

**计算机系统安全实验报告**

**实验三：数据库权限管理**

学　　 号 1190201308

班　　 级 1903202

学 生 陈东鑫

日 期 2021年12月8日

**1. 实验环境搭建**

操作系统 Win10

数据库 MySQL 8.0.26

编译器 IntelliJ IDEA 2021.2

语言 Java 8.0.291

**2. 实验步骤**

**2.1 一个简单的应用场景**

应用场景：银行收付款，系统包含以下功能

1. 用户： ①查看自身余额

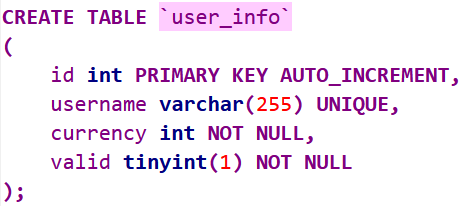
②存款/取款

1. 管理员： ①查看所有用户余额

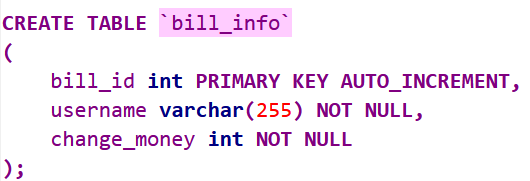
②批准/拒绝用户的存款/取款请求

**2.2 建立数据库**

建立表user\_info，存储所有用户的相关信息，包括id、用户名、密码、有效性等。脚本文件相关语句见下图。

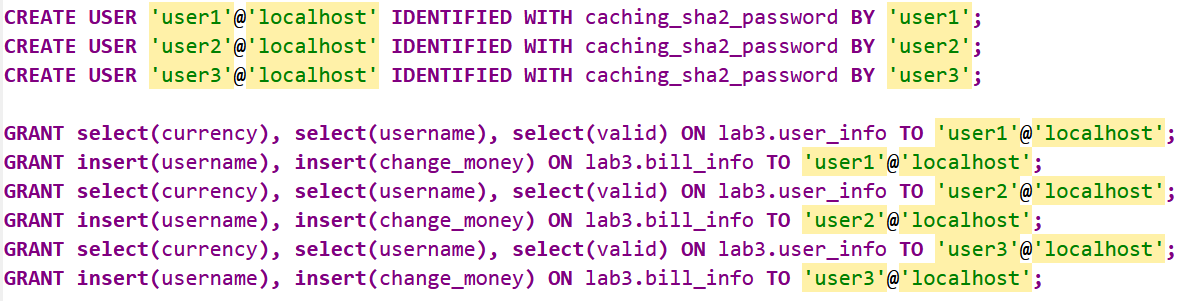


建立表bill\_info，存储所有由用户发出的存款/取款请求，字段包括账单id、请求用户名、请求转账数额。脚本文件相关语句见下图。



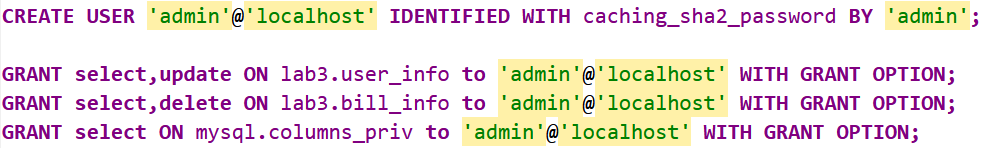
在数据库中创建两种用户：银行管理员(admin)和银行用户(user1\user2\user3)，密码均初始化为用户名

创建银行用户user\*：用户只能查看自己的余额，即只能对用户表的相关字段做查询，但不能对用户表做任何其他操作。



如上图，用户只能对user\_info的用户名、余额、有效性进行查询，不能做插入、删除和修改等操作；只能对bill\_info的用户名、请求钱数进行插入，不能进行查看、修改和删除，其他部分如id由数据库进行自增长，且不能进行查看，放置泄露银行用户数量数据。

创建银行管理员用户admin：用于查看所有账户信息和批准存款/取款请求，故管理员可以修改用户表的余额字段，并且能够在处理完请求后删除请求表的记录。



如上图，管理员能够查询、修改user\_info中用户的数据，以便修改用户余额；能够查询和删除bill\_info中的数据，以便处理用户请求并且删除处理后的请求信息。

