电子科技大学计算机科学与工程学院

标 准 实 验 报 告

（实验）课程名称 数据库原理及应用

电子科技大学教务处制表

电 子 科 技 大 学

实　验　报　告

学生姓名：　张力群　　　学　号：2020080301001　　指导教师：　孙明

实验地点：立人楼 B106　　　　　　实验时间：2022 年 12 月 2 日

一、实验项目名称：SQL 实验

二、实验学时：4

三、实验内容：用 sql 语句完成给定题目

四、实验环境：openGauss (2.0.0)

五、实验数据及结果分析

六、实验结论

报告评分：

指导教师签字：

基于openGauss的sql实验

一、实验项目名称

基于openGauss的sql实验

二、实验学时

4个学时

三、实验内容

在给定数据集中，利用openGauss对数据集进行数据库简单数据管理和高级数据管理的应用，包括对函数，存储过程，触发器，索引的使用等

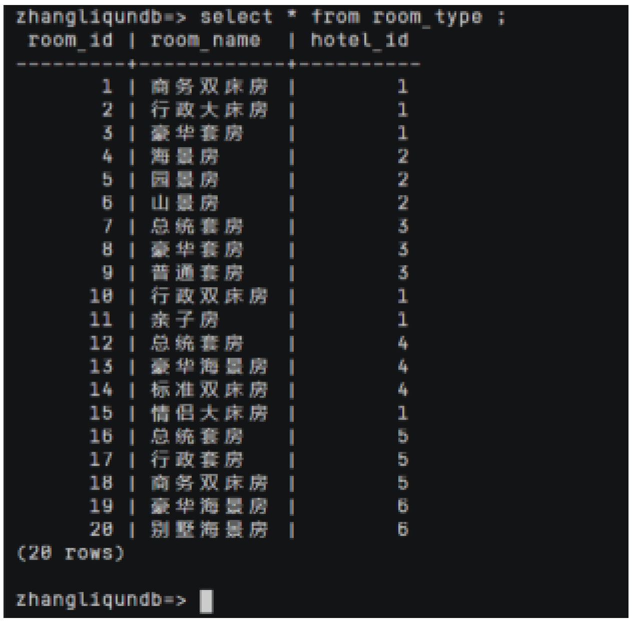
四、实验环境

openGauss(2.0.0)

五、实验数据及结果分析

1.查询所有房型的具体信息，包括room_id, Room_name, hotel_id。

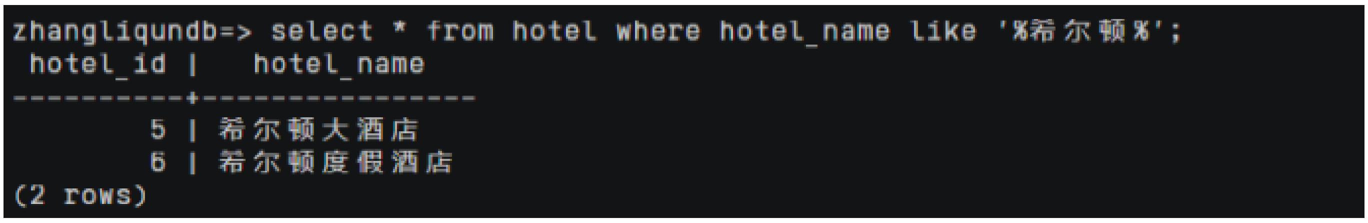
```
select * from room_type ;
```



```
zhangliqundb=> select * from room_type ;
 room_id | room_name    | hotel_id
---------+--------------+----------
       1 | 商务双床房   |        1
       2 | 行政大床房   |        1
       3 | 豪华套房     |        1
       4 | 海景房       |        2
       5 | 园景房       |        2
       6 | 山景房       |        2
       7 | 总统套房     |        3
       8 | 豪华套房     |        3
       9 | 普通套房     |        3
      10 | 行政双床房   |        1
      11 | 亲子房       |        1
      12 | 总统套房     |        4
      13 | 豪华海景房   |        4
      14 | 标准双床房   |        4
      15 | 情侣大床房   |        1
      16 | 总统套房     |        5
      17 | 行政套房     |        5
      18 | 商务双床房   |        5
      19 | 豪华海景房   |        6
      20 | 别墅海景房   |        6
(20 rows)

zhangliqundb=> 
```

