openGauss 场景化综合实验(金融场景)实验报告

姓名： 王娇妹 学号： 2012679

实验步骤：

* 前置环境设置
* 创建数据表
* 插入表数据
* 手工插入一条数据
* 添加约束
* 查询数据
* 视图
* 索引
* 数据的修改和删除
* 新用户的创建和授权
* 新用户连接数据库
* 删除Schema

实验报告

一、实验环境说明

此次实验我使用openGauss实验环境。选择openGauss的原因：

1、数据库系统这门课的前几次实验大多都是采用openGauss进行，我对openGauss更加熟悉，而且实验手册是以openGauss为例进行说明的。

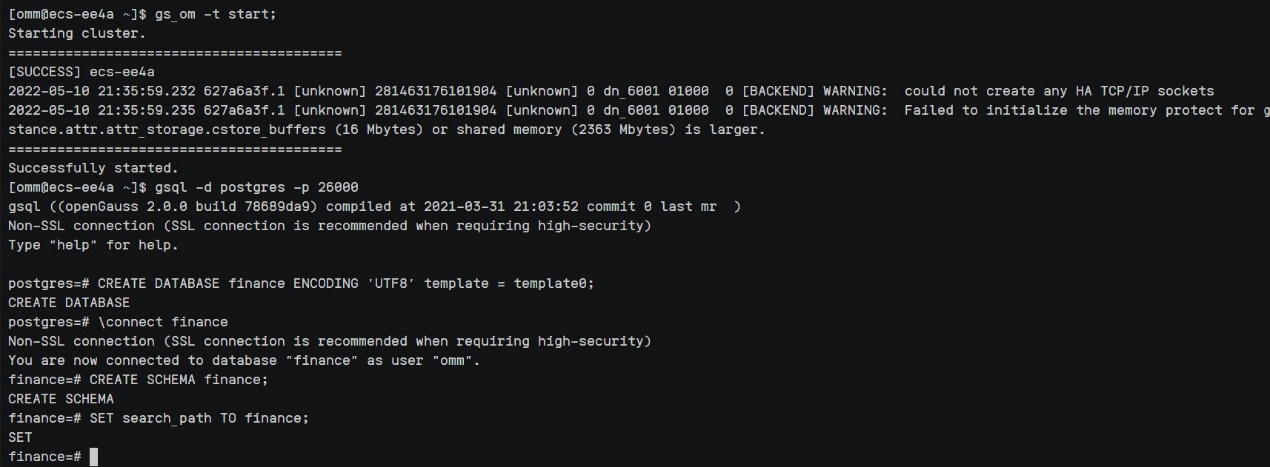
2、在实验前浏览实验指导手册时，手册里提到“基于GaussDB的操作可能没有权限进行1.1.11的实验步骤”，我想尝试一下1.1.11~1.1.13的实验，所以选择openGauss。