数据库具体构建信息，见SQL脚本文件db.sql。

**2.3 实验流程**

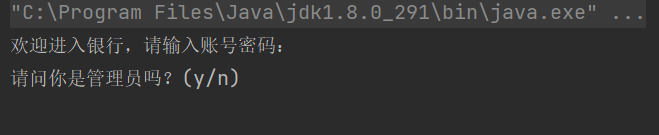
1. 登录

登陆时需要分为两种身份进行登录，即管理员或普通用户，编写两个类Admin和Client，当登陆输入账号密码时，对两个类进行实例化，并使用login方法进行登录。

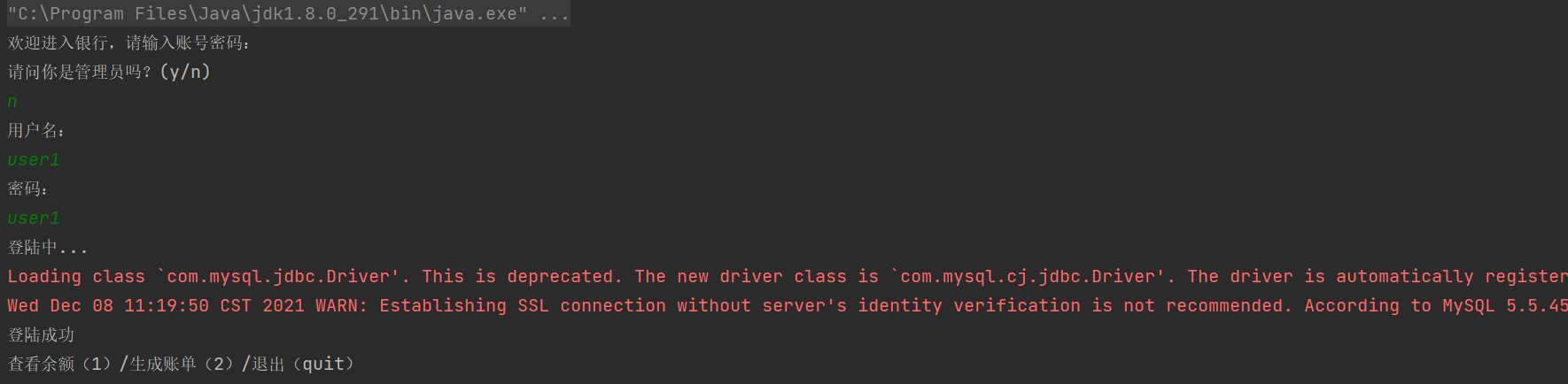
例如，管理员身份登陆时：

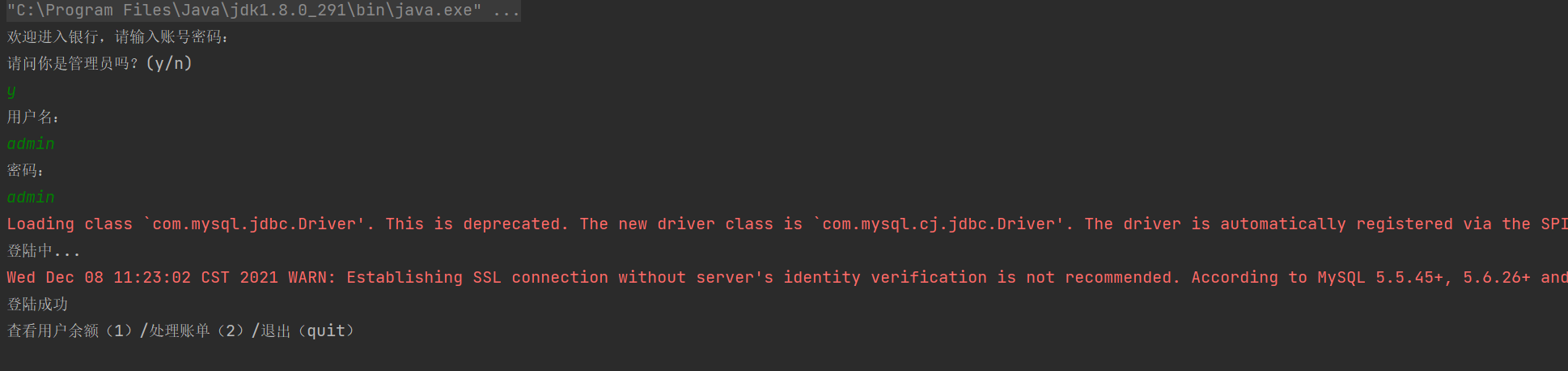
Admin admin = new Admin(username, password);  
if(admin.login()){  
 while (true) {  
 System.*out*.println("查看用户余额（1）/处理账单（2）/退出（quit）");  
 ack = in.nextLine();  
 if (ack.equals("1")) {  
 admin.lookupUsers();  
 } else if (ack.equals("2")) {  
 admin.dealBills();  
 } else if (ack.equals("quit")||ack.equals("q")) {  
 break;  
 }  
 }  
}

