实验一结果截图

一、实验截图

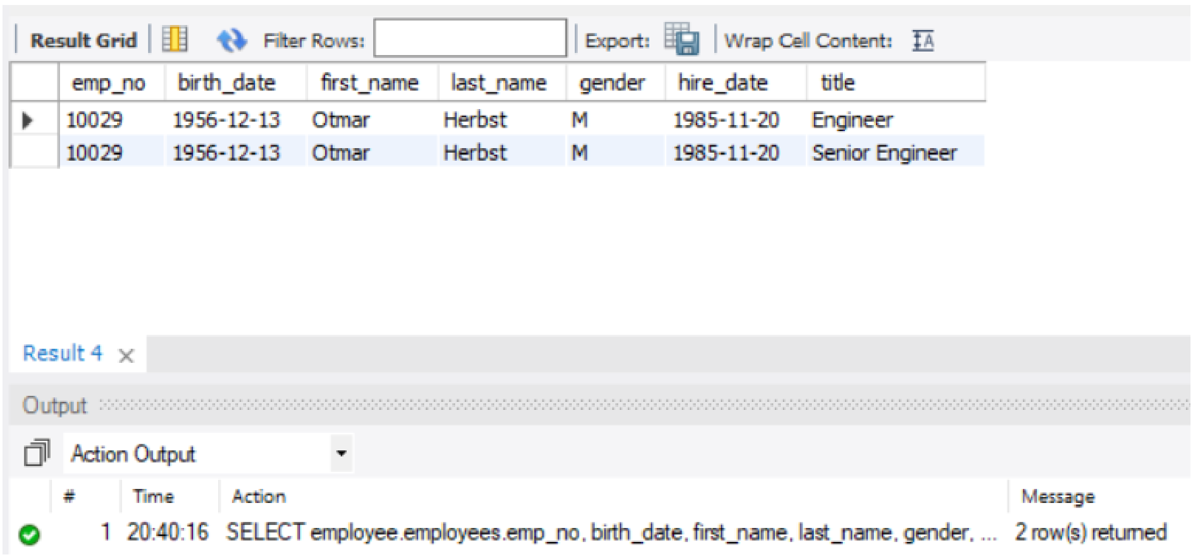
（注意截图清晰，包含完整的sql语句和运行结果）

（1）查询 emp_no 是“10029”的员工信息，显示其 emp_no, birth_date, first_name, last_name, gender, hire_date, title；

代码：

```
SELECT employee.employees.emp_no, birth_date, first_name, last_name, gender,
hire_date, title
FROM employee.employees,employee.titles
where employee.employees.emp_no = employee.titles.emp_no
and employee.titles.emp_no='10029'
```

截图：



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

emp_no	birth_date	first_name	last_name	gender	hire_date	title
10029	1956-12-13	Otmar	Herbst	M	1985-11-20	Engineer
10029	1956-12-13	Otmar	Herbst	M	1985-11-20	Senior Engineer

