

数据库系统课程实验报告

|  |  |
| --- | --- |
| 实验名称： | 实验六 数据库安全性 |
| 实验日期： | 2023-5-5 |
| 实验地点： | 文宣楼B311 |
| 提交日期： | 2023-5-5 |
|  | |
| 学号： | 33920212204567 |
| 姓名： | 任宇 |
| 专业年级： | 软工2021级 |
| 学年学期： | 2022-2023学年第二学期 |

1. 实验环境

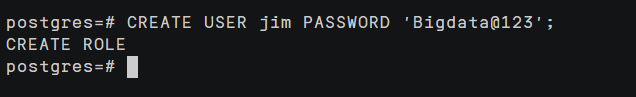
* 华为 ECS+openGauss 数据库服务器平台
* 前提：openGauss 数据库服务器正常运行
* 已建立带样例数据的 SALES 数据库

1. 实验目的

* 理解数据库系统用户(user)、权限(privilege)和角色(role)的概念和作用
* 熟练掌握用户的管理：创建、查看、删除和权限的授予与回收
* 熟练掌握通过数据字典查看用户权限、表和视图权限的方法
* 熟练掌握使用 Grant 命令给用户、角色授权的方法
* 熟练掌握使用 Revoke 命令回收已授权限的方法
* 熟练掌握角色定义、重命名和删除的方法
* 熟练掌握修改角色中权限的方法
* 理解视图的安全性作用

