实验一 OLTP实验

一、实验目的

学会安装、配置、使用Mysql,借助一个简单的银行核心业务数据库，通过对该数据库中用户对自己的活期存款账户进行操作，包括开户、存款、取款、转账、查询账户余额等，来进一步加深对关系数据库、OLTP、事务处理、SQL语句等的理解，并熟练掌握对Mysql数据库的使用。

1. 实验环境

安装并配置有Mysql的电脑。

1. 实验内容

**（1）创建数据库模型**

在本实验中，我们需要对客户、账户、交易明细进行管理，所以需要三张表，对这些信息进行管理，这些表格的字段及其主外键关系，请参考下图。

客户表，包括id、姓名、性别、出生日期、居住地等字段。Id为主键，一般用身份证号即可。

账户表是活期账户，字段包括客户id、开户日期、当前余额等字段。客户id是账户表的主键，同时它指向客户表，是一个外键。

交易明细表记录历次交易，包括开户、销户、存钱、取钱、转账等。包括客户id、操作日期时间、操作类型、金额、上次余额、本次余额等。其中客户id、操作日期时间为主键，客户id指向客户表，是一个外键。

**Savings:**

**cust\_id** char(18),

open\_date char(8),

cur\_balance decimal(16,3)

**Customer**:

**id** char(18),

name char(16),

gender char(1),

birth\_day char(8),

residence\_place char(16)

**Transaction\_history:**

**cust\_id** char(18),

**operation\_datetime** char(12),

operation\_type char(1),

amount decimal(16,3),

last\_balance decimal(16,3),

cur\_balance decimal(16,3)

**（2）建表**

**客户表**

客户表包含如下字段。表名为customer。

|  |  |  |
| --- | --- | --- |
| 字段 | 数据类型 | 备注 |
| id | Char(18) | 主键，唯一标识一个客户。 |
| name | Char(16) | 姓名 |
| gender | Char(1) | 性别，M表示男，F表示女 |
| birth\_day | Char(8) | 用字符串表示日期，比如“19900101”表示1990年1月1日，月和日都用两个字符表示 |
| residence\_place | Char(16) | 居住地，具体到省份 |

建表的sql语句如下。

|  |
| --- |
| Create table **customer**  (  id char(18),  name char(16),  gender char(1),  birth\_day char(8),  residence\_place char(16),  primary key(id)  ); |

**账户表**

账户表是活期账户，其字段如下。表名为savings。

由于一个客户只有一个账户，所以cust\_id为外键指向customer表，同时也作为账户表的主键，唯一标识这个账户是谁的。

|  |  |  |
| --- | --- | --- |
| 字段 | 数据类型 | 备注 |
| cust\_id | Char(18) | 这是外键，也是主键。 |
| open\_date | Char(8) | 开户日期 |
| cur\_balance | Numeric(16,3) | 当前余额，小数点后到厘 |

建表的sql语句如下。

|  |
| --- |
| Create table **savings**  (  cust\_id char(18),  open\_date char(8),  cur\_balance decimal(16,3),  primary key(cust\_id),  foreign key(cust\_id) references customer(id) on delete cascade on update cascade  ); |

我们用numeric来表示金额，定义这种字段类型的时候，第一个参数表示精度，即包括小数点的十进制数字的个数，第二个参数表示小数点后的数字位数。

比如salary定义为decimal(5,2)，那么它的范围为-99.99 到 99.99，注意包括小数点。 NUMERIC 和 DECIMAL 类型被 MySQL 以同样的类型实现，在 SQL92 标准中是允许的。

**交易明细表**

这个表记录用户的交易明细，包括存钱、取钱、转账等。

表名为transaction\_history。

|  |  |  |
| --- | --- | --- |
| 字段 | 数据类型 | 备注 |
| cust\_id | char(18) | 这是外键  和operation\_datetime一起构成主键 |
| operation\_datetime | Char(14) | 用字符串表示日期时间，比如“19900101151515”表示1990年1月1日15点15分15秒 |
| Operation\_type | Char(1) | O表示open开户, C表示close销户, D表示deposit存钱,W表示withdraw取钱, T表示transfer转账 |
| amount | 操作金额 | 正数表示增加  负数表示减少 |
| last\_balance | 余额 | 上一个余额 |
| cur\_balance | 余额 | 当前余额 |

建表的sql语句如下。

|  |
| --- |
| Create table transaction\_history  (  cust\_id char(18),  operation\_datetime char(14),  operation\_type char(1),  amount decimal(16,3),  last\_balance decimal(16,3),  cur\_balance decimal(16,3),  primary key(cust\_id, operation\_datetime),  foreign key(cust\_id) references customer(id) on delete cascade on update cascade  ); |