Result 4

Output

Action Output

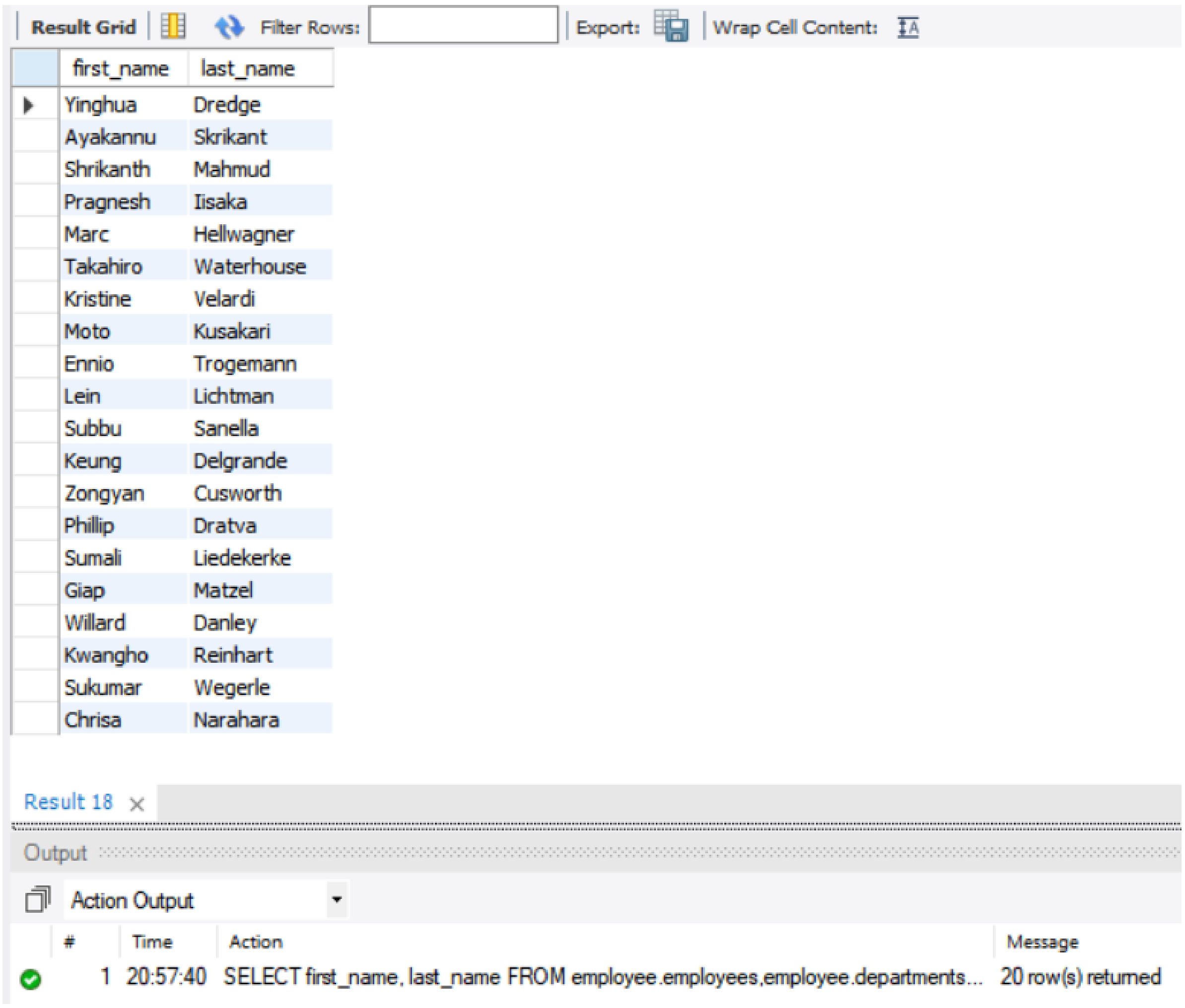
#	Time	Action	Message
1	20:40:16	SELECT employee.employees.emp_no, birth_date, first_name, last_name, gender, ...	2 row(s) returned

（2）查询入职时间在 1990 年后且在“Finance”部门工作过的男员工姓名；

代码：

```
SELECT first_name, last_name
FROM employee.employees,employee.departments,employee.dept_emp
where employee.employees.emp_no = employee.dept_emp.emp_no
and employee.dept_emp.dept_no = employee.departments.dept_no
and dept_name = 'Finance'
and gender = 'M'
and hire_date >= '1990-01-01'
```

截图：



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

first_name	last_name
Yinghua	Dredge
Ayakannu	Skrikant
Shrikanth	Mahmud
Pragnesh	Iisaka
Marc	Hellwagner
Takahiro	Waterhouse
Kristine	Velardi
Moto	Kusakari
Ennio	Trogemann
Lein	Lichtman
Subbu	Sanella
Keung	Delgrande
Zongyan	Cusworth
Phillip	Dratva
Sumali	Liedekerke
Giap	Matzel
Willard	Danley
Kwangho	Reinhart
Sukumar	Wegerle
Chrisa	Narahara

