数据库系统课程实验报告

实验名称：	数据基本查询
实验日期：	2022.10.28
实验地点：	四号楼
提交日期：	2022.10.30

学号：	22920202202877
姓名：	陈鑫蕾
专业年级：	数媒 2020 级
学年学期：	2022-2023 学年第一学期

1. 实验目的

- 熟练掌握 openGauss 单表查询的语法结构及其使用方法
- 掌握设计正确查询语句以实现查询要求的方法

  -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）

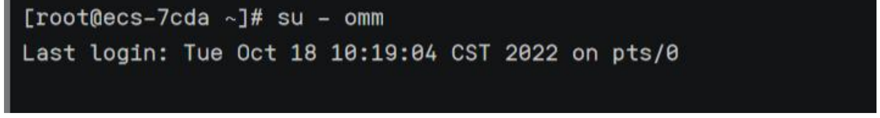
  -模糊查询、聚集函数、分组统计和排序
- 掌握 Group by 的使用
- 正确区分元组过滤条件（WHERE 子句）和分组过滤条件（HAVING 短语）的异同
- 掌握 Order by 的使用
- 掌握使用 DISTINCT 实现查询结果的去重方法
- 掌握空值 NULL 的使用方法
- 掌握表别名的使用场合及方法
- 掌握自身连接的使用方法

2. 实验内容和步骤

（1）连接到数据库

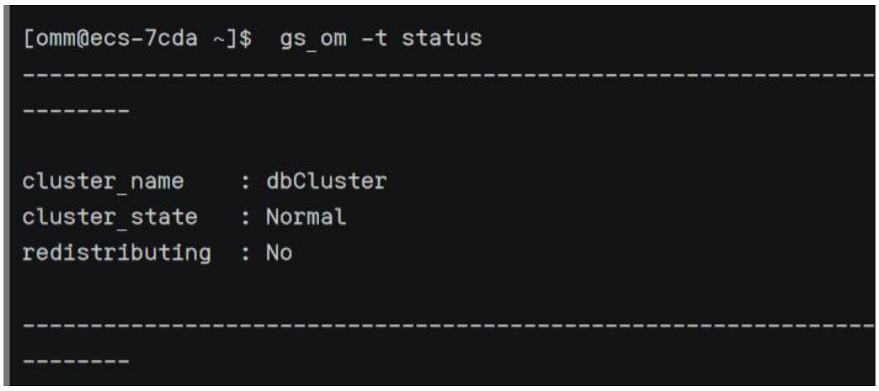
步骤如下:

1.在数据库主节点服务器上，切换至 omm 操作系统用户环境。



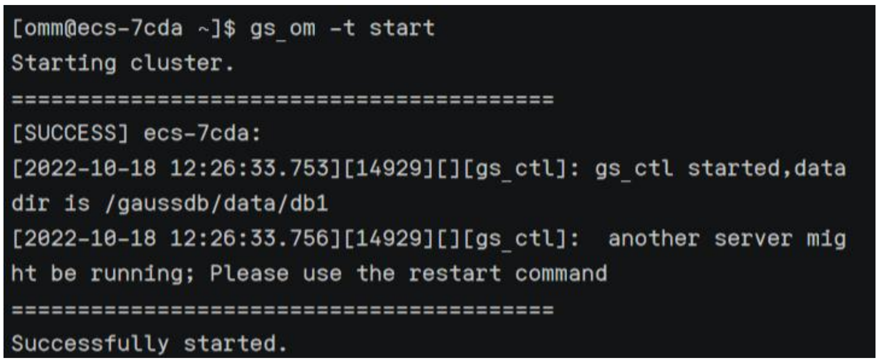
```
-bash: gsql: command not found
[root@ecs-7cda ~]# su - omm
Last login: Tue Oct 18 10:19:04 CST 2022 on pts/0
```

2.查看服务是否启动。



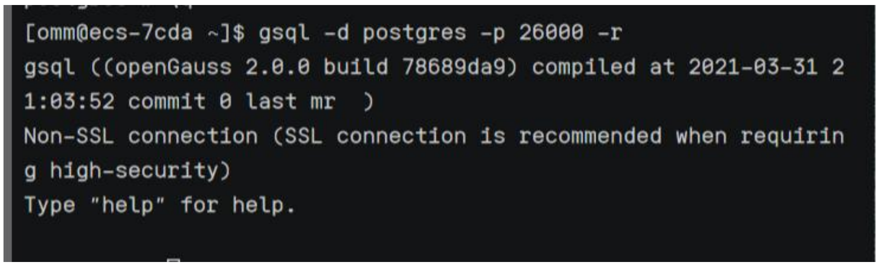
```
[omm@ecs-7cda ~]$  gs_om -t status
-----------------------------------------------------------------------------
--------

cluster_name    : dbCluster
cluster_state   : Normal
redistributing  : No

-----------------------------------------------------------------------------
--------
```

3.启动数据库服务



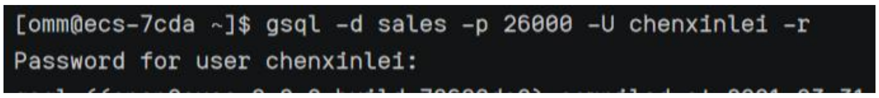
```
[omm@ecs-7cda ~]$ gs_om -t start
Starting cluster.
=========================================
[SUCCESS] ecs-7cda:
[2022-10-18 12:26:33.753][14929][][gs_ctl]: gs_ctl started,data
dir is /gaussdb/data/db1
[2022-10-18 12:26:33.756][14929][][gs_ctl]:  another server mig
ht be running; Please use the restart command
=========================================
Successfully started.
```

4.连接数据库



```
[omm@ecs-7cda ~]$ gsql -d postgres -p 26000 -r
gsql ((openGauss 2.0.0 build 78689da9) compiled at 2021-03-31 2
1:03:52 commit 0 last mr  )
Non-SSL connection (SSL connection is recommended when requirin
g high-security)
Type "help" for help.
```

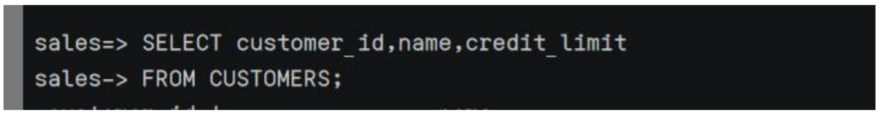
5.使用自己创建的用户连接到此数据库（此用户下已包含创建好的数据库，表，对应的约束等，详细操作如实验二。）



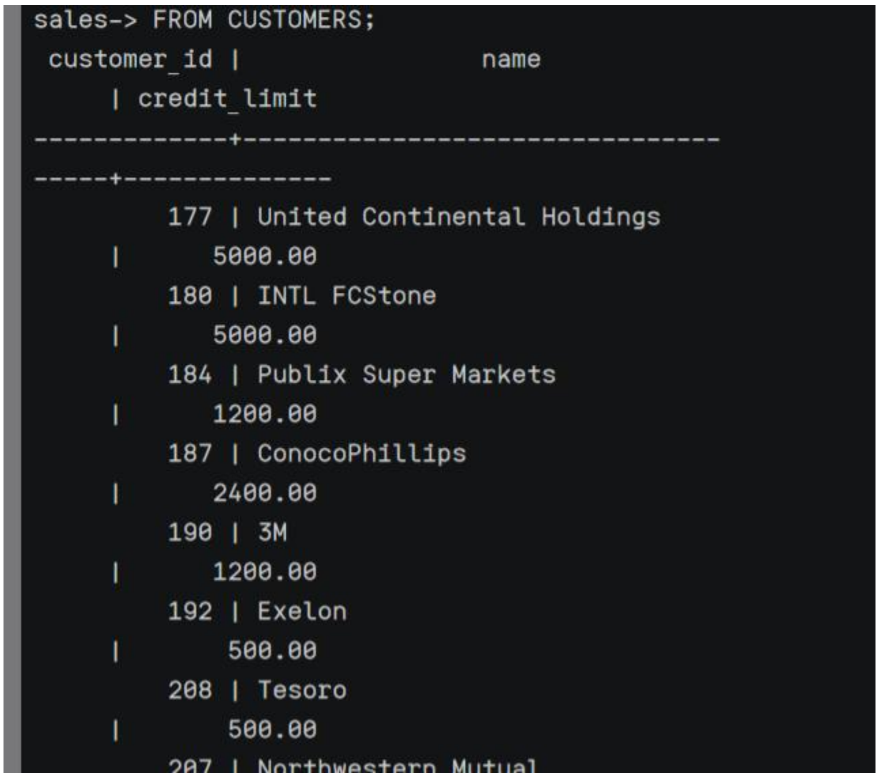
```
[omm@ecs-7cda ~]$ gsql -d sales -p 26000 -U chenxinlei -r
Password for user chenxinlei:
```

（2）查询

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）

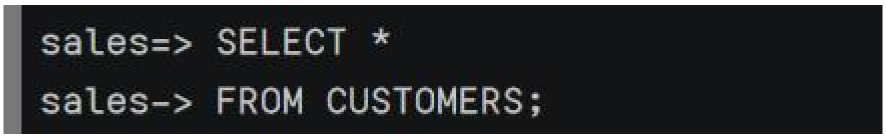


```
sales=> SELECT customer_id,name,credit_limit
sales-> FROM CUSTOMERS;
```

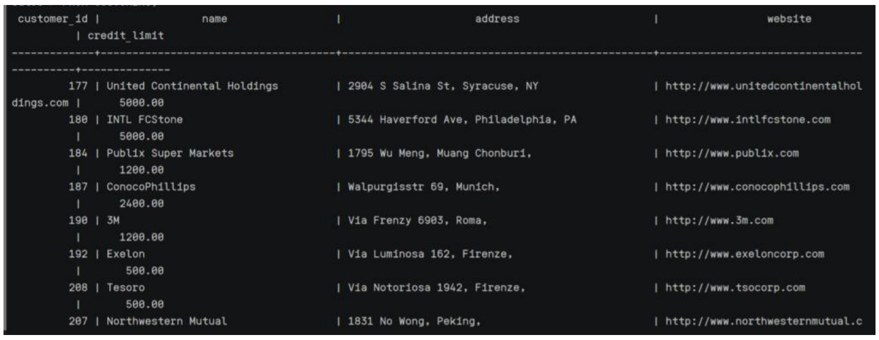


```
sales-> FROM CUSTOMERS;
 customer_id |                name
     | credit_limit
-------------+--------------------------------
-----+--------------
         177 | United Continental Holdings
     |      5000.00
         180 | INTL FCStone
     |      5000.00
         184 | Publix Super Markets
     |      1200.00
         187 | ConocoPhillips
     |      2400.00
         190 | 3M
     |      1200.00
         192 | Exelon
     |       500.00
         208 | Tesoro
     |       500.00
         207 | Northwestern Mutual
```

