计算机系统安全实验

实验四

姓名：卢兑玳

学号：L170300901

实验 4　完整性访问控制系统设计与实现

一、系统设计说明：

设计完整性访问控制系统，实现系统，并满足某商业公司的完整性访问控制需求。

（1）配合第 7 章，为商业公司设计系统，提出针对该公司业务需求的应用系统安全策略。安全策略中要明确指明对公司的要求与约束,和对客户的要求与约束,区分各自的责任。（当出现商业公司与客户间意见分歧或法律纠纷时，安全策略可作为仲裁依据）

（2）配合第 9 章 为商业公司设计系统，应用系统满足完整性需求。需求中包含责任分离、功能分离、审计。

（3）具体指明是哪类应用系统，应用背景范围不限，可以是银行、股票等，符合商业系统完整性需求即可。

（4）4 学时，每人独立完成。

二、系统要求：

（1）给出应用系统的安全策略文档。

（2）提供交互界面，能够完成录入、查询等功能。

（3）满足责任分离、功能分离原则。

（4）保存审计日志。

（5）遵循 Clark-Wilson 模型，定义应用系统的完整性限制条件。

（6)遵循 Clark-Wilson 模型的证明规则和实施规则,并在设计报告中有所体现。

三、编制实验报告，给出系统安全策略文档、系统设计报告，源代码、实现过程说明，及心得体会。

3.1 实验设计

本实验设计实现了一个简单的银行存取款系统，实现了存款，取款，查询的功能，满足责任分离，功能分离的原则。

1.数据库的构建

数据库 bank 由三张表组成，分别是管理员表，用户表，用户待存取款表。

```
mysql> use bank
Database changed
mysql> create table client_info(
    -> id int(5) not null primary key,
    -> name char(10) not null,
    -> password int(6) not null,
    -> money int(9) not null);
Query OK, 0 rows affected, 3 warnings (0.04 sec)

mysql> create table manager_info(
    -> id int(5) not null primary key,
    -> name char(10) not null,
    -> password int(6) not null,
    -> ;
ERROR 1064 (42000): You have an error in your SQL syntax; check the manual that corresponds to your MySQL serv
 for the right syntax to use near '' at line 4
mysql> create table manager_info(
    -> id int(5) not null primary key,
    -> name char(10) not null,
    -> password int(6) not null);
Query OK, 0 rows affected, 2 warnings (0.03 sec)

mysql> create table bill_waiting_deal(
    -> id int(3) not null primary key,
    -> client_id int(5) not null,
    -> name char(10) not null,
    -> money int(9) not null);
Query OK, 0 rows affected, 3 warnings (0.04 sec)
```

```
mysql> desc bill_waiting_deal;
+-----------+----------+------+-----+---------+-------+