2.查询所有酒店名称中包含“希尔顿”的酒店，返回酒店名称和酒店id。

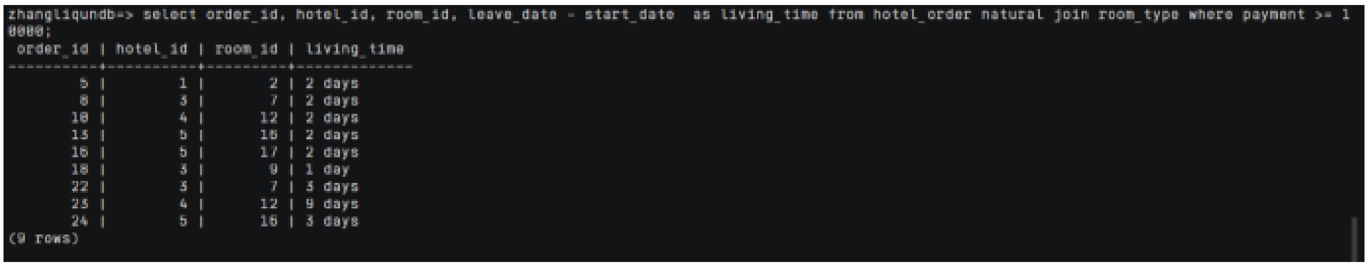
```
select * from hotel where hotel_name like '%希尔顿%';
```



```
zhangliqundb=> select * from hotel where hotel_name like '%希尔顿%';
 hotel_id |    hotel_name
----------+-----------------
        5 | 希尔顿大酒店
        6 | 希尔顿度假酒店
(2 rows)
```

3.查询订单总价在10000元及以上的所有订单详情，包括订单编号、 酒店编号、房型编号及居住时长。

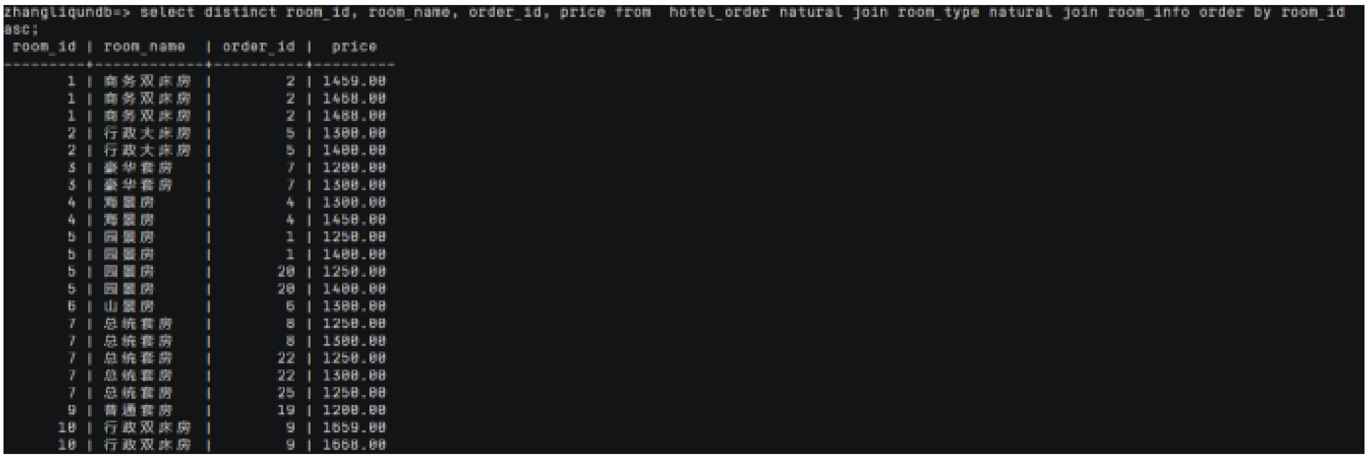
```sql
select order_id, hotel_id, room_id, leave_date - start_date  as living_time
from hotel_order natural join room_type where payment >= 10000;
```



```
zhangliqundb=> select order_id, hotel_id, room_id, leave_date - start_date  as living_time from hotel_order natural join room_type where payment >= 1
0000;
 order_id | hotel_id | room_id | living_time
----------+----------+---------+-------------
        5 |        1 |       2 | 2 days
        8 |        3 |       7 | 2 days
       10 |        4 |      12 | 2 days
       15 |        5 |      16 | 2 days
       16 |        5 |      17 | 2 days
       18 |        3 |       9 | 1 day
       22 |        3 |       7 | 3 days
       23 |        4 |      12 | 9 days
       24 |        5 |      16 | 3 days
(9 rows)
```