2.查询顾客的所有信息

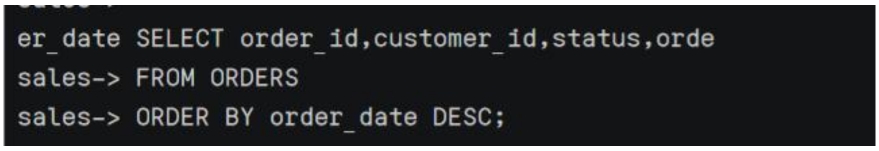


```
sales=> SELECT *
sales-> FROM CUSTOMERS;
```

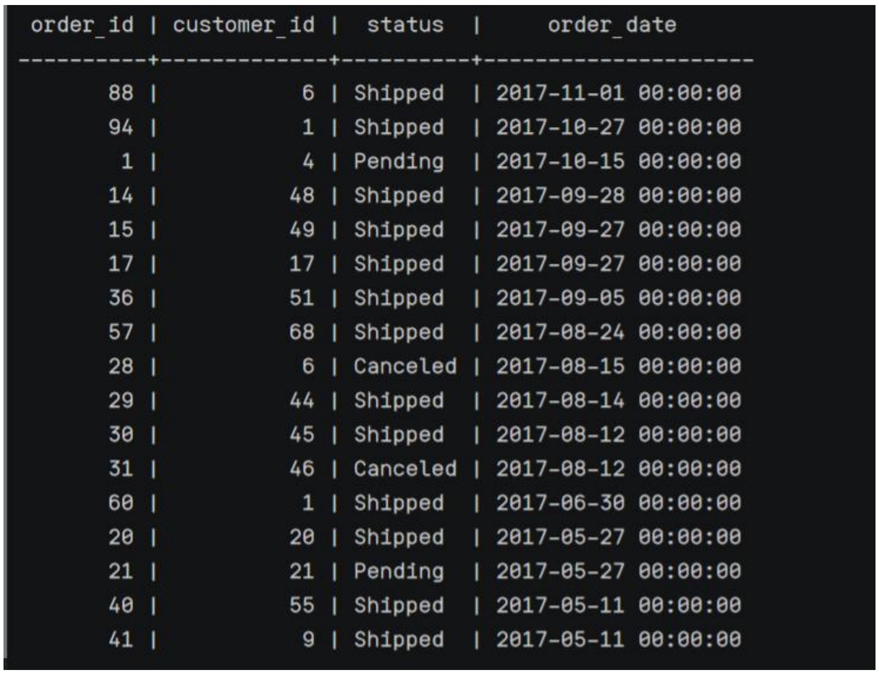


```
 customer_id |                name                 |                     address                      |              website
         | credit_limit
-------------+-------------------------------------+--------------------------------------------------+--------------------------------
----------+--------------
         177 | United Continental Holdings         | 2904 S Salina St, Syracuse, NY                   | http://www.unitedcontinentalhol
dings.com |      5000.00
         180 | INTL FCStone                        | 5344 Haverford Ave, Philadelphia, PA             | http://www.intlfcstone.com
         |      5000.00
         184 | Publix Super Markets                | 1795 Wu Meng, Muang Chonburi,                    | http://www.publix.com
         |      1200.00
         187 | ConocoPhillips                      | Walpurgisstr 69, Munich,                         | http://www.conocophillips.com
         |      2400.00
         190 | 3M                                  | Via Frenzy 6903, Roma,                           | http://www.3m.com
         |      1200.00
         192 | Exelon                              | Via Luminosa 162, Firenze,                       | http://www.exeloncorp.com
         |       500.00
         208 | Tesoro                              | Via Notoriosa 1942, Firenze,                     | http://www.tsocorp.com
         |       500.00
         207 | Northwestern Mutual                 | 1831 No Wong, Peking,                            | http://www.northwesternmutual.c
```

3.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

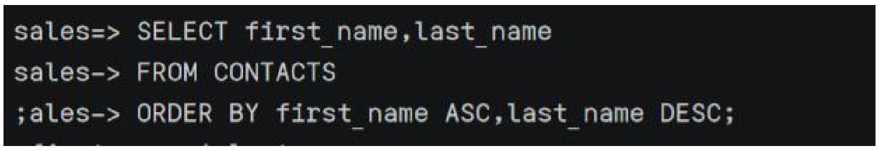


```
er_date SELECT order_id,customer_id,status,orde
sales-> FROM ORDERS
sales-> ORDER BY order_date DESC;
```

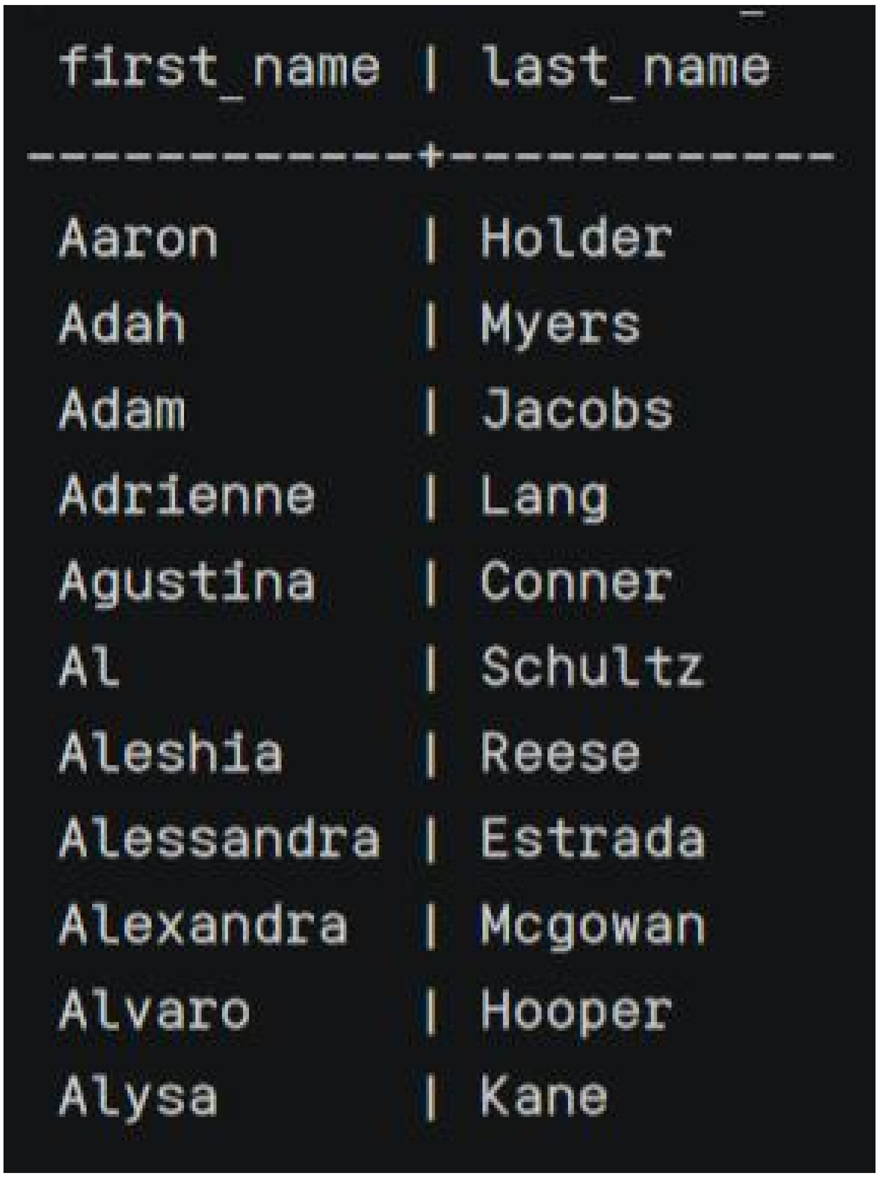


```
 order_id | customer_id |  status  |     order_date
----------+-------------+----------+---------------------
       88 |           6 | Shipped  | 2017-11-01 00:00:00
       94 |           1 | Shipped  | 2017-10-27 00:00:00
        1 |           4 | Pending  | 2017-10-15 00:00:00
       14 |          48 | Shipped  | 2017-09-28 00:00:00
       15 |          49 | Shipped  | 2017-09-27 00:00:00
       17 |          17 | Shipped  | 2017-09-27 00:00:00
       36 |          51 | Shipped  | 2017-09-05 00:00:00
       57 |          68 | Shipped  | 2017-08-24 00:00:00
       28 |           6 | Canceled | 2017-08-15 00:00:00
       29 |          44 | Shipped  | 2017-08-14 00:00:00
       30 |          45 | Shipped  | 2017-08-12 00:00:00
       31 |          46 | Canceled | 2017-08-12 00:00:00
       60 |           1 | Shipped  | 2017-06-30 00:00:00
       20 |          20 | Shipped  | 2017-05-27 00:00:00
       21 |          21 | Pending  | 2017-05-27 00:00:00
       40 |          55 | Shipped  | 2017-05-11 00:00:00
       41 |           9 | Shipped  | 2017-05-11 00:00:00
```

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



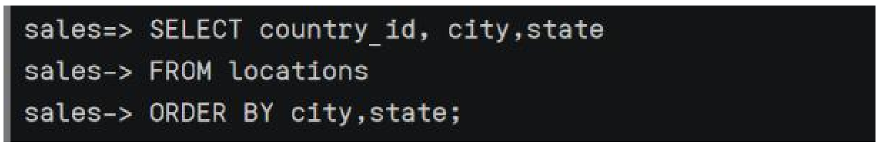
```
sales=> SELECT first_name,last_name
sales-> FROM CONTACTS
;ales-> ORDER BY first_name ASC,last_name DESC;
```



