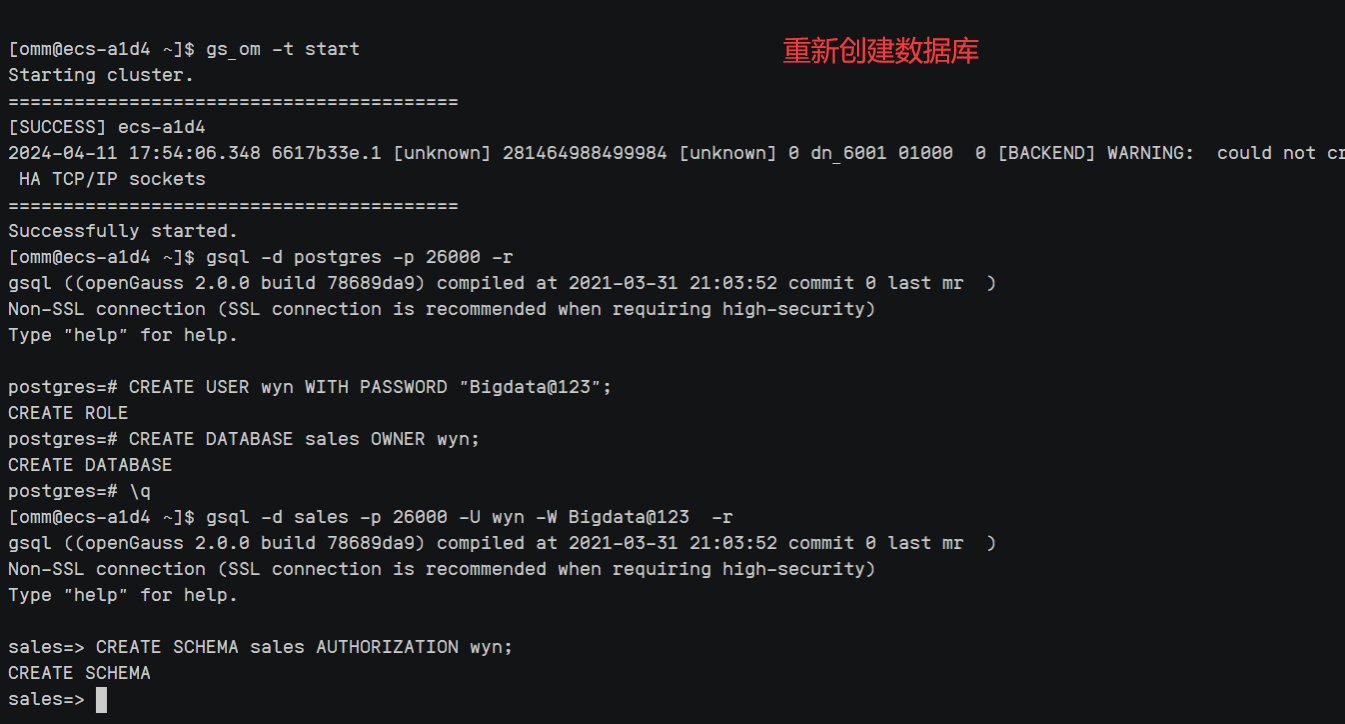


数据库系统课程实验报告

|  |  |
| --- | --- |
| 实验名称： | 实验三：数据定义 |
| 实验日期： | 2024.4.11—2024.4.13 |
| 实验地点： | 文宣楼B313 |
| 提交日期： | 2024.4.13 |
|  | |
| 学号： | 37220222203791 |
| 姓名： | 魏一诺 |
| 专业年级： | 软工2022级 |
| 学年学期： | 2023-2024学年第二学期 |

1. 实验目的
   1. 熟练掌握 openGauss 单表查询的语法结构及其使用方法
   2. 掌握设计正确查询语句以实现查询要求的方法
   3. 简单单表查询 (此处指不涉及模糊、集合、聚集、分组、排序的查询)
   4. 模糊查询、聚集函数、分组统计和排序
   5. 掌握 Group by 的使用
   6. 正确区分元组过滤条件 (WHERE 子) 和分组过滤条件 (HAVING 短语) 的异同
   7. 掌握 Order by的使用
   8. 掌握使用 DISTINCT 实现查询结果的去重方法
   9. 掌握空值 NULL 的使用方法
   10. 掌握表别名的使用场合及方法
   11. 掌握自身连接的使用方法
   12. 掌握使用 LIMIT m [offset n]命令来控制输出行数的方法
2. 实验内容和步骤
3. 连接到第二次实验的数据库，重新创建用户、schema、表，重新添加表的数据

