

《数据库概论》实验一：用 SQL 进行数据操作 实验报告

高辰潇

181220014

QQ: 1910342119

Email: DerekGaocx@outlook.com

实验环境

操作系统：Ubuntu 18.04

数据库管理系统：Mysql 8.0

软件：JetBrains DataGrip

实验过程

1 使用 SQL 语句建立基本表

```
1. CREATE DATABASE IF NOT EXISTS HW1;
2. USE HW1;
3.
4. CREATE TABLE Customers (
5.     cid CHAR(4) NOT NULL,
6.     cname CHAR(20) NOT NULL,
7.     city CHAR(20) NOT NULL,
8.     discnt REAL,
9.     PRIMARY KEY (cid)
10.);
11.
12. CREATE TABLE Agents (
13.     aid CHAR(3) NOT NULL,
14.     aname CHAR(20) NOT NULL,
15.     city CHAR(20) NOT NULL,
16.     perc SMALLINT,
17.     PRIMARY KEY (aid)
18.);
19.
20. CREATE TABLE Products (
21.     pid CHAR(3) NOT NULL,
22.     pname CHAR(20) NOT NULL,
```

```

23.    city CHAR(20),
24.    quantity INT NOT NULL,
25.    price REAL NOT NULL,
26.    PRIMARY KEY (pid)
27. );
28.
29. CREATE TABLE Orders (
30.    ordno INT NOT NULL,
31.    orddate DATE NOT NULL,
32.    cid CHAR(4) NOT NULL,
33.    aid CHAR(3) NOT NULL,
34.    pid CHAR(3) NOT NULL,
35.    qty INT,
36.    dols REAL,
37.    PRIMARY KEY (ordno)
38. );

```

2 使用 SQL 插入数据

2.1 向 Customers 表插入数据

```

39. INSERT INTO Customers (cid, cname, city, discnt)
40. VALUES
41. ('c001', 'Tiptop', 'Duluth', 10.00),
42. ('c002', 'Basics', 'Dallas', 12.00),
43. ('c003', 'Allied', 'Dallas', 8.00),
44. ('c004', 'ACME', 'Duluth', 8.00),
45. ('c006', 'ACME', 'Kyoto', 0.00);

```

2.2 向 Agents 表插入数据

```

46. INSERT INTO Agents (aid, aname, city, perc)
47. VALUES
48. ('a01', 'Smith', 'New York', 6),
49. ('a02', 'Jones', 'Newark', 6),
50. ('a03', 'Brown', 'Tokyo', 7),
51. ('a04', 'Gray', 'New York', 6),
52. ('a05', 'Otasi', 'Duluth', 5),
53. ('a06', 'Smith', 'Dallas', 5);

```

2.3 向 Product 插入数据

```
54. INSERT INTO Products (pid, pname, city, quantity, price)
```

```
55. VALUES
```

```
56. ('p01', 'comb', 'Dallas', 111400, 0.50),
```

```
57. ('p02', 'brush', 'Newark', 203000, 0.50),
```

```
58. ('p03', 'razor', 'Duluth', 150600, 1.00),
```

```
59. ('p04', 'pen', 'Duluth', 125300, 1.00),
```

```
60. ('p05', 'pencil', 'Dallas', 221400, 1.00),
```

```
61. ('p06', 'folder', 'Dallas', 123100, 2.00),
```

```
62. ('p07', 'case', 'Newark', 100500, 1.00);
```

2.4 向 Orders 插入数据

```
1. INSERT INTO Orders (ordno, orddate, cid, aid, pid, qty, dols)
```

```
2. VALUES
```

```
3. (1011, '2016-01-08', 'c001', 'a01', 'p01', 1000, 450.00),
```

```
4. (1012, '2016-01-12', 'c001', 'a01', 'p01', 1000, 450.00),
```

```
5. (1019, '2016-02-24', 'c001', 'a02', 'p02', 400, 180.00),
```

```
6. (1017, '2016-02-10', 'c001', 'a06', 'p03', 600, 540.00),
```

```
7. (1018, '2016-02-16', 'c001', 'a03', 'p04', 600, 540.00),
```

```
8. (1023, '2016-03-12', 'c001', 'a04', 'p05', 500, 450.00),
```

```
9. (1022, '2016-03-08', 'c001', 'a05', 'p06', 400, 720.00),
```

```
10. (1025, '2016-04-07', 'c001', 'a05', 'p07', 800, 720.00),
```

```
11. (1013, '2016-01-13', 'c002', 'a03', 'p03', 1000, 880.00),
```

```
12. (1026, '2016-05-20', 'c002', 'a05', 'p03', 800, 704.00),
```

```
13. (1015, '2016-01-23', 'c003', 'a03', 'p05', 1200, 1104.00),
```

```
14. (1014, '2016-01-18', 'c003', 'a03', 'p05', 1200, 1104.00),
```

```
15. (1021, '2016-02-28', 'c004', 'a06', 'p01', 1000, 460.00),
```

```
16. (1016, '2016-01-25', 'c006', 'a01', 'p01', 1000, 500.00),
```