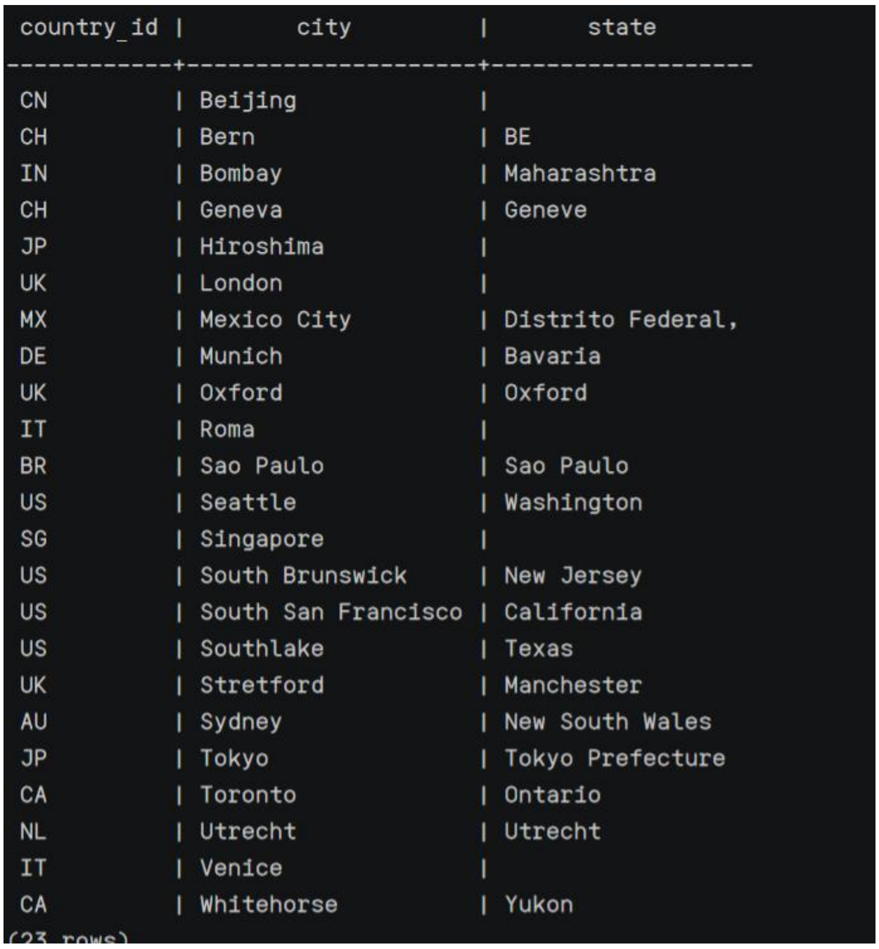
```
 first_name | last_name
------------+------------
 Aaron      | Holder
 Adah       | Myers
 Adam       | Jacobs
 Adrienne   | Lang
 Agustina   | Conner
 Al         | Schultz
 Aleshia    | Reese
 Alessandra | Estrada
 Alexandra  | Mcgowan
 Alvaro     | Hooper
 Alysa      | Kane
```

5. 执行以下语句并观察 state 列 NULL 值的显示位置，得出结论

（1）NULL 值穿插在表中

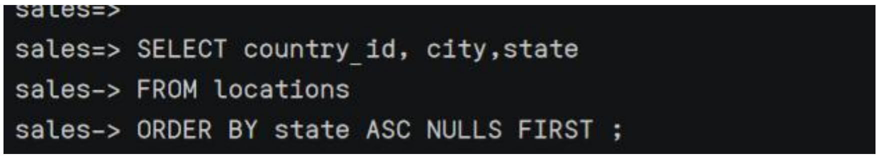


```
sales=> SELECT country_id, city,state
sales-> FROM locations
sales-> ORDER BY city,state;
```

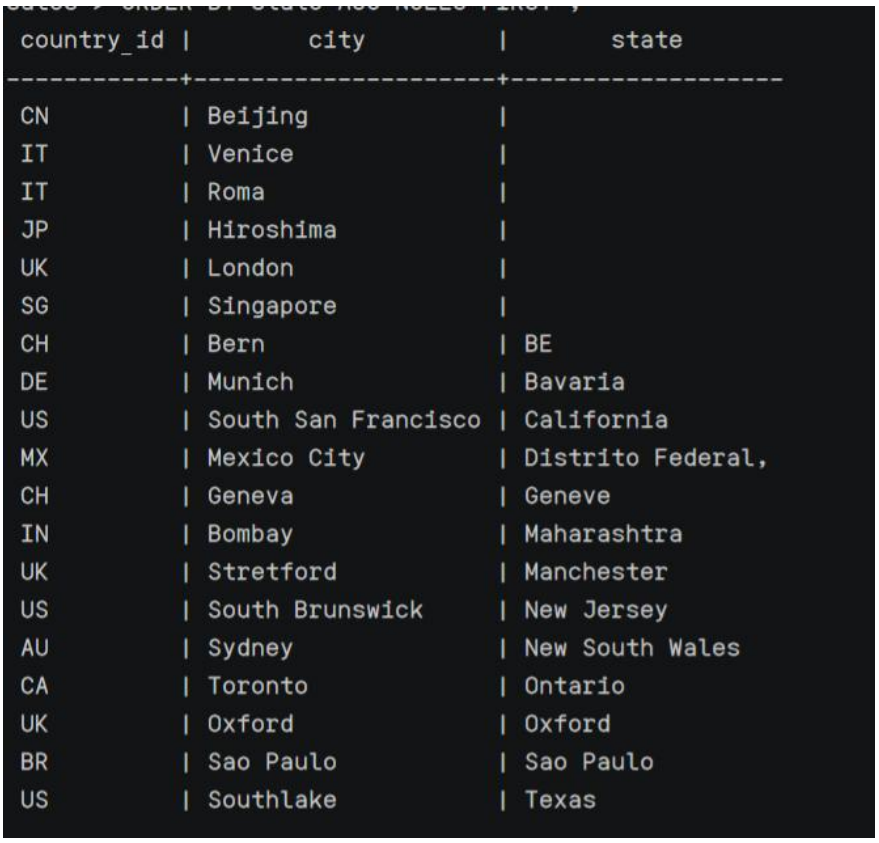


```
 country_id |        city         |       state
------------+---------------------+--------------------
 CN         | Beijing             |
 CH         | Bern                | BE
 IN         | Bombay              | Maharashtra
 CH         | Geneva              | Geneve
 JP         | Hiroshima           |
 UK         | London              |
 MX         | Mexico City         | Distrito Federal,
 DE         | Munich              | Bavaria
 UK         | Oxford              | Oxford
 IT         | Roma                |
 BR         | Sao Paulo           | Sao Paulo
 US         | Seattle             | Washington
 SG         | Singapore           |
 US         | South Brunswick     | New Jersey
 US         | South San Francisco | California
 US         | Southlake           | Texas
 UK         | Stretford           | Manchester
 AU         | Sydney              | New South Wales
 JP         | Tokyo               | Tokyo Prefecture
 CA         | Toronto             | Ontario
 NL         | Utrecht             | Utrecht
 IT         | Venice              |
 CA         | Whitehorse          | Yukon
(23 rows)
```

（2）NULL 值出现在开头

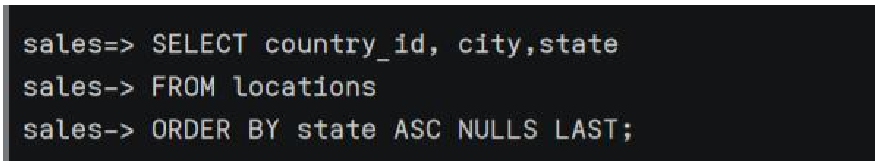


```
sales=>
sales=> SELECT country_id, city,state
sales-> FROM locations
sales-> ORDER BY state ASC NULLS FIRST ;
```

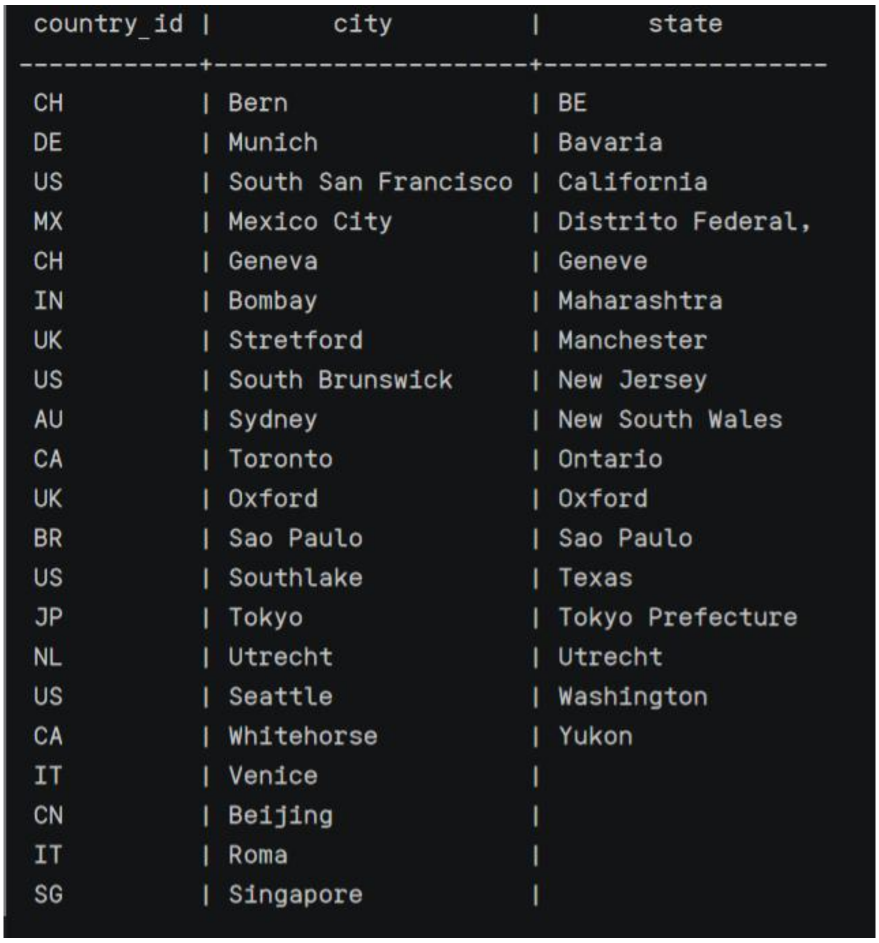


```
 country_id |         city         |       state
------------+---------------------+-------------------
 CN         | Beijing             |
 IT         | Venice              |
 IT         | Roma                |
 JP         | Hiroshima           |
 UK         | London              |
 SG         | Singapore           |
 CH         | Bern                | BE
 DE         | Munich              | Bavaria
 US         | South San Francisco | California
 MX         | Mexico City         | Distrito Federal,
 CH         | Geneva              | Geneve
 IN         | Bombay              | Maharashtra
 UK         | Stretford           | Manchester
 US         | South Brunswick     | New Jersey
 AU         | Sydney              | New South Wales
 CA         | Toronto             | Ontario
 UK         | Oxford              | Oxford
 BR         | Sao Paulo           | Sao Paulo
 US         | Southlake           | Texas
```

(3)NULL 出现在末尾

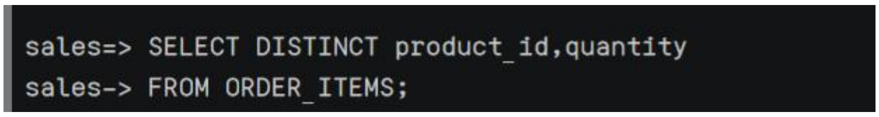


```
sales=> SELECT country_id, city,state
sales-> FROM locations
sales-> ORDER BY state ASC NULLS LAST;
```