Result 18

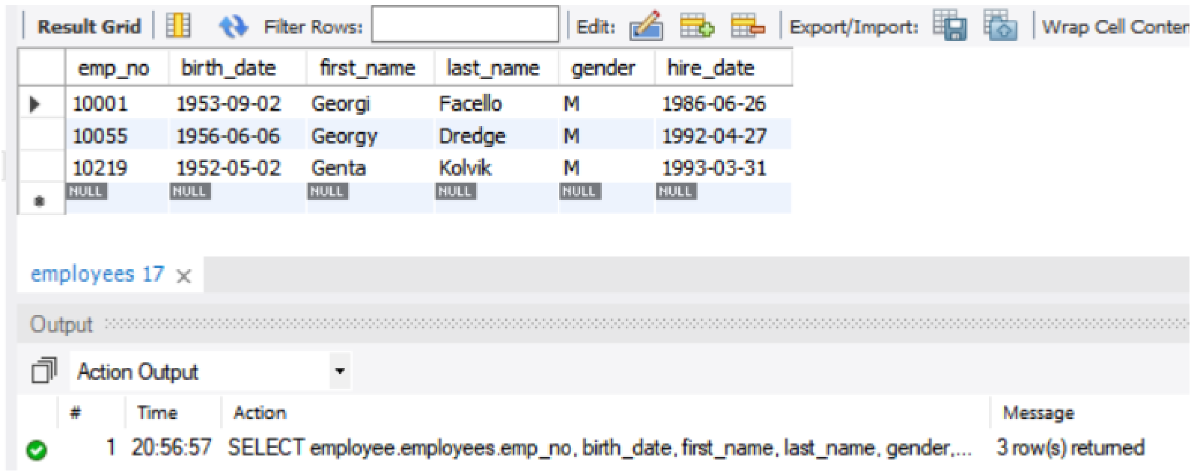
Output

Action Output

```
#   Time      Action                                                                  Message
1   20:57:40  SELECT first_name, last_name FROM employee.employees,employee.departments...  20 row(s) returned
```

（3）查询没有在“Production”部门工作过且 first_name 是“Ge”开头的的员工信息，显示其 emp_no, birth_date, first_name, last_name, gender, hire_date；

```sql
SELECT employee.employees.emp_no, birth_date, first_name, last_name,
gender,hire_date
FROM employee.employees
where employee.employees.first_name like 'Ge%'
and employee.employees.emp_no not in(
    SELECT employee.employees.emp_no
    FROM employee.employees,employee.departments,employee.dept_emp
    where employee.employees.emp_no = employee.dept_emp.emp_no
    and employee.dept_emp.dept_no = employee.departments.dept_no
    and employee.departments.dept_name = 'Production'
)
```



emp_no	birth_date	first_name	last_name	gender	hire_date
10001	1953-09-02	Georgi	Facello	M	1986-06-26
10055	1956-06-06	Georgy	Dredge	M	1992-04-27
10219	1952-05-02	Genta	Kolvik	M	1993-03-31
NULL	NULL	NULL	NULL	NULL	NULL

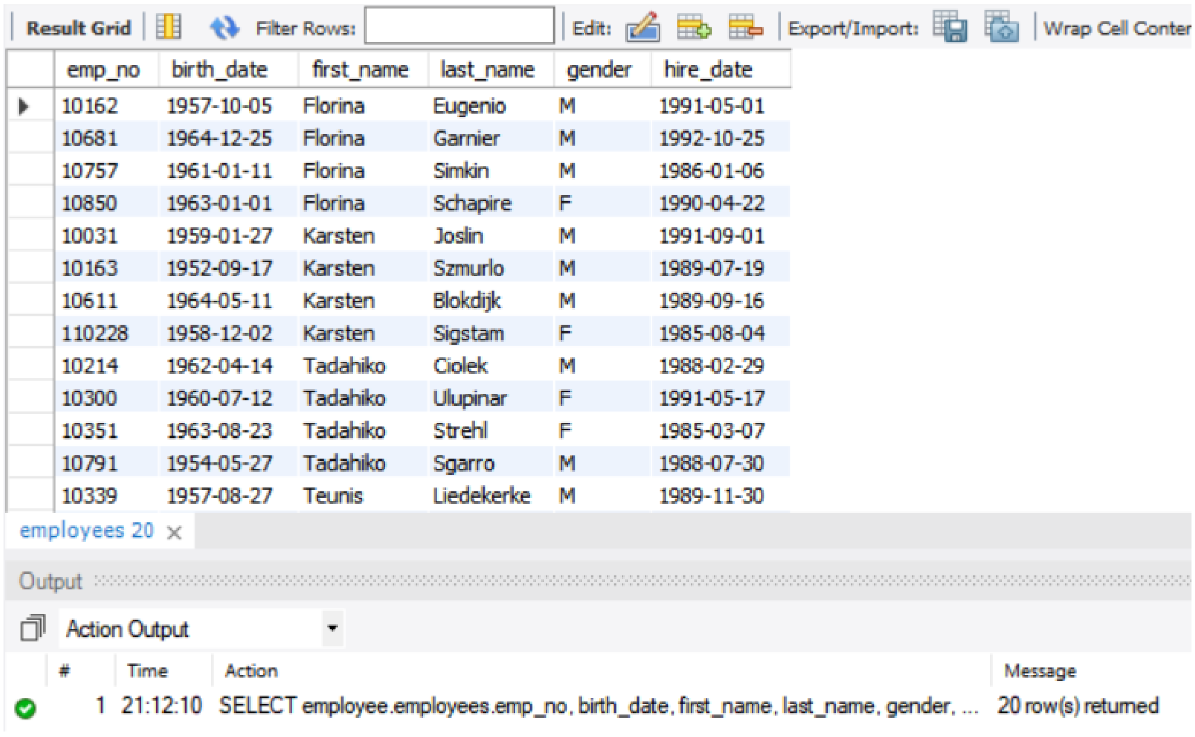
```
#   Time       Action                                                                       Message
1   20:56:57   SELECT employee.employees.emp_no, birth_date, first_name, last_name, gender,...   3 row(s) returned
```