仿照第二次实验中的步骤：

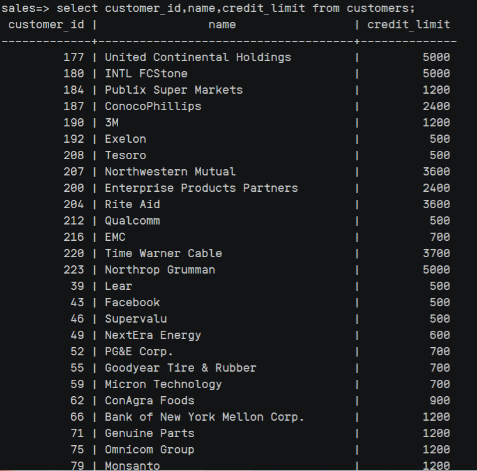


1. 查询顾客表中的顾客号（customer\_id）、顾客名（name）和信用卡额度（credit\_limit）。

步骤如下：

select customer\_id,name,credit\_limit from customers

结果展示：



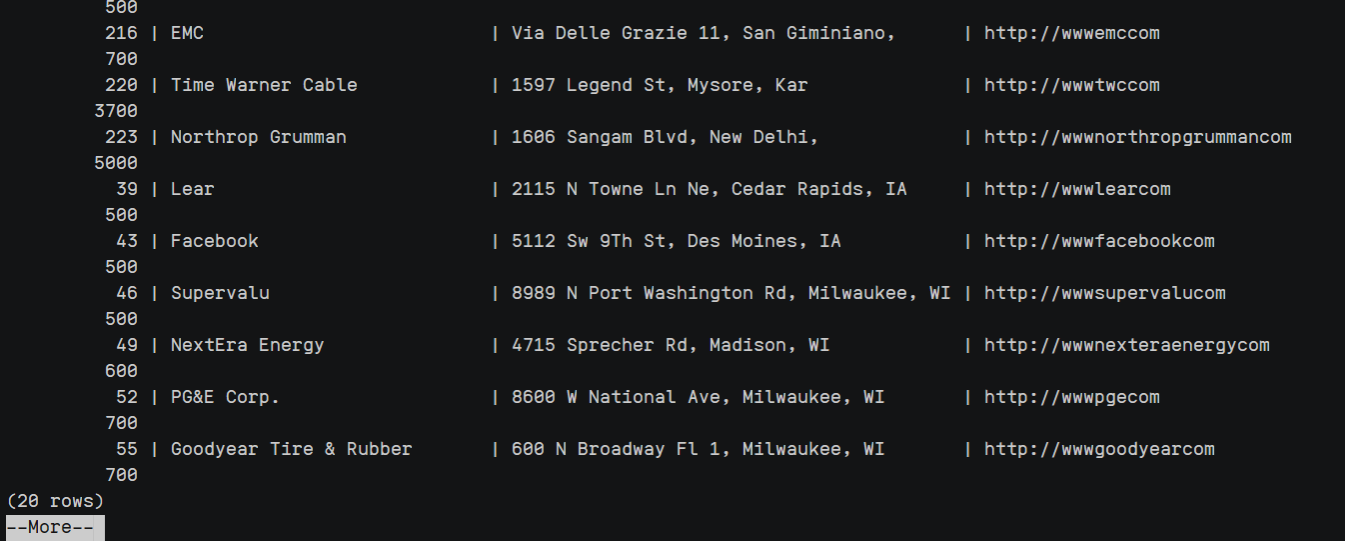
1. 查询顾客的所有信息，且只显示前 20 条记录。

步骤如下：

SELECT \* FROM customers LIMIT 20;  
//退出查询用Ctrl+c

结果展示：

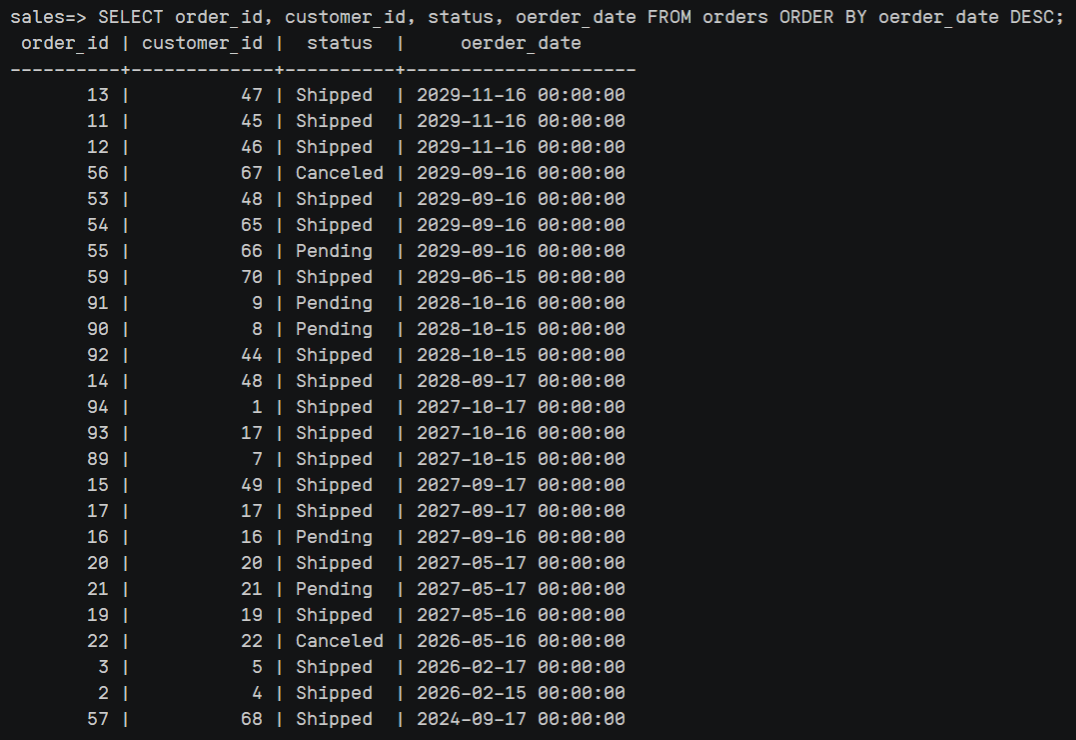




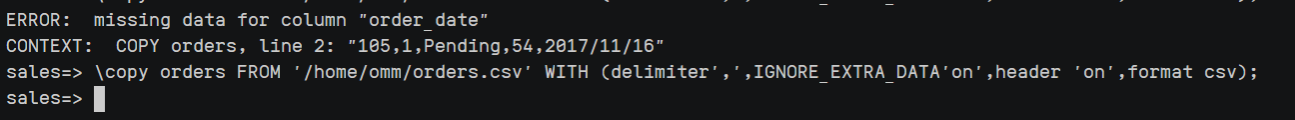
1. 查询订单表中的订单号，顾客号，状态，订单日期，并按订单日期降序显示结果。

SELECT order\_id, customer\_id, status, oerder\_date FROM orders ORDER BY oerder\_date DESC;