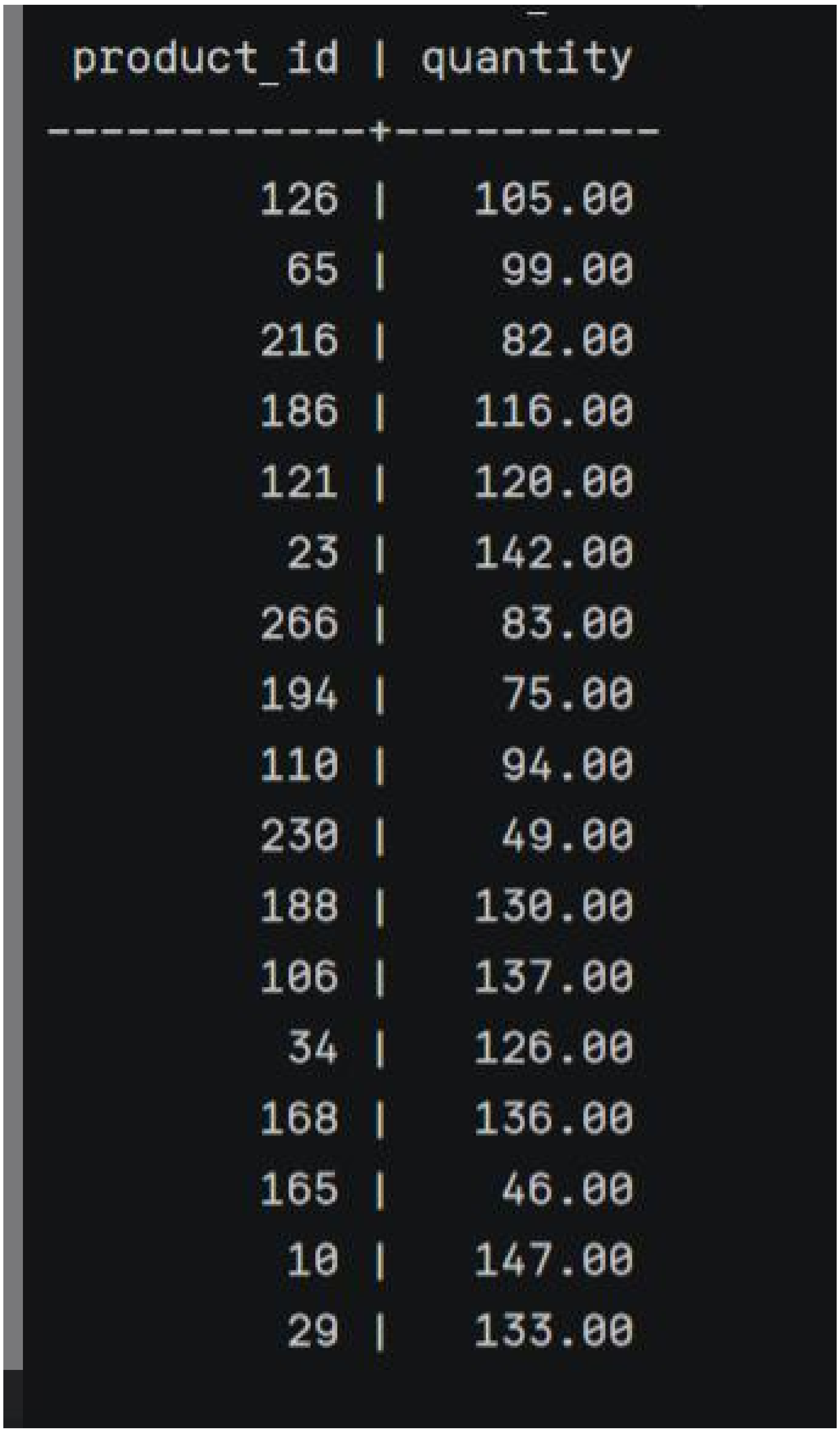


```
 country_id |         city         |        state
------------+----------------------+--------------------
 CH         | Bern                 | BE
 DE         | Munich               | Bavaria
 US         | South San Francisco  | California
 MX         | Mexico City          | Distrito Federal,
 CH         | Geneva               | Geneve
 IN         | Bombay               | Maharashtra
 UK         | Stretford            | Manchester
 US         | South Brunswick      | New Jersey
 AU         | Sydney               | New South Wales
 CA         | Toronto              | Ontario
 UK         | Oxford               | Oxford
 BR         | Sao Paulo            | Sao Paulo
 US         | Southlake            | Texas
 JP         | Tokyo                | Tokyo Prefecture
 NL         | Utrecht              | Utrecht
 US         | Seattle              | Washington
 CA         | Whitehorse           | Yukon
 IT         | Venice               |
 CN         | Beijing              |
 IT         | Roma                 |
 SG         | Singapore            |
```

6.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

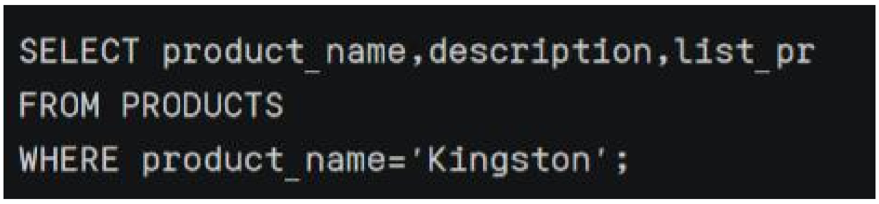


```
sales=> SELECT DISTINCT product_id,quantity
sales-> FROM ORDER_ITEMS;
```

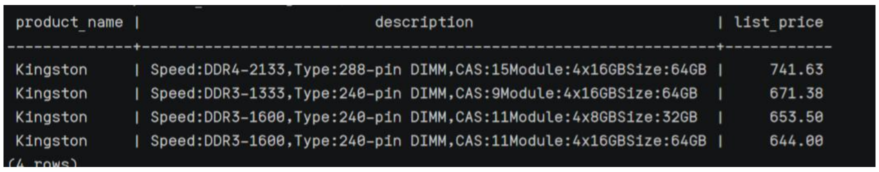


```
 product_id | quantity
------------+----------
        126 |   105.00
         65 |    99.00
        216 |    82.00
        186 |   116.00
        121 |   120.00
         23 |   142.00
        266 |    83.00
        194 |    75.00
        110 |    94.00
        230 |    49.00
        188 |   130.00
        106 |   137.00
         34 |   126.00
        168 |   136.00
        165 |    46.00
         10 |   147.00
         29 |   133.00
```

7.查询产品表中的产品名为‘Kingston’的产品名，产品描述和价格

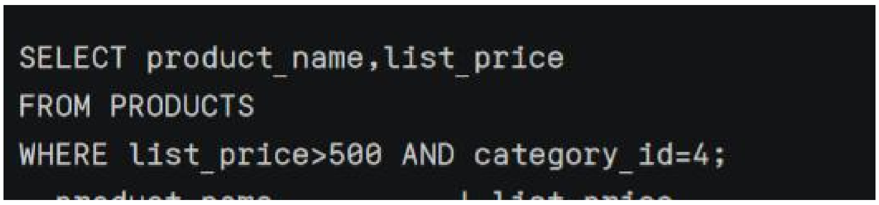


```
SELECT product_name,description,list_pr
FROM PRODUCTS
WHERE product_name='Kingston';
```

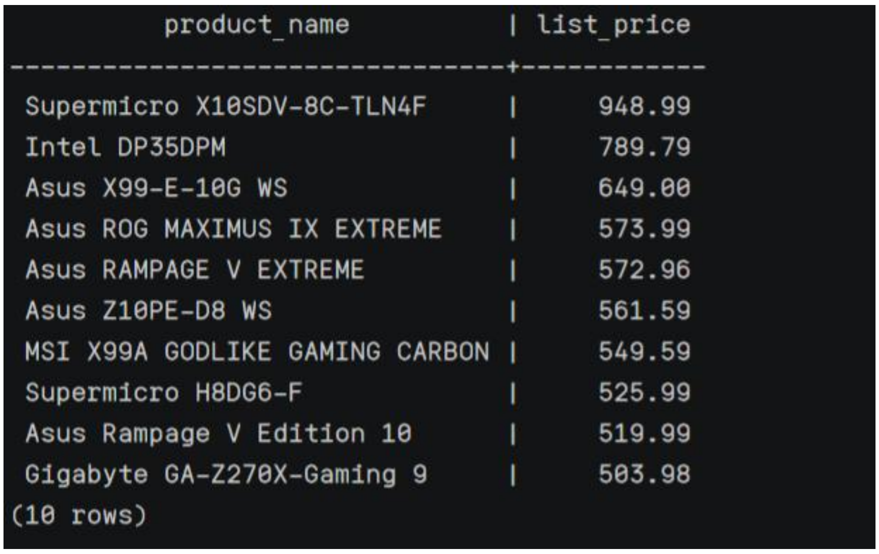


```
product_name |                    description                              | list_price
-------------+-------------------------------------------------------------+-----------
Kingston     | Speed:DDR4-2133,Type:288-pin DIMM,CAS:15Module:4x16GBSize:64GB |     741.63
Kingston     | Speed:DDR3-1333,Type:240-pin DIMM,CAS:9Module:4x16GBSize:64GB  |     671.38
Kingston     | Speed:DDR3-1600,Type:240-pin DIMM,CAS:11Module:4x8GBSize:32GB  |     653.50
Kingston     | Speed:DDR3-1600,Type:240-pin DIMM,CAS:11Module:4x16GBSize:64GB |     644.00
(4 rows)
```

8.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

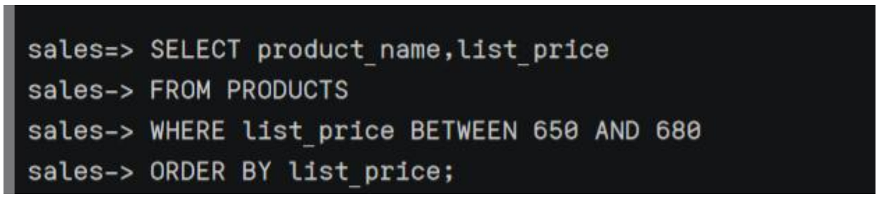


```
SELECT product_name,list_price
FROM PRODUCTS
WHERE list_price>500 AND category_id=4;
```

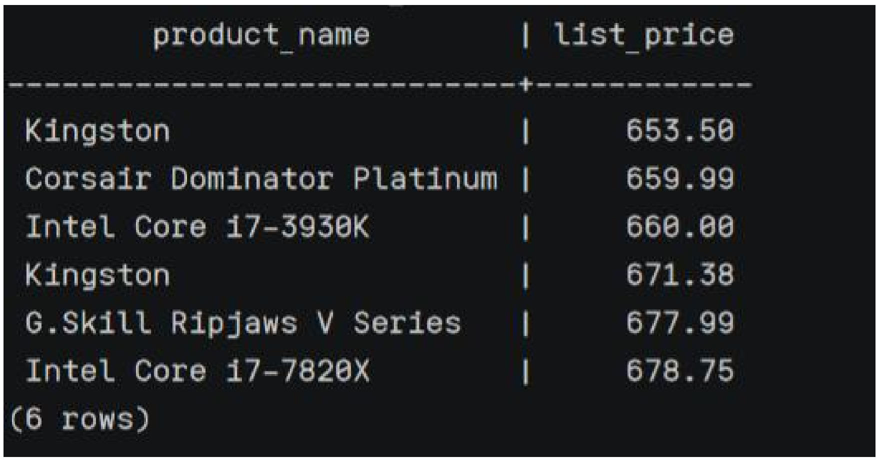


```
          product_name          | list_price
--------------------------------+------------
 Supermicro X10SDV-8C-TLN4F     |     948.99
 Intel DP35DPM                  |     789.79
 Asus X99-E-10G WS              |     649.00
 Asus ROG MAXIMUS IX EXTREME    |     573.99
 Asus RAMPAGE V EXTREME         |     572.96
 Asus Z10PE-D8 WS               |     561.59
 MSI X99A GODLIKE GAMING CARBON |     549.59
 Supermicro H8DG6-F             |     525.99
 Asus Rampage V Edition 10      |     519.99
 Gigabyte GA-Z270X-Gaming 9     |     503.98
(10 rows)
```