| Field     | Type     | Null | Key | Default | Extra |
+-----------+----------+------+-----+---------+-------+
| id        | int      | NO   | PRI | NULL    |       |
| client_id | int      | NO   |     | NULL    |       |
| name      | char(10) | NO   |     | NULL    |       |
| money     | int      | NO   |     | NULL    |       |
+-----------+----------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql> desc client_info;
+----------+----------+------+-----+---------+-------+
| Field    | Type     | Null | Key | Default | Extra |
+----------+----------+------+-----+---------+-------+
| id       | int      | NO   | PRI | NULL    |       |
| name     | char(10) | NO   |     | NULL    |       |
| password | int      | NO   |     | NULL    |       |
| money    | int      | NO   |     | NULL    |       |
+----------+----------+------+-----+---------+-------+
4 rows in set (0.00 sec)

mysql> desc manager_info;
+----------+----------+------+-----+---------+-------+
| Field    | Type     | Null | Key | Default | Extra |
+----------+----------+------+-----+---------+-------+
| id       | int      | NO   | PRI | NULL    |       |
| name     | char(10) | NO   |     | NULL    |       |
| password | int      | NO   |     | NULL    |       |
+----------+----------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

其中，管理员表包含管理员 id，姓名（name），账户密码（password），本实验一共创建了两个管理员 admin1，admin。

用户表包含用户 id，name，password，和用户账户存款额（money），本实

验创建的用户为 client1

用户待存取款表包含存取款账单 id，该账单的用户 client_id，用户姓名 name，用户存取款数（money）。当用户发起申请存取款时，在管理员未处理之前，会在此表中记录账单，其中 money 列为负数表示取款，正数表示存款。

2.系统安全策略文档

用户拥有存取款功能和查询功能。用户在申请存取款后，会生成一个账单，提交到管理员处。待管理员同意后，才能执行操作，对数据库中的用户存款金额进行修改。

在登录界面提供两种操作：登录和注册。新用户（管理员和用户）只有注册以后并且以注册的用户名和密码登录，才能进行相关操作。