结果展示：



报错显示没有orderdate的数据 我改了格式 添加成功了



1. 查询联系表中的名 (first name) 和姓 (last name)，并按名升序，姓降序显示。

步骤如下：

SELECT first\_name, last\_name FROM contacts ORDER BY first\_name ASC, last\_name DESC;

结果展示：



1. 执行以下语句并观察 state 列 NULL 值的显示位置，得出结论。

### SELECT country\_id, city,state FROM locations ORDER BY city,state;

### SELECT country\_id, city,state FROM locations ORDER BY state ASC NULLS FIRST ;（

### SELECT country\_id, city,state FROM locations ORDER BY state ASC NULLS LAST;

结果展示：

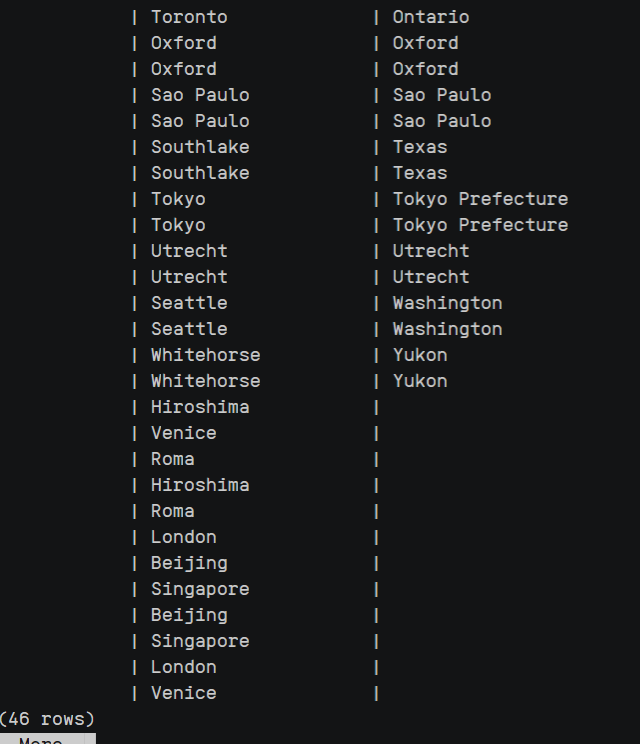
（第一个）



第二个（ASC NULLS FIRST）



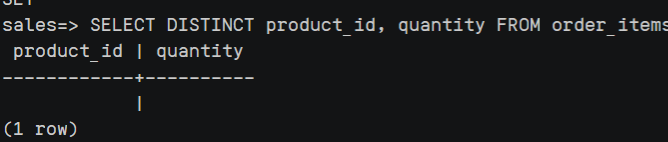
第三个（ASC NULLS LAST）：



结论：ASC NULLS FIRST就是空值的在前面排列，ASC NULLS FLAST是空值在后面排列

### （7） 查询订单细节表中（order\_items）的产品号和数量，查询结果应无重复元组。

SELECT DISTINCT product\_id, quantity FROM order\_items;



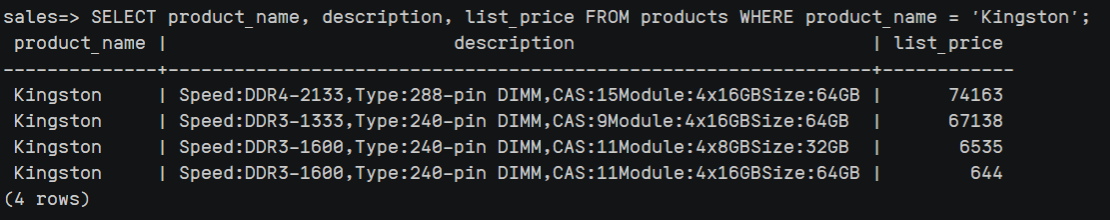
结果展示：

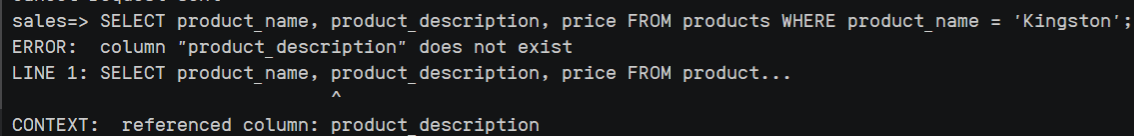


### （8）查询产品表中的产品名为‘Kingston’的产品名，产品描述和价格。

SELECT product\_name, description, list\_price FROM products WHERE product\_name = 'Kingston';