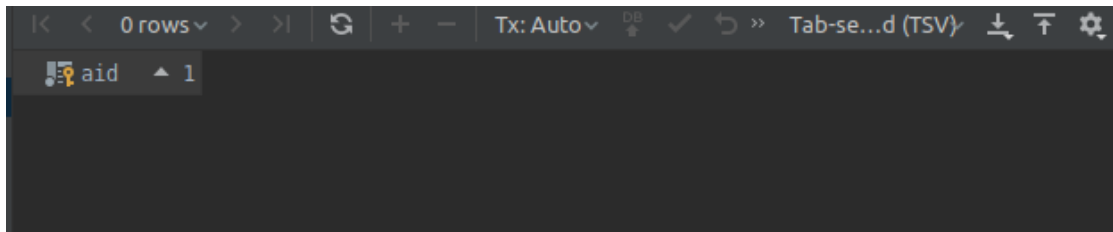
```
17. (1020, '2016-02-05', 'c006', 'a03', 'p07', 600, 600.00),
```

```
18. (1024, '2016-03-12', 'c006', 'a06', 'p01', 800, 400.00);
```

3 使用 SQL 语言写出下列查询，并将查询结果保留在实验报告中

3.1 检索没有为居住在 **Duluth** 的任何客户订购过任何商品的经销商的编号

```
1. SELECT Agents.aid
2. FROM Agents
3. WHERE Agents.aid NOT IN (
4.     SELECT DISTINCT Agents.aid
5.     FROM Agents, Orders, Customers
6.     WHERE Orders.aid=Agents.aid AND Orders.cid=Customers.cid AND Customers.city='Duluth'
7. );
```



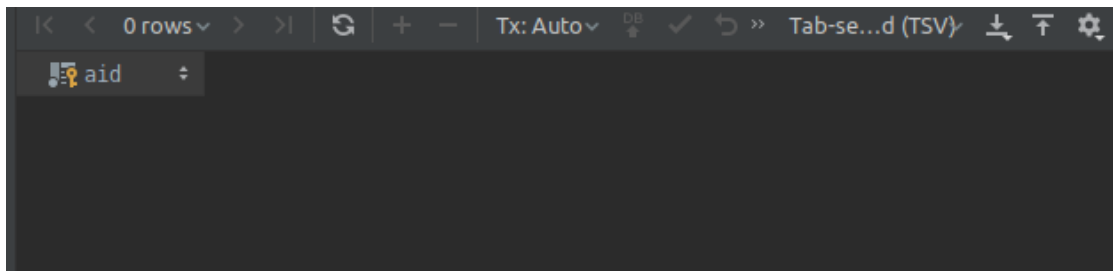
3.2 检索为居住在 **Duluth** 和 **Kyoto** 的所有客户订购过同一种商品的经销商的编号

```
1. SELECT Agents.aid
2. FROM Agents
3. WHERE EXISTS(
4.     SELECT Products.pid
5.     FROM Products
6.     WHERE NOT EXISTS (
7.         SELECT *
8.         FROM Customers
9.         WHERE Customers.city IN ('Duluth', 'Kyoto') AND NOT EXISTS (
10.             SELECT *
11.             FROM Orders
12.             WHERE Orders.aid=Agents.aid AND Orders.pid=Products.pid AND Orders.cid=Customers.cid
```

```

13.         )
14.     )
15. );

