

**数据库系统原理实践报告**

|  |  |
| --- | --- |
| 专 业： | 计算机科学与技术 |
| 班 级： | CS2007 |
| 学 号： | U202015512 |
| 姓 名： | 孙溪 |
| 指导教师： | 赵小松 |

|  |  |
| --- | --- |
| 分数 |  |
| 教师签名 |  |

2022 年12 月30 日

**教师评分页**

|  |  |
| --- | --- |
| 子目标 | 子目标评分 |
| 1 |  |
| 2 |  |
| 3 |  |
| 4 |  |
| 5 |  |
| 6 |  |
|  |  |

|  |  |
| --- | --- |
| 总分 |  |

目 录

[1 课程任务概述 1](#_Toc123569654)

[2 任务实施过程与分析 2](#_Toc123569655)

[2.1 数据库、表与完整性约束的定义 (Create) 2](#_Toc123569656)

[2.2 表结构与完整性约束的修改 (ALTER) 3](#_Toc123569657)

[2.3 数据查询 (Select) 4](#_Toc123569658)

[2.4 数据的插入、修改与删除 13](#_Toc123569659)

[2.5 视图 14](#_Toc123569660)

[2.6 存储过程与事务 14](#_Toc123569661)

[2.7 触发器 17](#_Toc123569662)

[2.8 用户自定义函数 18](#_Toc123569663)

[2.9 安全性控制 19](#_Toc123569664)

[2.10 并发控制与事务的隔离级别 20](#_Toc123569665)

[2.11 数据库应用开发（JAVA 篇） 20](#_Toc123569666)

[2.12 备份+日志：介质故障与数据库恢复 25](#_Toc123569667)

[2.13 数据库设计与实现 25](#_Toc123569668)

[2.14 数据库的索引B+树实现 28](#_Toc123569669)

[3 课程总结 30](#_Toc123569670)

# 1 课程任务概述

课程的重点是将数据库系统原理和数据库系统原理实践，理论与实践相结合，使用OpenGauss作为例子。课程包括一系列实践任务，涵盖以下内容：

1. 管理和编程数据库对象，如表、索引、视图、约束、存储过程、函数、触发器和游标。
2. 与数据处理相关的数据查询、插入、删除和修改任务。
3. 对数据库安全控制、完整性控制、恢复机制以及并发控制机制等核心系统进行实验。
4. 数据库的设计和实现。
5. 数据库应用系统的开发（Java版）。
6. 数据库内核实验（B+树）。

该课程使用头歌实践教学平台，实践课程网址可在相关教室教师发布的链接及邀请码中找到。实验环境为Linux操作系统下的OpenGauss 2.1。在数据库应用开发阶段，使用Java 1.8。

# 2 任务实施过程与分析

## 2.1 数据库、表与完整性约束的定义 (Create)

关卡主要关于数据库和表的创建，包括创建数据库、创建表以及在表中建立主键、外键、CHECK、DEFAULT和UNIQUE约束的任务。

### 2.1.1 创建数据库

该关卡任务已完成，实施情况本报告略过。

### 2.1.2 创建表及主码约束

在指定的数据库中创建一个表，并为表指定主码。使用create table关键语句进行表创建，并用primary key指定主码。代码如下：

create table t\_emp(

id INT PRIMARY KEY,

name VARCHAR(32),

deptld INT,

salary FLOAT

);

### 2.1.3 创建外码约束

外码约束主要通过foreign key来指定，创建两表之间的关联。关键代码为staff中的CONSTRAINT语句，指定外码与dept表的关联。代码如下：

create table dept(

deptNo int primary key,

deptName varchar(32)

);

create table staff(

staffNo int primary key,

staffName varchar(32),

gender char(1),

dob date,

salary numeric(8,2),

deptNo int,

CONSTRAINT FK\_staff\_deptNo foreign key(deptNo) references dept(deptNo)

);

### 2.1.4 CHECK 约束

为表创建CHECK约束，创建表products，并分别实现对品牌和价格的约束，两个CHECK约束的名称分别为CK\_products\_brand和CK\_products\_price，主码约束不显示命名。通过constraint语句设置约束，关键代码如下：

create table products(

pid char(10) primary key,

name varchar(32),

brand char(10),

price int,

constraint CK\_products\_brand check(brand in ('A','B')),

constraint CK\_products\_price check(price > 0)

);

### 2.1.5 DEFAULT 约束

该关卡任务已完成，实施情况本报告略过。

### 2.1.6 UNIQUE 约束

该关卡任务已完成，实施情况本报告略过。

## 2.2 表结构与完整性约束的修改 (ALTER)

这一节包含四个任务，重点是数据库中表的基本修改操作。这些任务涉及使用alter语句来修改表的定义，如更换/修改表名、列名、列的类型、列约束、表约束、添加或删除列和约束。

### 2.2.1 修改表名

该关卡任务已完成，实施情况本报告略过。

### 2.2.2 添加与删除字段

该关卡任务已完成，实施情况本报告略过。

### 2.2.3 修改字段

该关卡任务已完成，实施情况本报告略过。

### 2.2.4 添加、删除与修改约束

包括删除和添加主码约束；删除和添加外码约束；删除和添加CHECK约束；删除和添加UNIQUE约束。代码如图2.2.1所示：

图 2.2.1 添加、修改与删除约束

## 2.3 数据查询 (Select)

本实训采用的是某银行的一个金融场景应用的模拟数据库，数据库中表，表结构以及所有字段的说明。

### 2.3.1 金融应用场景介绍，查询客户主要信息

该关卡任务已完成，实施情况本报告略过。

### 2.3.2 邮箱为 null 的客户

该关卡任务已完成，实施情况本报告略过。

### 2.3.3 既买了保险又买了基金的客户

本关任务： 请用一条语句查询既买了保险又买了基金的客户的名称和邮箱。

通过select语句和嵌套查询、多条件查询。通过限制pro\_type来找到对应的c\_id，进而获得对应的客户信息，最后按照c\_id排序。代码如图2.2.2所示：

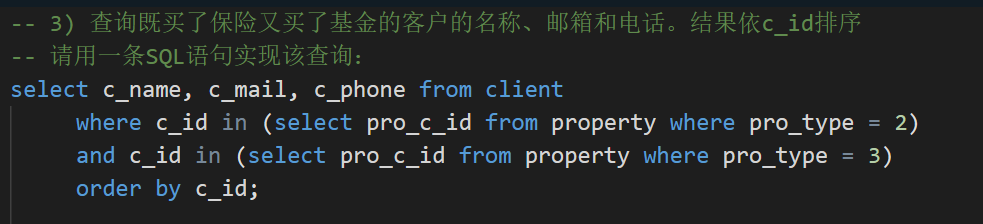


图 2.3.1 既买了保险又买了基金的客户

### 2.3.4 办理了储蓄卡的客户信息

本关任务： 查询办理了储蓄卡的客户名称、手机号、银行卡号。

本关的关键在多表的连接，将client和bank\_card表进行内连接，查询对应客户办理了储蓄卡的个人信息。代码如图2.2.3所示：

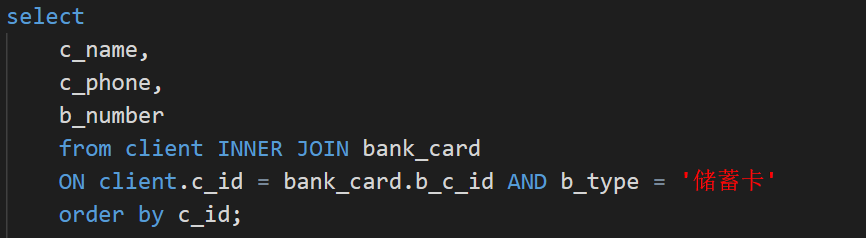


图 2.3.2 办理了储蓄卡的客户信息

### 2.3.5 每份金额在 30000∼50000 之间的理财产品

本关任务： 查询每份金额在30000～50000之间的理财产品，并对结果进行排序。

关键在于通过between语句进行条件限定，降序排列的效果。代码如图一2.2.4所示：

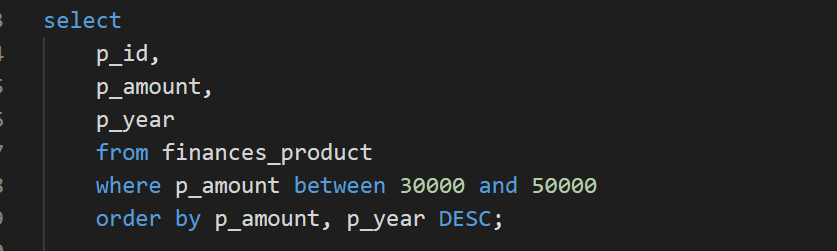


图 2.3.3 每份金额在 30000∼50000 之间的理财产品

### 2.3.6 商品收益的众数

本关任务： 查询资产表中所有资产记录里商品收益的众数和它出现的次数。

通过pro\_icome进行分组，用count()函数来统计出现次数，需要找出众数则需要找到出现次数最多的一个，可以通过ALL关键词来与所有的数据进行比较，筛选出出现次数最多的一个。代码如图2.2.5所示：

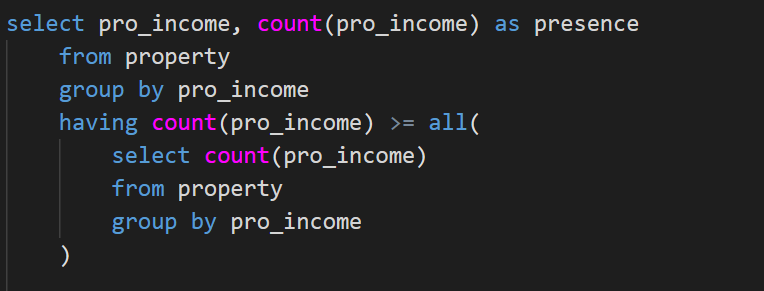


图 2.3.4 商品收益的众数

### 2.3.7 未购买任何理财产品的武汉居民

本关任务： 查询未购买任何理财产品的武汉居民的信息。

本关关键在于通过like函数用表达式匹配来查找符合条件的居民，并且通过not in()来筛选。代码如图2.3.5所示：

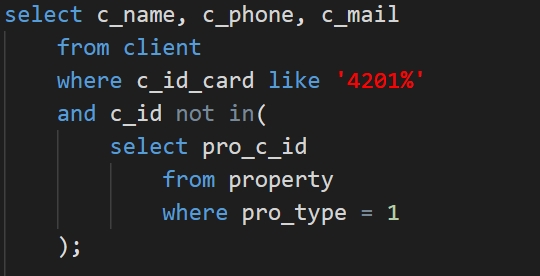


图 2.3.5 未购买任何理财产品的武汉居民

### 2.3.8 持有两张信用卡的用户

该关卡任务已完成，实施情况本报告略过。

### 2.3.9 购买了货币型基金的客户信息

该关卡任务已完成，实施情况本报告略过。

### 2.3.10 投资总收益前三名的客户

本关任务： 查询投资总收益前三名的客户。

本关通过sum函数计算投资总收益，将client和property两表右连接。并且通过c\_id和pro\_status进行分类，筛选pro\_status状态为‘可用’，最后通过total\_icome进行排序，只取top3。代码如图2.3.6所示：