**（3）初始数据**

插入3个用户

|  |
| --- |
| insert into customer values ('110108197012190014', 'wang tao', 'M', '19701219', 'beijing');  insert into customer values ('110108197108290016', 'li ming', 'M', '19710829', 'tianjin');  insert into customer values ('110108197509050018', 'li li', 'F', '19750905', 'beijing'); |

备注：M/F表示male和female。注意身份证里面的日期和出生日期的对应关系。

插入3个开户记录

|  |
| --- |
| insert into **savings** values ('110108197012190014', '20180301', 13000.00);  insert into **savings** values ('110108197108290016', '20180101', 35000.00);  insert into **savings** values ('110108197509050018', '20180101', 150000.00); |

备注：3个人开户的日期不一样，开户时存入的金额不一样。

插入一些交易记录，具体如下。

开户、开户、开户：三个人的开户应该在交易历史记录里有一笔记录

|  |
| --- |
| insert into transaction\_history values('110108197012190014', '20180301093030',  'O', 13000.00, 0, 13000.00);  insert into transaction\_history values('110108197108290016', '20180101103030',  'O', 35000.00, 0, 35000.00);  insert into transaction\_history values('110108197509050018', '20180101111515',  'O', 150000.00, 0, 150000.00); |

备注：开户的时候，交易金额就是开户存入的钱数，原来的余额是0，现在的余额就是刚刚存进去的钱。

存钱、取钱、存钱、取钱、存钱、取钱：每个账户存钱500，取钱200。

|  |
| --- |
| insert into transaction\_history values('110108197012190014', '20180302093030',  'D', 500.00, 13000.00, 13500.00);  update **savings set** cur\_balance **=**13500.00 where cust\_id='110108197012190014';  insert into transaction\_history values('110108197012190014', '20180302093230',  'W', -200.00, 13500.00, 13300.00);  update **savings set** cur\_balance **=**13300.00 where cust\_id ='110108197012190014';  insert into transaction\_history values('110108197108290016', '20180102103030',  'D', 500.00, 35000.00, 35500.00);  update **savings set** cur\_balance **=**35500.00 where cust\_id ='110108197108290016';  insert into transaction\_history values('110108197108290016', '20180102103330',  'W', -200.00, 35500.00, 35300.00);  update **savings set** cur\_balance **=**35300.00 where cust\_id ='110108197108290016';  insert into transaction\_history values('110108197509050018', '20180102111515',  'D',500.00, 150000.00, 150500.00);  update **savings set** cur\_balance **=**150500.00 where cust\_id ='110108197509050018';  insert into transaction\_history values('110108197509050018', '20180102131515',  'W', -200.00, 150500.00, 150300.00);  update **savings set** cur\_balance **=**150300.00 where cust\_id ='110108197509050018'; |

备注：开户的第二天做了存钱和取钱。

转账：从'110108197012190014'转账200到'110108197509050018'，注意两者的交易记录和账户余额。

|  |
| --- |
| insert into transaction\_history values('110108197012190014', '20180503093030',  'T', -200.00, 13300.00, 13100.00);  update **savings set** cur\_balance **=**13100.00 where cust\_id ='110108197012190014';  insert into transaction\_history values('110108197509050018', '20180503093030',  'T', 200.00, 150300.00, 150500.00);  update **savings set** cur\_balance **=**150500.00 where cust\_id ='110108197509050018'; |

**（4）事务处理**

存款：开始事务，修改余额，增加流水帐，提交事务。

'110108197012190014'存了100进去。

|  |
| --- |
| set @@autocommit=0;  start transaction;  select cur\_balance into @cur\_balance from **savings** where cust\_id='110108197012190014';  set @old\_balance = @cur\_balance;  set @cur\_balance = @cur\_balance +100;  update **savings** set cur\_balance=@cur\_balance where cust\_id='110108197012190014';  set @cur\_datetime = date\_format(now(),'%Y%m%d%H%i%s');  insert into **transaction\_history** values('110108197012190014', @cur\_datetime,  'D', +100.00, @old\_balance, @cur\_balance);  commit;  set @@autocommit=1; |

对上述代码的解释如下：首先启动事务，然后读取当前的账户余额。接着更新账户余额，同时增加交易历史记录。最后，提交事务。

备注1：@打头的标识符是变量，比如@ cur\_datetime是我们自己用的一个变量，目的是记录当前日期时间，为了记录存钱是什么时候发生的。

备注2：“set @@autocommit=0;”表示由用户掌控事务提交，而不是每个sql单独作为一个事务提交，“set @@autocommit=1;”则恢复单个sql语句作为事务提交的系统缺省配置。

取款：开始事务，修改余额，增加流水帐，提交事务

'110108197108290016'账户取款**100。**

|  |
| --- |
| set @@autocommit=0;  start transaction;  select cur\_balance into @cur\_balance from **savings** where cust\_id='110108197012190014';  set @old\_balance = @cur\_balance;  set @cur\_balance = @cur\_balance -100;  update **savings** set cur\_balance=@cur\_balance where cust\_id='110108197012190014';  set @cur\_datetime = date\_format(now(),'%Y%m%d%H%i%s');  insert into **transaction\_history** values('110108197012190014', @cur\_datetime,  'W', -100.00, @old\_balance, @cur\_balance);  commit;  set @@autocommit=1; |