```

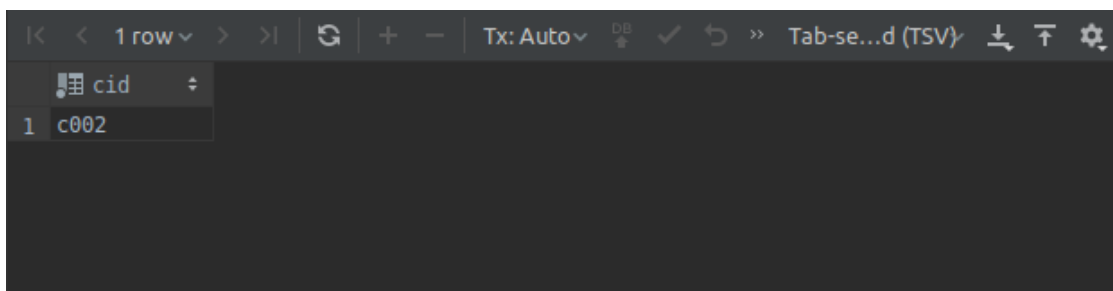


3.3 检索仅通过 a03 和 a05 两个经销商订购过商品的客户编号

```

63. SELECT DISTINCT o.cid
64. FROM Orders o
65. WHERE o.aid='a03'
66. AND o.cid IN (
67.     SELECT d.cid
68.     FROM Orders d
69.     WHERE d.aid='a05'
70. )
71. AND NOT EXISTS (
72.     SELECT *
73.     FROM Orders r
74.     WHERE r.cid=o.cid AND r.aid NOT IN ('a03', 'a05')
75. );

```



3.4 在所有有客户的城市中都被销售过的商品的编号

```

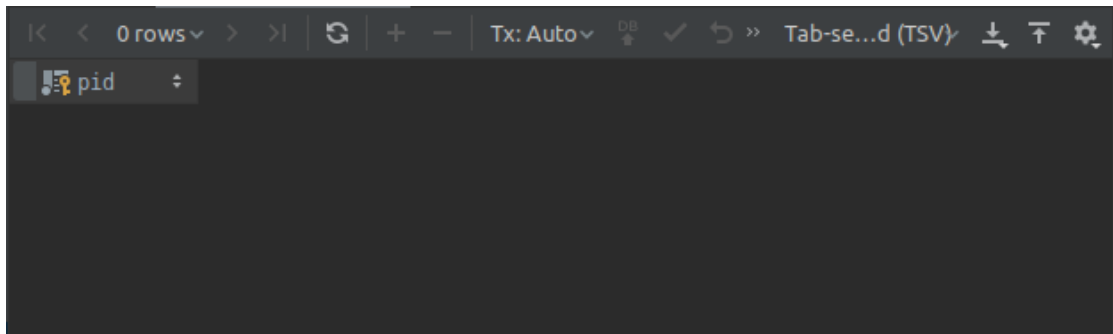
1. SELECT Products.pid
2. FROM Products
3. WHERE NOT EXISTS (
4.     SELECT Customers.city

```

```

5.     FROM Customers
6.     WHERE NOT EXISTS (
7.         SELECT *
8.         FROM Orders, Customers c
9.         WHERE Orders.cid=c.cid AND Orders.pid=Products.pid AND c.city=Custom
           ers.city
10.     )
11. );

```



3.5 返回每一个客户的编号及其最后两份订单的订购日期（按照订单编号的大小区分订单的先后）

```

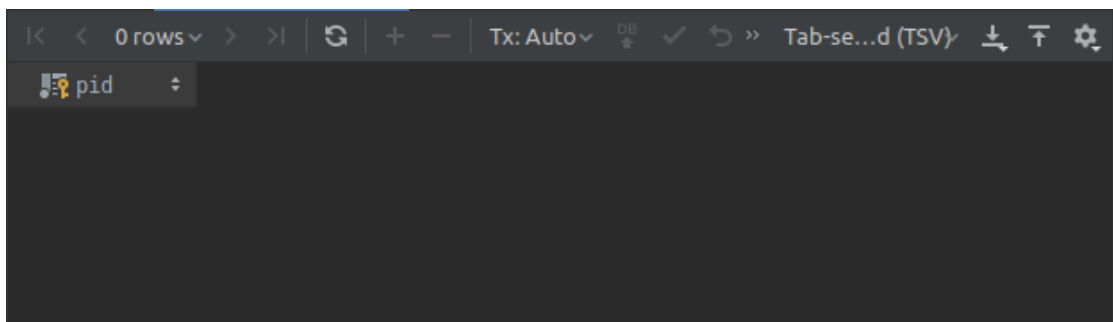
1. SELECT Orders.cid, Orders.orddate
2. FROM Orders
3. WHERE 2 > (
4.     SELECT COUNT(*)
5.     FROM Orders o
6.     WHERE o.cid=Orders.cid AND o.orddate < Orders.orddate
7. )
8. ORDER BY Orders.cid, Orders.ordno ASC;