结果展示：



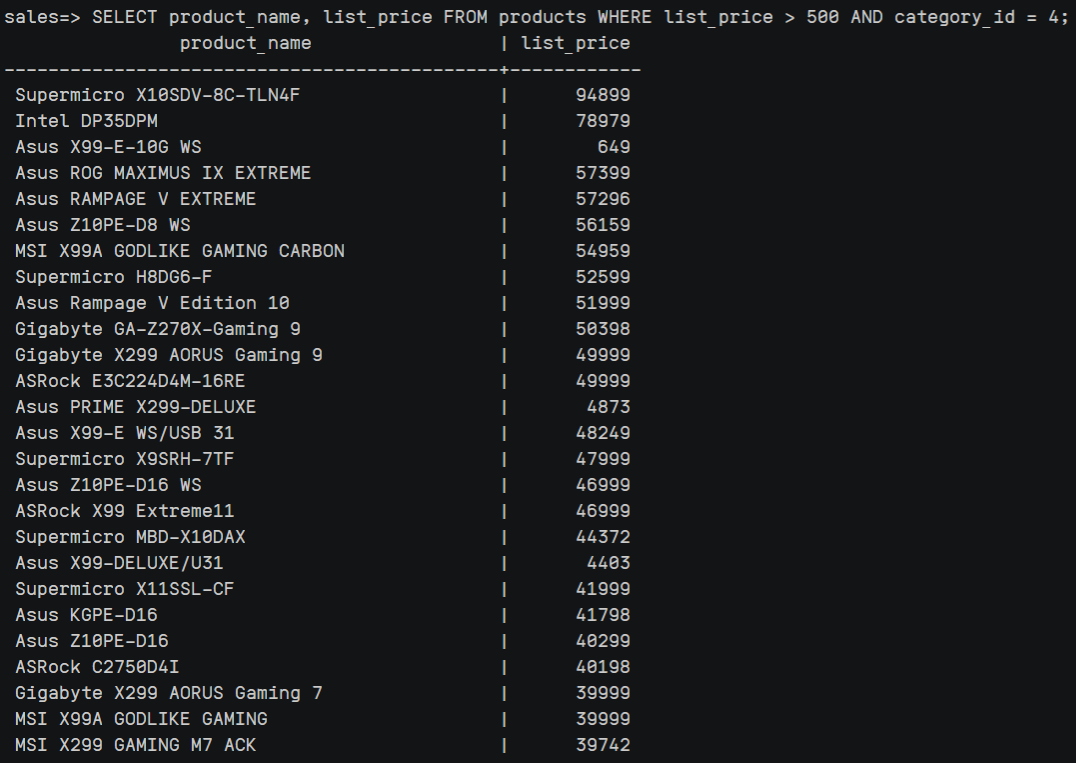


ALTER TABLE products ADD COLUMN description VARCHAR(2000);

### （9）查询产品表中所有价格大于 500 且 category\_id 为 4 的产品名和价格。

SELECT product\_name, list\_price FROM products WHERE list\_price > 500 AND category\_id = 4;

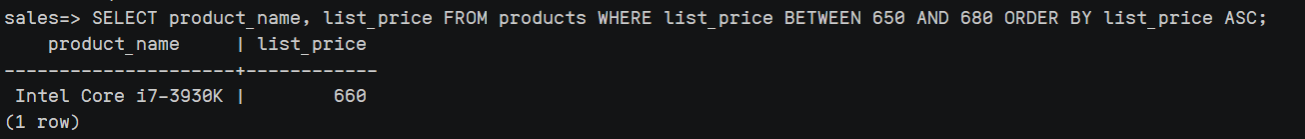
结果展示：



### （10）查询产品表中所有价格在 650 和 680 之间的产品名和价格并按价格升序显示结果。

SELECT product\_name, list\_price FROM products WHERE list\_price BETWEEN 650 AND 680 ORDER BY list\_price ASC;

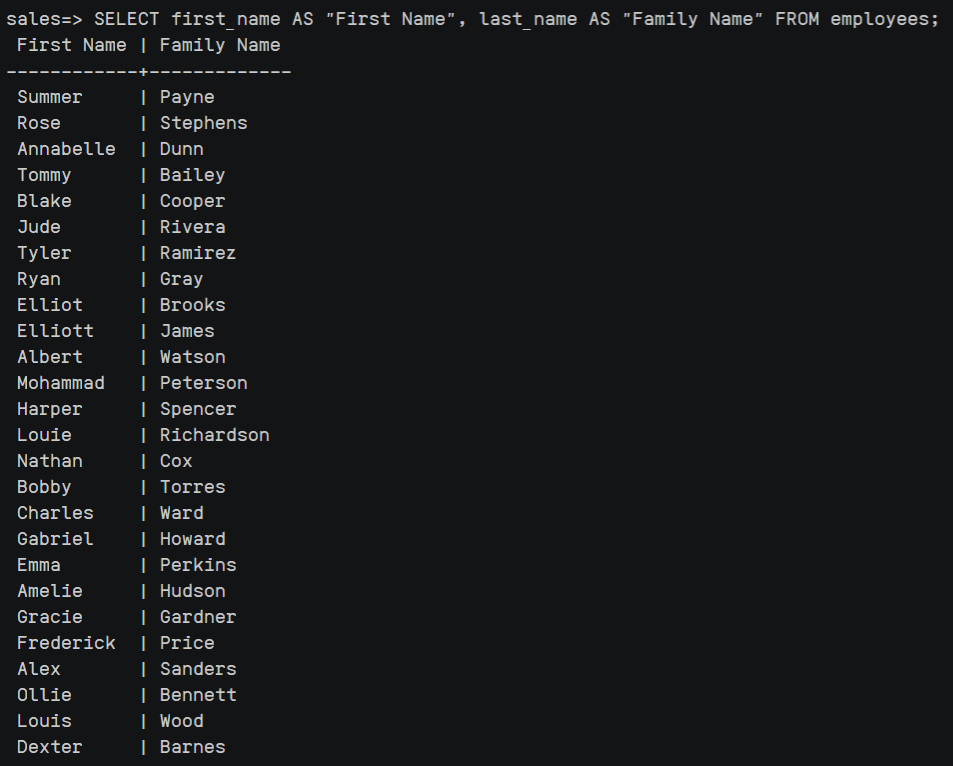
结果展示：



### （10）查询雇员表中的名和姓，名和姓的字段分别显示为”First Name”和”Family Name”。

SELECT first\_name AS "First Name", last\_name AS "Family Name" FROM employees;