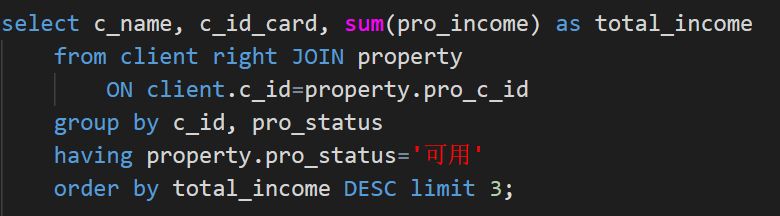


图 2.3.6 投资总收益前三名的客户

### 2.3.11 黄姓客户持卡数量

该关卡任务已完成，实施情况本报告略过。

### 2.3.12 客户理财、保险与基金投资总额

本关任务： 查询客户理财、保险、基金投资金额的总和，并排序。

本关的关键在于通过full join 和union all的多表连接，选出三类资产的购买客户，计算出资金总和作为依据从而排序。代码如图2.3.7所示：

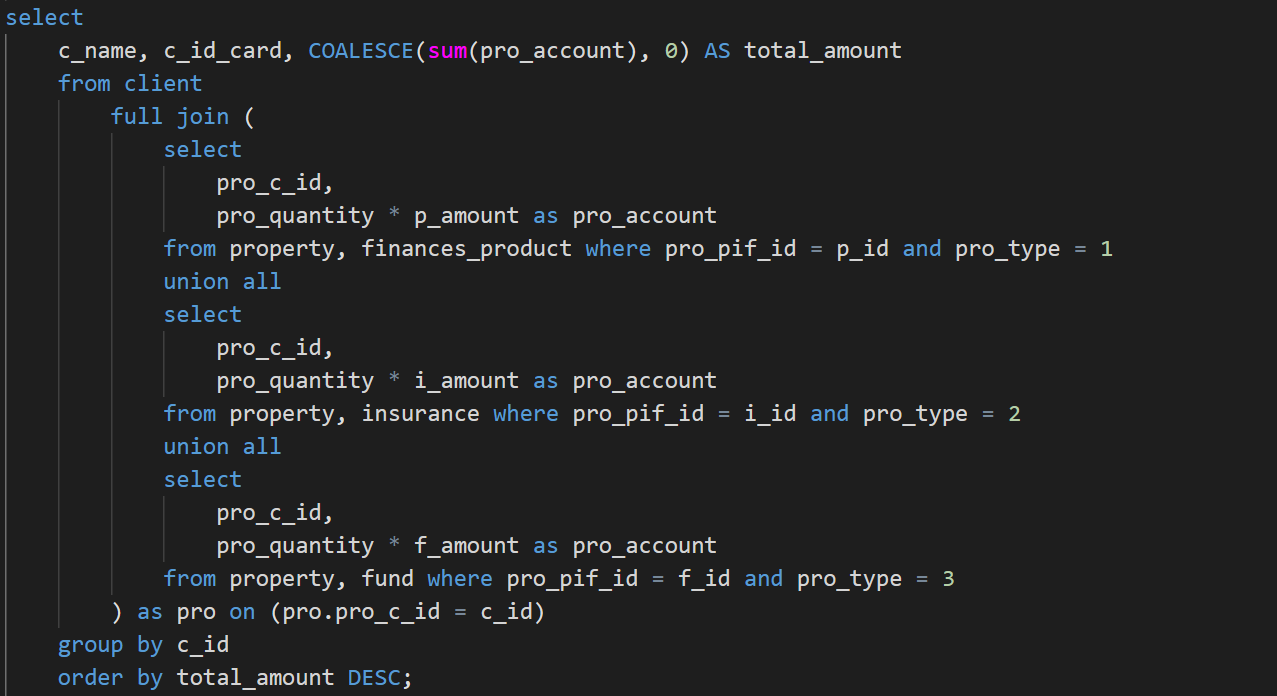


图 2.3.7 客户理财、保险与基金投资总额

### 2.3.13 客户总资产

该关卡任务已完成，实施情况本报告略过。

### 2.3.14 第 N 高问题

本关任务：查询每份保险金额第4高保险产品的编号和保险金额。

本关关键在于通过distinct消重，通过limit off, count来取特定排序的值。代码如图2.3.8所示：

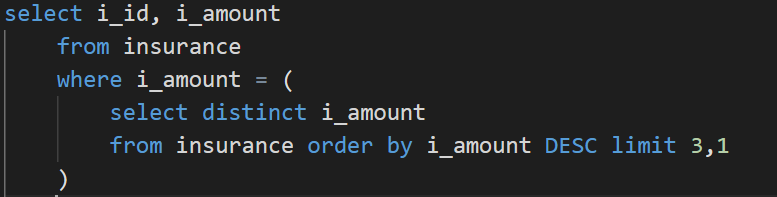


图 2.3.8 第 N 高问题

### 2.3.15 基金收益两种方式排名

本关任务： 对客户基金投资收益实现两种方式的排名次。

本关通过两种排序方式分别进行排序，通过rank() over()来统计名次不连续的排名，通过dense\_rank() over()统计名次连续的排名。代码如图2.3.9所示：

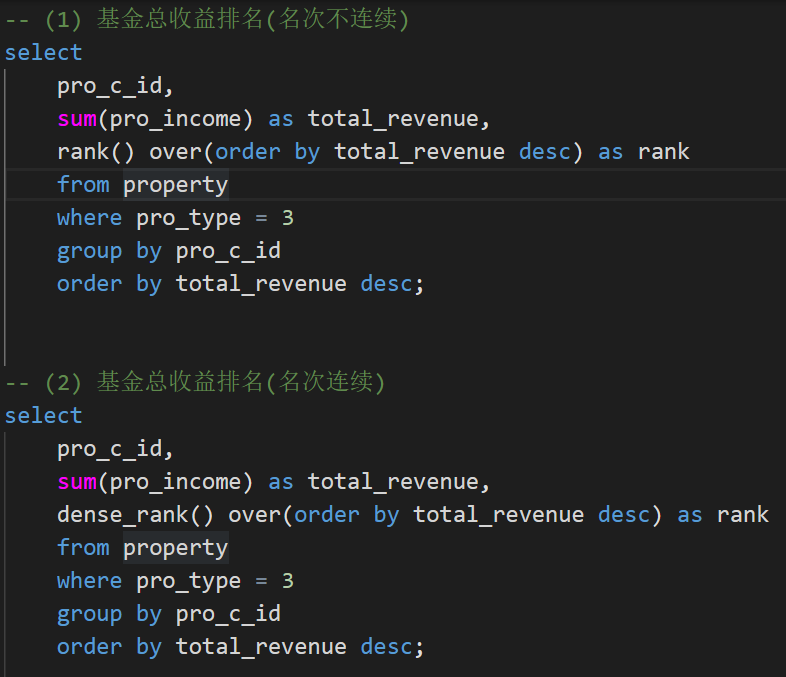


图 2.3.9 基金收益两种方式排名

### .3.16 持有完全相同基金组合的客户

本关任务：查询持有完全相同基金组合的客户。

思路：查询property表中pro\_type字段值为3的记录，并将它们按pro\_c\_id分组。对于每组记录，使用string\_agg函数将pro\_pif\_id值组合成一个字符串。然后，它连接两个子查询，并使用WHERE子句筛选出pifset和pifset2字段相同且pro\_c\_id字段小于pro\_c\_id2字段的记录。最后，使用ORDER BY子句将结果按pro\_c\_id字段排序。代码如图2.3.10所示：

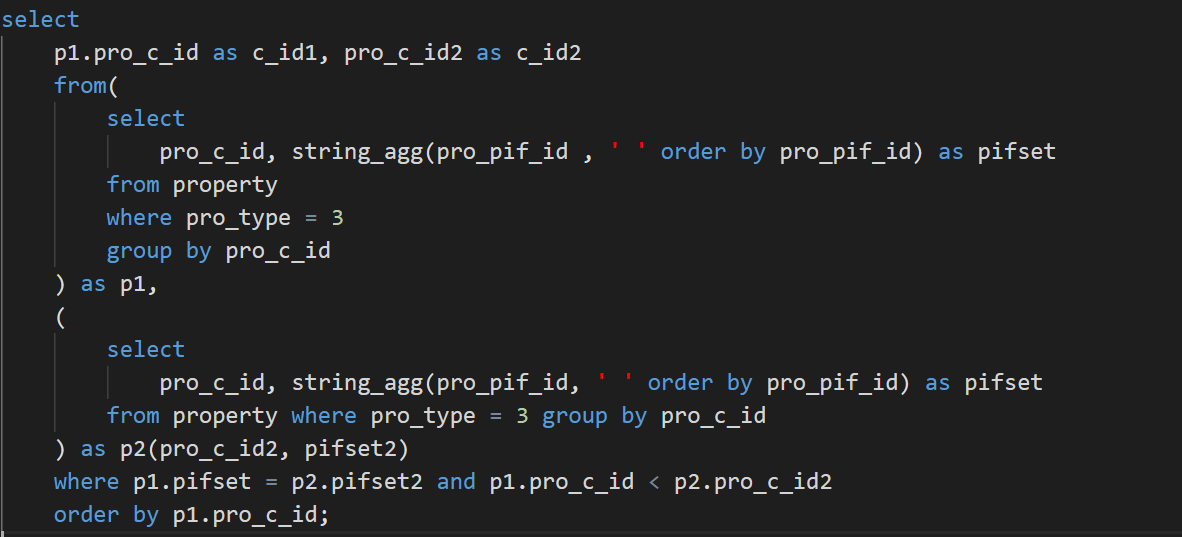


图 2.3.10 持有完全相同基金组合的客户

### 2.3.17 购买基金的高峰期

本关任务：查询2022年2月购买基金的高峰期，如果连续三个交易日，投资者购买基金的总金额超过100万，则称这连续的几日为投资者购买基金的高峰期。

此关采用了暴力解法，将2月所有可能成为高峰期的日期列举出来，进行查找。需要限定资产为基金类型和至少三个连续交易日。代码过长，穷举思路简单。

### 2.3.18 至少有一张信用卡余额超过 5000 元的客户信用卡总金额

该关卡任务已完成，实施情况本报告略过。

### 2.3.19 以日历表格式显示每日基金购买总金额

本关任务：以日历表格式显示2022年2月每周每日基金购买总金额。

查询property表中pro\_type是基金且pro\_purchase\_time字段在2022年2月份的记录，并将它们与fund表连接。然后，它使用DATE\_PART函数将pro\_purchase\_time字段的星期数减去5，并使用extract函数提取pro\_purchase\_time字段的星期几。查询使用SUM函数计算总额，使用CASE语句按星期几分类，最后使用GROUP BY和ORDER BY子句将结果按wk字段分组和排序。代码如图2.3.11所示：