则在登陆时需要确认是管理员还是普通用户：



用户和管理员登录时菜单功能不同：



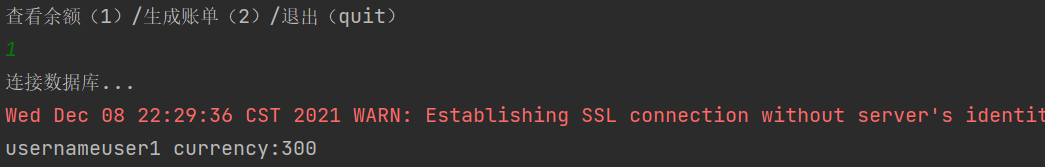


当普通用户冒充管理员或管理员冒充普通用户登陆时，由于没有权限，使用菜单中的功能时会因为越权而被数据库拒绝，可以及时发现。

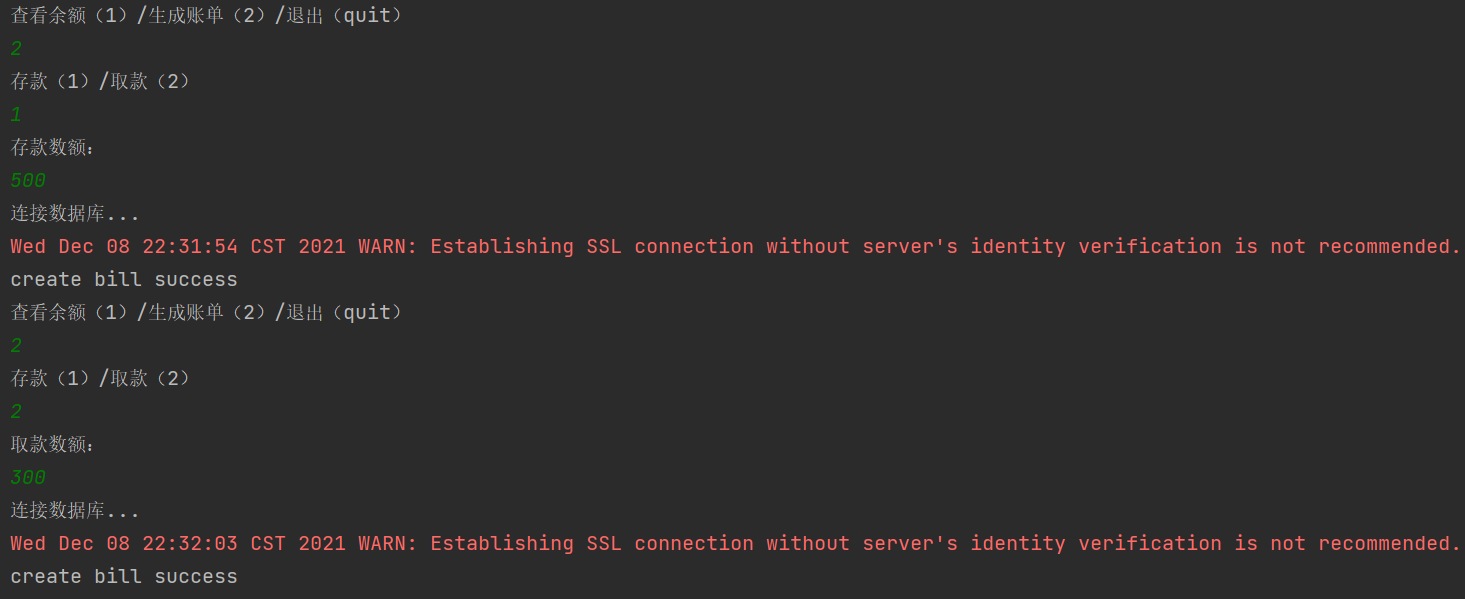
1. 用户查询自身余额和存款/取款

用户Client类拥有lookup和dealBills方法：lookup方法查看user\_info表中自身账户余额，而dealBills方法向bill\_info中插入转账请求。