二、1.1.3-1.1.13执行结果截图

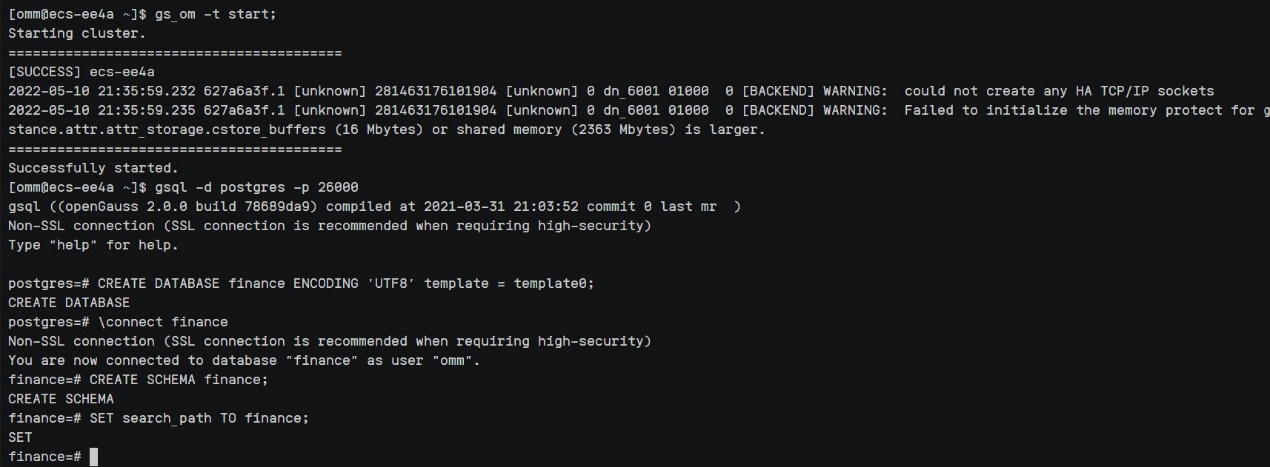
1.1.3 创建数据表

步骤 1 创建金融数据库finance。

使用gsql工具登陆数据库。

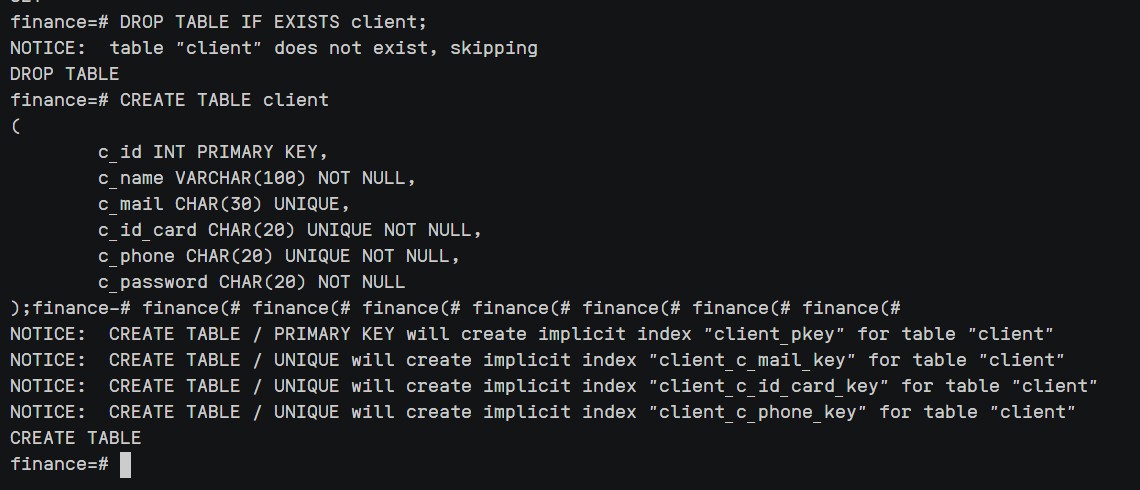


创建数据库financeL并连接，创建名为finance的schema，并设置finance为当前的schema，将默认搜索路径设为finance。



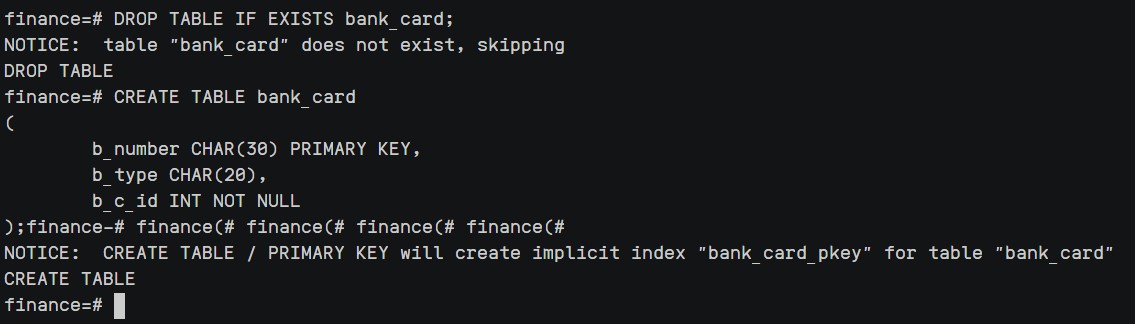
步骤 2 客户信息表的创建。

创建客户信息表client。



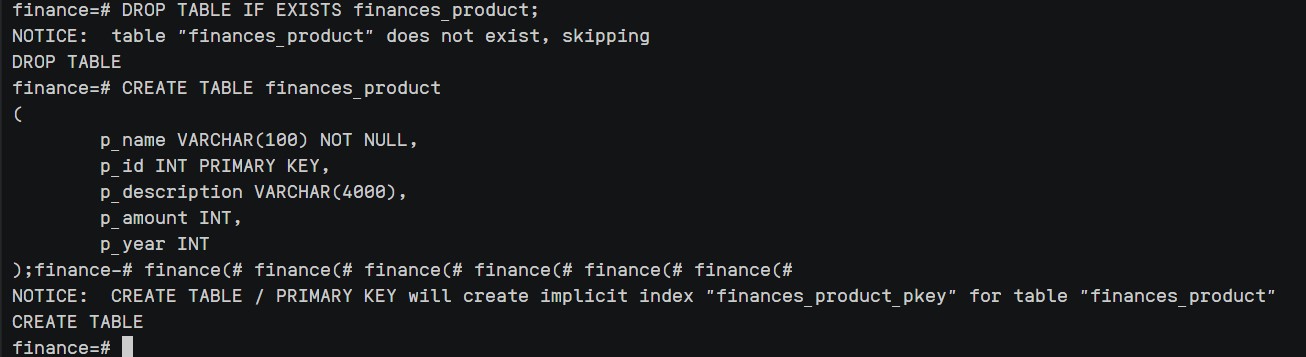
步骤 3 银行卡信息表的创建。

创建银行卡信息表bank\_card。



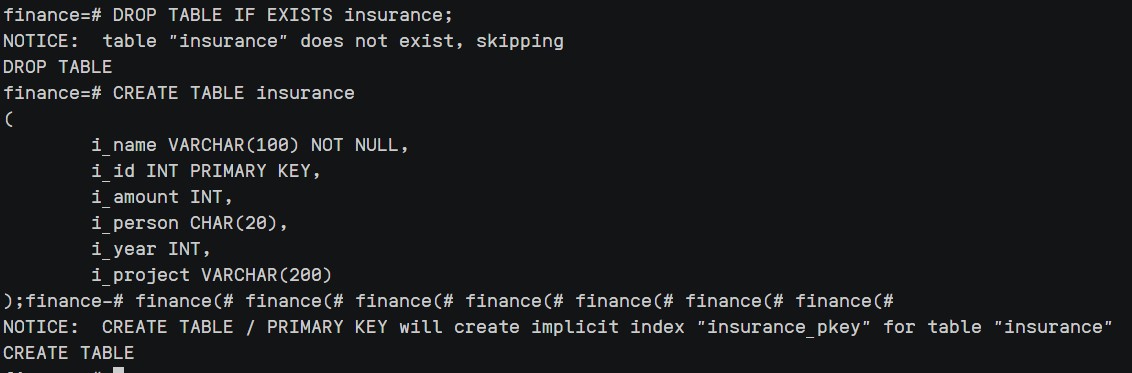
步骤 4 理财产品信息表的创建。

创建理财产品信息表finances\_product。



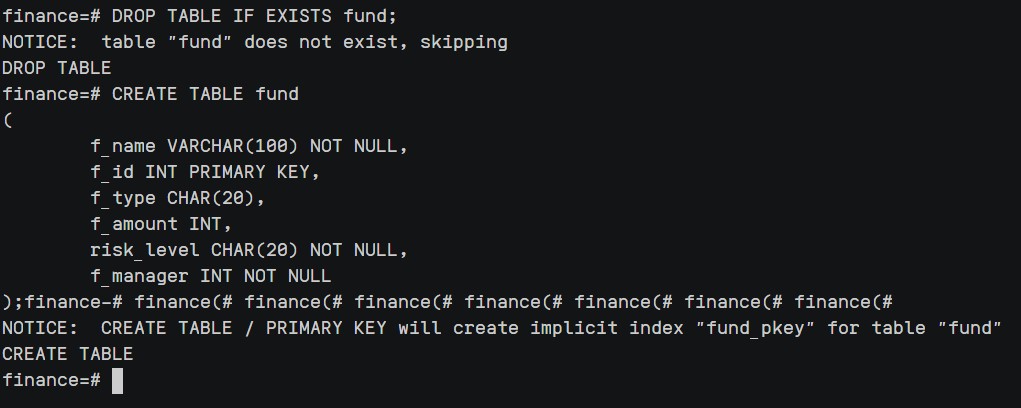
步骤 5 保险信息表的创建。

创建保险信息表insurance。



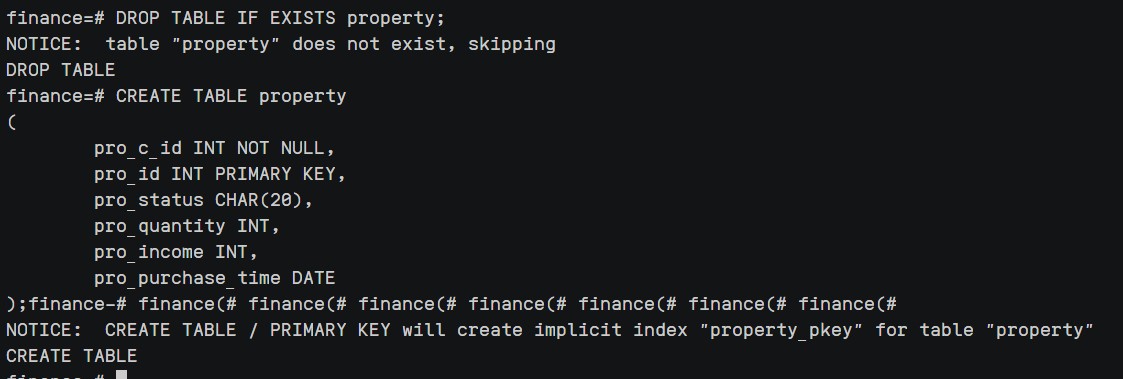
步骤 6 基金信息表的创建。

创建保险信息表fund。



步骤 7 资产信息表的创建。