取款和存款时，首先需要以用户身份和密码登录，选择存款或者取款功能，输入金额，然后生成账单。管理员登录后可以看见用户申请的账单信息，里面有用户的用户名和更改金额。当两个管理员登录并且认证通过后，该账单才可以被删除，该操作执行成功。如果有其中一个管理员没有同意，则直接取消账单。注意:取款金额不能超过当前拥有的金额。多管理员认证策略体现了责任分离制度，确保不会有个别管理员非法操作。账单只能由用户发起，由管理员认证同意，体现了功能分离制度。

查询是直接调用当前数据库中的数据，没有更改数据库中的信息，不需要管理员进行同意。

3.责任分离，功能分离原则

在完整性策略定义的授权方式里面，责任分离原则禁止一个实体完成单独完成一个功能操作。对于取款和存款两种操作，会对数据库中的内容进行修改，所以此时必须采用责任分离的原则。在这里，管理员 1 和管理员需要同时对存取款功能进行同意，才能实现用户存取款成功，然后改变数据库中用户的存款金额。否则用户存款金额不可以被修改，不影响数据的完整性。

在功能分离原则中，我们禁止任何用户执行不属于他的权力。我们的用户相当于一个实体，对于查询和退出登陆两种操作是用户自身的权力，而且不影响数据的完整性。对于管理员身份的用户。单独有管理员时他不能做任何事情，单独有用户他只能查询余额，只有在用户申请这两种然后管理员同意时操作才能完成，符合功能分离的原则，同时也能保证数据的完整性。

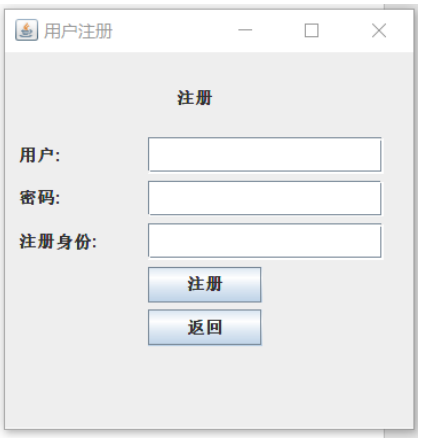
3.2 实验过程

1.交互界面演示

登录界面：

如果已有账号，则需要输入用户名，密码，用户身份（用户/管理员），点击登录，即可登录到对应用户的界面（图 1）。如果需要注册用户，则点击注册，出现注册界面（图 2），输入想要注册的用户名，密码，和注册身份（用户/管理员）.

（图 1）



（图 2）

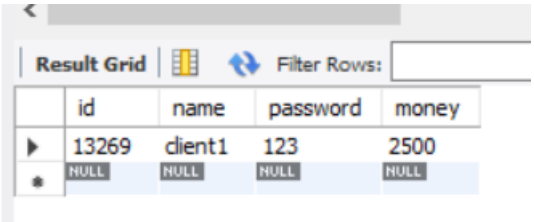
用户登录界面：

通过用户身份登录（client1 用户），出现用户界面（图 3）。用户有存款，取款，查询余额，注销登录功能。

（图 3）

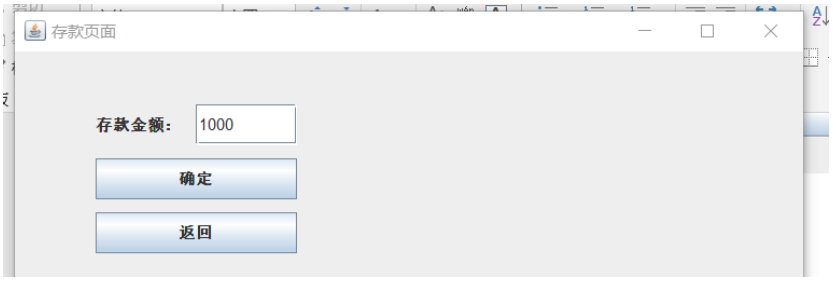
用户存款：

存款之前，数据库中用户的余额为 2500（图 4）

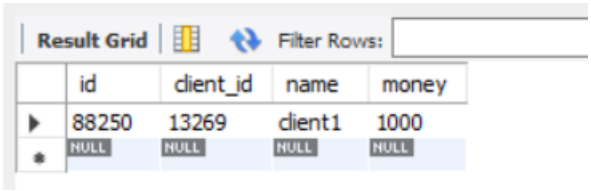


（图 4）

点击存款按钮，出现存款界面（图 5），输入存款额度 1000 元，点击确定，进入管理员审核阶段。同时观察数据库中的待处理账单，出现了一项存款+1000 元的账单（图 5.1）。



（图 5）



（图 5.1）

管理员审核阶段：

在用户点击存款确认按钮之后，会进行管理员审核阶段，管理员审核阶段分两步，需要两个管理员登录审核过程。

管理员 1 审核，首先会弹出登录按钮（图 6），输入管理员 admin1 的账号，密码登录，进入初步审核阶段。

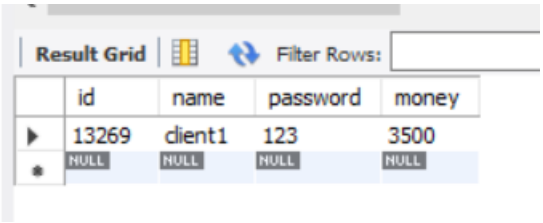
（图 6）

初步审核阶段，在管理员 1 登录后，出现初步审核阶段（图 7），如果管理员点击不同意，则该存款账单会从数据库中消除，同时存款行为也会取消。用户账户余额不变。如果点击同意，会进入二次确认阶段。

（图 7）

二次审核阶段，会出现第二个管理员登录界面，输入 admin 管理员用户名和密码后，出现二次审核界面（图 8）。点击不同意，则该存款账单会从数据库中消除，同时存款行为也会取消。点击同意，则该存款账单会从数据库中消除，同时存款行为正式完成，我们再去观察用户数据库（图 9），发现用户账户金额变成了 3500 元（增加了 1000 元）

（图 8）

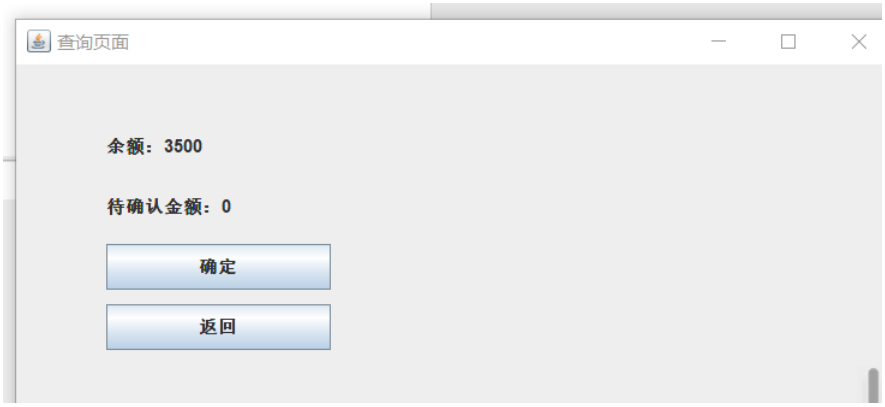


（图 9）

取款界面和上面操作一致，因此不再赘述。

用户查询余额阶段：

点击图 3 的查询按钮按钮，查询用户余额（图 10），我们可以发现用户余额和数据库中存款金额一致。



（图 10）

2. 审计日志

所有日志保存在 log.txt 文件夹下（图 11）