1. 实验内容和步骤
2. 学习并完成实验文档中提供链接中的内容：
3. 创建用户jim，登录密码为Bigdata@123

**CREATE USER jim PASSWORD 'Bigdata@123';**



1. 查看用户列表

**SELECT \* FROM pg\_user;**



1. 为用户jim追加有创建角色的CREATE ROLE权限

**ALTER USER jim CREATEROLE;**



1. 删除用户

**DROP USER jim CASCADE;**



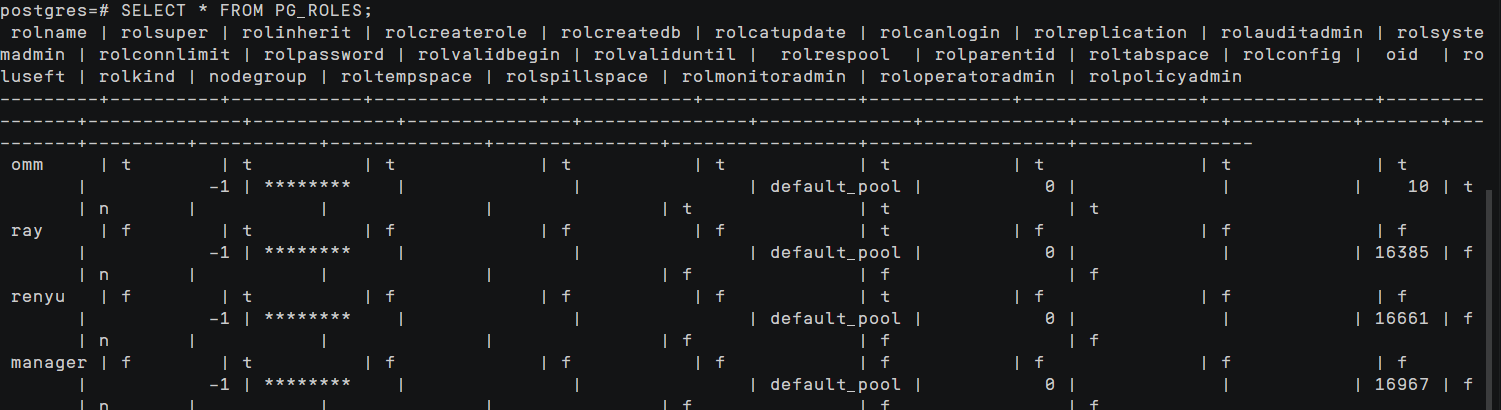
1. 创建一个角色，名为manager，密码为Bigdata@123

**CREATE ROLE manager IDENTIFIED BY 'Bigdata@123';**



1. 查看角色

**SELECT \* FROM PG\_ROLES;**



1. 修改角色manager的密码为abcd@123

**ALTER ROLE manager IDENTIFIED BY 'abcd@123' REPLACE 'Bigdata@123';**



1. 修改角色manager为系统管理员

**ALTER ROLE manager SYSADMIN;**



1. 删除角色manager

**DROP ROLE manager;**



1. 创建名为joe的用户

**CREATE USER joe PASSWORD 'Bigdata@123';**



1. 将sysadmin权限授权给joe

**GRANT ALL PRIVILEGES TO joe;**



1. 撤销joe用户的sysadmin权限

**REVOKE ALL PRIVILEGES FROM joe;**



1. 创建tpcds模式

**CREATE SCHEMA tpcds;**



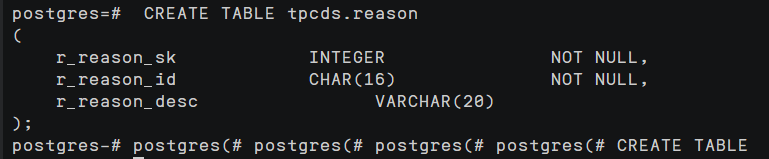
1. tpcds模式下创建一张reason表

**CREATE TABLE tpcds.reason(**

**r\_reason\_sk INTEGER NOT NULL,**

**r\_reason\_id CHAR(16) NOT NULL,**

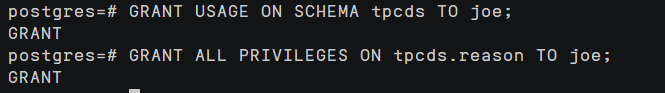
**r\_reason\_desc VARCHAR(20) );**



1. 将模式tpcds的使用权限和表tpcds.reason的所有权限授权给用户joe

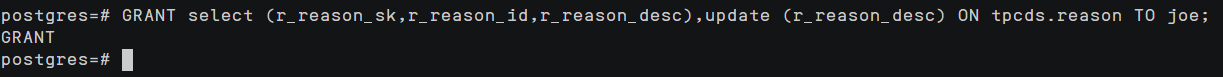
**GRANT USAGE ON SCHEMA tpcds TO joe;**

**GRANT ALL PRIVILEGES ON tpcds.reason TO joe;**



1. 将tpcds.reason表中r\_reason\_sk、r\_reason\_id、r\_reason\_desc列的查询权限，r\_reason\_desc的更新权限授权给joe

**GRANT select (r\_reason\_sk,r\_reason\_id,r\_reason\_desc),update (r\_reason\_desc) ON tpcds.reason TO joe;**



1. 将tpcds.reason表中r\_reason\_sk、r\_reason\_id、r\_reason\_desc列的查询权限，r\_reason\_desc的更新权限授权给joe