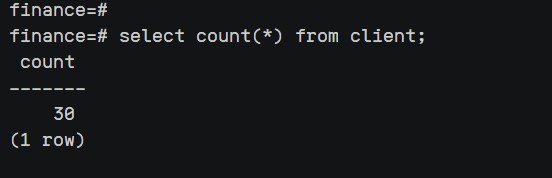
创建资产信息表property。



1.1.4 插入表数据

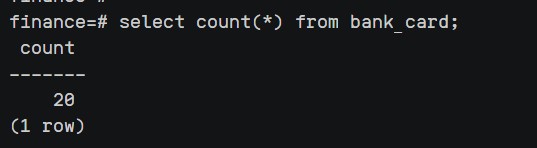
步骤 1 对client表进行数据初始化。

执行insert操作，查询插入结果。



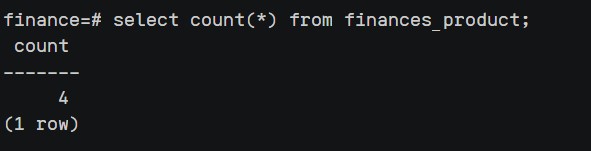
步骤 2 对bank\_card表进行数据初始化。

执行insert操作，查询插入结果。



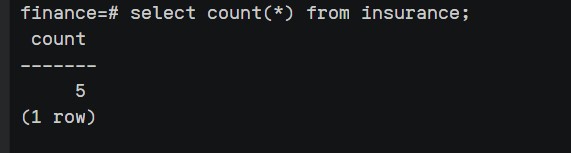
步骤 3 对finances\_product表进行数据初始化。

执行insert操作，查询插入结果。



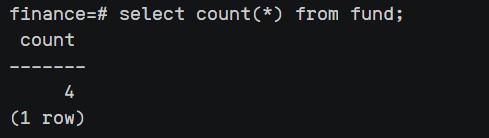
步骤 4 对insurance表进行数据初始化。

执行insert操作，查询插入结果。



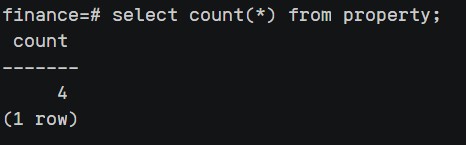
步骤 5 对fund表进行数据初始化。

执行insert操作，查询插入结果。



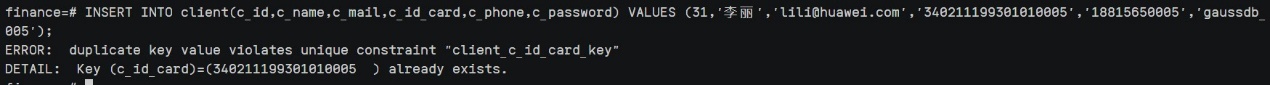
步骤 6 对property表进行数据初始化。

执行insert操作，查询插入结果。

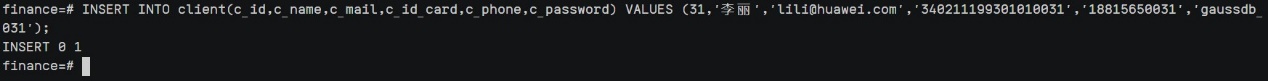


1.1.5 手工插入一条数据