（4）查询 first_name 相同且人数超过 3 人的员工信息，显示其 emp_no, birth_date, first_name, last_name, gender, hire_date，要求按 first_name升序显示；

代码：

```
SELECT employee.employees.emp_no, birth_date, first_name, last_name, gender,
hire_date
FROM employee.employees
WHERE first_name IN (
    SELECT first_name
    FROM employee.employees
    GROUP BY first_name
    HAVING COUNT(first_name)>3
    )
ORDER BY first_name;
```

截图：



emp_no	birth_date	first_name	last_name	gender	hire_date
10162	1957-10-05	Florina	Eugenio	M	1991-05-01
10681	1964-12-25	Florina	Garnier	M	1992-10-25
10757	1961-01-11	Florina	Simkin	M	1986-01-06
10850	1963-01-01	Florina	Schapire	F	1990-04-22
10031	1959-01-27	Karsten	Joslin	M	1991-09-01
10163	1952-09-17	Karsten	Szmurlo	M	1989-07-19
10611	1964-05-11	Karsten	Blokdijk	M	1989-09-16
110228	1958-12-02	Karsten	Sigstam	F	1985-08-04
10214	1962-04-14	Tadahiko	Ciolek	M	1988-02-29
10300	1960-07-12	Tadahiko	Ulupinar	F	1991-05-17
10351	1963-08-23	Tadahiko	Strehl	F	1985-03-07
10791	1954-05-27	Tadahiko	Sgarro	M	1988-07-30
10339	1957-08-27	Teunis	Liedekerke	M	1989-11-30

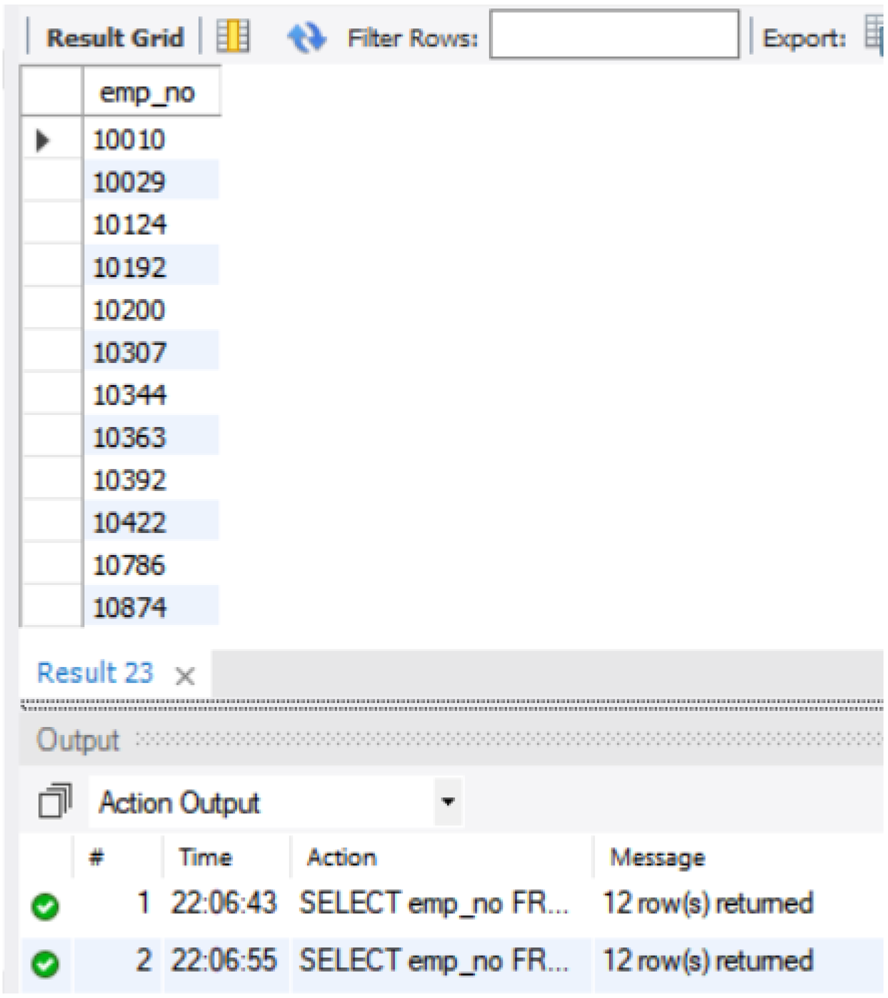
```
#   Time       Action                                                                        Message
1   21:12:10   SELECT employee.employees.emp_no, birth_date, first_name, last_name, gender, ...   20 row(s) returned
```