开户：开始事务，新增用户表，修改余额表，增加流水帐，提交事务

为身份证号为'110108197808080020'叫'wang tao'的一位男用户开户，存钱80000元。

|  |
| --- |
| set @@autocommit=0;  start transaction;  insert into customer values ('110108197808080020', 'wang tao', 'M', '19780808', 'beijing');  insert into savings values ('110108197808080020', '20180510', 80000.00);  insert into transaction\_history values('110108197808080020', '20180510093030',  'O', 80000.00, 0, 80000.00);  commit;  set @@autocommit=1; |

销户：请自行设计。

提示：开始事务，删除用户表，删除余额表，删除所有的流水帐，提交事务。

delete from customer where id=…;

delete from savings where cust\_id=…;

delete from transaction\_history where cust\_id=…;

如果这么做，历史记录就都没有了。

可以在用户表里面增加一个字段，为“账户激活状态”，销户的时候，把这个字段的值置为“N”即可。

查询交易明细：查询'110108197808080020'的'20180101'到'20180510'的所有交易明细。

|  |
| --- |
| select cust\_id, operation\_datetime, operation\_type, amount, last\_balance, cur\_balance  from transaction\_history  where cust\_id='110108197808080020'  and operation\_datetime >= '20180101' and operation\_datetime <='20180511'; |

备注：由于把日期时间表示为字符串，我们可以利用字符串的字典序，查询一定时间范围里面的交易记录，即operation\_datetime >= '20180101' and operation\_datetime <='20180510'所表达的意思，就是限定查找'20180101'到'20180510'之间的所有交易明细，由于是字符串比较，没有时间部分信息也没有关系，如果一定要加上，可以写成operation\_datetime >= '20180101000000' and operation\_datetime <='20180510235959'。

备注：由于只有一个sql语句，所以它自己就作为一个事务运行。无需我们指定事务开始和结束。

查询余额：查询'110108197808080020'的余额。

|  |
| --- |
| select cur\_balance  from savings  where cust\_id = '110108197808080020'; |

转账：开始事务，修改余额1，修改余额2，增加流水帐1，增加流水帐2，提交事务。

从'110108197012190014'到'110108197509050018'转100元钱。

为了确认转账成功，需要在下面转账事务之前、之后，通过“select cur\_balance from savings where cust\_id='110108197012190014' or cust\_id='110108197509050018'”来查询余额。

|  |
| --- |
| set @@autocommit=0;  start transaction;  select cur\_balance into @cur\_balance\_A from **savings** where cust\_id='110108197012190014';  set @old\_balance\_A = @cur\_balance\_A;  set @cur\_balance\_A = @cur\_balance\_A -100;  update **savings** set cur\_balance=@cur\_balance\_A where cust\_id='110108197012190014';  set @cur\_datetime = date\_format(now(),'%Y%m%d%H%i%s');  insert into **transaction\_history** values('110108197012190014', @cur\_datetime,  'T', -100.00, @old\_balance\_A, @cur\_balance\_A);  select cur\_balance into @cur\_balance\_B from **savings** where cust\_id='110108197509050018';  set @old\_balance\_B = @cur\_balance\_B;  set @cur\_balance\_B = @cur\_balance\_B +100;  update **savings** set cur\_balance=@cur\_balance\_B where cust\_id='110108197509050018';  set @cur\_datetime = date\_format(now(),'%Y%m%d%H%i%s');  insert into **transaction\_history** values('110108197509050018', @cur\_datetime,  'T', +100.00, @old\_balance\_B, @cur\_balance\_B);  commit;  set @@autocommit=1; |

转账和回滚：只要把转账事务中的commit改成rollback，就可以实现回滚。转账就没有发生，两个账户仍然是原来的钱。读者可以通过select cur\_balance from savings;语句来确认。

1. 实验要求

请大家仔细阅读实验大纲，借助参考资料自行完成实验操作，并完成实验报告。

考查方式：助教提出有关的操作，学生给助教现场编码演示助教提出的操作。

五、实验报告

学院 数据科学概论 课程实验报告

|  |  |  |
| --- | --- | --- |
| 实验题目： | | 学号：201500000000 |
| 日期：2019.3.20 | 班级：2015级1班/菁英班 | 姓名：张三 |
| Email：zhangsan@qq.com | | |
| 实验目的： | | |
| 实验软件和硬件环境： | | |
| 实验原理和方法： | | |
| 实验步骤：（不要求罗列完整源代码） | | |
| 结论分析与体会： | | |
| 就实验过程中遇到和出现的问题，你是如何解决和处理的，自拟1－3道问答题： | | |