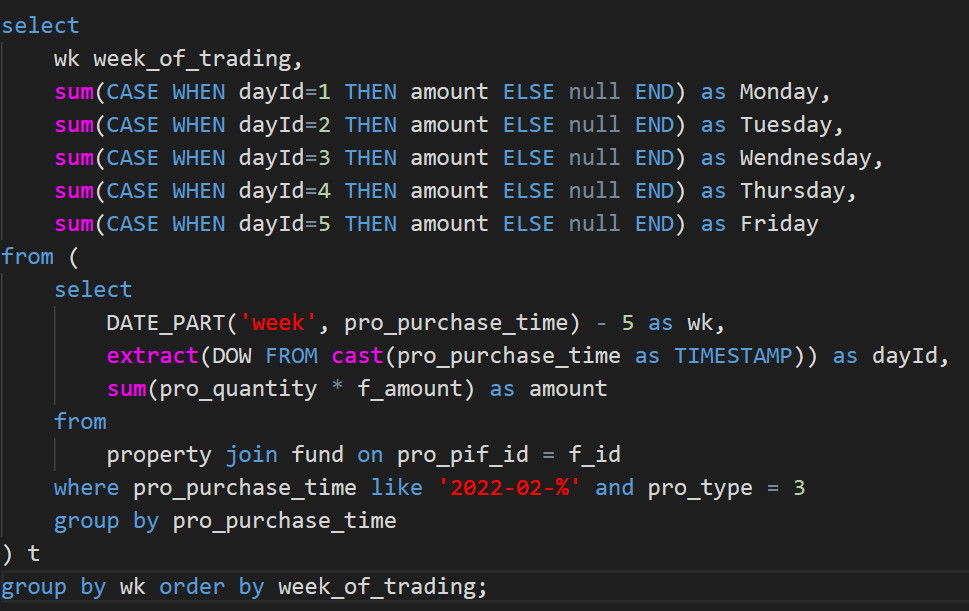


图 2.3.11 以日历表格式显示每日基金购买总金额

### 2.3.20 查询销售总额前三的理财产品

本关任务：查询销售总额前三的理财产品。

思路：使用to\_char函数将pro\_purchase\_time字段转换为yyyy格式的字符串，并计算每个产品在每年的销售额。接着，使用rank函数为每年的每个产品分配排名，并使用order by子句将结果按sumamount字段降序排序。最后，使用UNION连接两个子查询，并使用WHERE子句筛选出每年排名前3的产品。代码如图2.3.12所示：

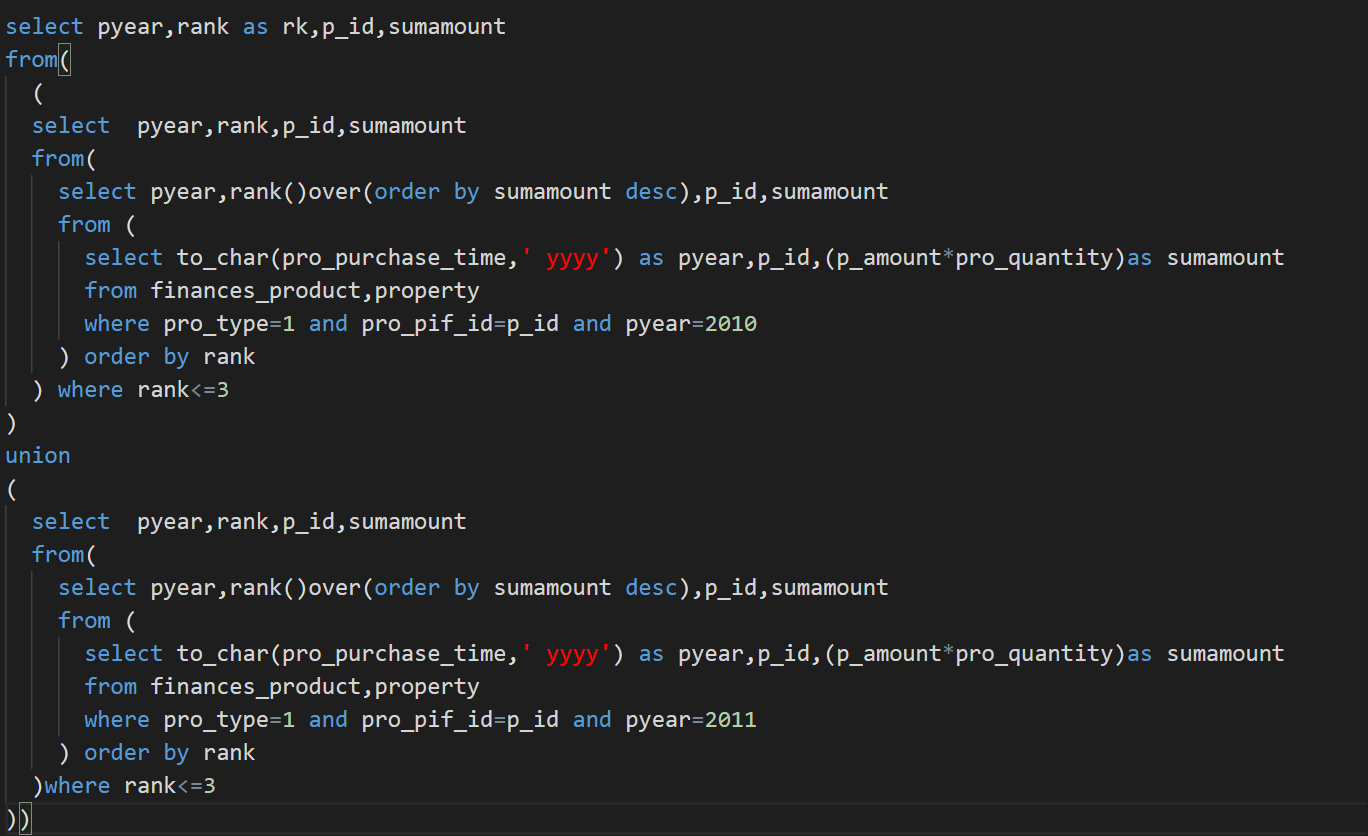


图 2.3.12 查询销售总额前三的理财产品

### 2.3.21 投资积极且偏好理财类产品的客户

本关任务：投资积极且偏好理财类产品的客户。若该客户持有基金产品种类数小于其持有的理财产品种类数，则认为该客户为投资积极且偏好理财产品的客户。

查询property表中pro\_type字段值为1或3的记录，并将它们与finances\_product和fund表连接。然后，使用COUNT(DISTINCT pro\_pif\_id)函数统计每个客户购买的不同产品/基金的数量，并使用GROUP BY子句将结果按pro\_c\_id字段分组。接着，使用HAVING子句筛选出购买产品数量大于3的客户，用INNER JOIN连接两个子查询，并使用WHERE子句筛选出购买产品数量大于基金数量的客户。代码如图2.3.13所示：

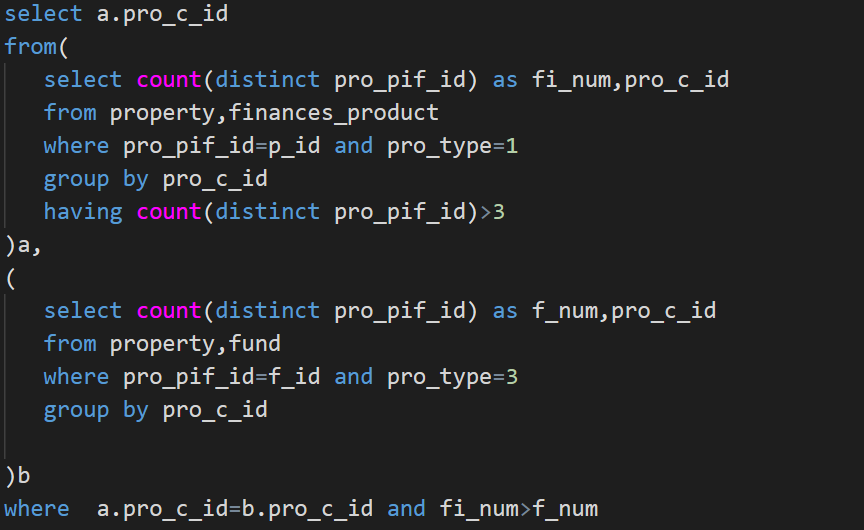


图 2.3.13 投资积极且偏好理财类产品的客户

### 2.3.22 查询购买了所有畅销理财产品的客户

本关任务：查询购买了所有畅销理财产品的客户。持有人数超过2的理财产品称为畅销理财产品。

子查询来找出所有类型为 1 的记录，并去重，再找出找出所有记录数大于 2 的 pro\_pif\_id，将所有类型为 1 的记录分组，并使用 HAVING 子句来筛选出所有记录数等于"sumid" 的 pro\_c\_id。代码如图2.3.14所示：