步骤 1 在金融数据库的客户信息表中添加一个客户的信息。（属性冲突的场景）



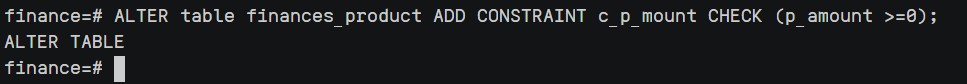
步骤 2 在金融数据库的客户信息表中添加一个客户的信息。（插入成功的场景）。



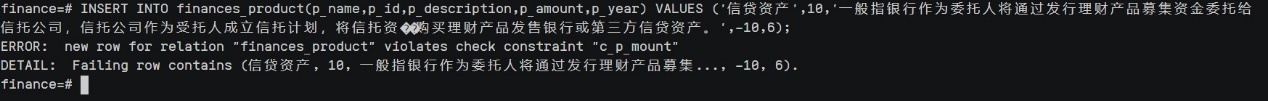
1.1.6 添加约束

步骤 1 在理财产品表、保险信息表和基金信息表中，都存在金额这个属性，在现实生活中，金额不会存在负数。因此针对表中金额的属性，增加大于0的约束条件。

为finances\_product表的p\_amount列添加大于等于0的约束。

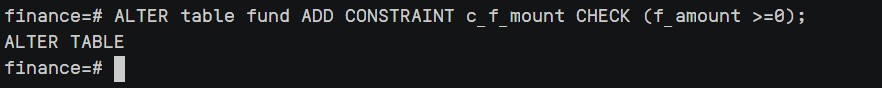


步骤 2 尝试手工插入一条金额小于0的记录。



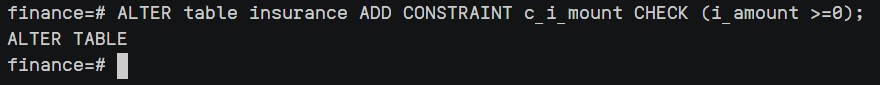
步骤 3 向fund表添加约束。

为fund表的f\_amount列添加大于等于0的约束。



步骤 4 向insurance表添加约束。

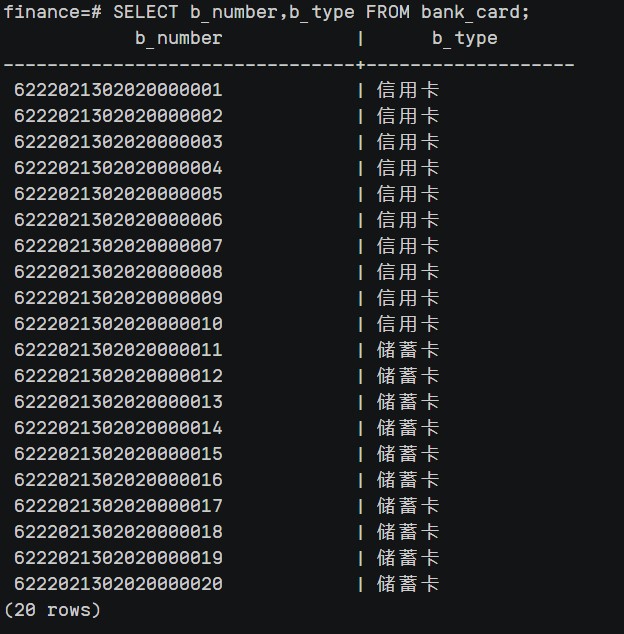
为insurance表的i\_amount列添加大于等于0的约束。



1.1.7 查询数据

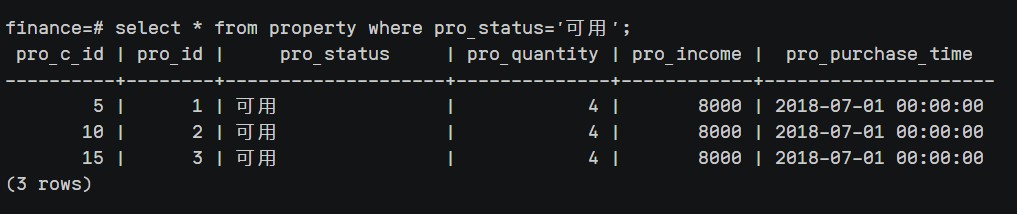
步骤 1 单表查询。

查询银行卡信息表。



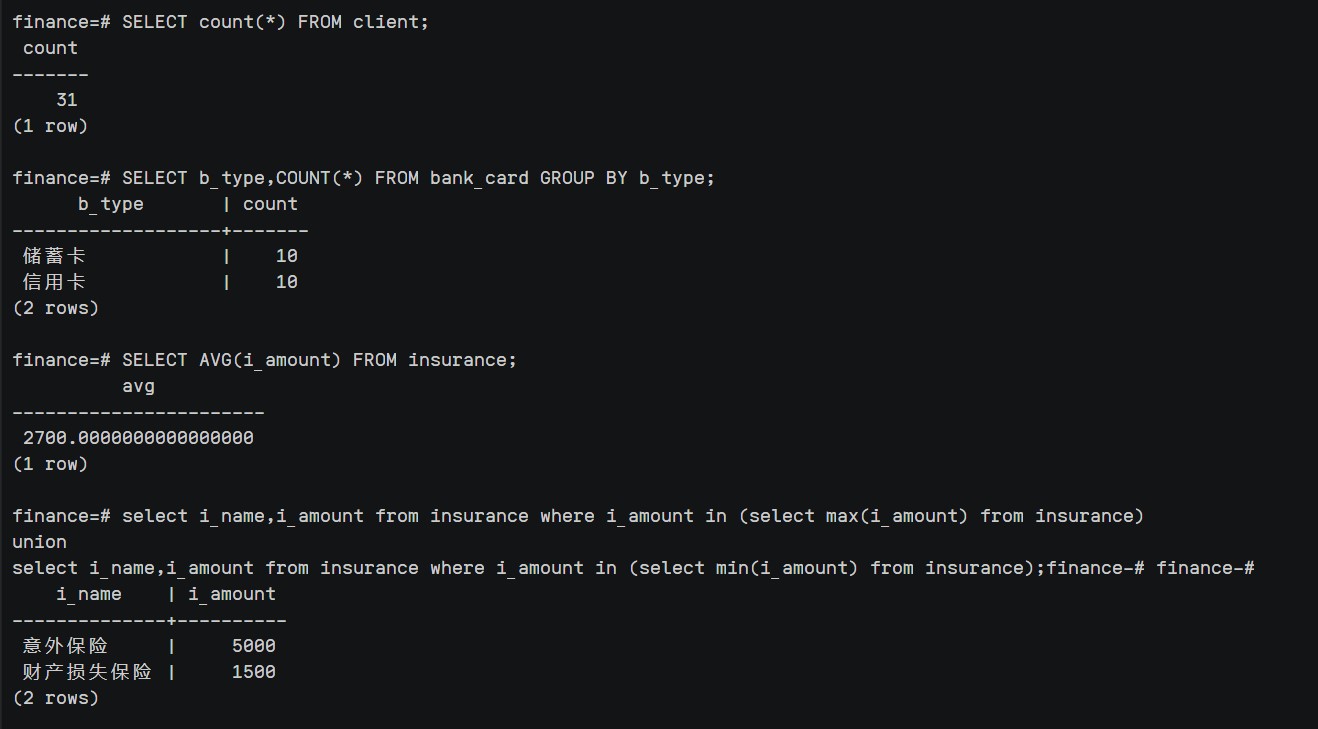