①查看余额：



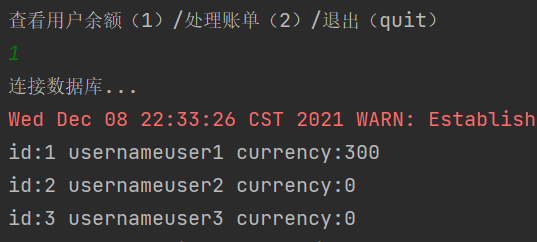
②生成存款/取款请求：



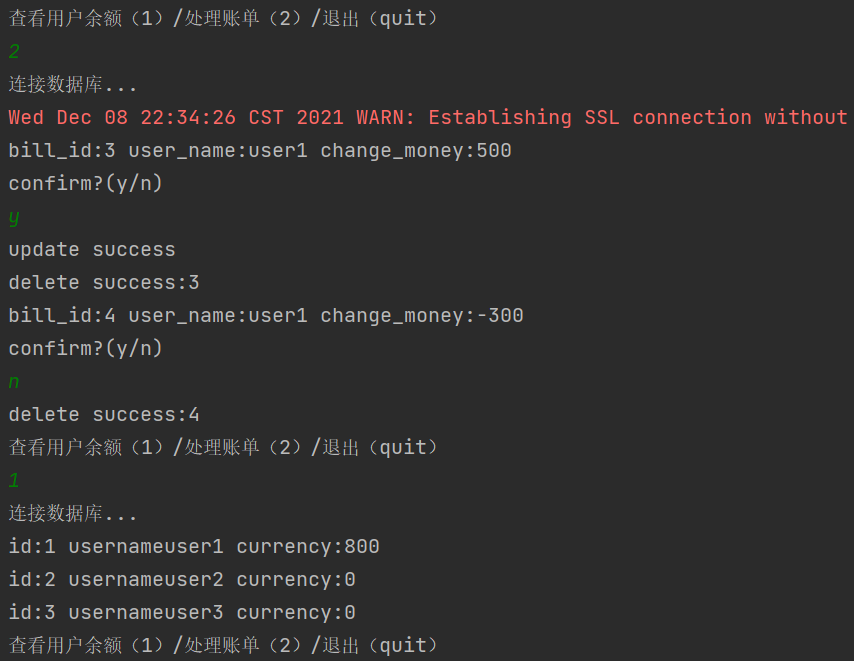
1. 管理员查询所有用户余额和批量处理存款/取款请求

管理员Admin类有lookupUsers和dealBills方法：lookupUsers方法查看所有用户的余额，而dealBills方法批量处理bill\_info中的请求记录。

①查看所有账户余额：



②批准/拒绝来自普通用户的存款/取款请求：



1. 日志文件编写

编写Log类，在程序开始时，初始化向日志文件Log.txt的输出流。

try {  
 File file = new File("log.txt");  
 FileOutputStream output = new FileOutputStream(file, true);  
 Log.*output* = output;  
} catch (IOException e) {  
 System.*out*.println("open log.txt failed");  
}

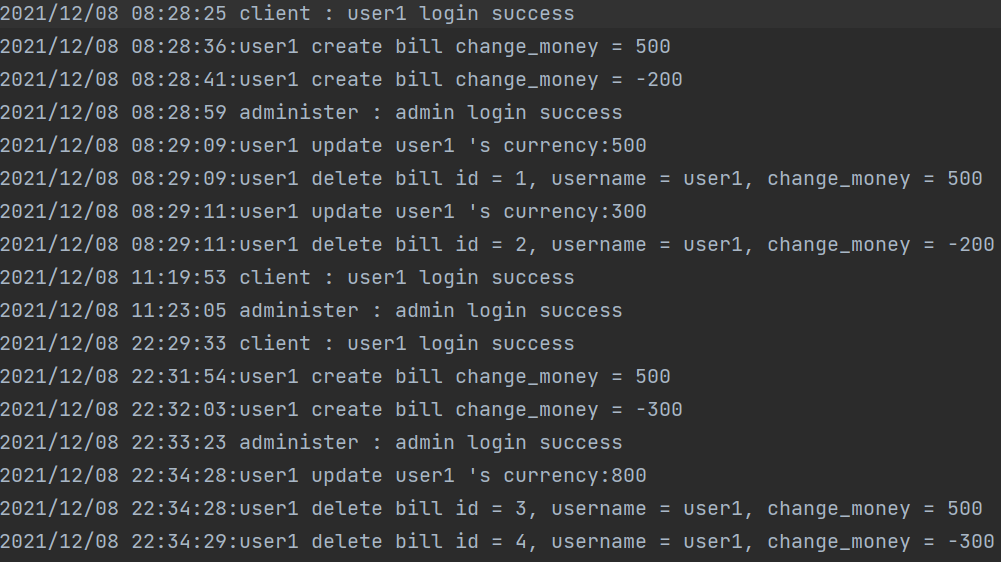
得到当前时间:

Date now = new Date();  
SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy/MM/dd HH:mm:ss");  
String current\_time = dateFormat.format(now);

将输出流作为static成员变量，在写入日志时直接调用类的输出流进行写入。

try {  
 Log.*output*.write((current\_time + ":" + username + " delete bill id = " + bill\_id + ", username = " + username + ", change\_money = " + change\_money + "\n").getBytes(StandardCharsets.*UTF\_8*));  
} catch (IOException e) {  
 System.*out*.println("写入失败");  
}

查看Log.txt文件：



**4. 心得体会**

掌握了MySQL数据库基本权限管理命令、SQL语言以及数据库系统的设计、数据库用户权限管理的实现，学习了在特定场景下对数据表进行创建和管理、对数据库用户的创建和合理的权限分配，权限分配细化到数据库、表、列。