**GRANT create,connect on database postgres TO joe WITH GRANT OPTION;**



1. 创建角色tpcds\_manager

**CREATE ROLE tpcds\_manager PASSWORD 'Bigdata@123';**



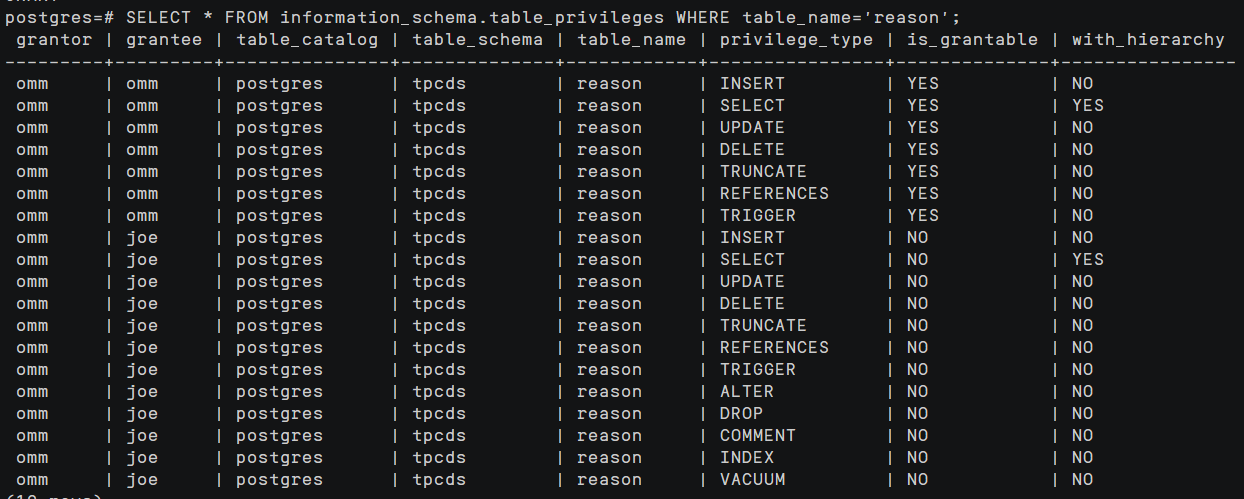
1. 将模式tpcds的访问权限授权给角色tpcds\_manager，并授予该角色在tpcds下创建对象的权限，不允许该角色中的用户将权限授权给其人

**GRANT USAGE,CREATE ON SCHEMA tpcds TO tpcds\_manager;**



1. 查看表reason权限

**SELECT \* FROM information\_schema.table\_privileges WHERE table\_name='reason';**



1. 创建角色manager

**CREATE ROLE manager PASSWORD 'Bigdata@123';**



1. 将joe的权限授权给manager，并允许该角色将权限授权给其他人

**GRANT joe TO manager WITH ADMIN OPTION;**



1. 创建用户senior\_manager

**CREATE ROLE senior\_manager PASSWORD 'Bigdata@123';**



1. 将用户manager的权限授权给该用户

**GRANT manager TO senior\_manager;**

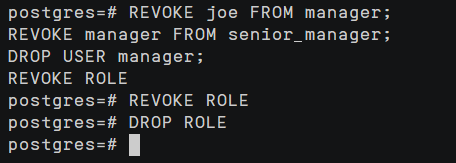


1. 逐步回收manager权限，删除manager用户

**REVOKE joe FROM manager;**

**REVOKE manager FROM senior\_manager;**

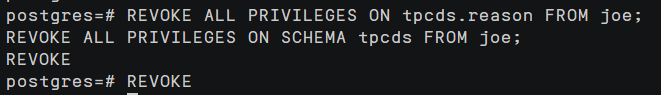
**DROP USER manager;**



1. 逐步回收joe权限

**REVOKE ALL PRIVILEGES ON tpcds.reason FROM joe;**

**REVOKE ALL PRIVILEGES ON SCHEMA tpcds FROM joe;**



1. 逐步回收tpcds\_manager权限

**REVOKE USAGE,CREATE ON SCHEMA tpcds FROM tpcds\_manager;**



1. 删除 tpcds\_manager用户

**DROP ROLE tpcds\_manager;**



1. 删除senior\_manager用户

**DROP ROLE senior\_manager;**



1. 删除joe用户

**DROP USER joe CASCADE;**



1. 创建视图 salesman，该视图只保存 employees 表中所有 job\_title 为'Sales Representative'的雇员：

CREATE VIEW salesman AS

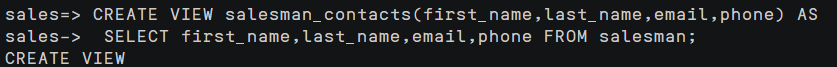
SELECT \* FROM employees WHERE job\_title='Sales Representative';



1. 创建基于salesman的视图salesman\_contacts (first\_name,last\_name,email,phone)，该视图存储salesman 的联系方式:

CREATE VIEW salesman\_contacts(first\_name,last\_name,email,phone) AS

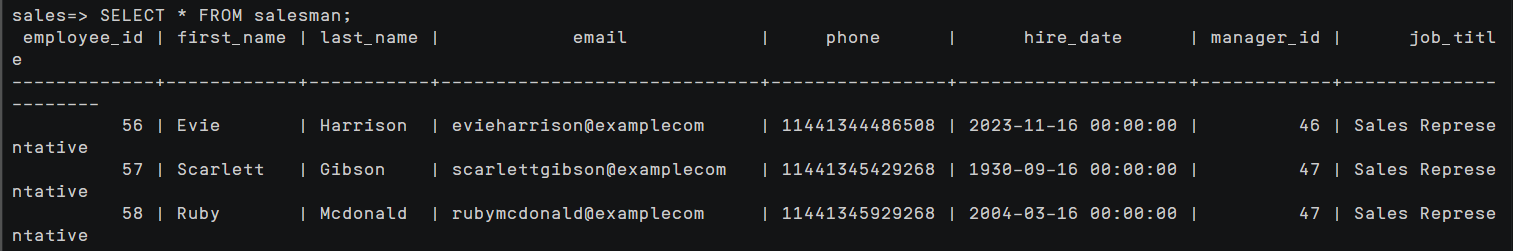
SELECT first\_name,last\_name,email,phone FROM salesman;

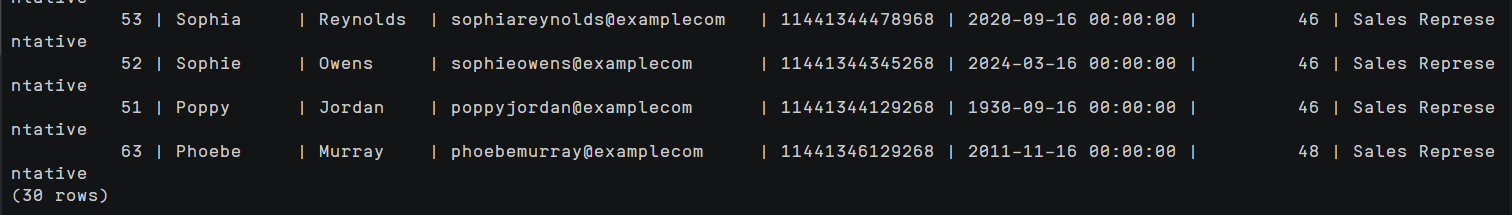


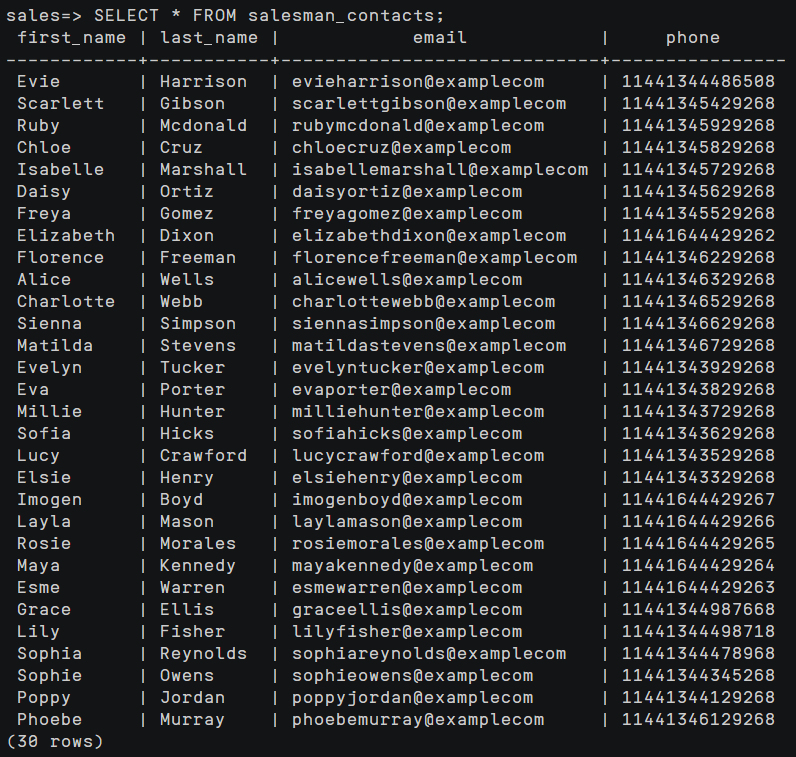
1. 查询视图 salesman 和 salesman\_contacts:

SELECT \* FROM salesman;

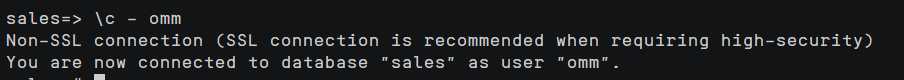
SELECT \* FROM salesman\_contacts;







1. 在当前窗口输入命令：\c - omm 切换到 omm 用户:

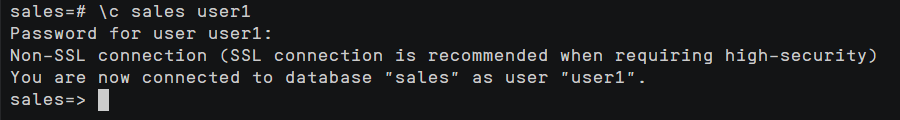


1. 创建新用户 user1:

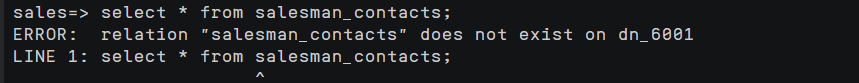
CREATE USER user1 WITH PASSWORD '123456abc!';



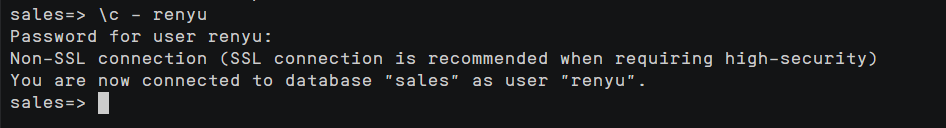
1. 在当前窗口输入命令：\c – user1 切换到 user1 用户:



1. 发布查询命令：select \* from salesman\_contacts;观察结果:



1. 发布命令：\c – renyu 切换到 renyu 用户:



1. 在当前用户下输入命令：grant select on salesman\_contacts to user1; 实现授权操作:

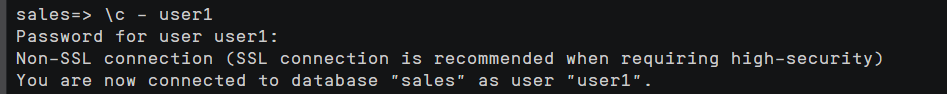
grant usage on schema sales to user1;

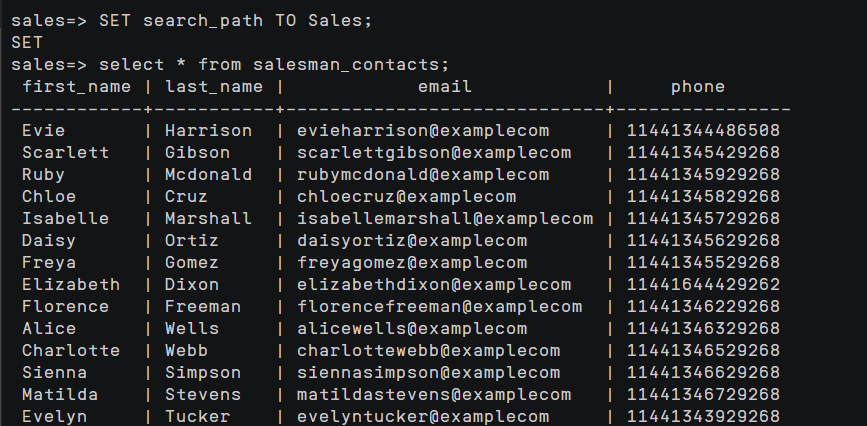
grant select on salesman\_contacts to user1;