（5）查询至少在“Production”和“Quality Management”两个部门都工作过的员工编号；

代码：

```sql
SELECT emp_no
FROM employee.dept_emp INNER JOIN employee.departments
ON employee.dept_emp.dept_no = employee.departments.dept_no
WHERE dept_name = 'Production'
AND emp_no IN (
    SELECT emp_no
    FROM employee.dept_emp INNER JOIN employee.departments
    ON employee.dept_emp.dept_no = employee.departments.dept_no
    WHERE dept_name = 'Quality Management');
```

截图：



emp_no
10010
10029
10124
10192
10200
10307
10344
10363
10392
10422
10786
10874

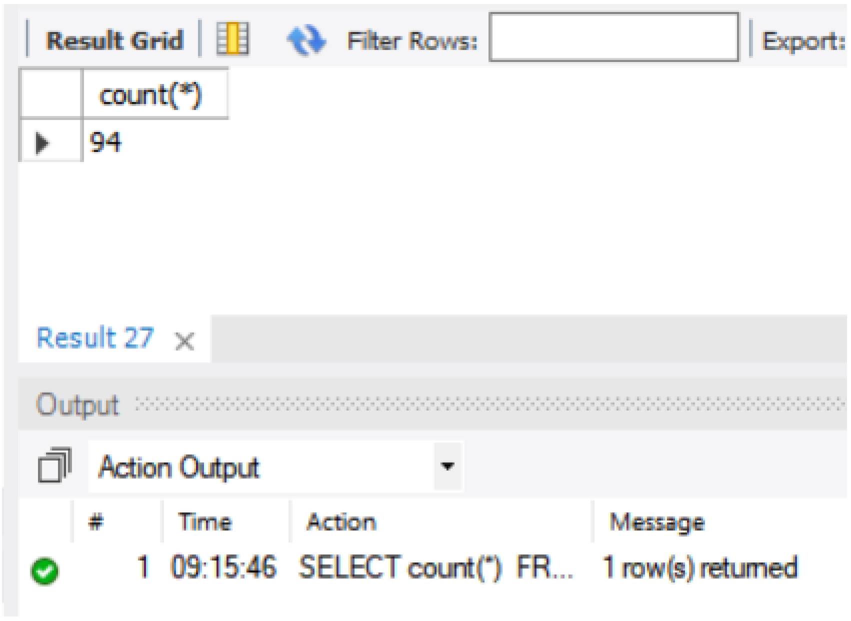
```
#  Time      Action            Message
1  22:06:43  SELECT emp_no FR...  12 row(s) returned
2  22:06:55  SELECT emp_no FR...  12 row(s) returned
```

（6）查询至少在 2 个部门工作过的员工人数；

代码：

```
SELECT count(*)
FROM employee.employees
WHERE emp_no in (
    SELECT emp_no
    FROM employee.dept_emp
    GROUP BY emp_no
    HAVING count(*)>=2);
```

截图:



```
count(*)
94
```

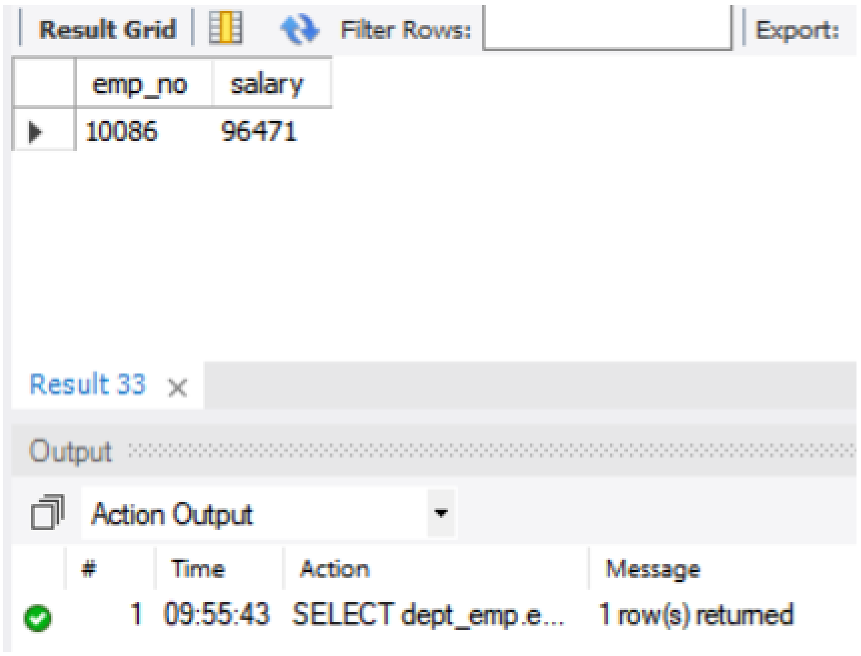
```
#   Time       Action                Message
1   09:15:46   SELECT count(*) FR...  1 row(s) returned
```