4.查询所有房型的订单情况，包括房型编号，房型名称，订单编号、 价格。

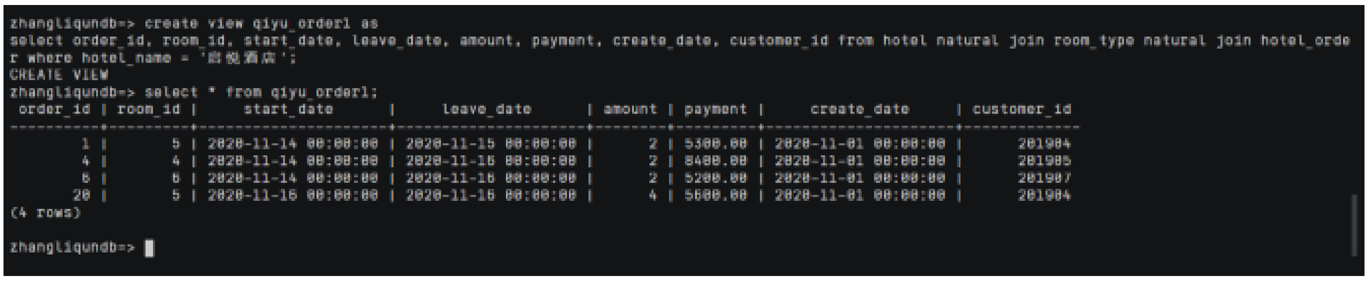
```sql
select distinct room_id, room_name, order_id, price
from  hotel_order natural join room_type natural join room_info
order by room_id asc;
```



```
zhangliqundb=> select distinct room_id, room_name, order_id, price from  hotel_order natural join room_type natural join room_info order by room_id
asc;
 room_id | room_name   | order_id |  price
---------+-------------+----------+---------
       1 | 商务双床房  |        2 | 1459.00
       1 | 商务双床房  |        2 | 1460.00
       1 | 商务双床房  |        2 | 1400.00
       2 | 行政大床房  |        5 | 1300.00
       2 | 行政大床房  |        5 | 1400.00
       3 | 豪华套房    |        7 | 1200.00
       3 | 豪华套房    |        7 | 1300.00
       4 | 海景房      |        4 | 1300.00
       4 | 海景房      |        4 | 1450.00
       5 | 园景房      |        1 | 1250.00
       5 | 园景房      |        1 | 1400.00
       5 | 园景房      |       20 | 1250.00
       5 | 园景房      |       20 | 1400.00
       6 | 山景房      |        6 | 1300.00
       7 | 总统套房    |        8 | 1250.00
       7 | 总统套房    |        8 | 1300.00
       7 | 总统套房    |       22 | 1250.00
       7 | 总统套房    |       22 | 1300.00
       7 | 总统套房    |       25 | 1250.00
       9 | 普通套房    |       19 | 1200.00
      10 | 行政双床房  |        9 | 1659.00
      10 | 行政双床房  |        9 | 1660.00
```