```

	cid	orddate
1	c001	2016-01-08
2	c001	2016-01-12
3	c002	2016-01-13
4	c002	2016-05-20
5	c003	2016-01-18
6	c003	2016-01-23
7	c004	2016-02-28
8	c006	2016-01-25
9	c006	2016-02-05

3.6 检索居住在 Dallas 的所有客户都订购过的商品编号

```
1. SELECT Products.pid
2. FROM Products
3. WHERE NOT EXISTS (
4.     SELECT *
5.     FROM Customers
6.     WHERE Customers.city='Dallas' AND NOT EXISTS (
7.         SELECT *
8.         FROM Orders
9.         WHERE Orders.pid=Products.pid AND Orders.cid=Customers.cid
10.    )
11. );
```



3.7 检索为居住在 Duluth 的所有客户订购过商品的经销商的编号及其佣金百分比，并按照佣金百分比的降序输出查询结果

```
1. SELECT Agents.aid, Agents.perc
2. FROM Agents
3. WHERE NOT EXISTS (
4.     SELECT *
5.     FROM Customers
6.     WHERE Customers.city='Duluth' AND NOT EXISTS(
7.         SELECT *
8.         FROM Orders
9.         WHERE Orders.aid=Agents.aid AND Orders.cid=Customers.cid
10.    )
11. )
12. ORDER BY Agents.perc DESC ;
```

Output HW1.Agents	
1	a06
	perc 5

3.8 检索符合下述条件的商品的编号：至少有一个客户通过与该客户位于同一个城市的经销商订购过该商品

```

1. SELECT Products.pid
2. FROM Products
3. WHERE EXISTS(
4.     SELECT *
5.     FROM Agents a, Customers c, Orders o
6.     WHERE o.aid=a.aid AND o.cid=c.cid AND c.city=a.city AND o.pid=Products.p
       id
7. );

```

Output HW1.Products	
1	p06
2	p07

3.9 检索享有最高销售提成比例的经销商（请分别写出使用统计函数和不使用统计函数的两种不同表示方法）

```

1. # 不使用统计函数
2. SELECT a.aname
3. FROM Agents a
4. WHERE a.perc >= ALL(
5.     SELECT Agents.perc
6.     FROM Agents

```



```

7. );
8.
9. # 使用统计函数
10. SELECT a.aname
11. FROM Agents a
12. WHERE a.perc = ALL(
13.     SELECT MAX(Agents.perc)
14.     FROM Agents
15. );