步骤 2 条件查询。

查询资产信息中‘可用’的资产数据。

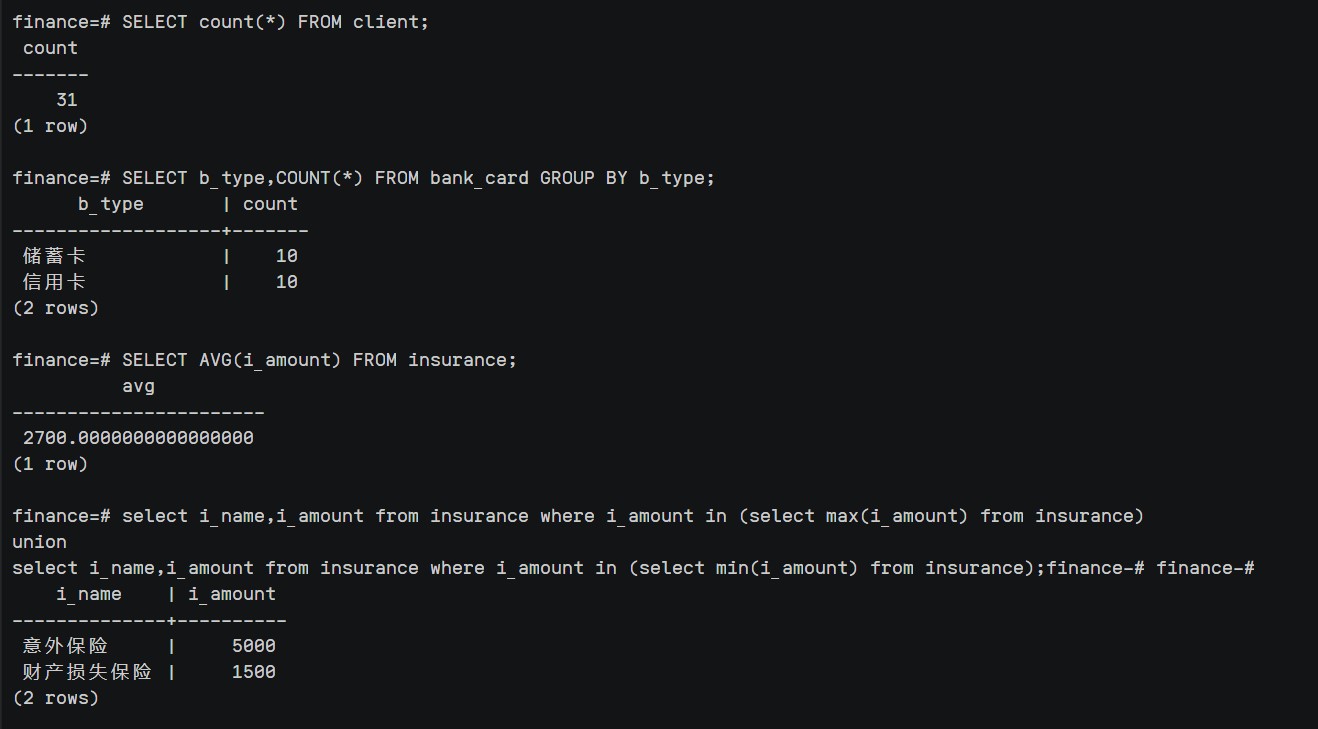


步骤 3 聚合查询。

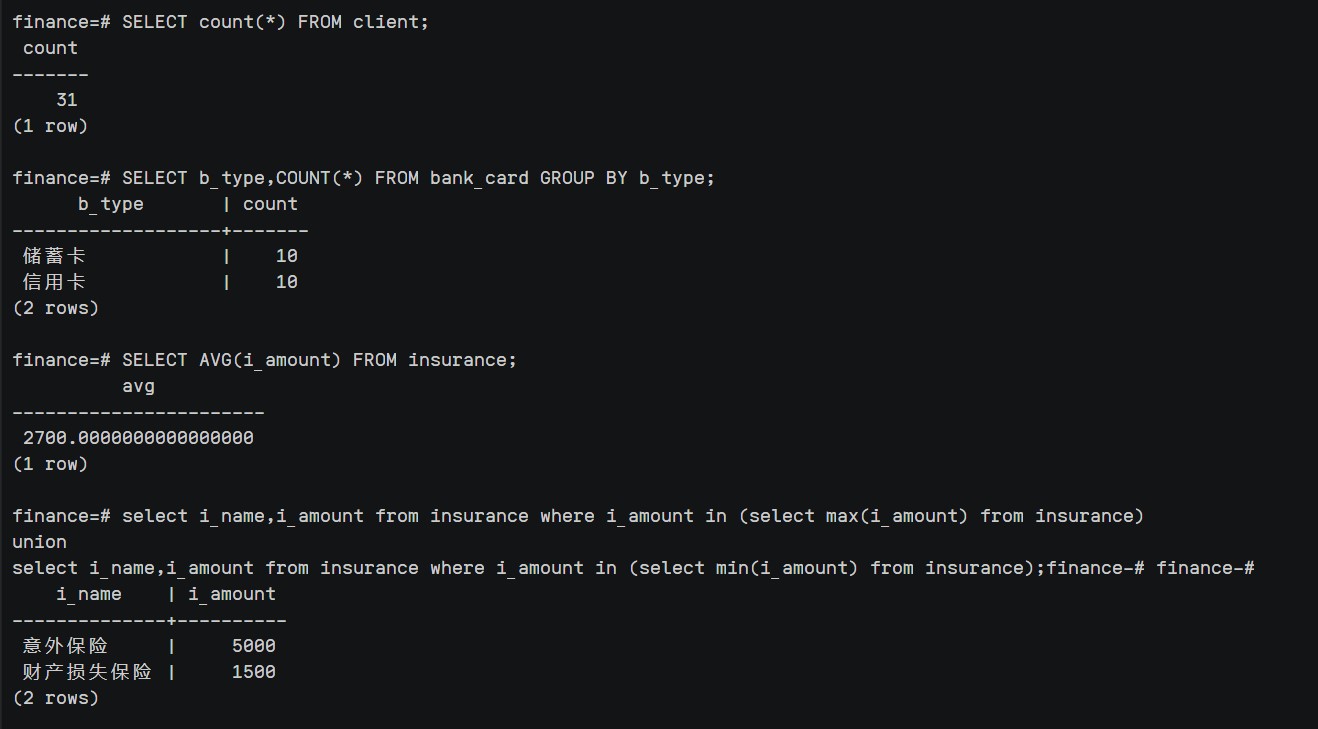
查询用户表中有多少个用户。



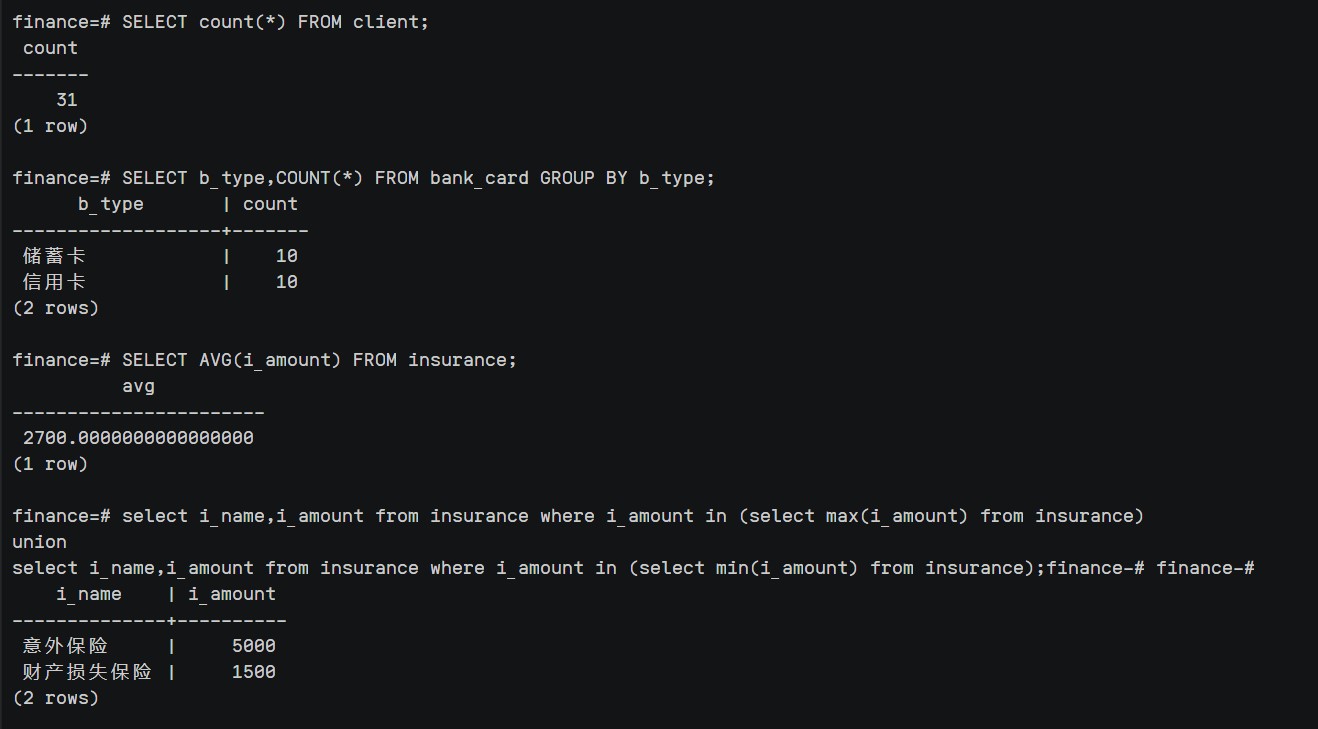
查询银行卡信息表中，储蓄卡和信用卡的个数。



查询保险信息表中，保险金额的平均值。



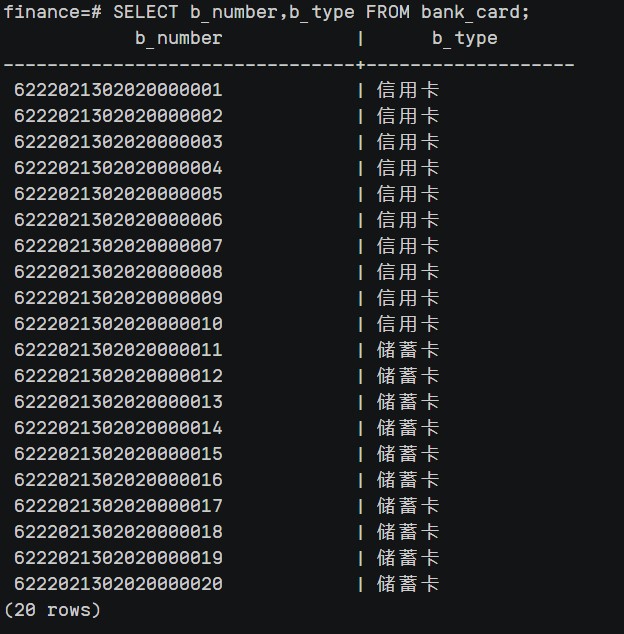
查询保险信息表中保险金额的最大值和最小值所对应的险种和金额。



步骤 4 连接查询。

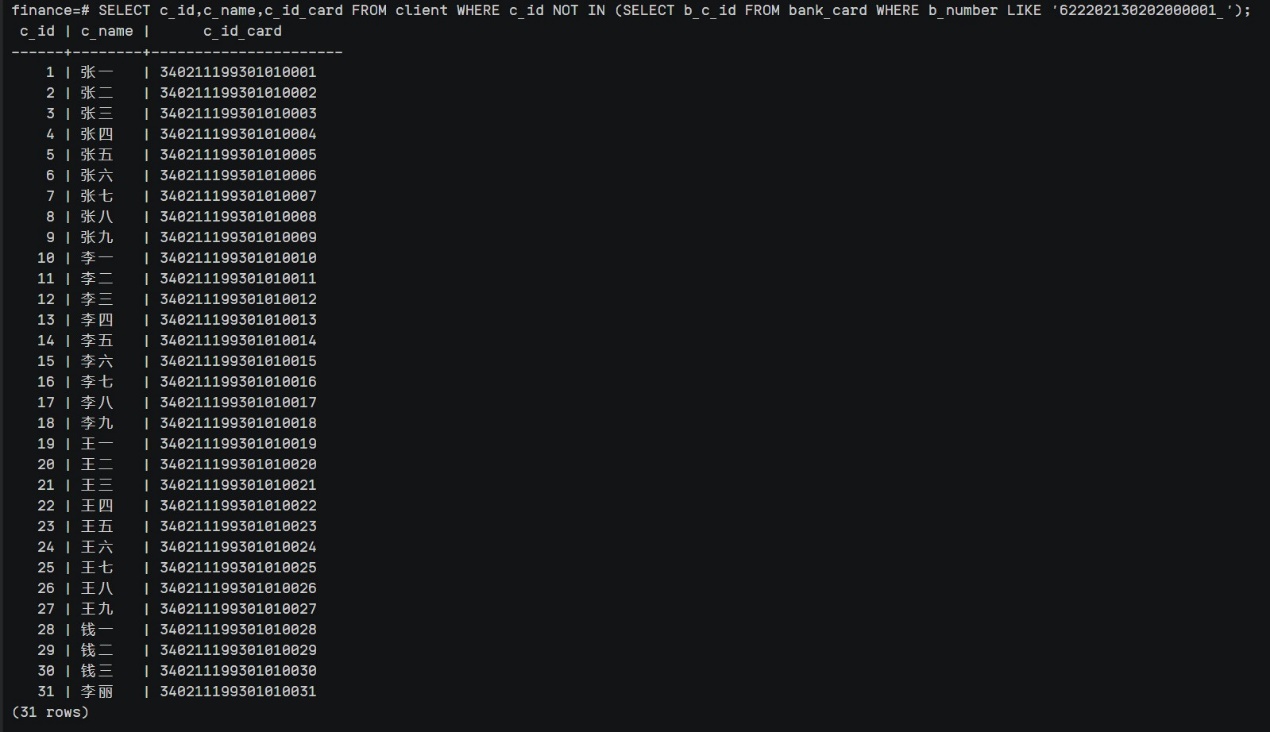
（1）半连接。

查询用户编号在银行卡表中出现的用户的编号，用户姓名和身份证。



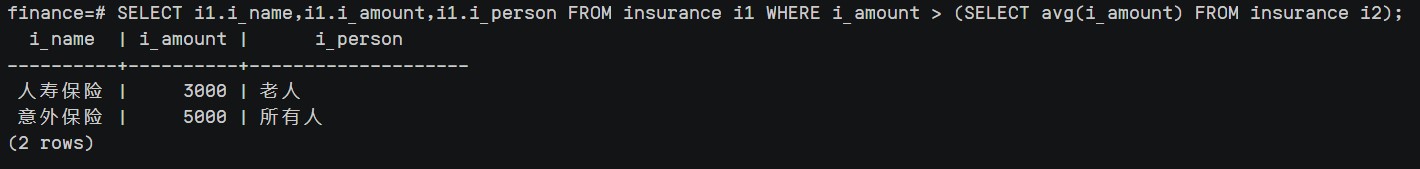
（2）反连接。

查询银行卡号不是‘622202130202000001\*‘的用户的编号，姓名和身份证。



步骤 5 子查询。

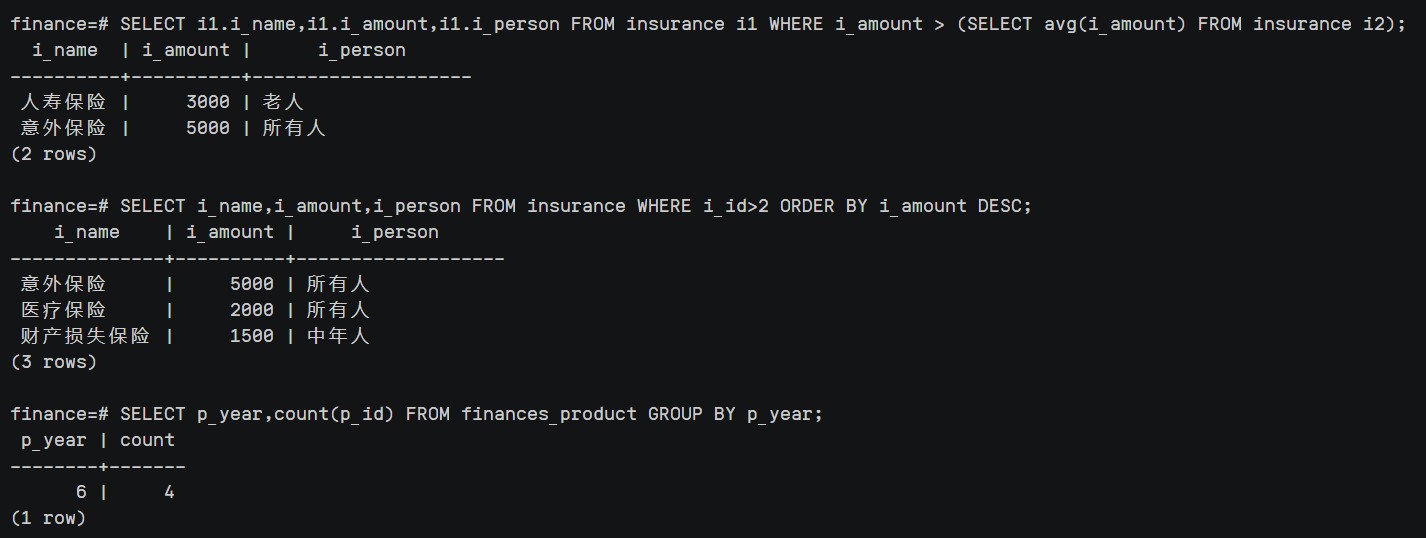
通过子查询，查询保险产品中保险金额大于平均值的保险名称和适用人群。



步骤 6 ORDER BY和GROUP BY。

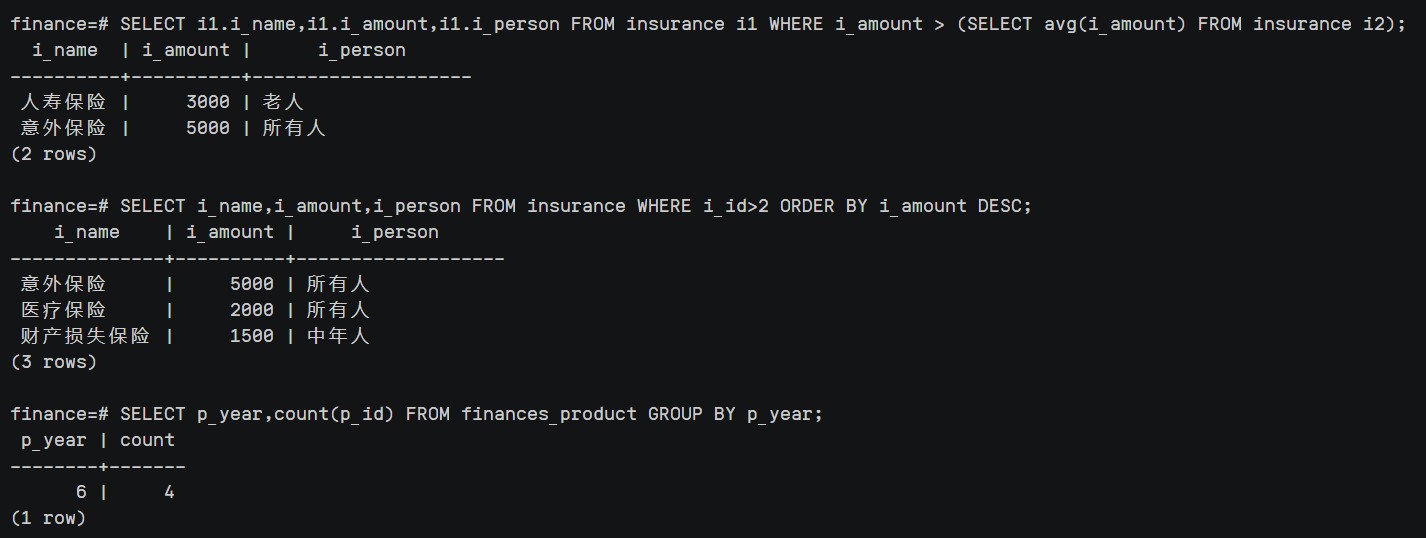