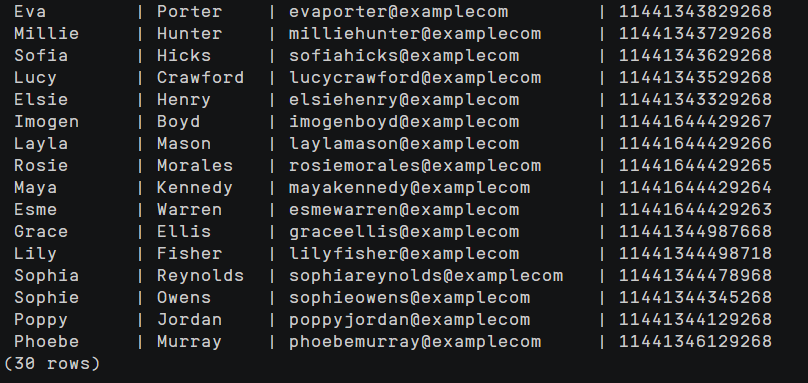




1. 依次重复步骤（7）和（8），比较两次查询的结果:

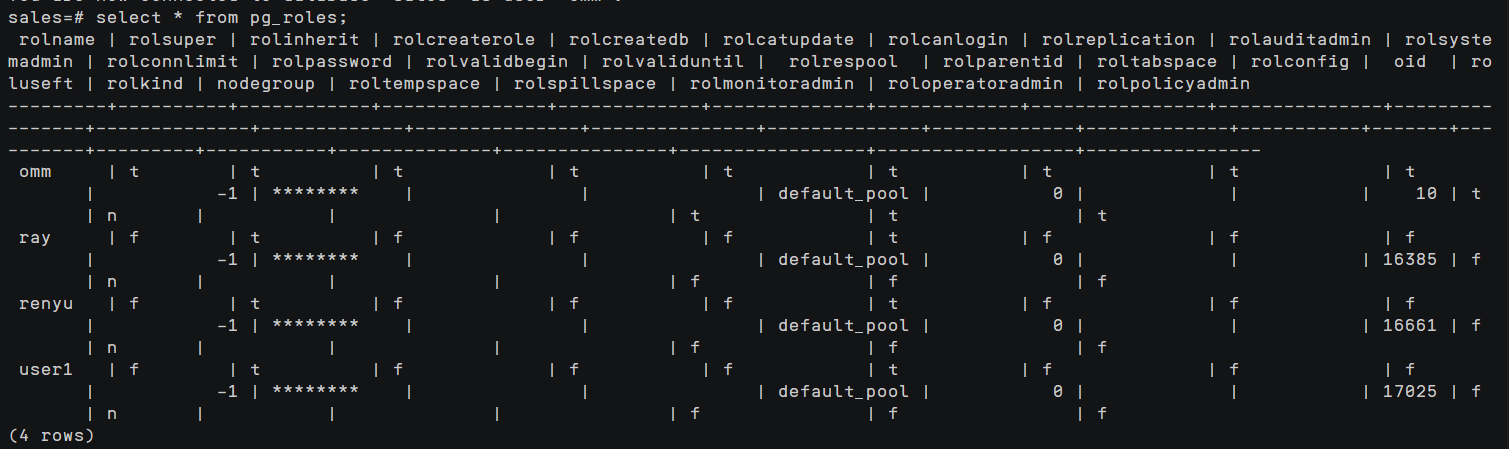


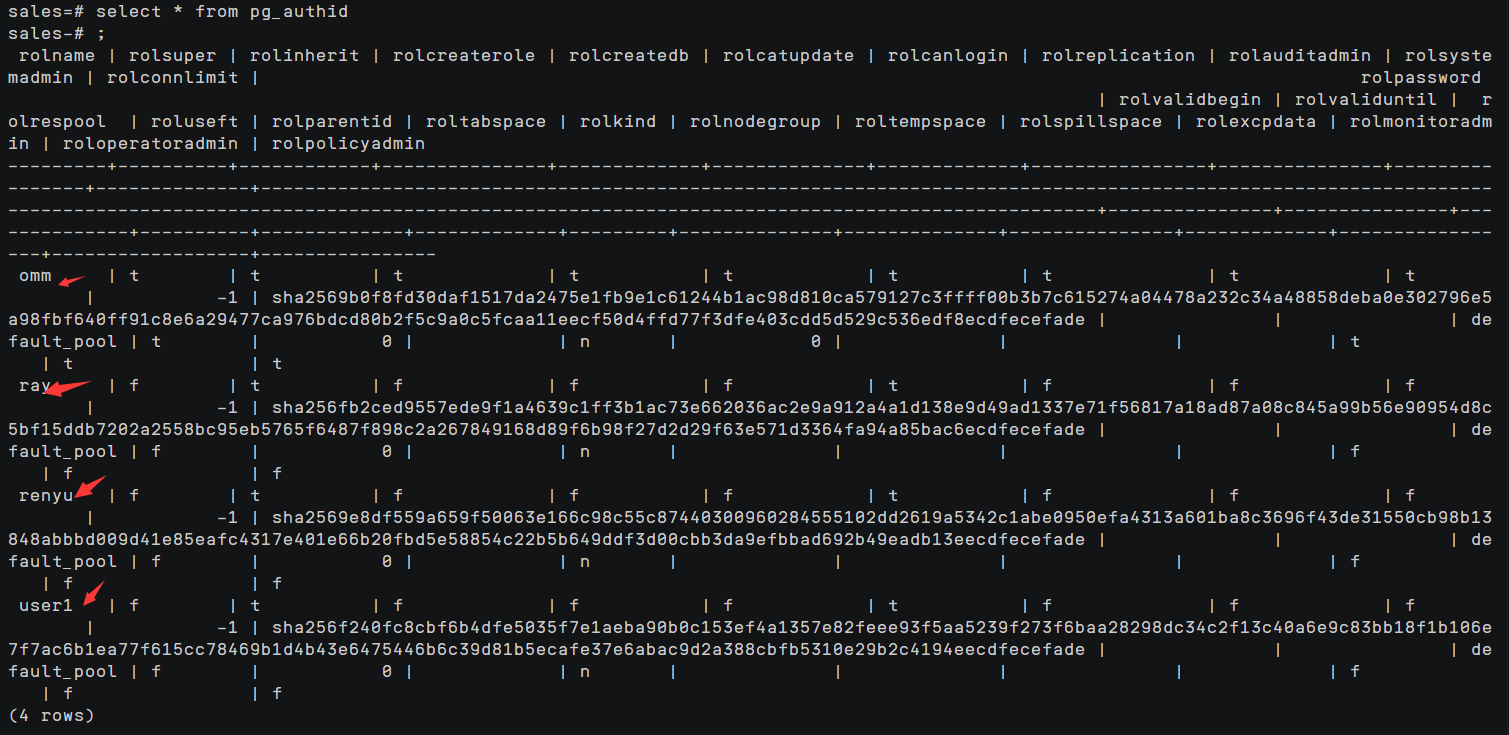




比较两次结果：第一次查询缺少对模式sales的使用权限和对视图的查询权限，因此找不到视图，而在授予对应的权限后，可以查询该视图。

1. 查看与角色、权限相关的系统表和系统视图：pg\_roles，pg\_authid:





1. 在完成（1）的基础上，重做教材中的[例 4.1-例 4.13]，因为 openGauss 的语法与教材上的不完全一致，可以通过以上实操加深对 openGauss 安全性控制机制的理解:
2. 把查询Student表的权限授给U1