9.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



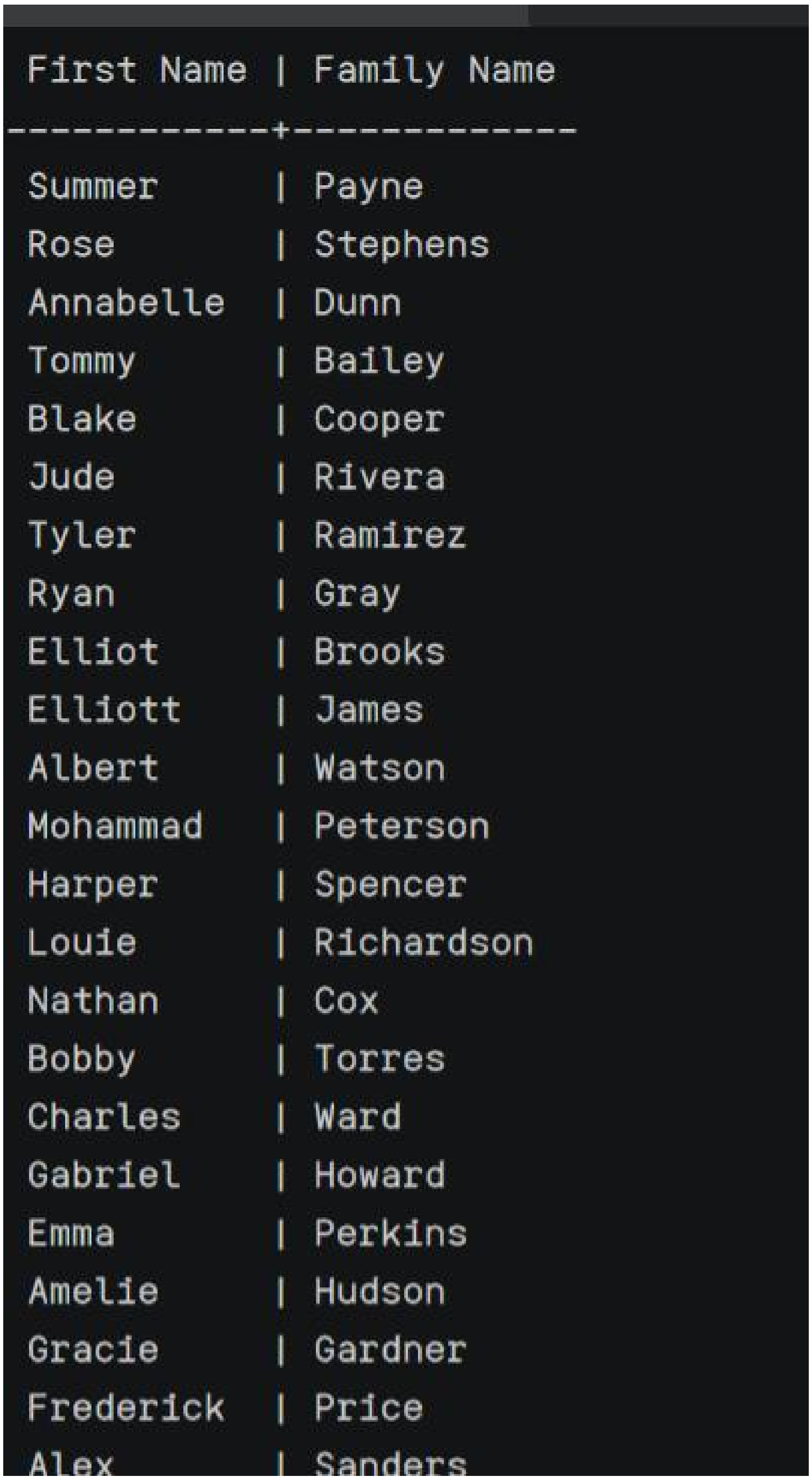
```
sales=> SELECT product_name,list_price
sales-> FROM PRODUCTS
sales-> WHERE list_price BETWEEN 650 AND 680
sales-> ORDER BY list_price;
```



```
        product_name         | list_price
-----------------------------+------------
 Kingston                    |     653.50
 Corsair Dominator Platinum  |     659.99
 Intel Core i7-3930K         |     660.00
 Kingston                    |     671.38
 G.Skill Ripjaws V Series    |     677.99
 Intel Core i7-7820X         |     678.75
(6 rows)
```

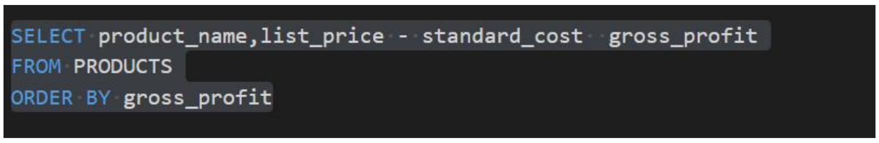
10.查询雇员表中的名和姓，名和姓的字段分别显示为"First Name"和"Family Name"

```
SELECT first_name AS "First Name",last_name AS "Family Name"
FROM EMPLOYEES;
```

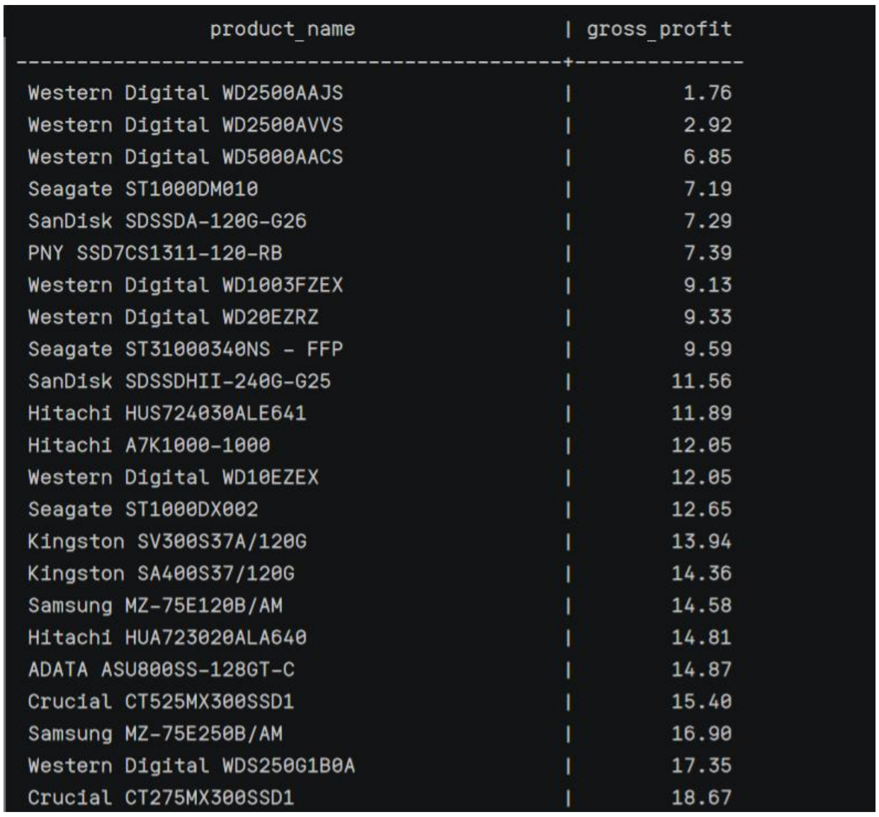


```
 First Name | Family Name
------------+-------------
 Summer     | Payne
 Rose       | Stephens
 Annabelle  | Dunn
 Tommy      | Bailey
 Blake      | Cooper
 Jude       | Rivera
 Tyler      | Ramirez
 Ryan       | Gray
 Elliot     | Brooks
 Elliott    | James
 Albert     | Watson
 Mohammad   | Peterson
 Harper     | Spencer
 Louie      | Richardson
 Nathan     | Cox
 Bobby      | Torres
 Charles    | Ward
 Gabriel    | Howard
 Emma       | Perkins
 Amelie     | Hudson
 Gracie     | Gardner
 Frederick  | Price
 Alex       | Sanders
```

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

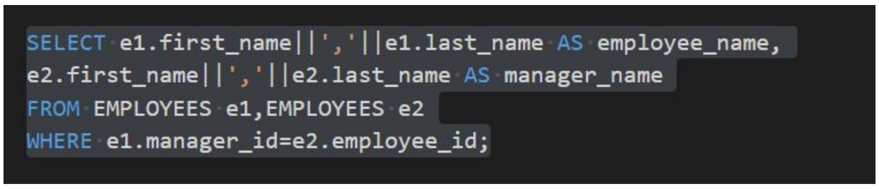


```
SELECT product_name,list_price - standard_cost  gross_profit
FROM PRODUCTS
ORDER BY gross_profit
```