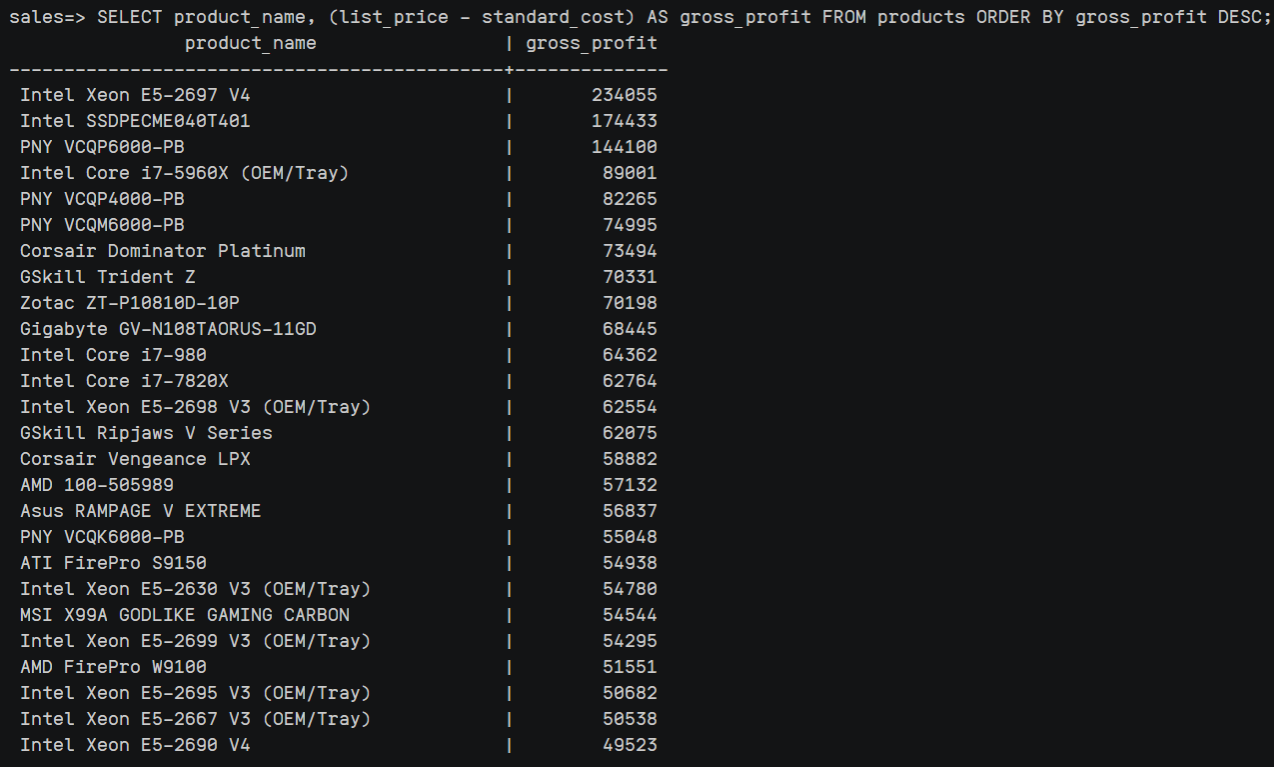
结果展示：



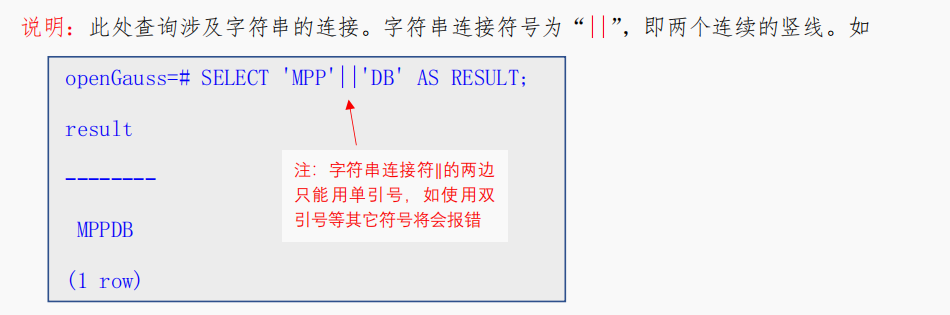
### （11）查询产品表中的产品名及毛利，并按毛利结果降序显示，毛利名为 gross\_profit，毛利=list\_price - standard\_cost。

SELECT product\_name, (list\_price - standard\_cost) AS gross\_profit FROM products ORDER BY gross\_profit DESC;

结果展示：



### （12）查询雇员表中每个雇员对应的经理名，要求第一列字段名为 employee\_name，第二列字段名为 manager\_name（雇员和经理的姓名同一格式为‘first\_name, last\_name’）。

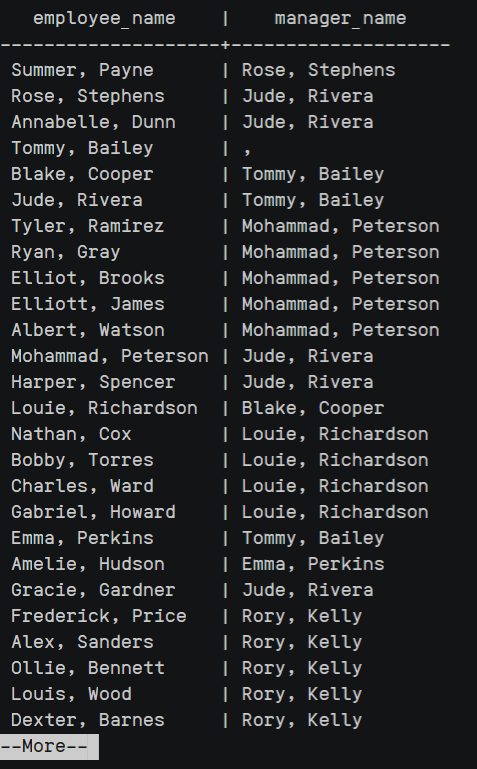


语法参考：[opengauss.org](https://opengauss.org/zh/docs/3.0.0/docs/Developerguide/字符处理函数和操作符.html)

步骤如下：

SELECT e.first\_name || ', ' || e.last\_name AS employee\_name, m.first\_name || ', ' ||m.last\_name AS manager\_name FROM employees e LEFT JOIN employees m ON e.manager\_id = m.employee\_id;