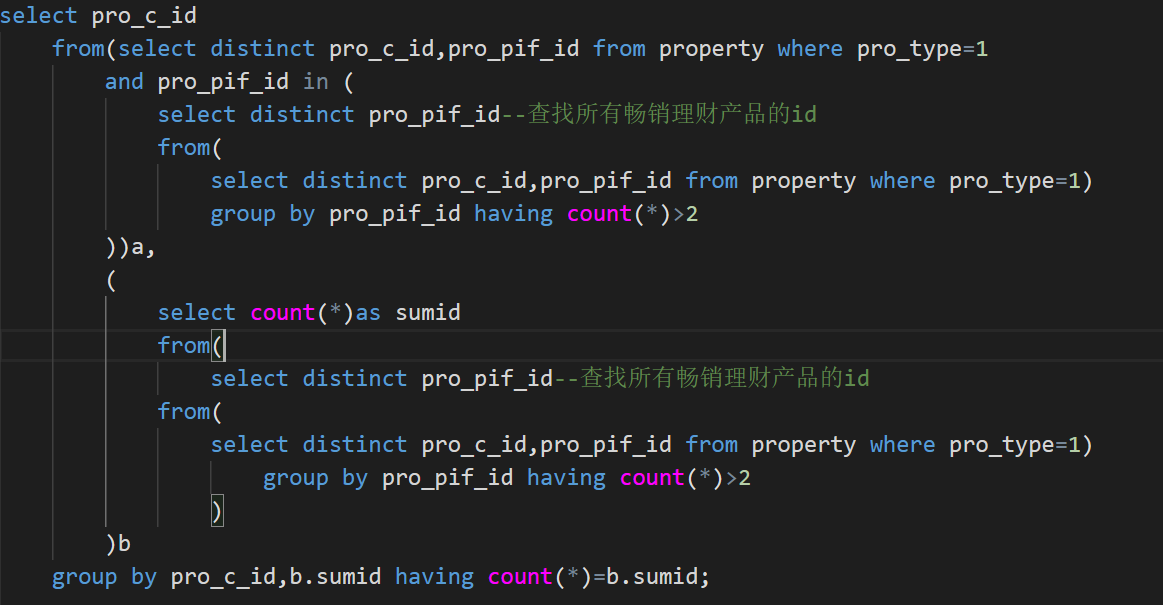


图 2.3.14 查询购买了所有畅销理财产品的客户

### 2.3.23 查找相似的理财产品

本关任务：查找相似的理财产品。结果输出要求：按照相似度值降序排列，相同相似度的理财产品之间则按照产品编号的升序排列。

使用三层嵌套的子查询来实现。通过对 pro\_type=1 的记录进行分组并计数，得到每个 pro\_pif\_id 的数量。然后，在外层查询中使用 dense\_rank() 函数按照 pro\_pif\_id 的数量排序，并得到每个 pro\_pif\_id 的排名（rank）。最后，通过在外层查询中筛选出 rank<=3 的结果，取出前三名的 pro\_pif\_id 和对应的数量。

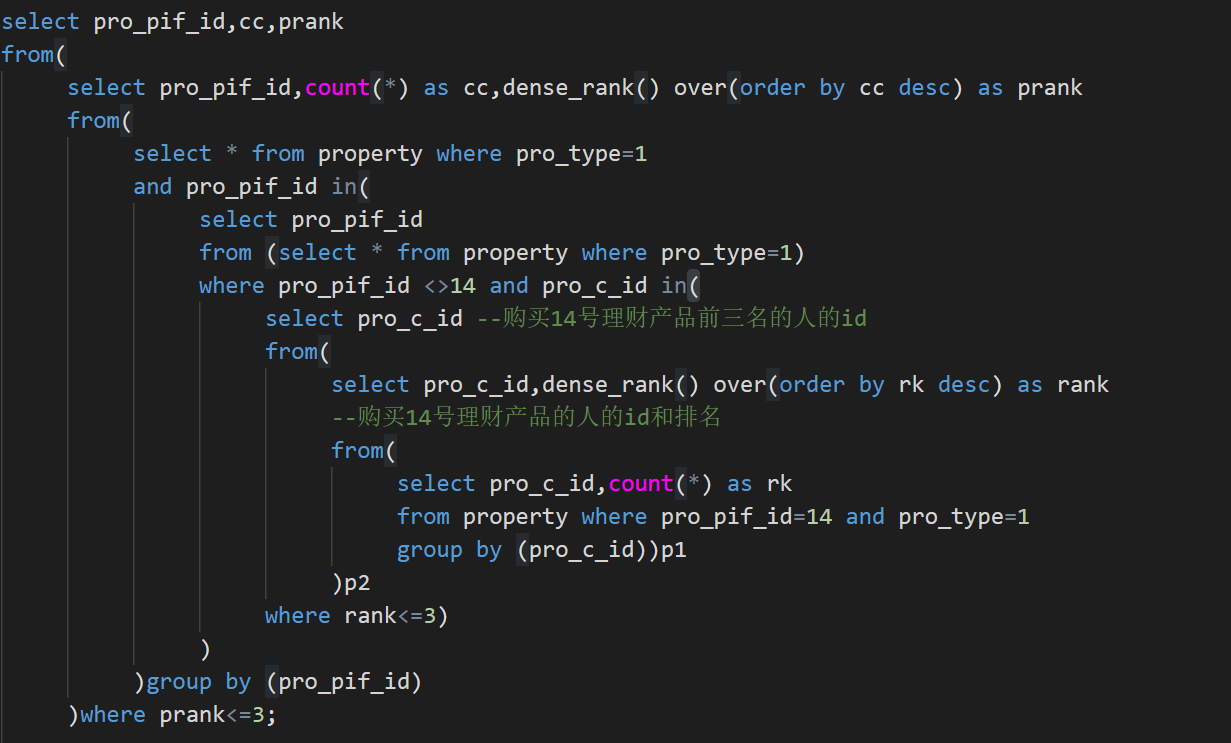


图 2.3.15 查找相似的理财产品

### 2.3.24 查询任意两个客户的相同理财产品数

该关卡任务已完成，实施情况本报告略过。

### 2.3.25 查找相似的理财客户

该关卡任务已完成，实施情况本报告略过。

## 2.4 数据的插入、修改与删除

该小节内容关于Insert, Update, Delete 语句在不同场景下的应用。

### 2.4.1 插入多条完整的客户信息

该关卡任务已完成，实施情况本报告略过。

### 2.4.2 插入不完整的客户信息

该关卡任务已完成，实施情况本报告略过。

### 2.4.3 批量插入数据

本关任务：向客户表client批量插入数据。向client表中插入所有new\_client表中的记录。代码如下：

insert into client

select \* from new\_client

### 2.4.4 删除没有银行卡的客户信息

本关任务：删除在本行没有银行卡的客户信息。

使用NOT EXISTS子句检查是否存在与client表c\_id字段值匹配的bank\_card表b\_c\_id字段值。如果不存在，则删除client表中的该记录。代码如下：

delete from client

where not exists (

select 1

from bank\_card

where client.c\_id = bank\_card.b\_c\_id

);

### 2.4.5 冻结客户资产

该关卡任务已完成，实施情况本报告略过。

### 2.4.6 连接更新

本关任务：根据客户表的内容修改资产表的内容。

用一条update语句，根据client表中提供的身份证号(c\_id\_card)，填写property表中对应的身份证号信息(pro\_id\_card)。通过set语句将property表中的pro\_id\_card字段更新为与pro\_c\_id字段匹配的client表中c\_id\_card字段的值。代码如下：

update property

set pro\_id\_card=(

select c\_id\_card from client where pro\_c\_id=c\_id

);

## 2.5 视图

此小节关于视图的创建与使用。

### 2.5.1 创建所有保险资产的详细记录视图

该关卡任务已完成，实施情况本报告略过。

### 2.5.2 基于视图的查询

本关任务：基于视图v\_insurance\_detail查询每位客户保险资产的总额和保险总收益。

从视图 "v\_insurance\_detail" 中查询数据，并按照客户的姓名和身份证号分组。对于每组数据，计算保险总金额 (由产品数量和产品单价组成) 和保险总收益，最后按照保险总金额降序排列。代码如图2.5.1所示：

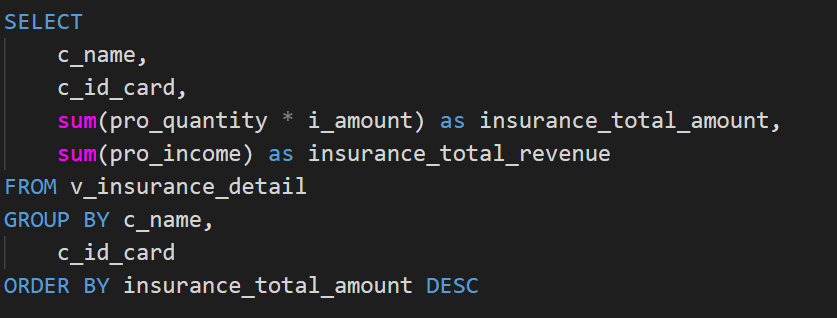


图 2.5.1 基于视图的查询

## 2.6 存储过程与事务

使用流程控制语句的存储过程、使用游标的存储过程和使用事务的存储过程。

### 2.6.1 使用流程控制语句的存储过程

本关任务：创建一个存储过程，向表fibonacci插入斐波拉契数列的前n项。

存储过程的目的是在 "fibonacci" 表中生成一个长度为 "m" 的斐波那契数列。首先将数列的第一和第二个数插入到 "fibonacci" 表中，使用一个 WHILE 循环，计算数列中的剩余数。每次循环，计算斐波那契数列中的下一个数，并插入到 "fibonacci" 表中。代码如图2.6.1所示：

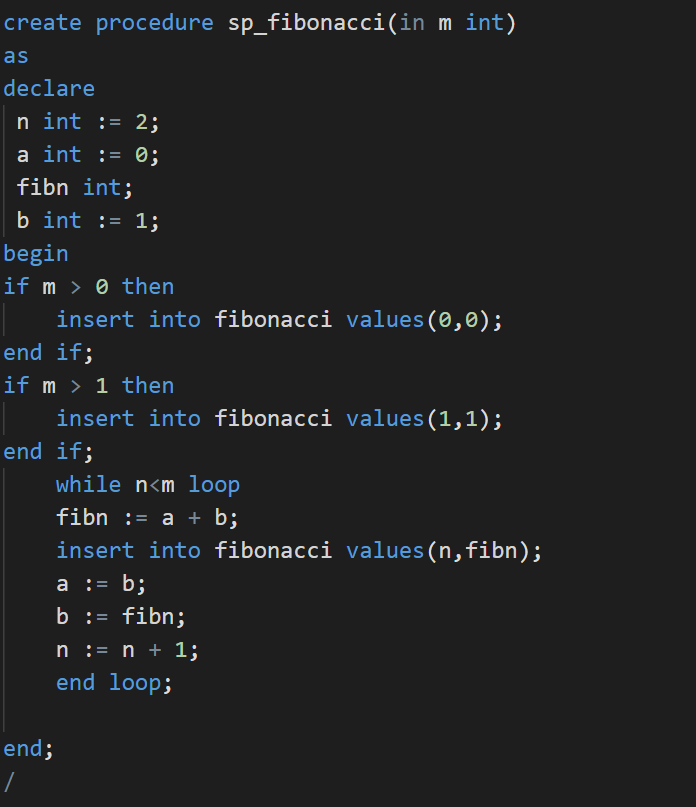


图 2.6.1 使用流程控制语句的存储过程

### 2.6.2 使用游标的存储过程

本关任务：使用游标编程存储过程为医院的某科室排夜班值班表。

使用两个游标来遍历员工表，第一个游标遍历所有类型为护士的员工，第二个游标遍历所有类型不为护士的员工。使用 WHILE 循环，遍历每一天。每次循环，把两个护士的名字和一个医生的名字插入到 "night\_shift\_schedule" 表中。通过EXTRACT 函数和 CAST 函数来获取每一天是星期几（0 表示星期天，1 表示星期一，以此类推）。因为当周末轮至科主任时，主任的夜班调至周一，由排在主任后面的医生依次递补值周末的夜班。所以如果这一天是星期天，则使用之前保存的 "head" 变量；如果这一天不是星期天，则使用游标 cur2 遍历的医生。代码如图2.6.2所示：