```



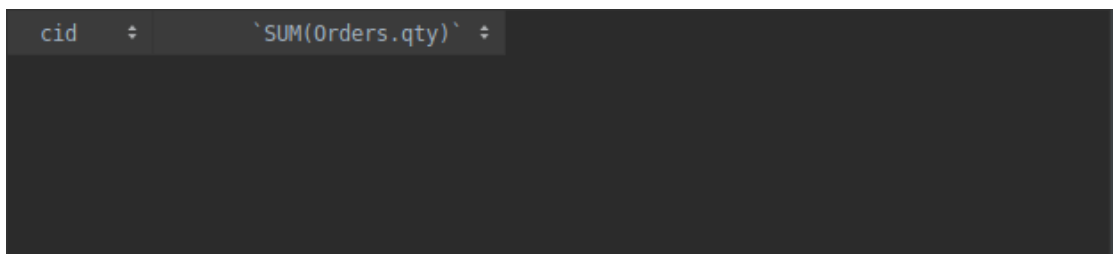
	aname
1	Brown

3.10 检索仅仅通过 a04 号经销商订购过商品的客户编号，并给出每个客户的订购总金额

```

1. SELECT Orders.cid, SUM(Orders.qty)
2. FROM Orders
3. WHERE Orders.aid='a04' AND Orders.cid NOT IN (
4.     SELECT o.cid
5.     FROM Orders o
6.     WHERE o.aid<>'a04'
7. )
8. GROUP BY Orders.cid;

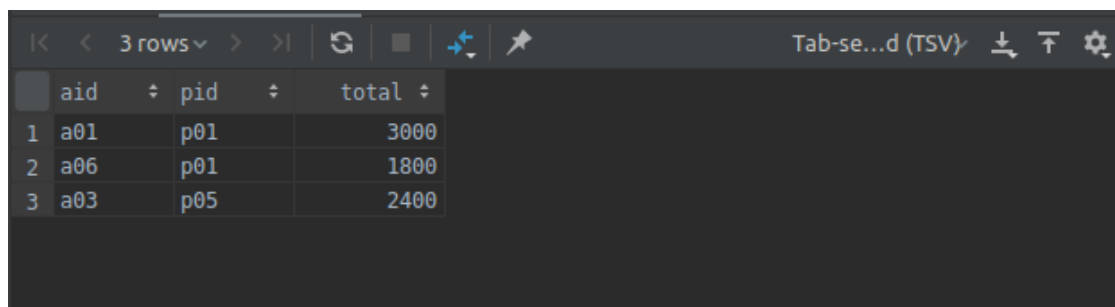
```



cid	SUM(Orders.qty)
-----	-----------------

3.11 检索每个经销商销售每一种产品的总数量，结果只需要返回每一种商品中，销售总数量排名前三且数量统计超过 1000 的

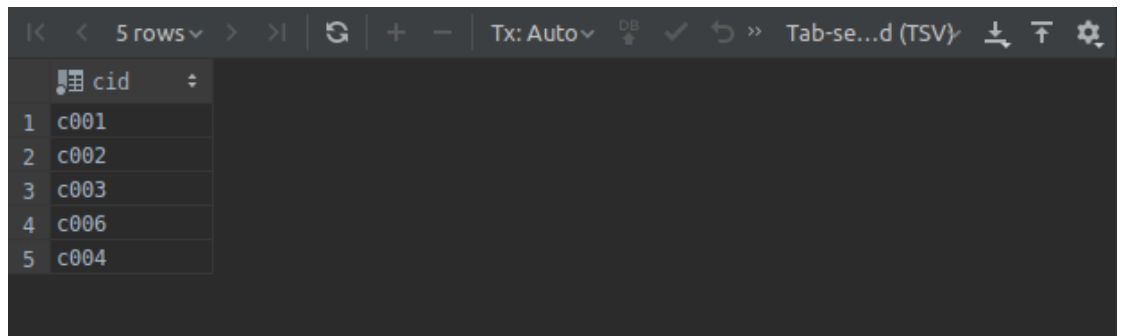
```
1. SELECT t.aid, t.pid, t.total
2. FROM
3.     (SELECT Orders.aid, Orders.pid, SUM(Orders.qty) AS total
4.     FROM Orders
5.     GROUP BY Orders.aid, Orders.pid) t
6. WHERE t.total>1000 AND 3>(
7.     SELECT COUNT(*)
8.     FROM
9.         (SELECT Orders.aid, Orders.pid, SUM(Orders.qty) AS total
10.        FROM Orders
11.        GROUP BY Orders.aid, Orders.pid) t1
12.     WHERE t1.pid=t.pid AND t1.total>t.total
13. )
14. ORDER BY t.pid ASC;
```



	aid	pid	total
1	a01	p01	3000
2	a06	p01	1800
3	a03	p05	2400

3.12 检索符合下述要求的客户的编号：在该客户订购过的所有商品中，每一种商品的平均每笔订单的订购数量均达到或超过 300

```
1. SELECT DISTINCT Orders.cid
2. FROM Orders
3. WHERE NOT EXISTS (
4.     SELECT o.pid
5.     FROM Orders o
6.     WHERE o.cid=Orders.cid
7.     GROUP BY o.pid
8.     HAVING AVG(o.qty)<300
9. );
```



	cid
1	c001
2	c002
3	c003
4	c006
5	c004

4 删除基本表

DROP TABLE Agents, Customers, Orders, Products;

实验中遇到的困难及解决办法

困难：3.11 查询中先按照 `pid` 和 `aid` 分组统计后进行排序的实现有些棘手。参考 [StackOverflow](#) 的一些问答后我采取了实验报告中的先 `COUNT` 筛选再分组统计的解决方案。

建议：可以增加输入数据的规模。