5.创建启悦酒店的订单视图。

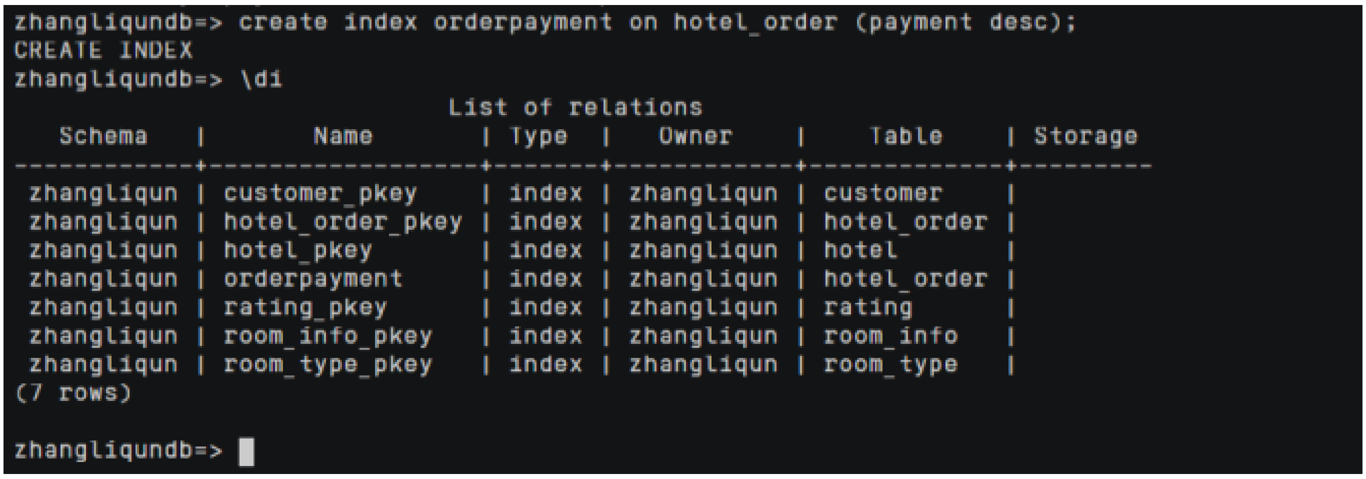
```sql
create view qiyu_order1 as
select order_id, room_id, start_date, leave_date, amount, payment, create_date,
customer_id
from hotel natural join room_type natural join hotel_order
where hotel_name = '启悦酒店';
select * from qiyu_order1;
```



```
zhangliqundb=> create view qiyu_orderl as
select order_id, room_id, start_date, leave_date, amount, payment, create_date, customer_id from hotel natural join room_type natural join hotel_orde
r where hotel_name = '启悦酒店';
CREATE VIEW
zhangliqundb=> select * from qiyu_orderl;
 order_id | room_id |      start_date       |      leave_date       | amount | payment |      create_date      | customer_id
----------+---------+-----------------------+-----------------------+--------+---------+-----------------------+-------------
        1 |       5 | 2020-11-14 00:00:00 | 2020-11-15 00:00:00 |      2 | 5300.00 | 2020-11-01 00:00:00 |      201904
        4 |       4 | 2020-11-14 00:00:00 | 2020-11-16 00:00:00 |      2 | 8400.00 | 2020-11-01 00:00:00 |      201905
        6 |       6 | 2020-11-14 00:00:00 | 2020-11-16 00:00:00 |      2 | 5200.00 | 2020-11-01 00:00:00 |      201907
       20 |       5 | 2020-11-16 00:00:00 | 2020-11-16 00:00:00 |      4 | 5600.00 | 2020-11-01 00:00:00 |      201904
(4 rows)

zhangliqundb=> █
```

6.在订单表的总价字段上 创建降序的普通索引。索引名为orderpayment. 用\di 命令查看创建的索引。

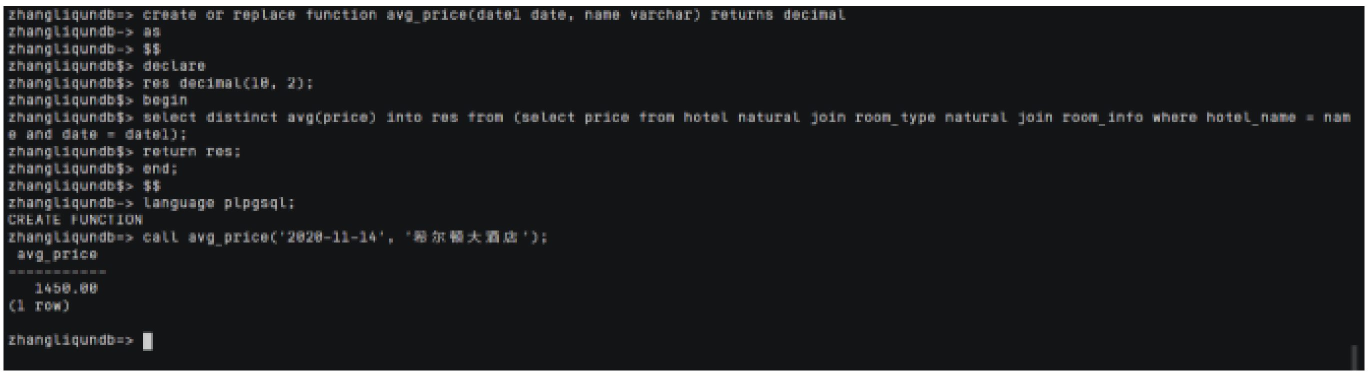
```
create index orderpayment on hotel_order (payment desc);
\di
```



```
zhangliqundb=> create index orderpayment on hotel_order (payment desc);
CREATE INDEX
zhangliqundb=> \di
                           List of relations
   Schema   |       Name       | Type  |   Owner    |    Table    | Storage
------------+------------------+-------+------------+-------------+---------
 zhangliqun | customer_pkey    | index | zhangliqun | customer    |
 zhangliqun | hotel_order_pkey | index | zhangliqun | hotel_order |
 zhangliqun | hotel_pkey       | index | zhangliqun | hotel       |
 zhangliqun | orderpayment     | index | zhangliqun | hotel_order |
 zhangliqun | rating_pkey      | index | zhangliqun | rating      |
 zhangliqun | room_info_pkey   | index | zhangliqun | room_info   |
 zhangliqun | room_type_pkey   | index | zhangliqun | room_type   |
(7 rows)

zhangliqundb=> █
```