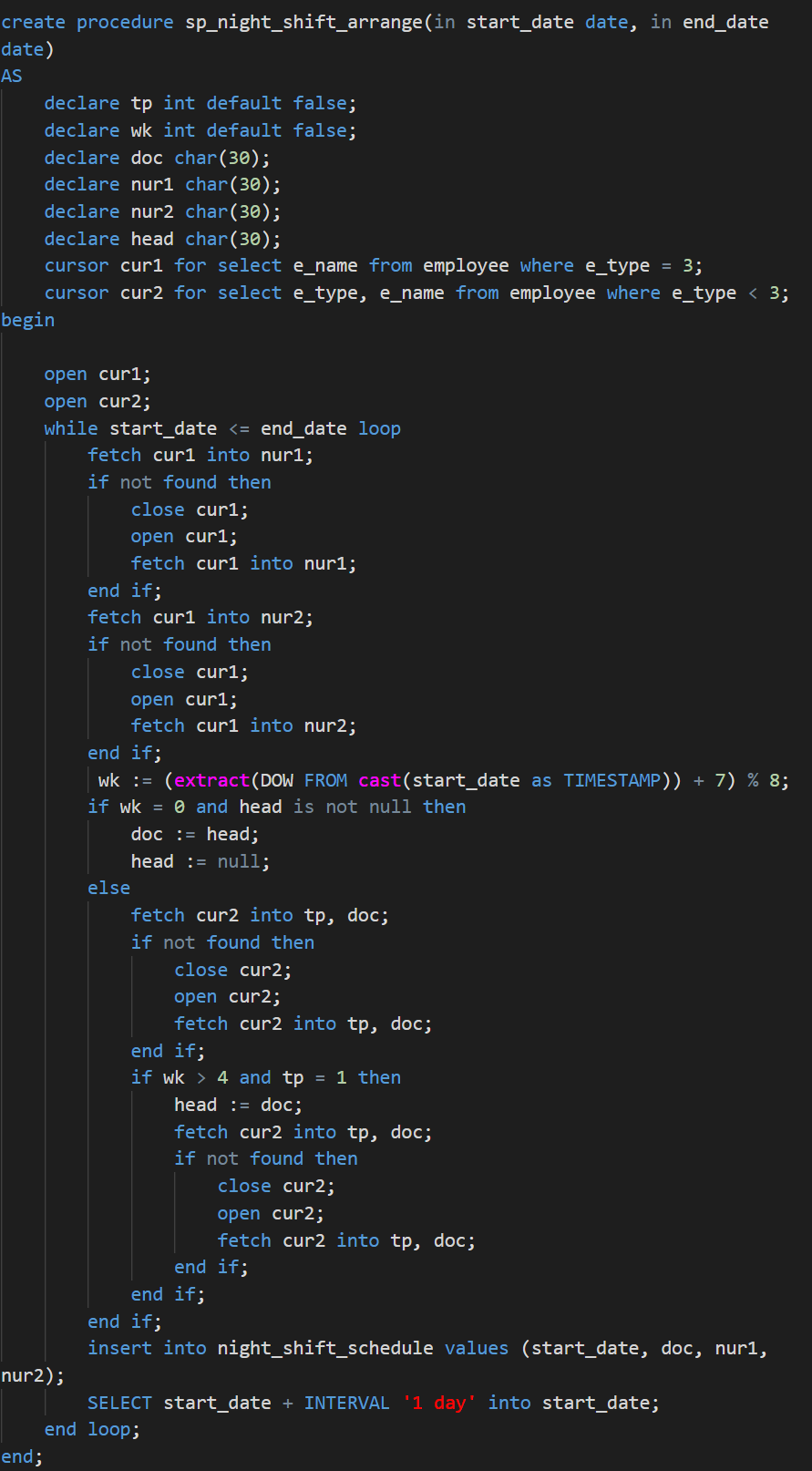


图 2.6.2 使用游标的存储过程

### 2.6.3 使用事务的存储过程

本关任务：编写实现转账功能的存储过程。

存储过程执行两条 update 语句，用于将转账金额从源账户和目标账户中分别扣除。然后进行判断，如果源账户余额不足或目标账户不存在，则回滚事务并将 return\_code 赋值为 0；否则提交事务并将 return\_code 赋值为 1。

这样的话，在调用这个存储过程时，可以通过 return\_code 的值来判断转账是否成功。代码如图2.6.3所示：

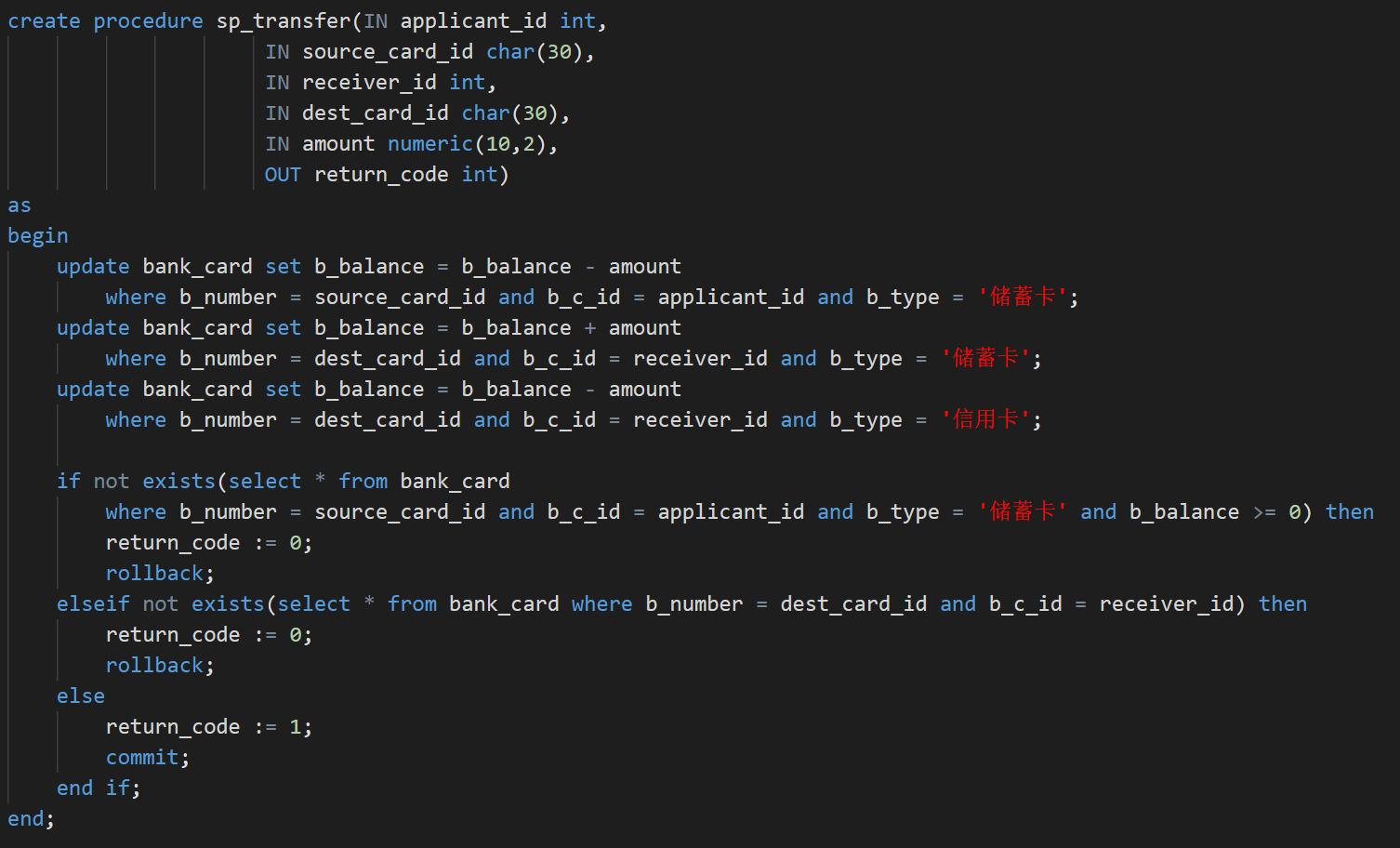


图 2.6.3 使用事务的存储过程

## 2.7 触发器

触发器的创建和使用场景。

### 2.7.1 为投资表 property 实现业务约束规则‐根据投资类别分别引用不同表的主码

本关任务： 为表property(资产表)编写一个触发器，以实现以下完整性业务规则：

* 如果pro\_type = 1, 则pro\_pif\_id只能引用finances\_product表的p\_id；
* 如果pro\_type = 2, 则pro\_pif\_id只能引用insurance表的i\_id；
* 如果pro\_type = 3, 则pro\_pif\_id只能引用fund表的f\_id；
* pro\_type不接受(1,2,3)以外的值。