（1）ORDER BY子句。

按照降序查询保险编号大于2的保险名称，保额和适用人群。



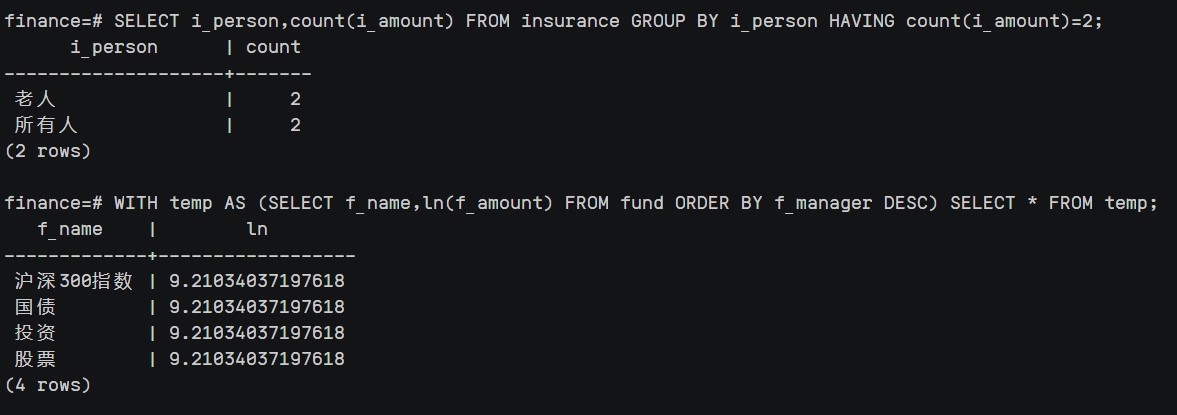
（2）GROUP BY子句。

查询各保险信息总数，按照p\_year分组。



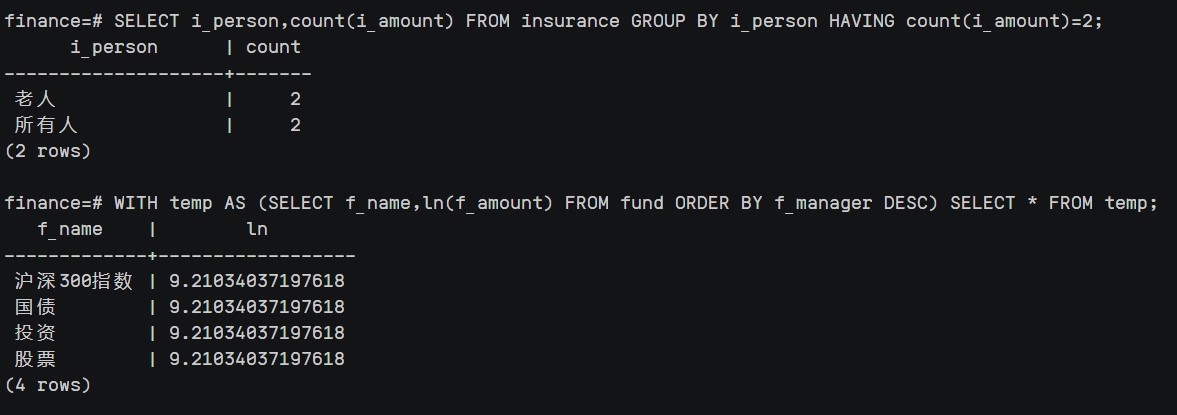
步骤 7 HAVING和WITH AS。

（1）HAVING子句。



（2）WITH AS子句。

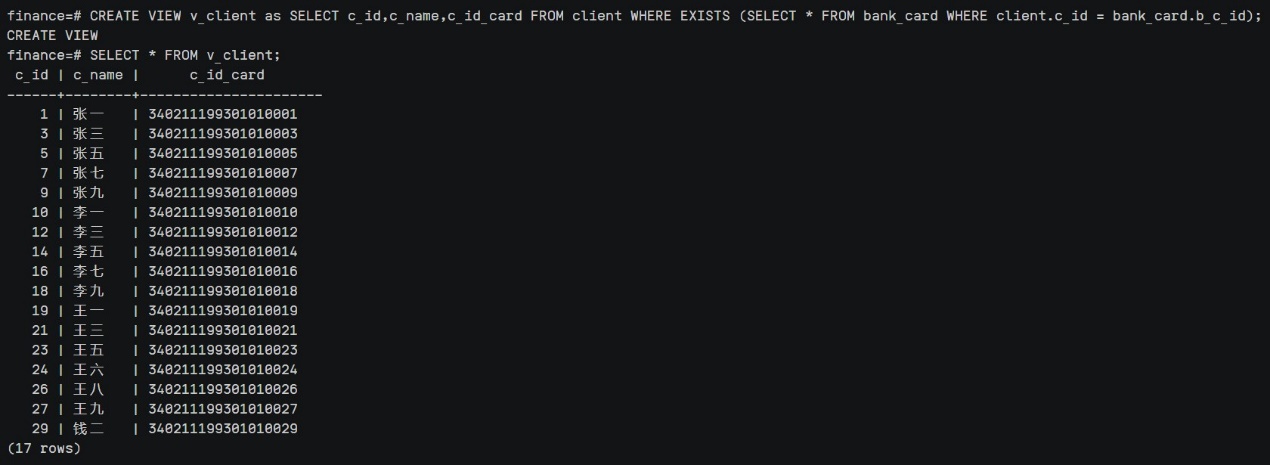
使用WITH AS查询基金信息表。



1.1.8 视图

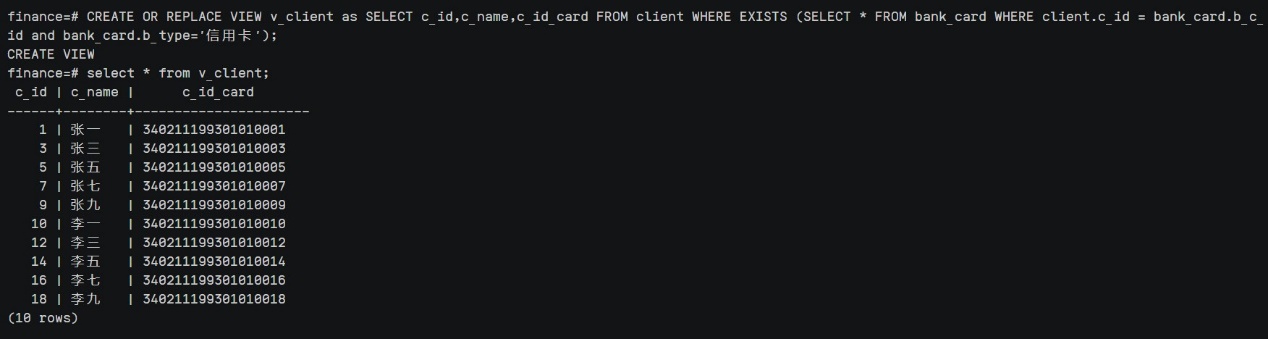
步骤 1 创建视图。

针对“查询用户编号在银行卡表中出现的用户的编号，用户姓名和身份证” 的查询，创建视图。使用视图进行查询。

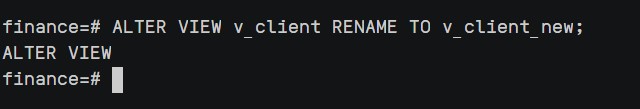


步骤 2 修改视图内容

修改视图，在原有查询的基础上，过滤出信用卡用户。使用视图进行查询。

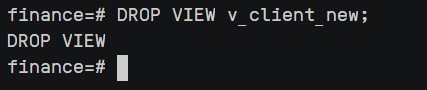


步骤 3 修改视图名称。



步骤 4 删除视图。

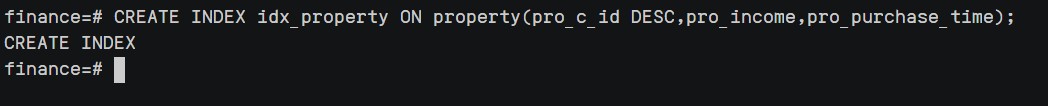
将v\_client视图删除，删除视图不影响基表。



1.1.9 索引