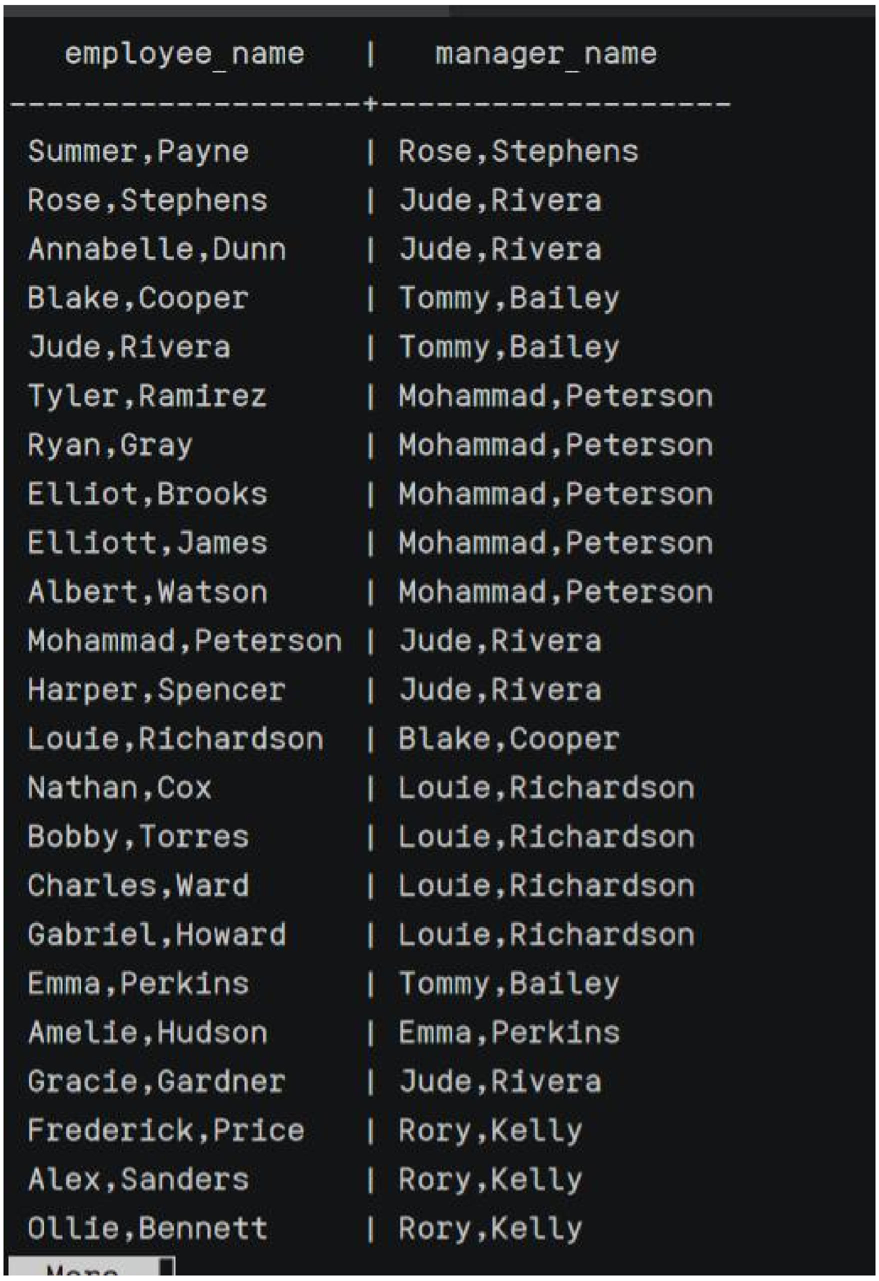


```
                product_name                  | gross_profit
----------------------------------------------+-------------
 Western Digital WD2500AAJS                   |         1.76
 Western Digital WD2500AVVS                   |         2.92
 Western Digital WD5000AACS                   |         6.85
 Seagate ST1000DM010                          |         7.19
 SanDisk SDSSDA-120G-G26                      |         7.29
 PNY SSD7CS1311-120-RB                        |         7.39
 Western Digital WD1003FZEX                   |         9.13
 Western Digital WD20EZRZ                     |         9.33
 Seagate ST31000340NS - FFP                   |         9.59
 SanDisk SDSSDHII-240G-G25                    |        11.56
 Hitachi HUS724030ALE641                      |        11.89
 Hitachi A7K1000-1000                         |        12.05
 Western Digital WD10EZEX                     |        12.05
 Seagate ST1000DX002                          |        12.65
 Kingston SV300S37A/120G                      |        13.94
 Kingston SA400S37/120G                       |        14.36
 Samsung MZ-75E120B/AM                        |        14.58
 Hitachi HUA723020ALA640                      |        14.81
 ADATA ASU800SS-128GT-C                       |        14.87
 Crucial CT525MX300SSD1                       |        15.40
 Samsung MZ-75E250B/AM                        |        16.90
 Western Digital WDS250G1B0A                  |        17.35
 Crucial CT275MX300SSD1                       |        18.67
```

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’）

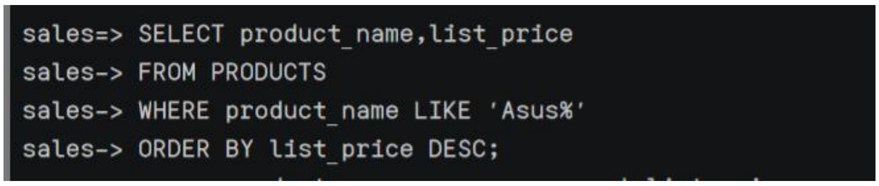


```
SELECT e1.first_name||','||e1.last_name AS employee_name,
e2.first_name||','||e2.last_name AS manager_name
FROM EMPLOYEES e1,EMPLOYEES e2
WHERE e1.manager_id=e2.employee_id;
```

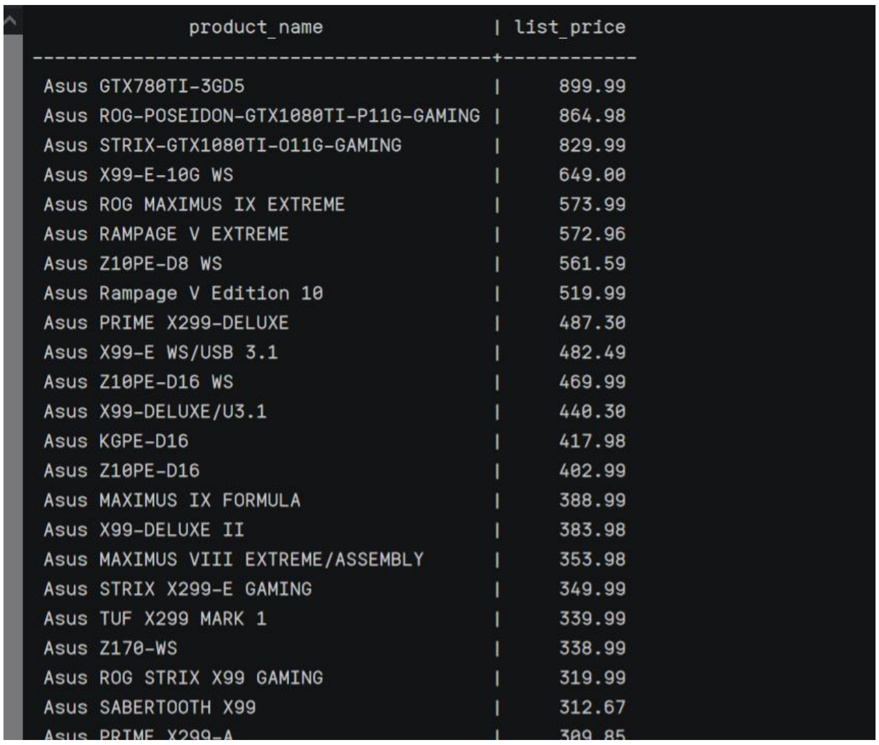


```
  employee_name    |   manager_name
-------------------+-------------------
 Summer,Payne      | Rose,Stephens
 Rose,Stephens     | Jude,Rivera
 Annabelle,Dunn    | Jude,Rivera
 Blake,Cooper      | Tommy,Bailey
 Jude,Rivera       | Tommy,Bailey
 Tyler,Ramirez     | Mohammad,Peterson
 Ryan,Gray         | Mohammad,Peterson
 Elliot,Brooks     | Mohammad,Peterson
 Elliott,James     | Mohammad,Peterson
 Albert,Watson     | Mohammad,Peterson
 Mohammad,Peterson | Jude,Rivera
 Harper,Spencer    | Jude,Rivera
 Louie,Richardson  | Blake,Cooper
 Nathan,Cox        | Louie,Richardson
 Bobby,Torres      | Louie,Richardson
 Charles,Ward      | Louie,Richardson
 Gabriel,Howard    | Louie,Richardson
 Emma,Perkins      | Tommy,Bailey
 Amelie,Hudson     | Emma,Perkins
 Gracie,Gardner    | Jude,Rivera
 Frederick,Price   | Rory,Kelly
 Alex,Sanders      | Rory,Kelly
 Ollie,Bennett     | Rory,Kelly
 More
```

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

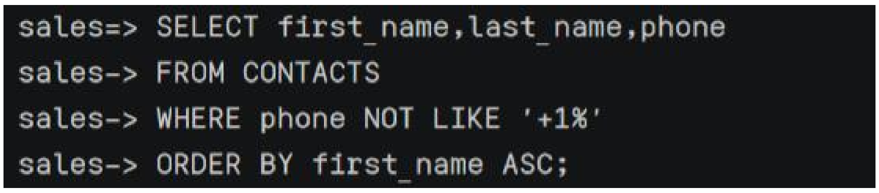


```
sales=> SELECT product_name,list_price
sales-> FROM PRODUCTS
sales-> WHERE product_name LIKE 'Asus%'
sales-> ORDER BY list_price DESC;
```