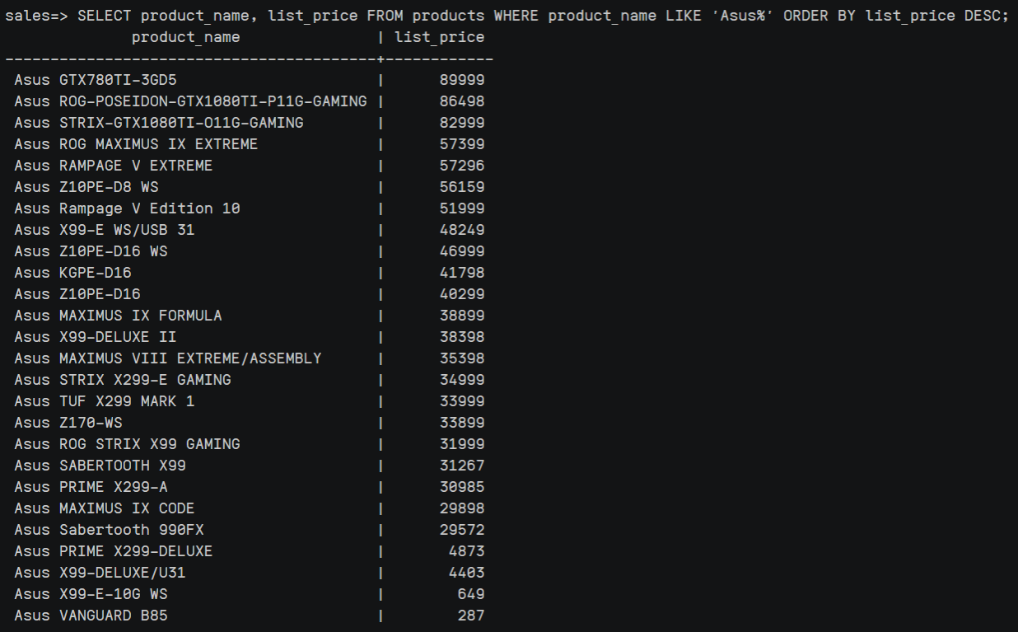
结果展示：



### （13）查询产品表中所有以 Asus 开头的产品名和价格，并以价格降序显示。

SELECT product\_name, list\_price FROM products WHERE product\_name LIKE 'Asus%' ORDER BY list\_price DESC;

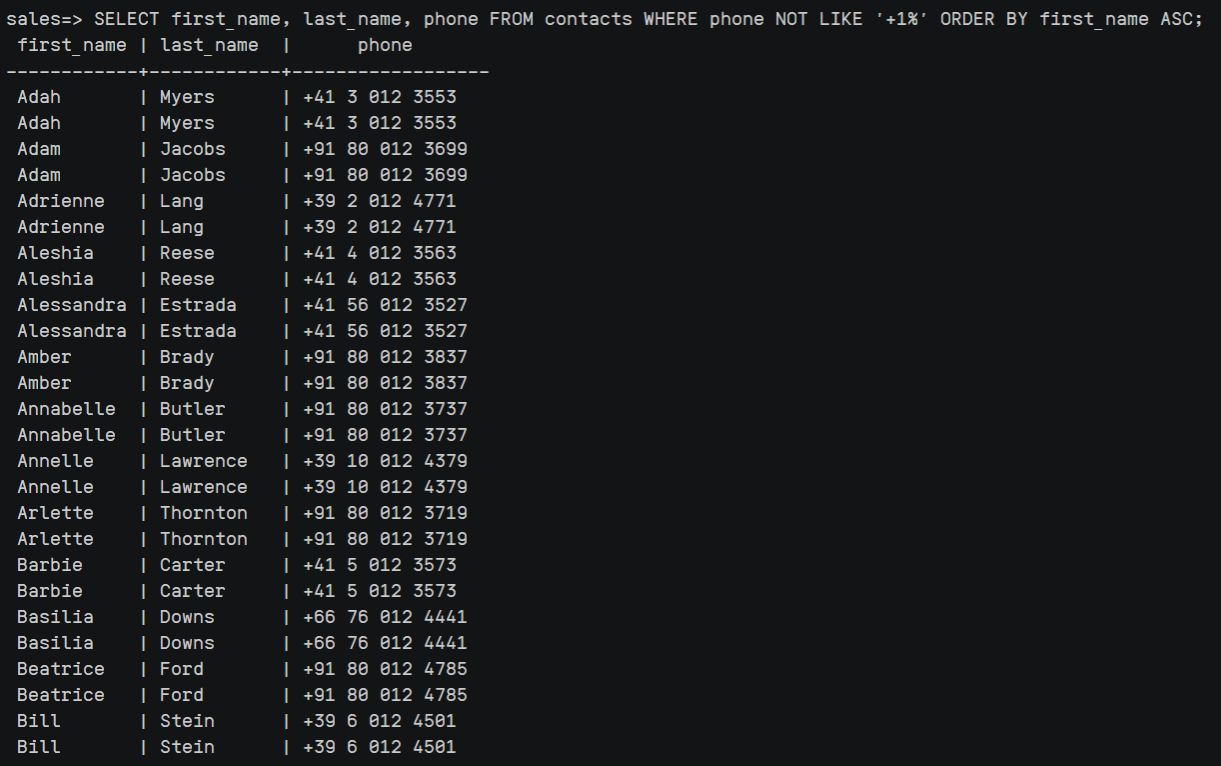
结果展示：



### （14）查询联系表中电话号码不是以‘+1’开头的名、姓和电话号码，并以名升序显示。

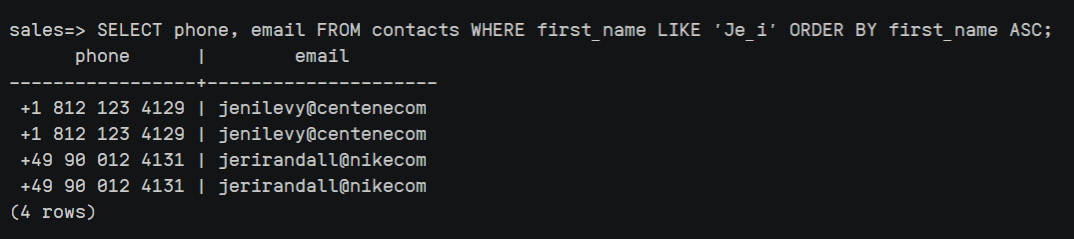
SELECT first\_name, last\_name, phone FROM contacts WHERE phone NOT LIKE '+1%' ORDER BY first\_name ASC;