7.创建函数：查询给定日期，给定酒店所有房型的平均价格。执行函数，输入参数为2020-11-14，希尔顿大酒店

```
create or replace function avg_price(date1 date, name varchar)
returns decimalas
$$
declare
    res decimal(10, 2);
begin
    select distinct avg(price) into res from
    (select price from hotel natural join room_type natural join room_info
    where hotel_name = name and date = date1);
    return res;
end;
$$
language plpgsql;
```

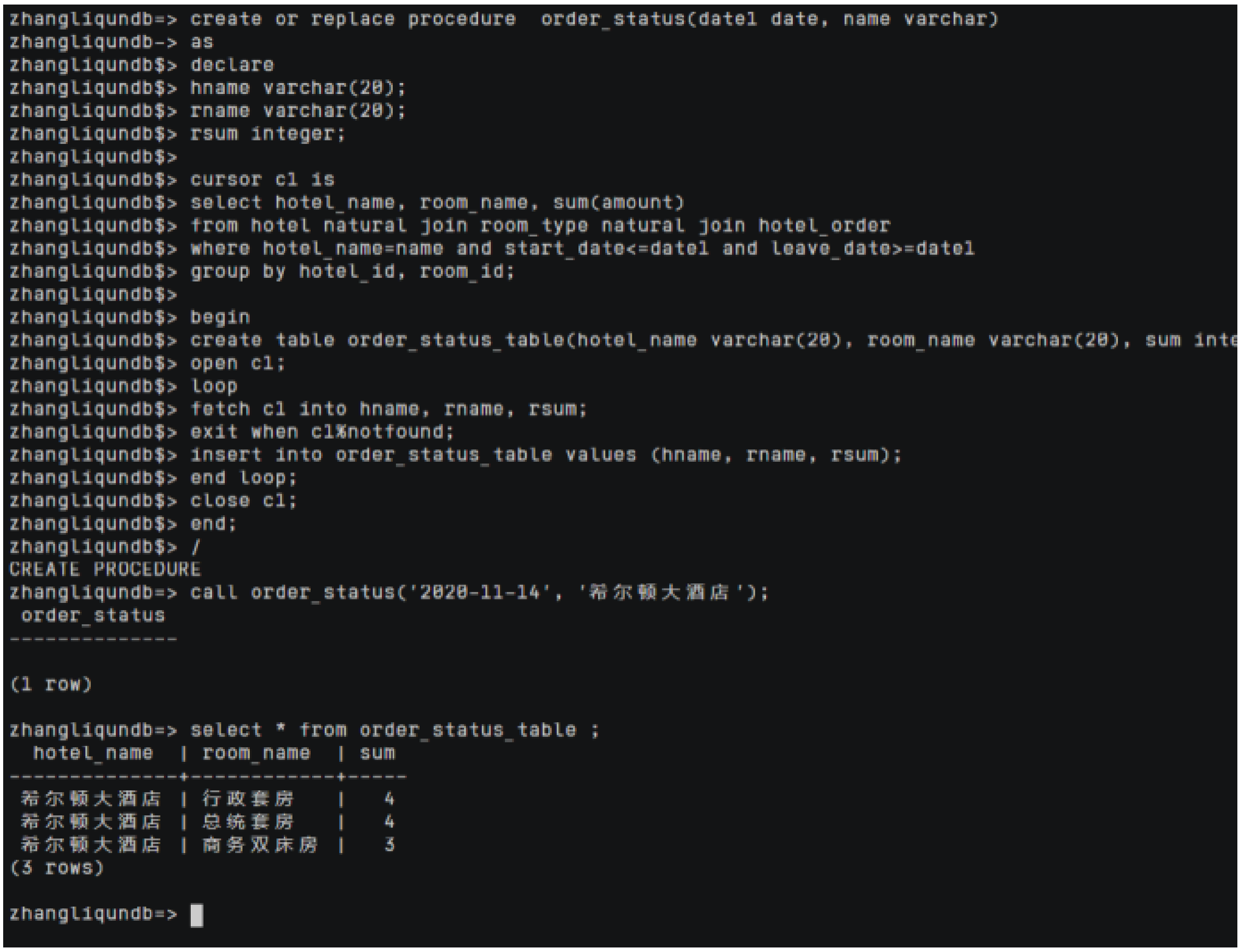


```
zhangliqundb=> create or replace function avg_price(date1 date, name varchar) returns decimal
zhangliqundb-> as
zhangliqundb-> $$
zhangliqundb$> declare
zhangliqundb$> res decimal(10, 2);
zhangliqundb$> begin
zhangliqundb$> select distinct avg(price) into res from (select price from hotel natural join room_type natural join room_info where hotel_name = nam
e and date = date1);
zhangliqundb$> return res;
zhangliqundb$> end;
zhangliqundb$> $$
zhangliqundb-> language plpgsql;
CREATE FUNCTION
zhangliqundb=> call avg_price('2020-11-14', '希尔顿大酒店');
 avg_price
-----------
   1450.00
(1 row)

zhangliqundb=> 
```

