customer表Customer\_ID列 |
| Account\_ID | 账户ID | char(32) | 是 | 否 | 否 |
| LastAccess | 最后访问时间 | date | 是 | 否 | 否 |

表14. Responsible

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空  (Null) | 是否主键  (Primary Key) | 是否外键  (若是则给出引用的表和列名) |
| Customer\_ID | 客户ID | char(18) | 否 | 是 | 是，Customer表Customer\_ID列 |
| Employee\_ID | 员工ID | char(18) | 否 | 是 | 是，Employee表Employee\_ID列 |
| ServiceType | 服务类型 | char(32) | 否 | 否 | 否 |

表15. SavingsAccount(储蓄账户)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 列名 | 中文含义 | 类型(长度) | 允许为空  (Null) | 是否主键  (Primary Key) | 是否外键  (若是则给出引用的表和列名) |
| Account\_ID | 账户编号 | char(32) | 否 | 是 | 是, Account表Account\_ID列 |
| Branch\_Name | 支行名 | char(32) | 是 | 否 | 否 |
| Account\_Balance | 账户余额 | float | 是 | 否 | 否 |
| Account\_Open\_Date | 开户日期 | date | 是 | 否 | 否 |
| Interest | 利率 | float | 是 | 否 | 否 |
| CurrencyType | 货币类型 | char(32) | 是 | 否 | 否 |

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）power designer的UI用起来有点难受，但是有些功能挺方便的，比如CDM->PDM,再从PDM自动生成建表的sql代码

（2）数据库的设计还是很重要的，如果不好好设计，在项目后期再去更改原来的设计的话，设计到的代码的更改会非常多而且复杂。