用触发器函数 TRI\_INSERT\_FUNC（）实现功能，如果执行过程中发现有不满

足条件的情况则利用 concat（）函数编辑报错信息 msg，并通过“raise exception‘%’,msg”抛出报错信息。代码如图 2.7.1 所示。

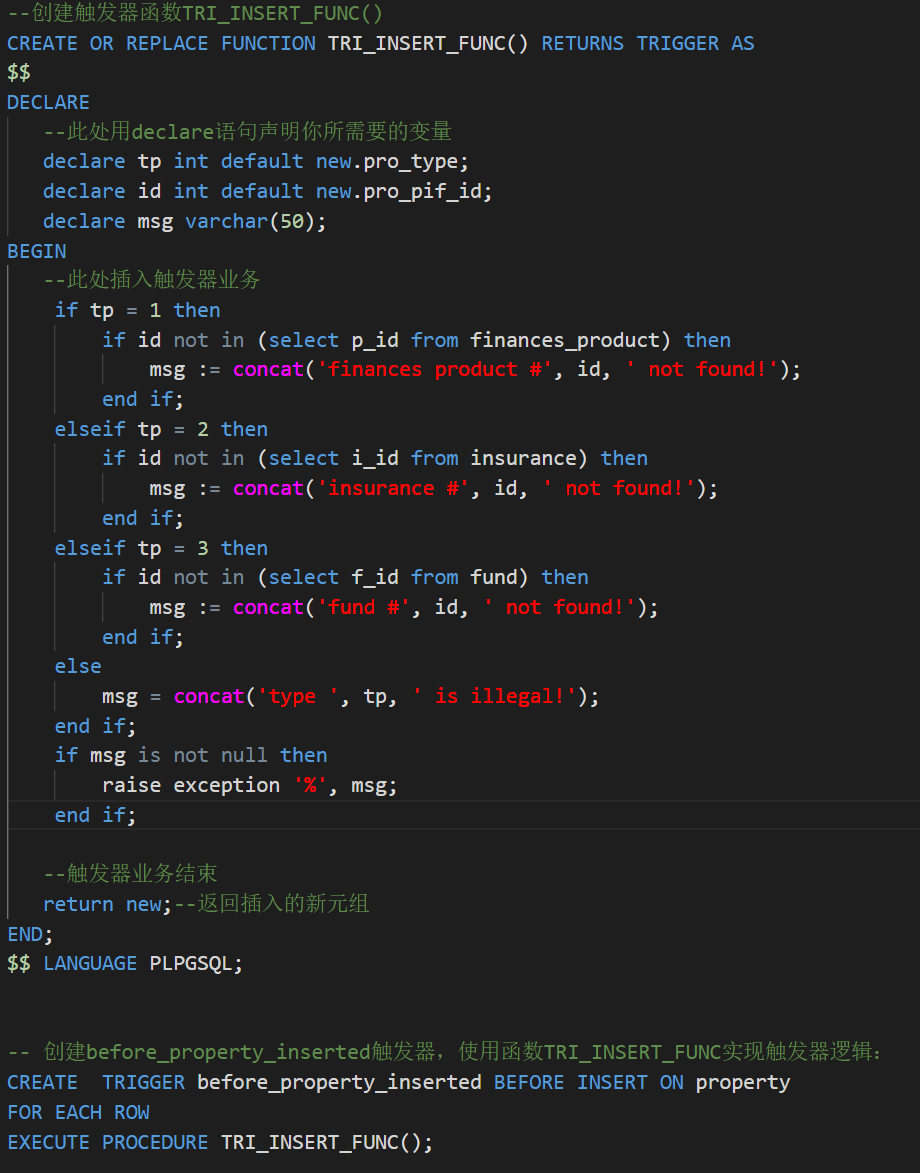


图 2.7.1 触发器

## 2.8 用户自定义函数

该小节需要创建符合要求的用户自定义函数和使用。

### 2.8.1 创建函数并在语句中使用它

本关任务： 编写一个依据客户编号计算其在本金融机构的存储总额的函数,并在SELECT语句使用这个函数。

用create function语句创建函数，利用创建的函数，仅用一条SQL语句查询存款总额在100万(含)以上的客户身份证号，姓名和存款总额(total\_deposit)，结果依存储总额从高到低排序。代码如图2.8.1所示：

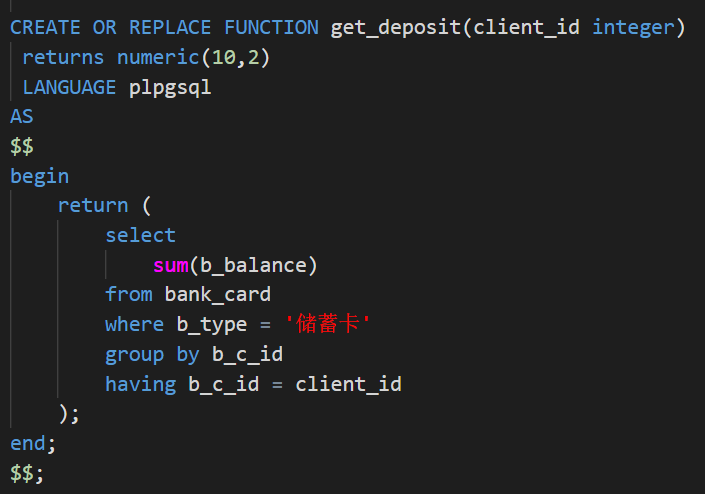


图 2.8.1 创建函数并在语句中使用它

## 2.9 安全性控制

涉及数据库中的用户、角色和权限等内容。

### 2.9.1 用户和权限

该关卡任务已完成，实施情况本报告略过。

### 2.9.2 用户、角色与权限

本关任务：创建角色，授予角色一组权限，并将角色代表的权限授予指定的一组用户。

通过create role创建角色，grant语句授予权限。代码如图2.9.1所示：

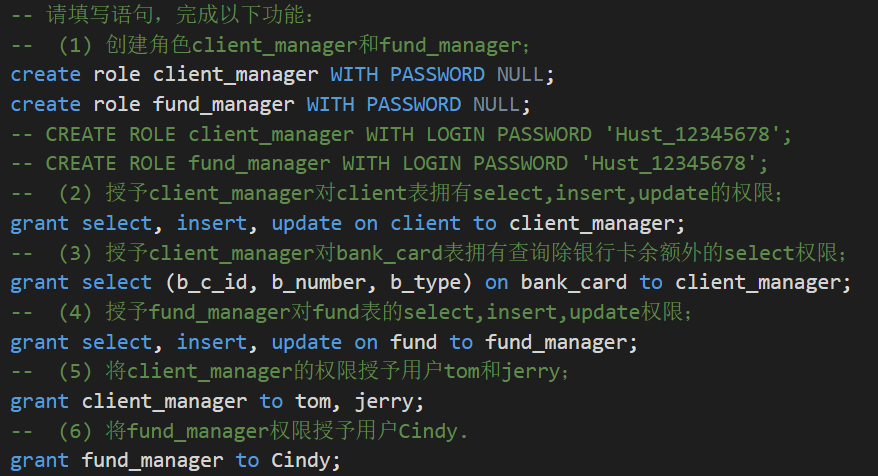


图 2.9.1 用户、角色与权限

## 2.10 并发控制与事务的隔离级别

有两个涉及该表的并发事务t1和t2，分别定义在t1.sql和t2.sql代码文件中。平台会让两个事务并发执行，通过修改代码文件来达到题目预期的并发执行效果。

### 2.10.1 不可重复读

本关任务： 选择合适的事务隔离级别，构造两个事务并发执行时，发生“不可重复读”现象。

不可重复读指在一个事务中两次查询之中，数据不一致。选择将事务的隔离级别设置为 read uncommitted，根据任务要求的执行顺序利用 pg\_sleep（）函数保证执行逻辑的准确，实现程序内部的等待。

### 2.10.2 幻读

本关任务：在read committed事务隔离级别，构造两个事务并发执行时，发生“幻读”现象。

两次查询余票超过 300 张的航班信息；在第 1 次查询之后，事务 t2 插入了一条航班信息并提交；第 2 次查询的记录数增多,发生“幻读”。该任务复现了这一场景。

### 2.10.3 主动加锁保证可重复读

本关任务： 在事务隔离级别较低的read committed情形下，通过主动加锁，保证事务的一致性。

OpenGauss的select语句支持for share和for update短语，分别表示对表加共享(Share)锁和写(write)锁，在事务结束时才释放。保证事务t1可重复读，在等待t2正常提交(commit)后，再查询一次全部航班的余票,MU2455的余票减少1张。

### 2.10.4 可串行化

该关卡任务已完成，实施情况本报告略过。

## 2.11 数据库应用开发（JAVA 篇）

结合 java 和数据库语言实现一个简单的数据库应用系统。

### 2.11.1 JDBC 体系结构和简单的查询

本关任务：正确使用JDBC查询client表中邮箱非空的客户信息，列出客户姓名，邮箱和电话。

调用DriverManager对象的getConnection（）方法来建立实际的数据库连接。旦获得了连接，就可以与数据库进行交互。通过statement接口发送SQL命令并从数据库接收数据的方法和属性。通过ResultSet executeQuery(String SQL)获取结果集，代码如图2.11.1所示：