(7) 查询在“d003”部门工作过的且工资最高的员工编号及其最高工资;

代码:

```
SELECT dept_emp.emp_no,salary
FROM employee.dept_emp INNER JOIN employee.salaries
ON dept_emp.emp_no = salaries.emp_no
WHERE dept_emp.dept_no = 'd003'
ORDER BY salary DESC
LIMIT 1;
```

截图:


Result Grid | Filter Rows: | Export:

	emp_no	salary
▶	10086	96471

Result 33 ×

Output

Action Output

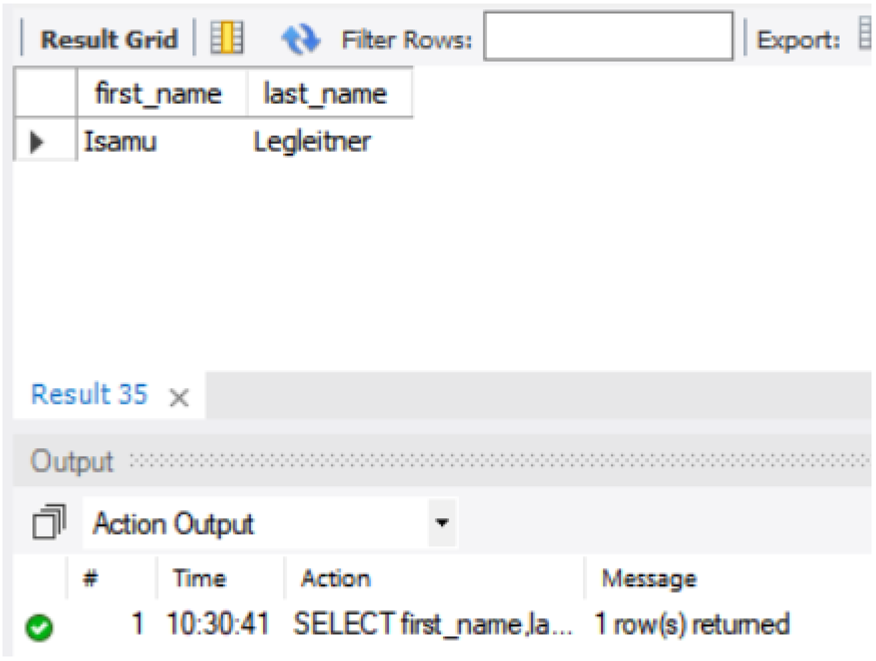
```
   #  Time      Action                Message
✓  1  09:55:43  SELECT dept_emp.e...  1 row(s) returned
```

（8）查询“d002”部门的当前领导姓名；

代码：

```
SELECT first_name,last_name
FROM employee.dept_manager INNER JOIN employee.employees
ON dept_manager.emp_no = employees.emp_no
WHERE to_date = '9999-01-01'
AND dept_no = 'd002';
```

截图：


Result Grid | Filter Rows: | Export:

	first_name	last_name
▶	Isamu	Legleitner

Result 35 ×

Output

Action Output

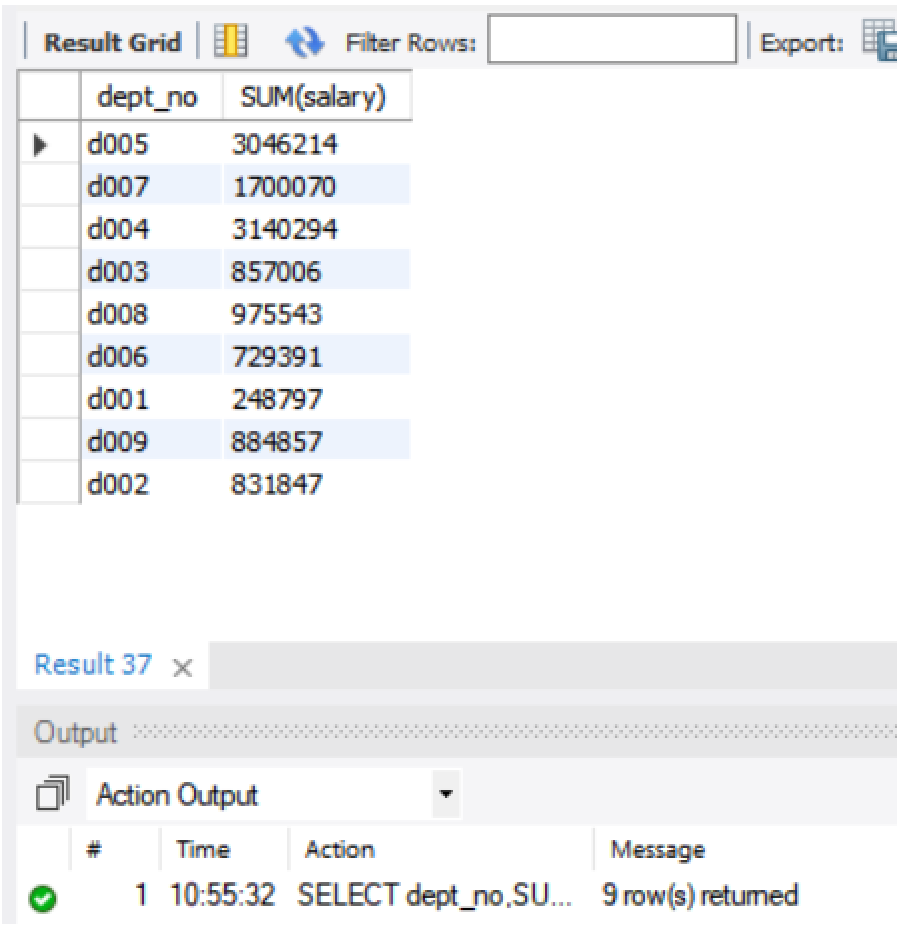
```
   #  Time      Action                Message
✓  1  10:30:41  SELECT first_name,la...  1 row(s) returned
```

（9）查询当前每个部门的部门编号和员工总工资；

代码：

```
SELECT dept_no,SUM(salary)
FROM employee.salaries,employee.dept_emp
WHERE salaries.emp_no = dept_emp.emp_no
AND employee.salaries.to_date = '9999-01-01'
GROUP BY dept_no;
```

截图:



Result Grid | Filter Rows: | Export:

dept_no	SUM(salary)
d005	3046214
d007	1700070
d004	3140294
d003	857006
d008	975543
d006	729391
d001	248797
d009	884857
d002	831847

Result 37

Output

Action Output

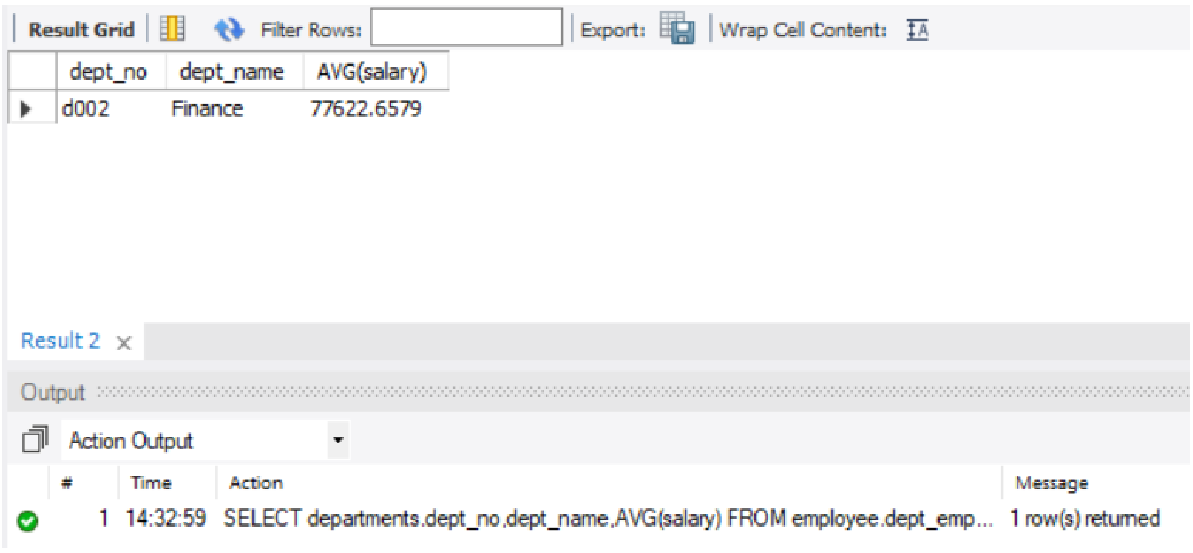
```
#   Time      Action               Message
1   10:55:32  SELECT dept_no,SU...  9 row(s) returned
```