8.创建存储过程：从订单表中统计指定酒店、指定日期的各种房型的预订情况，返回酒店名，房型，预定数量。 执行存储过程：统计希尔顿大酒店2020-11-14当天各个房型预定情况

> 利用存储过程创建一个新表，将结果插入到新表中

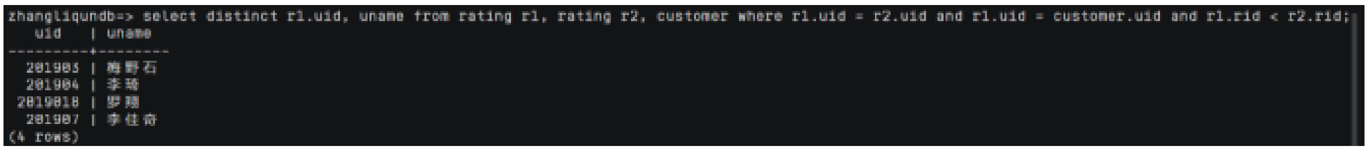
```
create or replace procedure  order_status(date1 date, name varchar)
as
declare
    hname varchar(20);
    rname varchar(20);
    rsum integer;

    cursor c1 is
    select hotel_name, room_name, sum(amount)
    from hotel natural join room_type natural join hotel_order
    where hotel_name=name and start_date<=date1 and leave_date>=date1
    group by hotel_id, room_id;

begin
    create table order_status_table(hotel_name varchar(20), room_name varchar(20),
sum integer);
    open c1;
    loop
        fetch c1 into hname, rname, rsum;
        exit when c1%notfound;
        insert into order_status_table values (hname, rname, rsum);
    end loop;
    close c1;
end;
/
call order_status('2020-11-14', '希尔顿大酒店');
select * from order_status_table;
```



```
zhangliqundb=> create or replace procedure  order_status(date1 date, name varchar)
zhangliqundb-> as
zhangliqundb$> declare
zhangliqundb$> hname varchar(20);
zhangliqundb$> rname varchar(20);
zhangliqundb$> rsum integer;
zhangliqundb$>
zhangliqundb$> cursor c1 is
zhangliqundb$> select hotel_name, room_name, sum(amount)
zhangliqundb$> from hotel natural join room_type natural join hotel_order
zhangliqundb$> where hotel_name=name and start_date<=date1 and leave_date>=date1
zhangliqundb$> group by hotel_id, room_id;
zhangliqundb$>
zhangliqundb$> begin
zhangliqundb$> create table order_status_table(hotel_name varchar(20), room_name varchar(20), sum inte
zhangliqundb$> open c1;
zhangliqundb$> loop
zhangliqundb$> fetch c1 into hname, rname, rsum;
zhangliqundb$> exit when c1%notfound;
zhangliqundb$> insert into order_status_table values (hname, rname, rsum);
zhangliqundb$> end loop;
zhangliqundb$> close c1;
zhangliqundb$> end;
zhangliqundb$> /
CREATE PROCEDURE
zhangliqundb=> call order_status('2020-11-14', '希尔顿大酒店');
 order_status
--------------

(1 row)

zhangliqundb=> select * from order_status_table ;
  hotel_name  | room_name   | sum
--------------+-------------+-----
 希尔顿大酒店 | 行政套房    |   4
 希尔顿大酒店 | 总统套房    |   4
 希尔顿大酒店 | 商务双床房  |   3
(3 rows)

zhangliqundb=> 
```