**GRANT SELECT ON TABLE Student TO U1;**



1. 把对Student表和Course表的全部操作权限授予用户U2和U3

**GRANT ALL PRIVILEGES ON TABLE Student,Course TO U2,U3;**



1. 把对SC表的查询权限授予所有用户

**GRANT SELECT ON TABLE SC TO PUBLIC;**



1. 把查询Student表和修改学生学号的权限授予用户U4

**GRANT UPDATE(Sno),SELECT ON TABLE Student TO U4;**



1. 把对表SC的INSERT权限授予用户U5，并允许将此权限再授予其他用户

**GRANT INSERT ON TABLE SC TO U5 WITH GRANT OPTION;**



1. 把对表SC的INSERT权限授予用户U6，并允许将此权限再授予其他用户

**GRANT INSERT ON TABLE SC TO U6 WITH GRANT OPTION;**



1. 把对表SC的INSERT权限授予用户U7

**GRANT INSERT ON TABLE SC TO U7;**



1. 把用户U4修改学生学号的权限收回

**REVOKE UPDATE(Sno) ON TABLE Student FROM U4;**



1. 收回所有用户对表SC的查询权限

**REVOKE SELECT ON TABLE SC FROM PUBLIC ;**



1. 把用户U5对SC表的INSERT权限收回

**REVOKE INSERT ON TABLE SC FROM U5 CASCADE;**



1. 创建角色R1（与书中不同，openguass需要密码），然后使用GRANT语句，使R1具有Student表的SELECT、UPDATE、INSERT权限，将这个权限授予U1，U2，U3（对应书中的王平、张明、赵玲），通过R1一次性收回U1（王平）的这三个权限

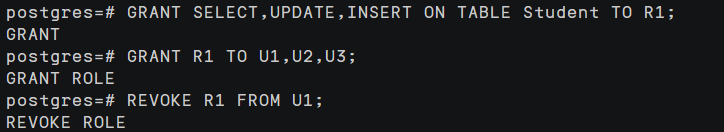
**CREATE ROLE R1 PASSWORD '123456abc!';**

**GRANT SELECT,UPDATE,INSERT ON TABLE Student TO R1;**

**GRANT R1 TO U1,U2,U3;**

**REVOKE R1 FROM U1;**





1. 角色权限的修改

**GRANT DELETE ON TABLE Student TO R1;**



1. 使R1减少SELECT权限

**REVOKE SELECT ON TABLE Student FROM R1;**



1. 实验总结

4.1 完成的工作

1. 登录到创建的Sales数据库中；

2. 进行了角色的创建、权限的授权与回收、查看权限等操作，学习了解了对应的SQL语句；

4.2 对实验的认识

通过本次实验，对于数据库存取控制中的自主存取控制有了更深的理解，用户对于不同数据库对象有着不同的存取权限，不同用户对同一数据库对象也有不同的存取权限，同时也学习了如何使用GRANT和REVOKE来控制权限，以及角色和视图的操作。

4.3 遇到的困难及解决方法

无。

1. 实验思考

 具有什么权限才能创建新用户？

答：拥有DBA权限的用户是系统中的超级用户，只有拥有DBA权限才可以创建新用户。

 角色的作用是什么？

答：角色是被命名的一组与数据库操作相关的权限，角色是权限的集合，因此，可以为一组具有相同权限的用户创建一个角色，使用角色来管理数据库可以简化授权的过程。

 如何实现角色所含权限的修改，请设计样例验证之。

答：对于角色，可以使用GRNAT授予R1权限，使用REVOKE来收回权限。

如角色R1，现在其具有对Student表的UPDATE，INSERT,DELETE权限，为其增加SELECT权限，再收回所有权限

**GRANT SELECT ON TABLE Student TO R1;**



**REVOKE ALL PRIVILEGES ON TABLE Student FROM R1;**