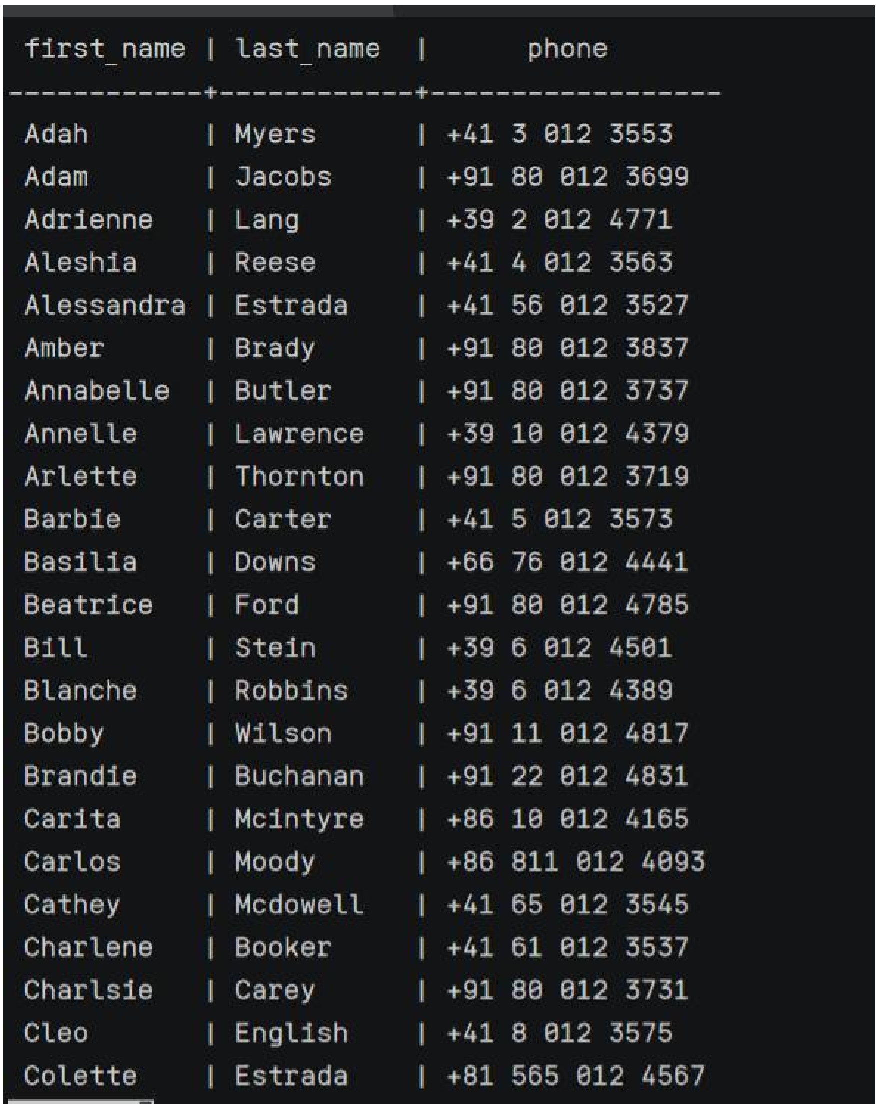


```
              product_name                | list_price
------------------------------------------+------------
 Asus GTX780TI-3GD5                       |     899.99
 Asus ROG-POSEIDON-GTX1080TI-P11G-GAMING |     864.98
 Asus STRIX-GTX1080TI-O11G-GAMING         |     829.99
 Asus X99-E-10G WS                        |     649.00
 Asus ROG MAXIMUS IX EXTREME              |     573.99
 Asus RAMPAGE V EXTREME                   |     572.96
 Asus Z10PE-D8 WS                         |     561.59
 Asus Rampage V Edition 10                |     519.99
 Asus PRIME X299-DELUXE                   |     487.30
 Asus X99-E WS/USB 3.1                    |     482.49
 Asus Z10PE-D16 WS                        |     469.99
 Asus X99-DELUXE/U3.1                     |     440.30
 Asus KGPE-D16                            |     417.98
 Asus Z10PE-D16                           |     402.99
 Asus MAXIMUS IX FORMULA                  |     388.99
 Asus X99-DELUXE II                       |     383.98
 Asus MAXIMUS VIII EXTREME/ASSEMBLY       |     353.98
 Asus STRIX X299-E GAMING                 |     349.99
 Asus TUF X299 MARK 1                     |     339.99
 Asus Z170-WS                             |     338.99
 Asus ROG STRIX X99 GAMING                |     319.99
 Asus SABERTOOTH X99                      |     312.67
 Asus PRIME X299-A                        |     309.85
```

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

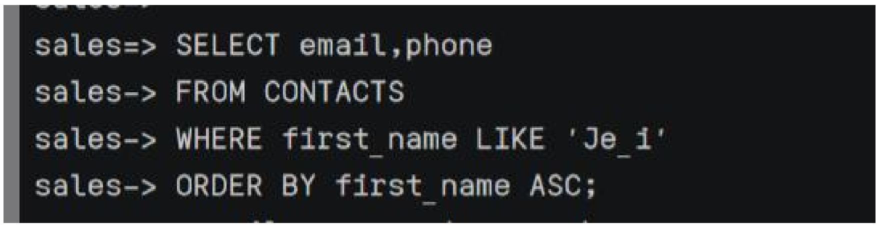


```
sales=> SELECT first_name,last_name,phone
sales-> FROM CONTACTS
sales-> WHERE phone NOT LIKE '+1%'
sales-> ORDER BY first_name ASC;
```

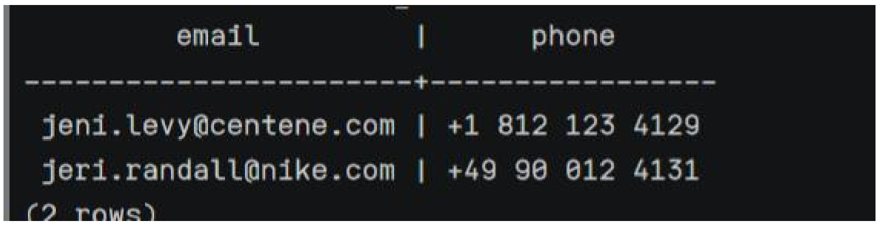


```
 first_name | last_name  |      phone
------------+------------+-----------------
 Adah       | Myers      | +41 3 012 3553
 Adam       | Jacobs     | +91 80 012 3699
 Adrienne   | Lang       | +39 2 012 4771
 Aleshia    | Reese      | +41 4 012 3563
 Alessandra | Estrada    | +41 56 012 3527
 Amber      | Brady      | +91 80 012 3837
 Annabelle  | Butler     | +91 80 012 3737
 Annelle    | Lawrence   | +39 10 012 4379
 Arlette    | Thornton   | +91 80 012 3719
 Barbie     | Carter     | +41 5 012 3573
 Basilia    | Downs      | +66 76 012 4441
 Beatrice   | Ford       | +91 80 012 4785
 Bill       | Stein      | +39 6 012 4501
 Blanche    | Robbins    | +39 6 012 4389
 Bobby      | Wilson     | +91 11 012 4817
 Brandie    | Buchanan   | +91 22 012 4831
 Carita     | Mcintyre   | +86 10 012 4165
 Carlos     | Moody      | +86 811 012 4093
 Cathey     | Mcdowell   | +41 65 012 3545
 Charlene   | Booker     | +41 61 012 3537
 Charlsie   | Carey      | +91 80 012 3731
 Cleo       | English    | +41 8 012 3575
 Colette    | Estrada    | +81 565 012 4567
```

15.查询联系表中的电话号码和电子邮件，要求名(first_name) 的长为4 且以'Je'开头，以'i'结尾， 按名升序显示

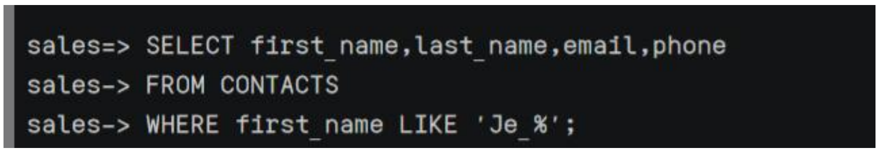


```
sales=> SELECT email,phone
sales-> FROM CONTACTS
sales-> WHERE first_name LIKE 'Je_i'
sales-> ORDER BY first_name ASC;
```

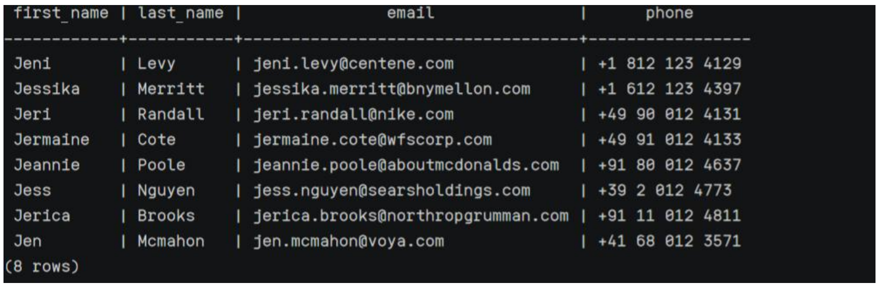


```
         email          |      phone
------------------------+-----------------
 jeni.levy@centene.com  | +1 812 123 4129
 jeri.randall@nike.com  | +49 90 012 4131
(2 rows)
```

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

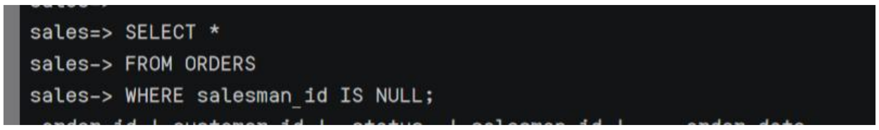


```
sales=> SELECT first_name,last_name,email,phone
sales-> FROM CONTACTS
sales-> WHERE first_name LIKE 'Je_%';
```

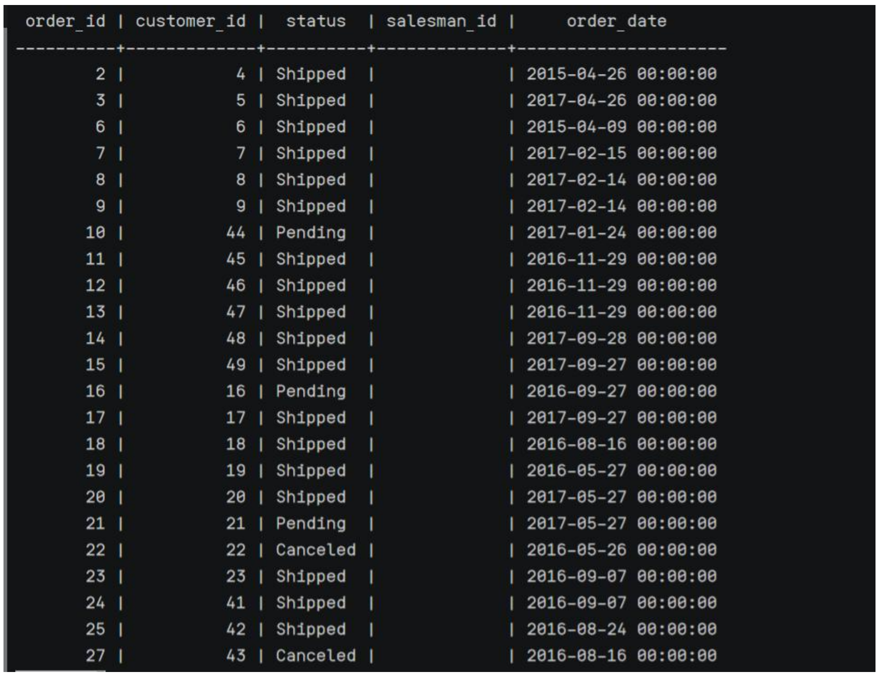


```
 first_name | last_name |               email               |      phone
------------+-----------+-----------------------------------+-----------------
 Jeni       | Levy      | jeni.levy@centene.com             | +1 812 123 4129
 Jessika    | Merritt   | jessika.merritt@bnymellon.com     | +1 612 123 4397
 Jeri       | Randall   | jeri.randall@nike.com             | +49 90 012 4131
 Jermaine   | Cote      | jermaine.cote@wfscorp.com         | +49 91 012 4133
 Jeannie    | Poole     | jeannie.poole@aboutmcdonalds.com  | +91 80 012 4637
 Jess       | Nguyen    | jess.nguyen@searsholdings.com     | +39 2 012 4773
 Jerica     | Brooks    | jerica.brooks@northropgrumman.com | +91 11 012 4811
 Jen        | Mcmahon   | jen.mcmahon@voya.com              | +41 68 012 3571
(8 rows)
```

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 not have a responsible salesman）