9.查找同时评价了2次及以上的用户信息。

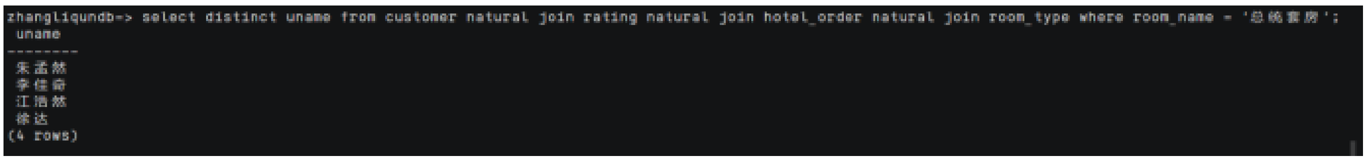
```sql
select distinct r1.uid, uname
from rating r1, rating r2, customer
where r1.uid = r2.uid and r1.uid = customer.uid and r1.rid < r2.rid;
```



```
zhangliqundb=> select distinct r1.uid, uname from rating r1, rating r2, customer where r1.uid = r2.uid and r1.uid = customer.uid and r1.rid < r2.rid;
   uid    | uname
----------+--------
  201903  | 梅野石
  201904  | 李琦
 2019018  | 罗翔
  201907  | 李佳奇
(4 rows)
```

10.查询评价过所有总统套房的顾客姓名。

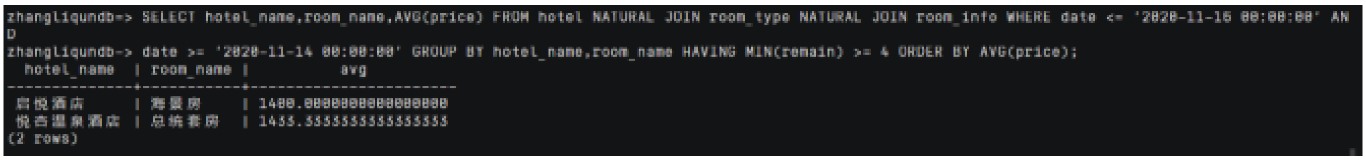
```sql
select distinct uname
from customer natural join rating natural join hotel_order natural join room_type
where room_name = '总统套房';
```



```
zhangliqundb=> select distinct uname from customer natural join rating natural join hotel_order natural join room_type where room_name = '总统套房';
 uname
--------
 朱孟然
 李佳奇
 江浩然
 徐达
(4 rows)
```

11.若要预定11.14-16日每天房间数量4间。查询满足条件（时 间区间，将预定房间数）的房型及其平均价格，并按平均价格从 低到高进行排序。查询结果应包含酒店，房型及平均价格信息。

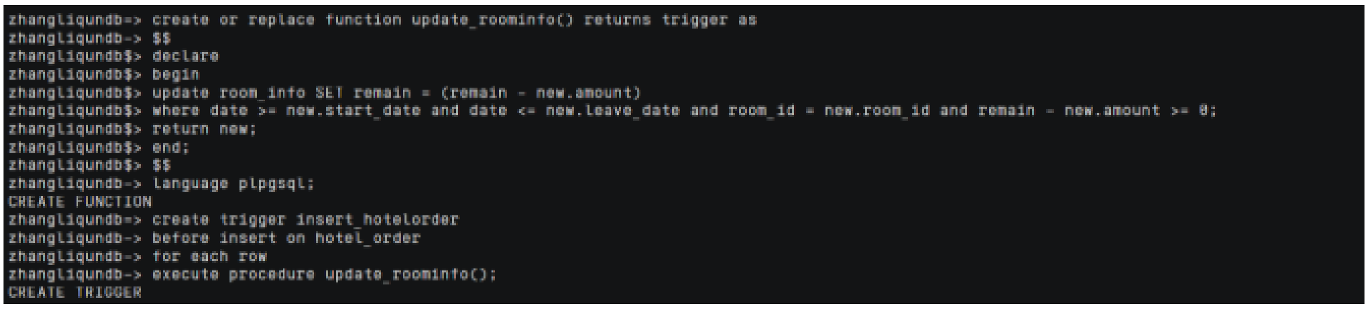
```sql
SELECT hotel_name,room_name,AVG(price)
FROM hotel NATURAL JOIN room_type NATURAL JOIN room_info WHERE date <= '2020-11-16
00:00:00' AND
date >= '2020-11-14 00:00:00'
GROUP BY hotel_name,room_name
HAVING MIN(remain) >= 4 ORDER BY AVG(price);
```