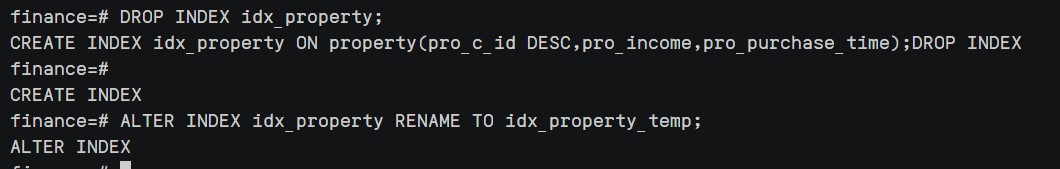
步骤 1 创建索引。

在普通表property上创建索引。



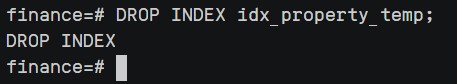
步骤 2 重命名索引。

在普通表property上重建及重命名索引。



步骤 3 删除索引。

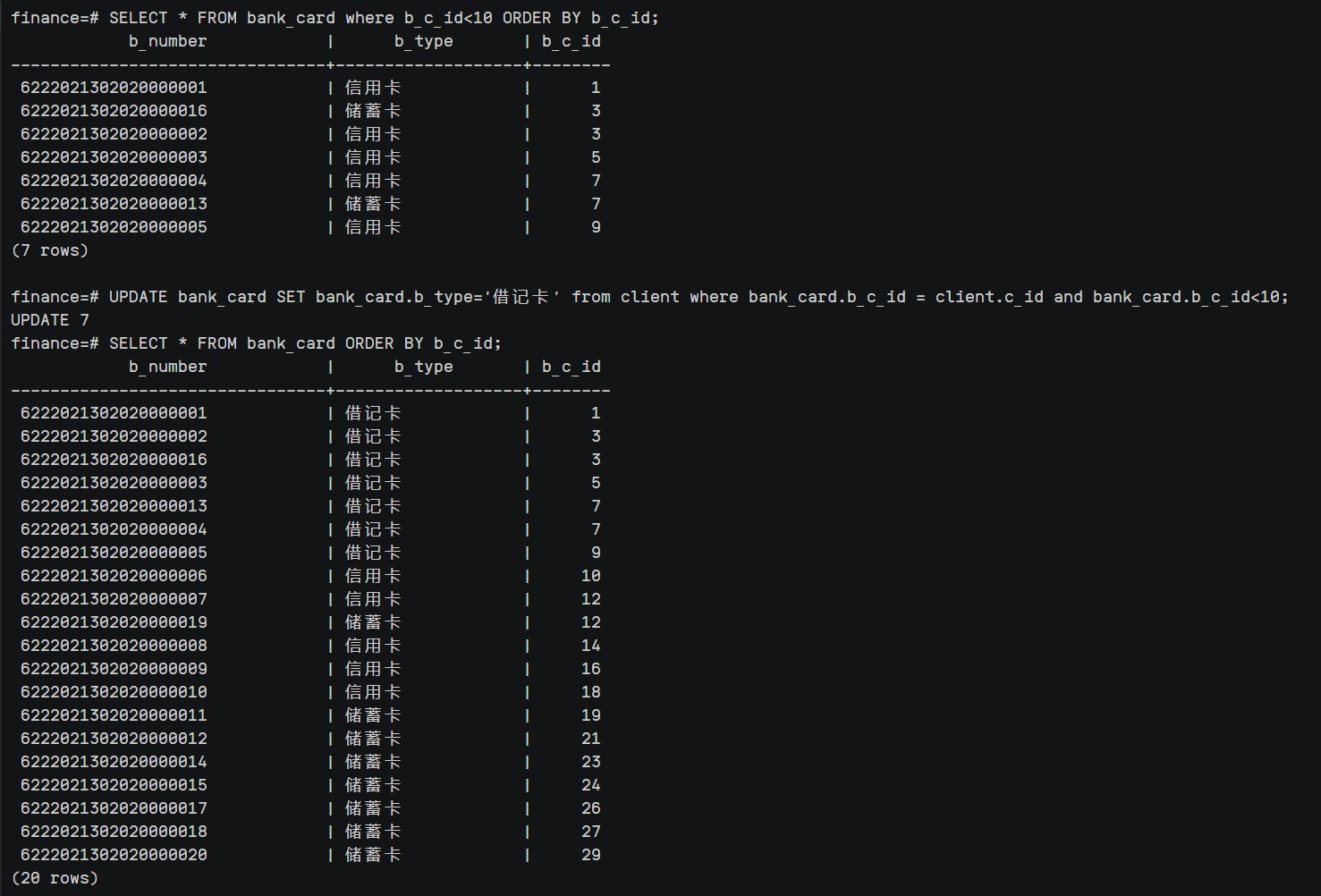
删除索引idx\_property\_temp。



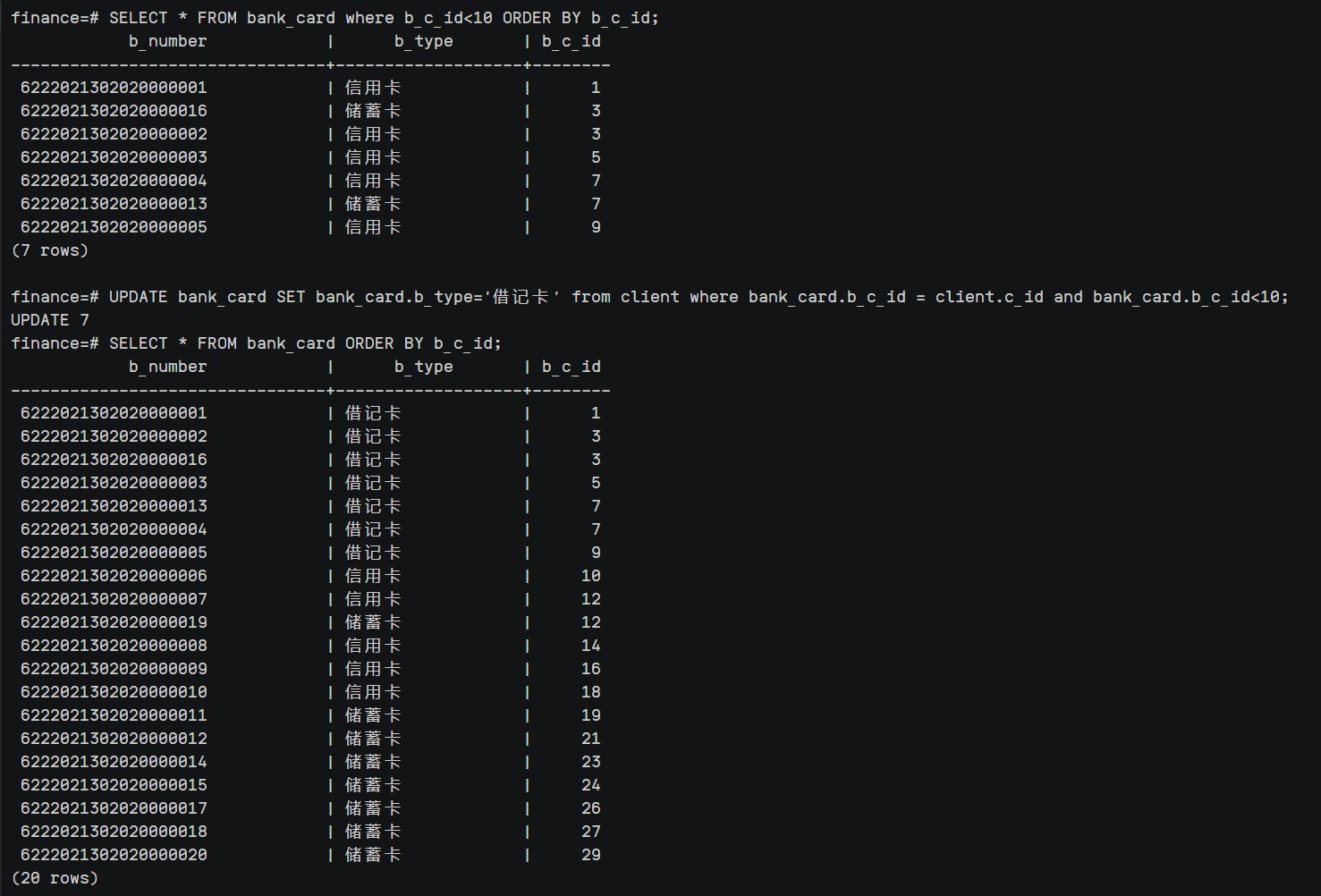
1.1.10 数据的修改和删除

步骤 1 修改数据。

修改/更新银行卡信息表中b\_c\_id小于10和客户信息表中c\_id相同的记录的b\_type字段。查看表数据。



更新数据，重新查询数据情况。



步骤 2 删除指定数据。

删除基金信息表中编号小于3的行，删除前查询结果。



开始删除数据



查询删除结果。

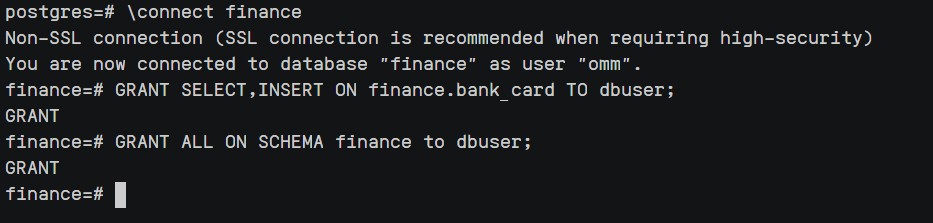


1.1.11 新用户的创建和授权

步骤 1 连接数据库后，进入SQL命令界面。创建用户dbuser，密码为Gauss#3demo。



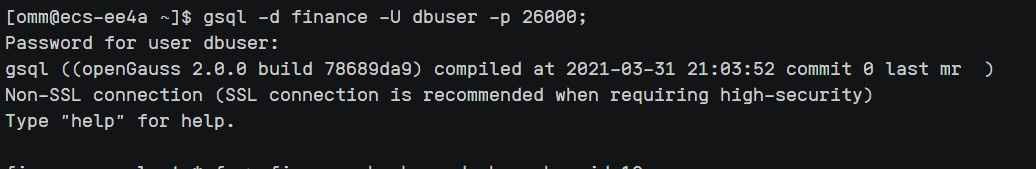
步骤 2 给用户dbuser授予finance数据库下bank\_card表的查询和插入权限，并将SCHEMA的权限也授予dbuser用户。