结果展示：



### （15）查询联系表中的电话号码和电子邮件，要求名(first\_name) 的长度为 4 且以’Je’开头，以’i’结尾，按名升序显示。

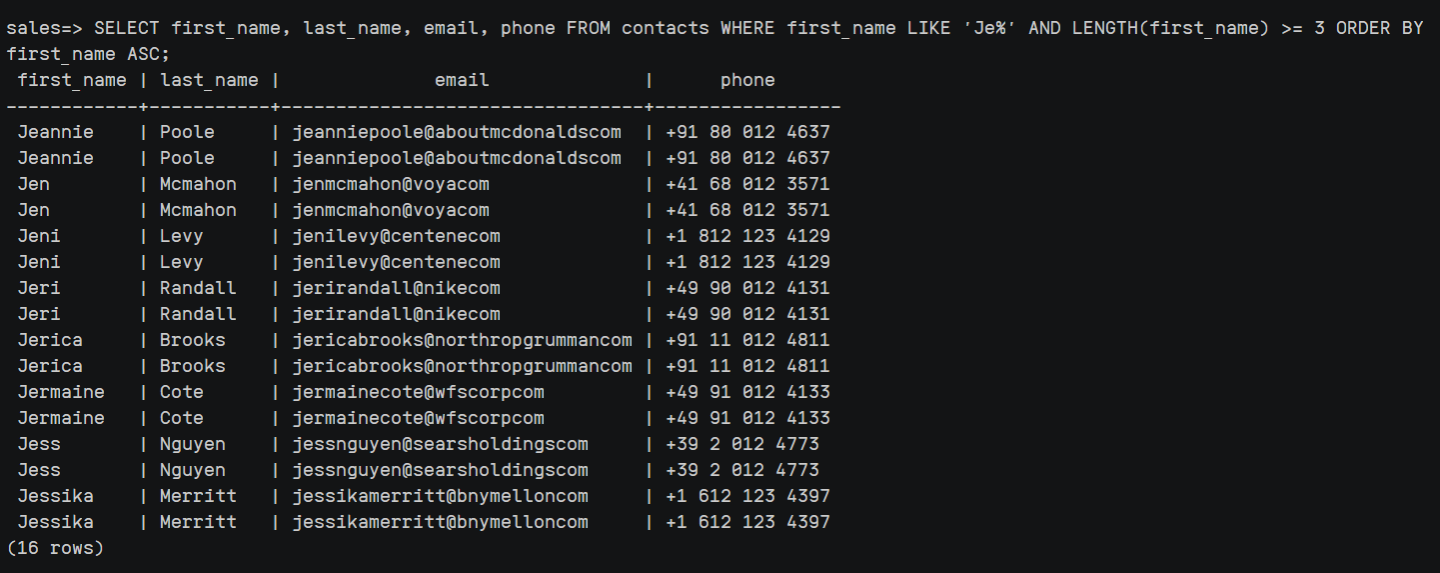
SELECT phone, email FROM contacts WHERE first\_name LIKE 'Je\_i' ORDER BY first\_name ASC;



### （16）查询联系表中所有以开头’Je’的名，且至少包含 3 个字符的名，姓，电子邮件和电话。

SELECT first\_name, last\_name, email, phone FROM contacts WHERE first\_name LIKE 'Je%' AND LENGTH(first\_name) >= 3 ORDER BY first\_name ASC;

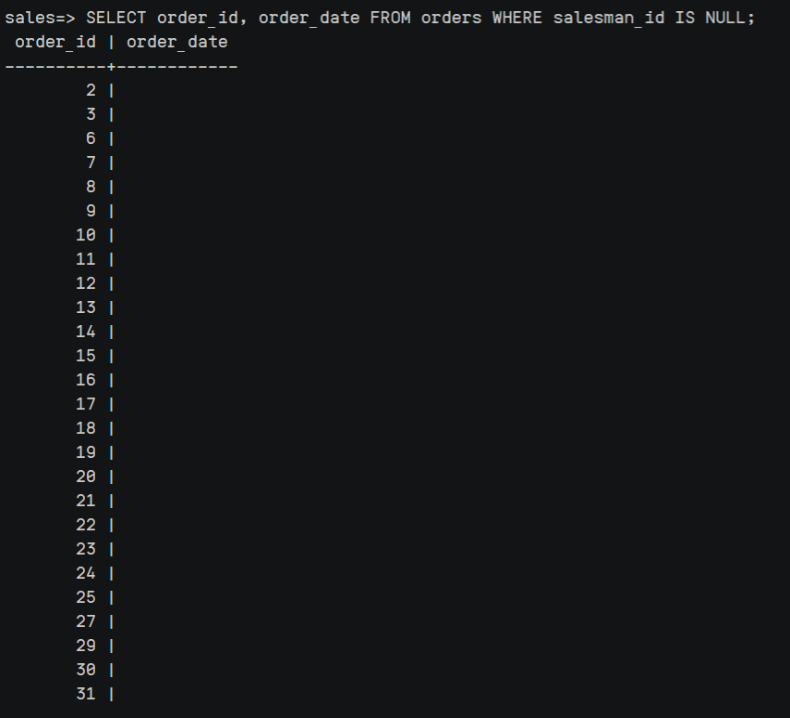
结果展示：



### （17）查询订单表中所有没有销售员负责的订单（i.e., query all sales orders that do nothave a responsible salesman）。

SELECT order\_id, order\_date FROM orders WHERE salesman\_id IS NULL;