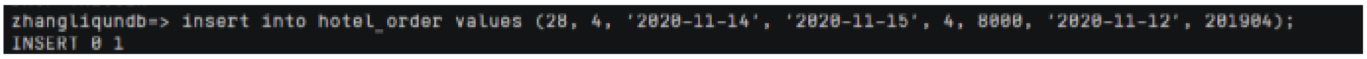
```
zhangliqundb=> SELECT hotel_name,room_name,AVG(price) FROM hotel NATURAL JOIN room_type NATURAL JOIN room_info WHERE date <= '2020-11-16 00:00:00' AN
D
zhangliqundb-> date >= '2020-11-14 00:00:00' GROUP BY hotel_name,room_name HAVING MIN(remain) >= 4 ORDER BY AVG(price);
  hotel_name  | room_name |          avg
--------------+-----------+-----------------------
 启悦酒店     | 海景房    | 1400.0000000000000000
 悦杏温泉酒店 | 总统套房  | 1433.3333333333333333
(2 rows)
```

12.编写触发器：完成预订房间，包括创建订单和更新房型信息。 该订单为预订11月14号-15号4号房型4间。

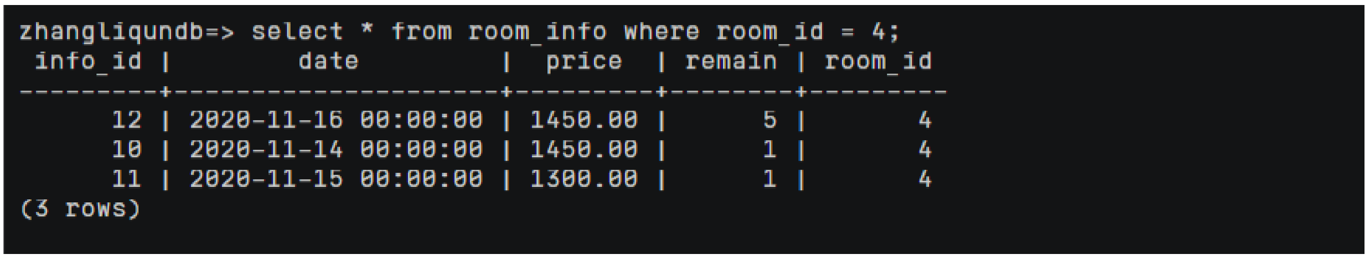
```sql
--触发器函数
CREATE OR REPLACE FUNCTION update_roominfo() RETURNS TRIGGER AS
$$
declare
begin
    update room_info set remain = remain - new.amount
    when date >= new.start_date and date <= new.leave_date and room_id =
new.room_id;
    return new;
end
$$
language plpgsql;

--触发器
create or replace trigger insert_hotelorder
after insert on hotel_order
for each row
execute procedure update_roominfo();

--执行函数
insert into hotel_order
values(28, 4, '2020-11-14', '2020-11-15', 4, 8000, '2020-11-12', 201904);
```



```
zhangliqundb=> create or replace function update_roominfo() returns trigger as
zhangliqundb-> $$
zhangliqundb$> declare
zhangliqundb$> begin
zhangliqundb$> update room_info SET remain = (remain - new.amount)
zhangliqundb$> where date >= new.start_date and date <= new.leave_date and room_id = new.room_id and remain - new.amount >= 0;
zhangliqundb$> return new;
zhangliqundb$> end;
zhangliqundb$> $$
zhangliqundb-> language plpgsql;
CREATE FUNCTION
zhangliqundb=> create trigger insert_hotelorder
zhangliqundb-> before insert on hotel_order
zhangliqundb-> for each row
zhangliqundb-> execute procedure update_roominfo();
CREATE TRIGGER
```



```
zhangliqundb=> insert into hotel_order values (28, 4, '2020-11-14', '2020-11-15', 4, 8000, '2020-11-12', 201904);
INSERT 0 1
```



```
zhangliqundb=> select * from room_info where room_id = 4;
 info_id |        date         |  price  | remain | room_id
---------+---------------------+---------+--------+---------
      12 | 2020-11-16 00:00:00 | 1450.00 |      5 |       4
      10 | 2020-11-14 00:00:00 | 1450.00 |      1 |       4
      11 | 2020-11-15 00:00:00 | 1300.00 |      1 |       4
(3 rows)
```

插入后，room_info中对应日期的remain也发生了相应的变化

六、实验结论

- 通过课堂上的限时测试与课下的思考，对openGauss中的触发器，函数，索引，存储过程都有了更深的理解，了解了用sql语句处理**更复杂的业务流程**
- 不同数据库间sql语句有一定的区别，高级数据的管理结构也有较大差异，需要根据不同的数据库做不同的操作