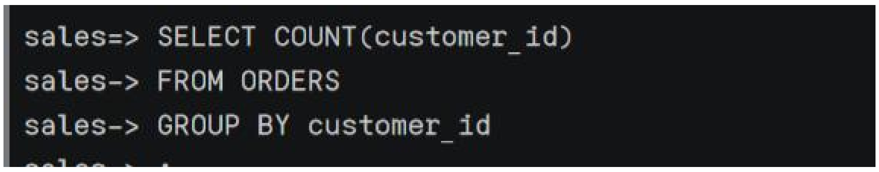


```
sales=> SELECT *
sales-> FROM ORDERS
sales-> WHERE salesman_id IS NULL;
```

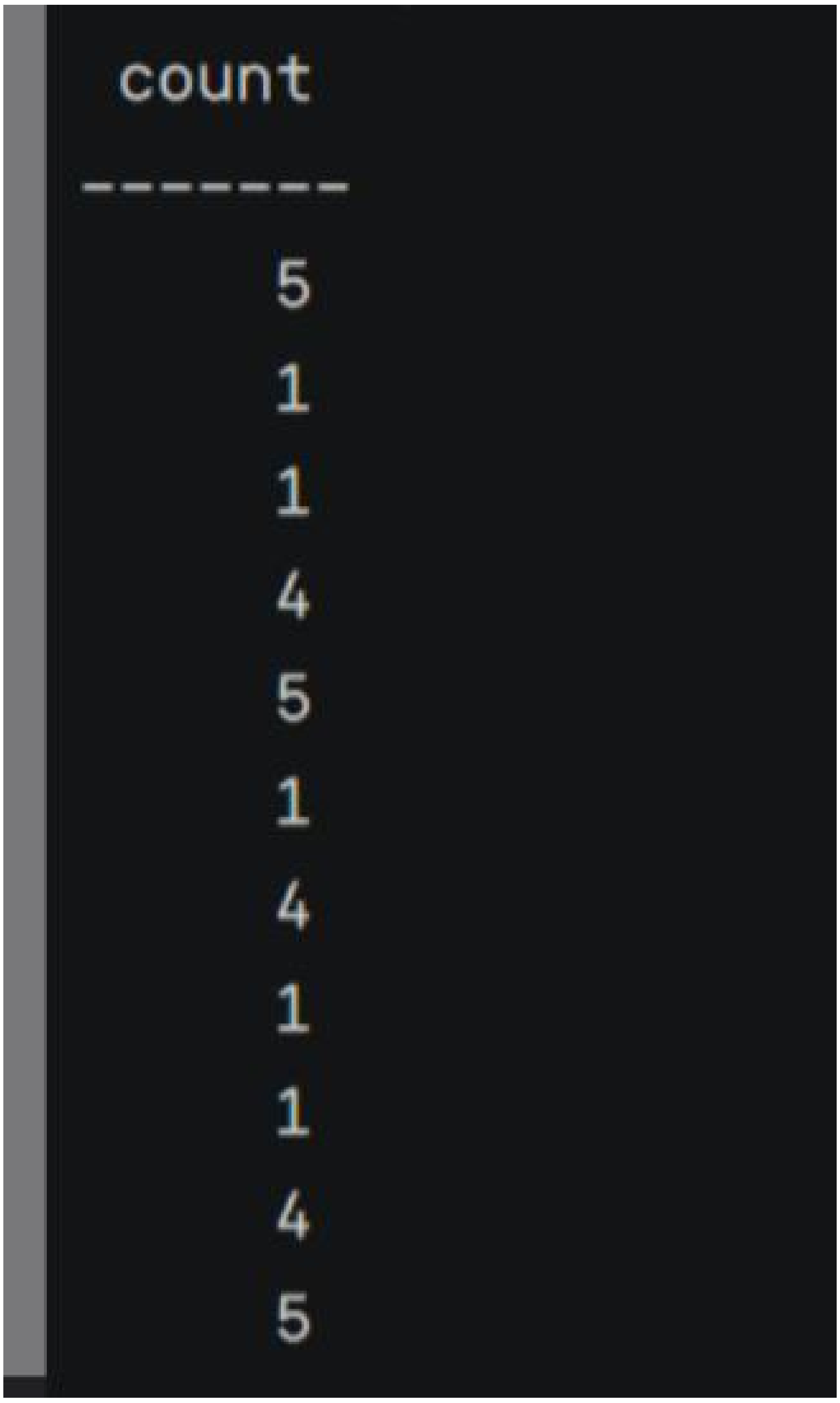


```
 order_id | customer_id |  status  | salesman_id |     order_date
----------+-------------+----------+-------------+---------------------
        2 |           4 | Shipped  |             | 2015-04-26 00:00:00
        3 |           5 | Shipped  |             | 2017-04-26 00:00:00
        6 |           6 | Shipped  |             | 2015-04-09 00:00:00
        7 |           7 | Shipped  |             | 2017-02-15 00:00:00
        8 |           8 | Shipped  |             | 2017-02-14 00:00:00
        9 |           9 | Shipped  |             | 2017-02-14 00:00:00
       10 |          44 | Pending  |             | 2017-01-24 00:00:00
       11 |          45 | Shipped  |             | 2016-11-29 00:00:00
       12 |          46 | Shipped  |             | 2016-11-29 00:00:00
       13 |          47 | Shipped  |             | 2016-11-29 00:00:00
       14 |          48 | Shipped  |             | 2017-09-28 00:00:00
       15 |          49 | Shipped  |             | 2017-09-27 00:00:00
       16 |          16 | Pending  |             | 2016-09-27 00:00:00
       17 |          17 | Shipped  |             | 2017-09-27 00:00:00
       18 |          18 | Shipped  |             | 2016-08-16 00:00:00
       19 |          19 | Shipped  |             | 2016-05-27 00:00:00
       20 |          20 | Shipped  |             | 2017-05-27 00:00:00
       21 |          21 | Pending  |             | 2017-05-27 00:00:00
       22 |          22 | Canceled |             | 2016-05-26 00:00:00
       23 |          23 | Shipped  |             | 2016-09-07 00:00:00
       24 |          41 | Shipped  |             | 2016-09-07 00:00:00
       25 |          42 | Shipped  |             | 2016-08-24 00:00:00
       27 |          43 | Canceled |             | 2016-08-16 00:00:00
```

18.统计每个顾客的订单总数（查询订单表）



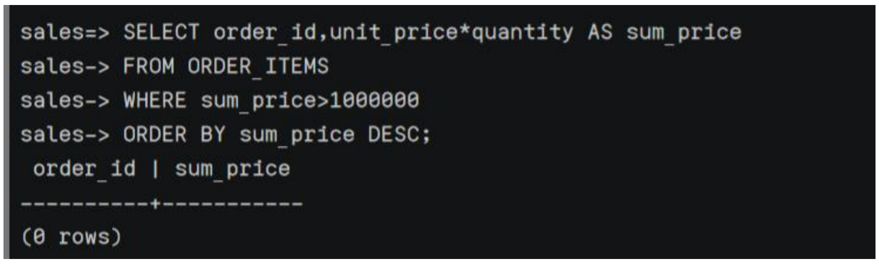
```
sales=> SELECT COUNT(customer_id)
sales-> FROM ORDERS
sales-> GROUP BY customer_id
```



```
 count
-------
     5
     1
     1
     4
     5
     1
     4
     1
     1
     4
     5
```

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）

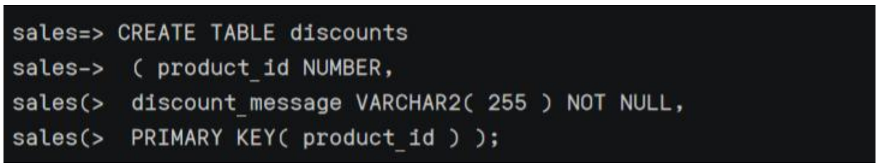


```
sales=> SELECT order_id,unit_price*quantity AS sum_price
sales-> FROM ORDER_ITEMS
sales-> WHERE sum_price>1000000
sales-> ORDER BY sum_price DESC;
 order_id | sum_price
----------+-----------
(0 rows)
```

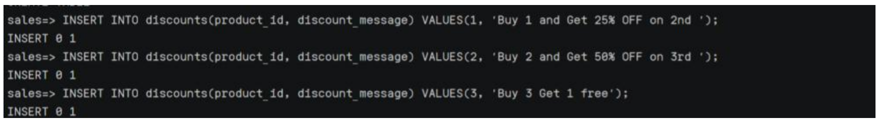
20.创建一个折扣表 discounts

插入 3 条数据：

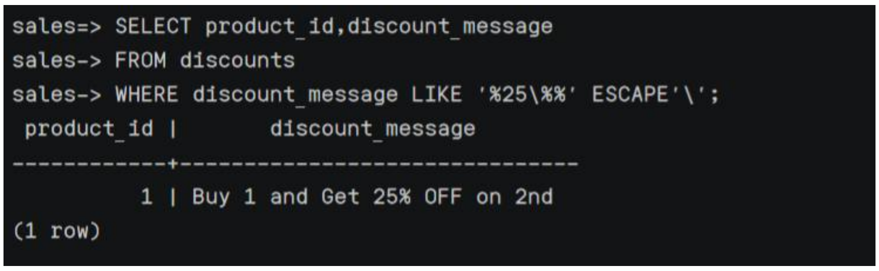
要求：查询折扣表中折扣信息出现“25%”的产品号和折扣信息。



```
sales=> CREATE TABLE discounts
sales->  ( product_id NUMBER,
sales(>  discount_message VARCHAR2( 255 ) NOT NULL,
sales(>  PRIMARY KEY( product_id ) );
```



```
sales=> INSERT INTO discounts(product_id, discount_message) VALUES(1, 'Buy 1 and Get 25% OFF on 2nd ');
INSERT 0 1
sales=> INSERT INTO discounts(product_id, discount_message) VALUES(2, 'Buy 2 and Get 50% OFF on 3rd ');
INSERT 0 1
sales=> INSERT INTO discounts(product_id, discount_message) VALUES(3, 'Buy 3 Get 1 free');
INSERT 0 1
```



```
sales=> SELECT product_id,discount_message
sales-> FROM discounts
sales-> WHERE discount_message LIKE '%25\%%' ESCAPE'\';
 product_id |       discount_message
------------+-------------------------------
          1 | Buy 1 and Get 25% OFF on 2nd
(1 row)
```

3. 实验总结

3.1 完成的工作

1.熟悉实验二中的操作，如数据库的创建，表的创建，约束，数据的插入等；

2.学习 sql 语句的基本语法和用法，对数据进行查询。

3.2 对实验的认识

熟练掌握 openGauss 单表查询的语法结构及其使用方法;

掌握设计正确查询语句以实现查询要求的方法如普通单表查询、模糊查询、聚集函数、分组统计和排序;

掌握 Group by 的使用 ;

掌握 Order by 的使用 ;

掌握使用 DISTINCT 实现查询结果的去重方法;

掌握空值 NULL 的使用方法 ;

掌握查询过程中别名的使用方法 ;

掌握自身连接的使用方法;

3.3 遇到的困难及解决方法

无