```
2020/01/01 19:58:30  管理员 admin1 登录成功
2020/01/01 19:59:32  管理员 admin1 同意了用户 client1 的 1000 元的存款请求
2020/01/01 20:00:30  管理员 admin 登录成功
2020/01/01 20:01:32  管理员 admin 同意了用户 client1 的 1000 元的存款请求
2020/01/01 20:10:32  用户 client1 存款 1000 成功
```

（图 11）

3. 遵循 Clark-Wilson 模型，定义应用系统的完整性限制条件

Clark-Wilson 模型考虑如下几点：

（1）主体必须被识别和认证

（2）客体只能通过规定的程序进行操作

（3）主体只能执行规定的程序

（4）必须维护正确的审计日志

（5）系统必须被证明能够正确工作

如果把管理员看作主体，用户为客体，则管理员登陆界面即为认证过程，管理员的用户和密码只有自己才能知道的，满足（1）。作为客体的用户只能进行取款、存款、查询、返回四种操作，满足（2）。主体管理员只能进行认证，不能单独进行存款取款等操作，满足（3）。根据之前的演示过程，可以证明系统正常工作，满足（4）（5）。

4. 遵循 Clark-Wilson 模型的证明规则和实施规则，并在设计报告中有所体现

（1）证明规则 1：当任意 IVP 运行时，它必须保证所有的 CDI 处于有效状态

当用户登录时，在没有管理员同意的情况下，不能对数据库中的数据进行操作。只能提交账单申请，待管理员同意后，才能更改数据库。

（2）证明规则 2：对相关联的 CDI，一个 TP 必须将这些 CDI 从一个有效状态转到另一个有效状态

在管理员同意后，账单就会被删除。这个交易过程，会将用户的申请状态，变为完成状态或错误状态（被管理员拒绝或读取/存入数据库失败）。

（3）证明规则 3：系统执行操作时，符合责任分离原则。

模型需要保证用户身份和执行代码身份一致。所以需要验证身份。这里设计的验证身份就是“登录”。

（4）实施规则 1：系统要维护关联关系，保证经过验证的 TP 操作相应的 CDI

在用户提出存取款申请后，管理员同意，就代表该账单已经被验证。被验证的这个账单可以对数据库中，相应的存款金额进行更改。

（5）实施规则 2：TP 操作 CDI 时，保证操作用户有权对相应 CDI 做操作，TP 所代表的用户是 CDI 的真实用户

经过验证的账单，即管理员同意后，可以对数据库中的 CDI（即用户的存款金额）进行更改。

（6）实施规则 3：系统执行操作时，符合责任分离原则，模型需要保证用户身份和执行代码身份一致

满足责任分离原则，用户和管理员都不能单独对存款金额进行更改，只有用户申请，管理员同意后，才能进行操作。

（7）实施规则 4：只有可以授予 TP 访问规则的主体才能修改列表中相应的表项，授权主体不能执行 TP 操作

只有用户提出申请，才能进行 TP 操作。授权的管理员，没有执行 TP 操作的能力。

4. 实验总结和反思

通过本次实验，我初步掌握了如何设计完整性访问控制系统，并且使其满足完整性访问控制需求，提出应用系统安全策略，并且在安全策略中指明各类用户的要求和约束，区分各自责任归属。