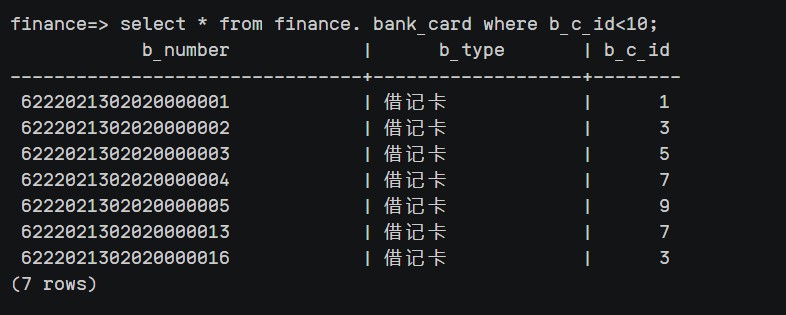
1.1.12 新用户连接数据库

步骤 1 在gsql登录数据库，使用新用户连接。

使用操作系统omm用户在新的窗口登陆并执行以下命令，并输入对应的密码。



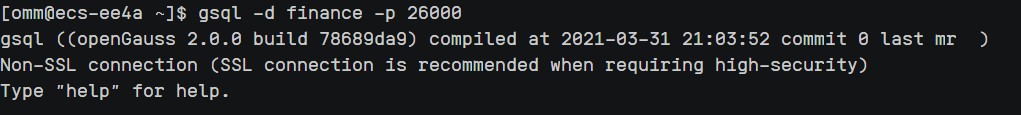
步骤 2 访问finance数据库的表bank\_card。



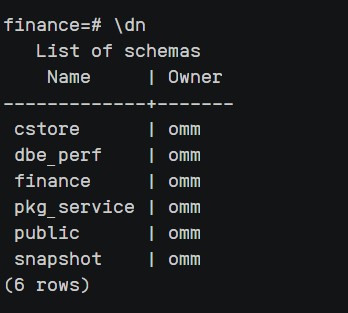
1.1.13 删除Schema

步骤 1 使用管理员用户登陆finance数据库。

使用操作系统omm用户使用gsql，新建session。



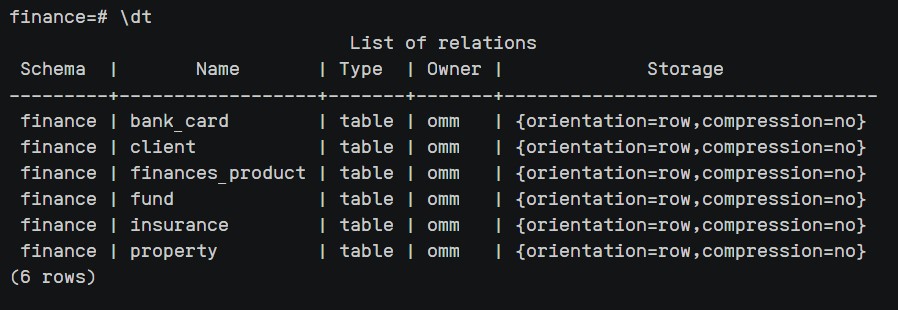
步骤 2 使用“\dn”查看数据库下的schema。



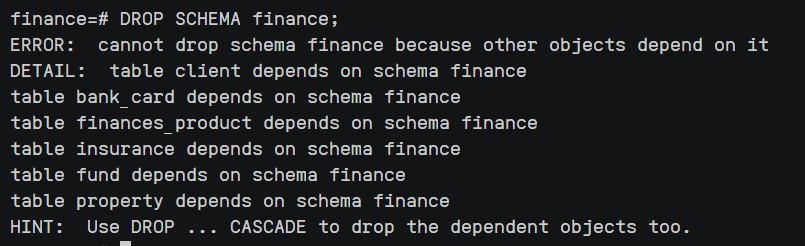
步骤 3 设置默认查询为finance。



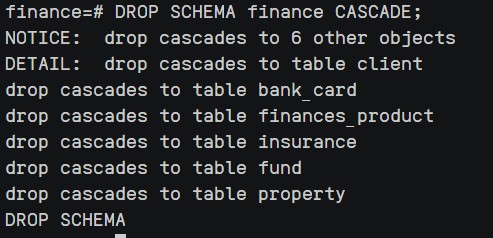
步骤 4 使用“\dt”命令可以看到在finance中的对象。



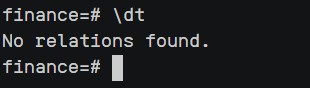
步骤 5 使用DROP SCHEMA 命令删除finance会有报错，因为finance下存在对象。



步骤 6 使用DROP SCHEMA…..CASCADE删除，会将finance连同下的对象一起删除。



步骤 7 使用“\dt”命令可以看到在finance和public中的对象，对象已删除。



三、对于能够用关系代数表达的SQL执行需求，请写出对应小节及步骤的关系代数查询语句（标明小节和步骤以表示对应关系）。

1.1.7 查询数据

步骤 1 单表查询：查询银行卡信息表。

SELECT b\_number,b\_type FROM bank\_card;

关系代数查询语句：

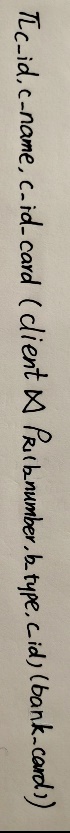
步骤 2 条件查询：查询资产信息中‘可用’的资产数据。

select \* from property where pro\_status='可用';

关系代数查询语句：

步骤 4 连接查询。

1. 半连接：查询用户编号在银行卡表中出现的用户的编号，用户姓名和身份证。

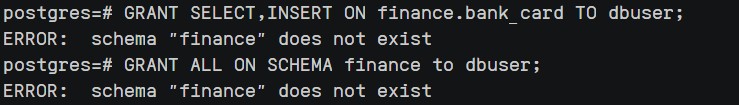
SELECT c\_id,c\_name,c\_id\_card FROM client WHERE EXISTS (SELECT \* FROM bank\_card WHERE client.c\_id = bank\_card.b\_c\_id);

关系代数查询语句：

四、如果你的初始SQL执行结果和要求的执行结果不符，其原因是什么？请就和要求结果不符的SQL执行内容分别进行说明。

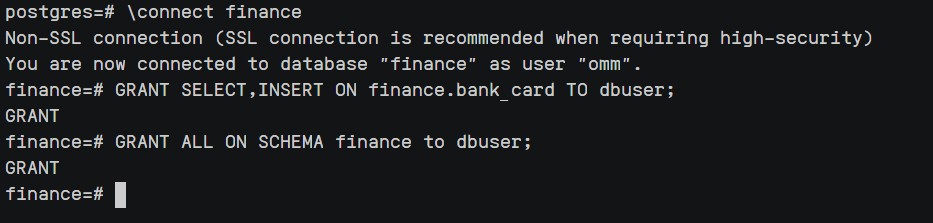
在此次实验中，我有两处SQL执行结果和要求的执行结果不符。

1、在执行1.1.11的步骤 2“给用户dbuser授予finance数据库下bank\_card表的查询和插入权限，并将SCHEMA的权限也授予dbuser用户”操作时，报错schema“finance”不存在。如下图所示：



经过查询资料，我发现是因为在1.1.11中，我重新登陆了数据库，但是没有连接finance数据库，所以会提示schema“finance” does not exist。

输入“\connect finance”命令后，操作可以正常进行。

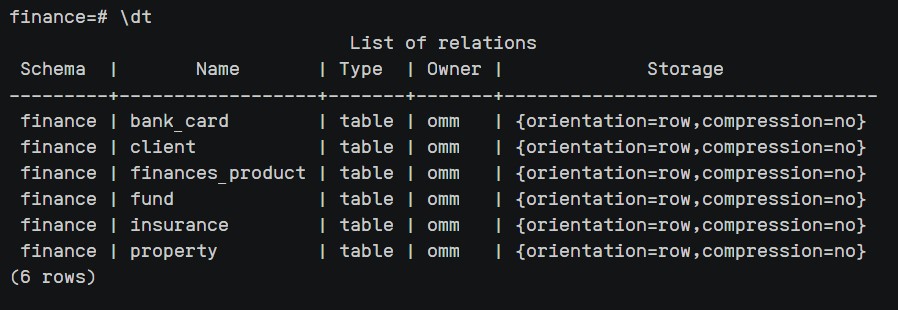


2、在执行1.1.13的步骤 4“使用‘\dt’命令可以看到在finance中的对象”操作时，报错No relation found。如下图：



我发现这是漏掉了步骤3“设置默认查询为finance”操作导致的。因为没有设置默认搜索路径，所以平台无法判断\dt命令查看谁的对象，产生错误。

输入“set search\_path to finance;”命令后，用“\dt”可以查询到finance中的对象。



五、实验总时长分析及遇到的问题、以及实验中学习到的知识点分析。

本次实验总用时58分钟。

通过这次实验，我发现自己对于SQL语句并不是很熟悉，一些SQL查询语句我需要思考一段时间才能写出来；甚至有些语句存在错误，要参考手册内容才能完成实验。在这次实验中，我复习了数据表的创建、插入数据、修改和删除数据、添加约束、查询数据等操作的基本SQL语句。

在openGuass中创建数据表不仅仅用一句“CREATE TABLE client（）;“，还要在前面添加“DROP TABLE IF EXISTS client;”，防止万一已经存在client数据表，影响到接下来的操作。插入数据使用“INSERT INTO ”语句，查询数据使用“SELECT FROM WHERE；”语句，修改数据使用“UPDATE … SET …”语句。添加约束使用“ALTER table 表名 ADD CONSTRAINT 列 CHECK (约束条件);”语句。视图与索引的基本操作类似，创建视图/索引使用“CREATE VIEW/ INDEX”语句，修改视图/索引名称使用“ALTER VIEW/ INDEX 旧名RENAME TO 新名”语句，删除视图/ INDEX使用“DROP VIEW/ INDEX 名称;”语句。