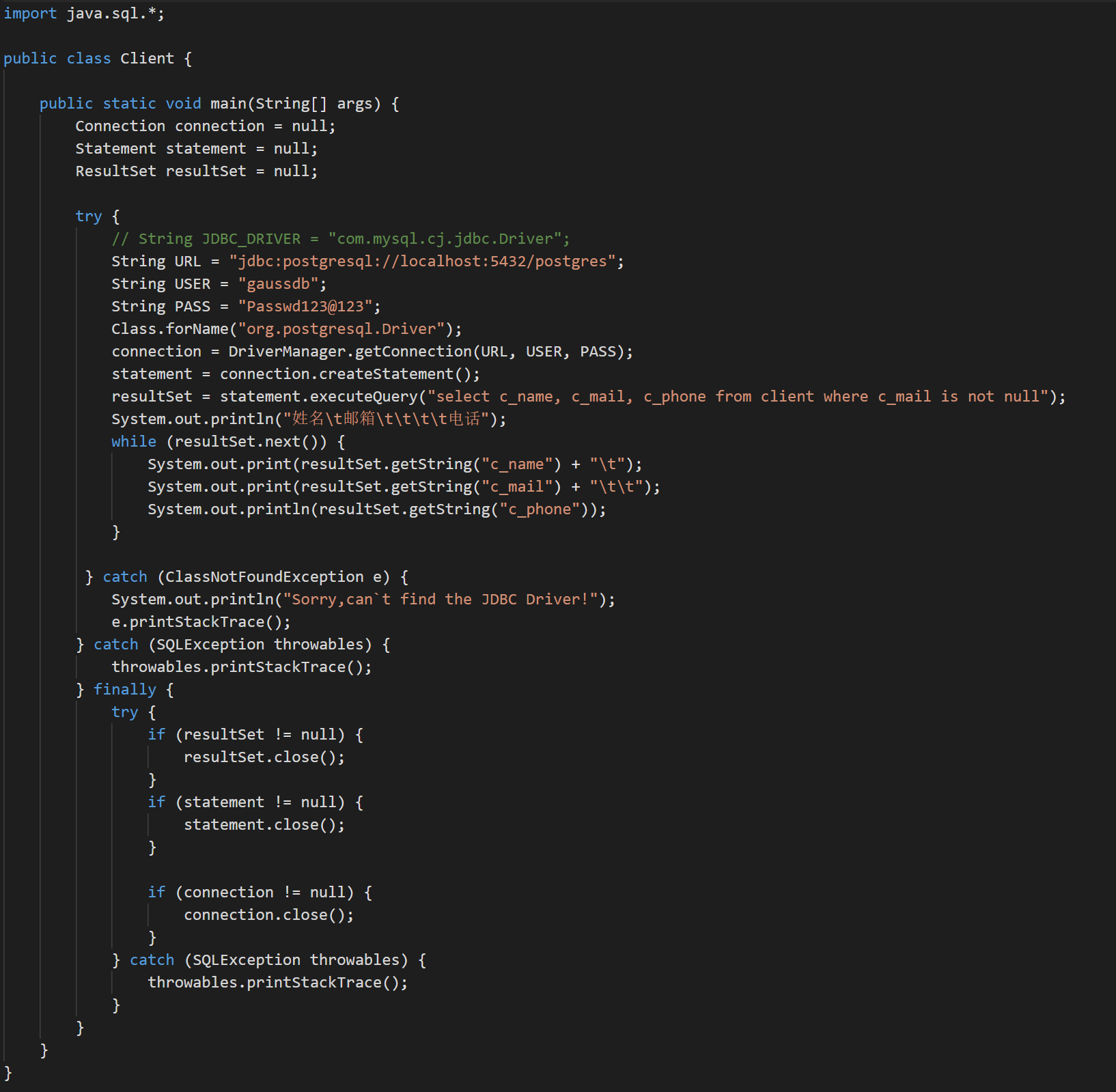


图 2.11.1 基于视图的查询

### 2.11.2 用户登录

该关卡任务已完成，实施情况本报告略过。

### 2.11.3 添加新客户

该关卡任务已完成，实施情况本报告略过。

### 2.11.4 银行卡销户

该关卡任务已完成，实施情况本报告略过。

### 2.11.5 客户修改密码

该关卡任务已完成，实施情况本报告略过。

### 2.11.6 转账与事务操作

补充代码，实现一个银行卡转账的方法，方法返回boolean值，true表示转帐成功，false表示转账失败，并不需要细分或解释失败的原因。

理清转账失败可能原因：转出或转入帐号不存在，转出账号是信用卡，转出帐号余额不足。处理对应情况返回转账状态为false，其余情况则返回true。main()不需要修改。代码如图2.11.2所示（节选）：



图 2.11.2 转账与事务操作

### 2.11.7 把稀疏表格转化为键值对存储

本关任务：将一个稀疏的表中有保存数据的列值，以键值对(列名，列值 )的形式转存到另一个表中，这样可以直接丢失没有值列。

依规则将entrance\_exam表的值转写到sc表。对每一行，从左至右依次考察每一列，转存非空列。转换为转存在大量的三元组中。代码如图2.11.3所示（节选）：

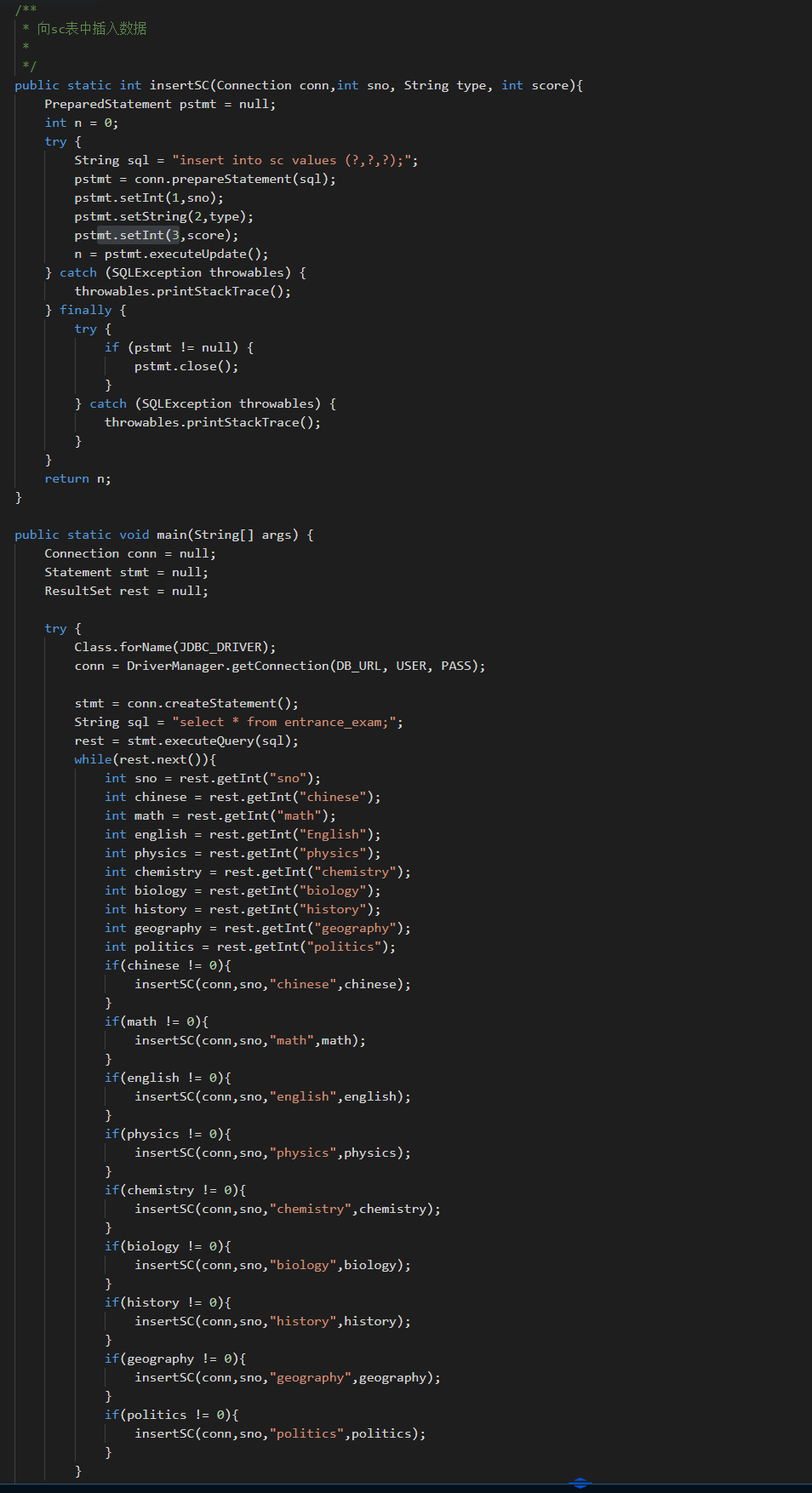


图 2.11.3 把稀疏表格转化为键值对存储

## 2.12 备份+日志：介质故障与数据库恢复

本节涉及数据库的备份与恢复，需要了解OpenGauss的恢复机制，提供的备份与恢复种类，gs\_dump和ds\_restore的使用。

### 2.12.1备份与恢复

本关任务: 备份数据库，然后再恢复它。用gs\_dump工具为该数据库做一次静态的(你一个人独享服务器)海量备份，备份文件为.tar格式文件； 然后再用gs\_restore工具，利用前述备份文件恢复数据库residents。