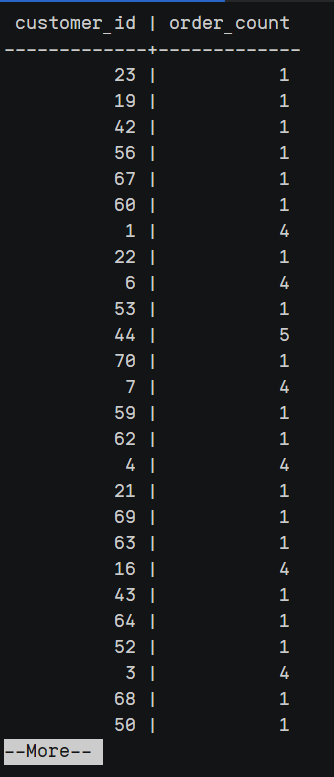
结果展示：



### （18）统计每个顾客的订单总数（查询订单表）。

SELECT customer\_id, COUNT(\*) AS order\_count FROM orders GROUP BY customer\_id;

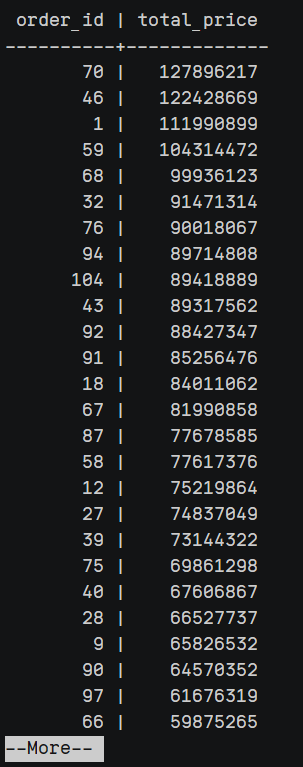
结果展示：



### （19）统计每个订单的总价格大于 1000000 的订单号和总价格，并按总价格降序显示结果。（查询订单细节表 order\_items，总价格=unit\_price\*quantity）

SELECT order\_id, SUM(unit\_price \* quantity) AS total\_price FROM order\_items GROUP BY order\_id HAVING SUM(unit\_price \* quantity) > 1000000 ORDER BY total\_price DESC;

结果展示：



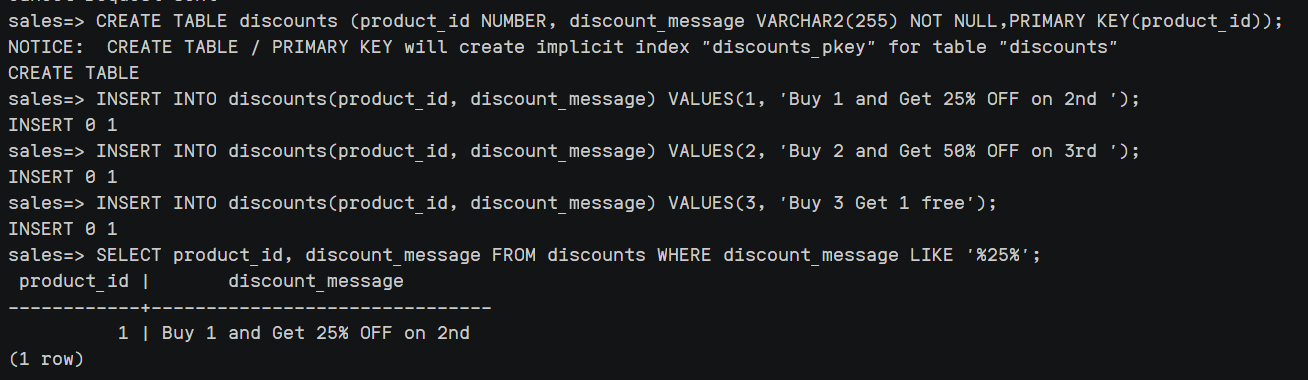
### （20）创建一个折扣表 discounts



CREATE TABLE discounts (product\_id NUMBER, discount\_message VARCHAR2(255) NOT NULL,PRIMARY KEY(product\_id));

SELECT product\_id, discount\_message FROM discounts WHERE discount\_message LIKE '%25%';

结果展示：



1. 实验总结

3.1 完成的工作

按照实验要求进行各项查询任务

学习了很多查询语句的写法

3.2 对实验的认识

对查询语句的使用方法有了更深的理解

3.3 遇到的困难及解决方法

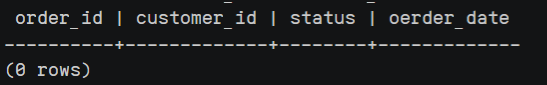
1. 做题目1（查询顾客表中的顾客号（customer\_id）、顾客名（name）和信用卡额度（credit\_limit））时

**遇到问题：**输入语句后并没有反应

**解决：**尝试了补充设置搜索路径 还是没用；发现语句里customers写成customer了，修改之后还是没有解决。尝试一下用SELECT \* FROM customers LIMIT 10;输出表里十条数据，结果莫名其妙就对了

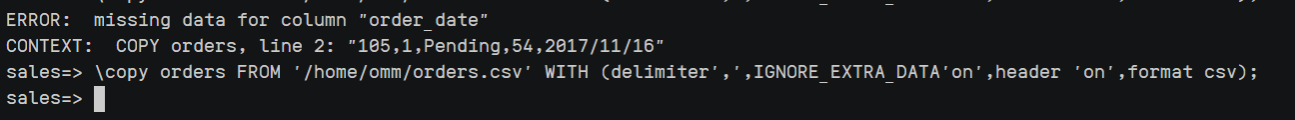
1. 做查询订单表中的订单号，顾客号，状态，订单日期，并按订单日期降序显示结果时：

**遇到问题：**



说明没添加数据成功呗 再加一次

报错显示没有order\_date的数据 我改了csv文件里那栏的格式就添加成功了。



还有一些错误 大多数是因为数据格式、指令拼写错误或者数据名写错了，column名写错了等原因，最终都解决了。