结合gs\_dump数据导出工具和gs\_restore数据导入工具，代码如图2.12.1所示：

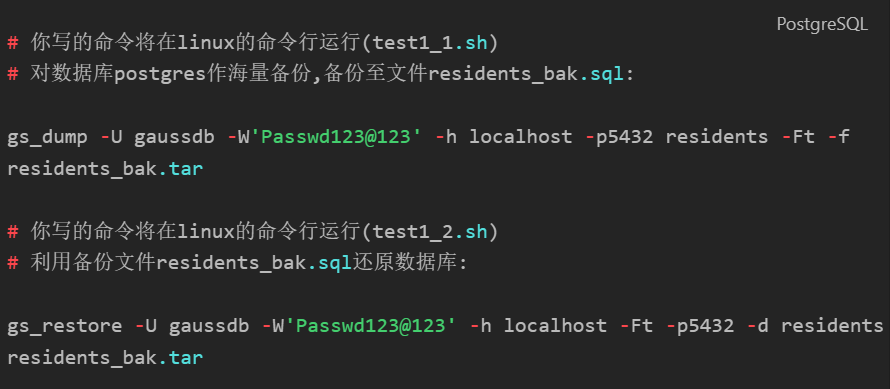


图 2.12.1 备份与恢复

## 2.13 数据库设计与实现

数据库设计与实现相关内容，包括从概念模型到 OpenGauss 实现、E‐R 图的构建、建模工具的使用。

### 2.13.1 从概念模型到 OpenGauss 实现

该关卡任务已完成，实施情况本报告略过。

### 2.13.2 从需求分析到逻辑模型

本关任务：根据应用场景业务需求描述，完成 E‐R 图，并转换成关系式。

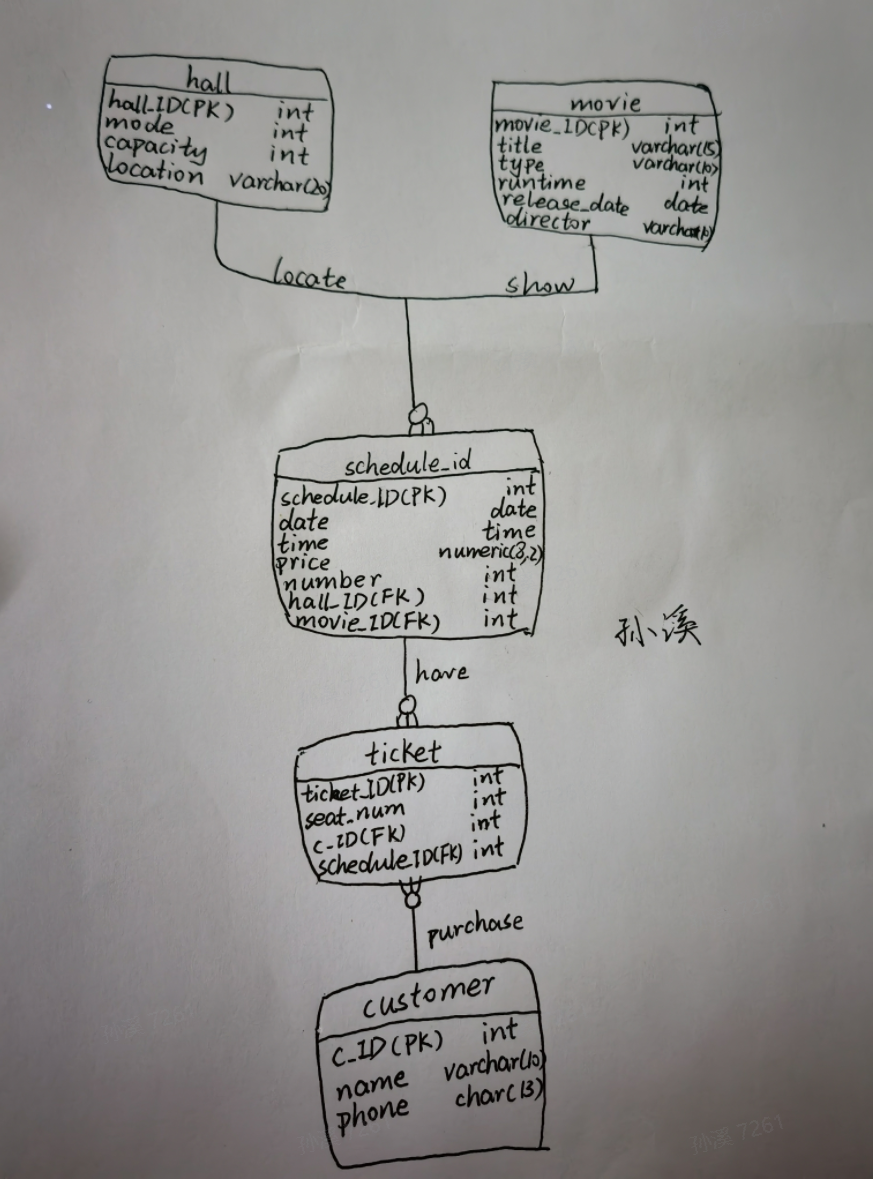
E-R图如图2.13.1所示：

图 2.13.1 E-R图

### 2.13.3建模工具的使用

该关卡任务已完成，实施情况本报告略过。

### 2.13.4制约因素分析与设计

建模是指将实际问题抽象成数据库能够表示的形式。这个过程分为两个步骤：概念建模和逻辑建模。

在概念建模阶段，我们需要考虑的制约因素包括：业务要求、数据要求、安全性、可扩展性、性能。

在逻辑建模阶段，我们需要考虑的制约因素包括：数据类型（需要确定每个属性的数据类型，并确定是否需要限制属性值的范围）、主键（以便在数据库中区分不同的记录）、外键（以便在数据库中维护数据之间的关系）、索引（需要考虑是否需要为表中的某些列建立索引，以提高查询效率）、数据冗余（需要考虑如何避免或最小化数据冗余，以保证数据的一致性和准确性）、数据分区（需要考虑如何将数据分区，以提高查询效率和系统的可扩展性）、数据约束（需要考虑如何通过设置数据约束来保证数据的一致性和准确性）。

在从实际问题到建立数据库模型的过程中，应该考虑若干制约因素，以机票订票系统为例，应考虑旅客的实际情况和系统的权限需求。在转账系统中需要考虑到实际场景下转账成功的必要条件。在把稀疏表格转化为键值对存储任务中，我们从数据结构的角度进行了存储结构优化，也进而提高了查询效率和空间利用率。

### 2.13.5工程师责任及其分析

工程师应该能够根据工程相关背景知识进行合理分析，评估专业工程实践和复杂工程问题解决方案对社会、健康、安全、法律和文化的影响，并理解应承担的责任。工程师应尽可能考虑系统中的安全漏洞，并使用科学方法对复杂工程问题进行研究，包括设计实验、分析和解释数据，并通过信息综合获得合理有效的结论。

具体来说，工程师应该：

1. 能够基于工程相关背景知识进行合理分析，评价专业工程实践和复杂工程问题解决方案对社会、健康、安全、法律以及文化的影响，并理解应承担的责任。

2. 能够尽可能考虑系统中存在的安全漏洞，保证系统的安全性，以免造成人身伤害和财产损失。

3. 能够基于科学原理并采用科学方法对复杂工程问题进行研究，包括设计实验、分析与解释数据、并通过信息综合得出合理有效的结论。

4. 能够考虑工程系统的可扩展性，以便在未来满足新的需求。

5. 能够考虑工程系统的性能，以便满足用户的需求。

工程师应该遵守所在国家/地区的法律法规和行业准则，并承担相应的责任。同时，工程师还应该遵守职业道德，尊重他人的知识产权和财产权，并不得在未经授权的情况下使用他人的知识产权和财产。

在解决实际问题时，工程师应该努力了解问题的背景和环境，并根据相关条件进行分析和决策。在实际实施过程中，工程师应该负责协调各方面的工作，保证工程项目的顺利实施。

在服务的过程中，工程师应该尽可能提供优质的服务，并对自己的工作负责。如果发生任何问题，工程师应该尽快解决问题，并对自己的工作负责。

## 2.14 数据库的索引B+树实现

进行 B+数基本数据结构的实现和简单操作。

### 2.14.1 BPlusTreePage的设计

本关任务：作为B+树索引结点类型的数据结构设计的第一部分：实现BPlusTreePage类，该类是B+树叶结点类型和内部结点类型的父类，提供B+树结点的基本功能。

完成主要功能，判断页类型是否为叶子结点、根结点；设置/获取页的类型；设置/获取页的大小；增加页的大小；设置/获取页的最大大小；获取页的最小大小；设置/获取页的父页id；设置/获取页的本身id；设置/获取页的层级。主要用于在B+树的各种操作中（插入、删除、查找等）维护页的相关信息，并能通过这些信息对B+树进行调整以保证B+树性质的维护。代码如图2.14.1所示：

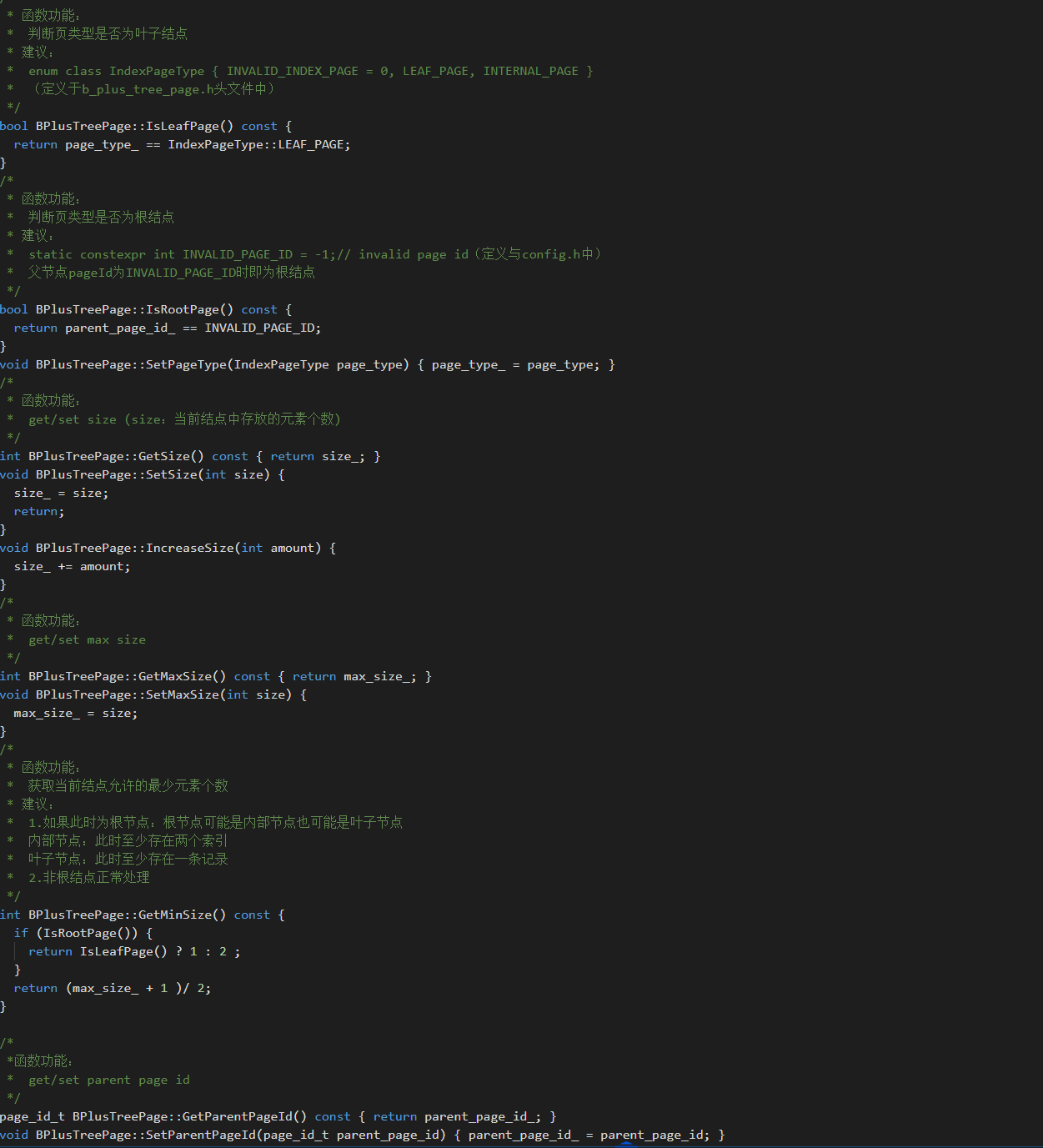


图 2.14.1 BPlusTreePage的设计

### 2.14.2 BPlusTreeInternalPage的设计

该关卡任务已完成，实施情况本报告略过。

### 2.14.3 BPlusTreeLeafPage 的设计

该任务关卡跳过。

### 2.14.4 B+树索引：Insert

该任务关卡跳过。

### 2.14.5 B+树索引：Remove

该任务关卡跳过。

# 3 课程总结

在这次实验中完成了14个不同的实验，涵盖了多种主题，包括创建和修改表及其完整性约束，数据查询和操作，视图，存储过程和事务，触发器，用户定义函数，安全控制，并发和事务隔离，数据库应用开发，备份和日志数据库，数据库设计和实现，以及 B+ 树实现。课程收获和体会如下：

1．使用 OpenGauss 数据库管理系统进行数据库管理。创建和修改表，以及如何在表中插入、修改和删除数据。

2．学会了如何使用视图和存储过程来简化数据库操作，以及如何使用事务来管理数据库操作。

3. 学会了如何使用触发器和用户定义函数来扩展数据库功能。

4. 学会了如何在数据库应用开发中使用 JDBC 来连接数据库，以及如何保证代码的安全性。

5. 学会了如何管理数据库的备份和日志，以及如何设计和实现数据库。

6. 基于B+树来实现数据库。

另外还学会了、如何使用 OpenGauss 复杂的语法来简化编程。还有在数据库应用开发中代码安全的重要性，以及 SQL 注入攻击可能造成的潜在后果。总的来说，这门课程可能会让你对数据库管理有更深入的了解，并且能够熟练使用 OpenGauss 数据库管理系统来实现各种数据库操作。