(10) 查询当前部门员工平均工资在 70000 元到 80000 元（包含70000，低于80000）的部门编号，部门名称和员工平均工资；

代码:

```
SELECT departments.dept_no,dept_name,AVG(salary)
FROM employee.dept_emp INNER JOIN employee.salaries ON
dept_emp.emp_no=salaries.emp_no
INNER JOIN employee.departments on departments.dept_no=dept_emp.dept_no
WHERE salaries.to_date='9999-01-01' AND dept_emp.to_date='9999-01-01'
GROUP BY departments.dept_no
HAVING AVG(salary)<80000 AND AVG(salary)>=70000;
```

截图:



dept_no	dept_name	AVG(salary)
d002	Finance	77622.6579

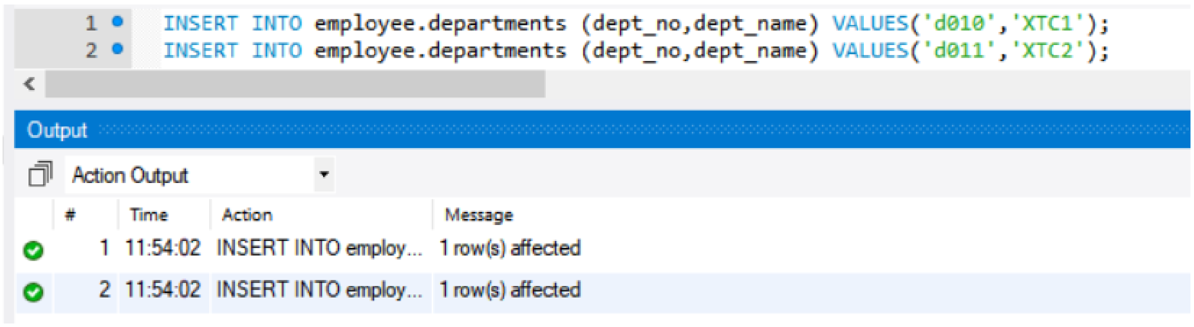
```
#  Time      Action                                                       Message
1  14:32:59  SELECT departments.dept_no,dept_name,AVG(salary) FROM employee.dept_emp...  1 row(s) returned
```

（11）在 departments 表新增 2 条记录（内容自定）；

代码:

```
INSERT INTO employee.departments (dept_no,dept_name) VALUES('d010','XTC1');
INSERT INTO employee.departments (dept_no,dept_name) VALUES('d011','XTC2');
```

截图:



```
INSERT INTO employee.departments (dept_no,dept_name) VALUES('d010','XTC1');
INSERT INTO employee.departments (dept_no,dept_name) VALUES('d011','XTC2');
```

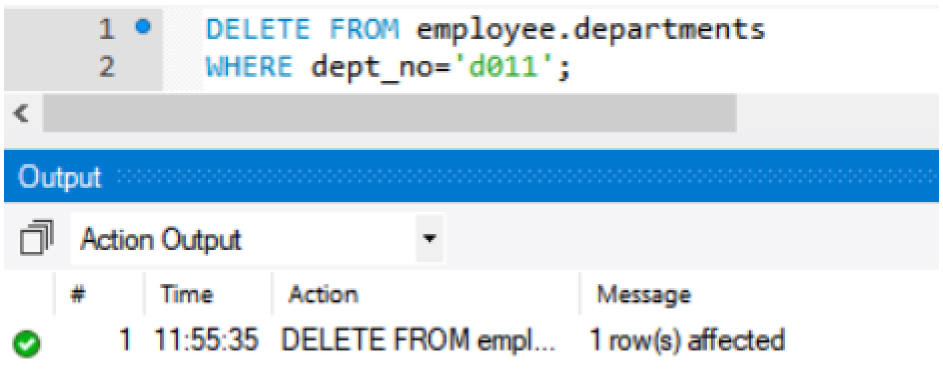
```
#  Time      Action                Message
1  11:54:02  INSERT INTO employ...  1 row(s) affected
2  11:54:02  INSERT INTO employ...  1 row(s) affected
```

（12）在 departments 表中删除刚才新增的 2 条记录中的 1 条;

代码:

```
DELETE FROM employee.departments
WHERE dept_no='d011';
```

截图:



```
DELETE FROM employee.departments
WHERE dept_no='d011';
```

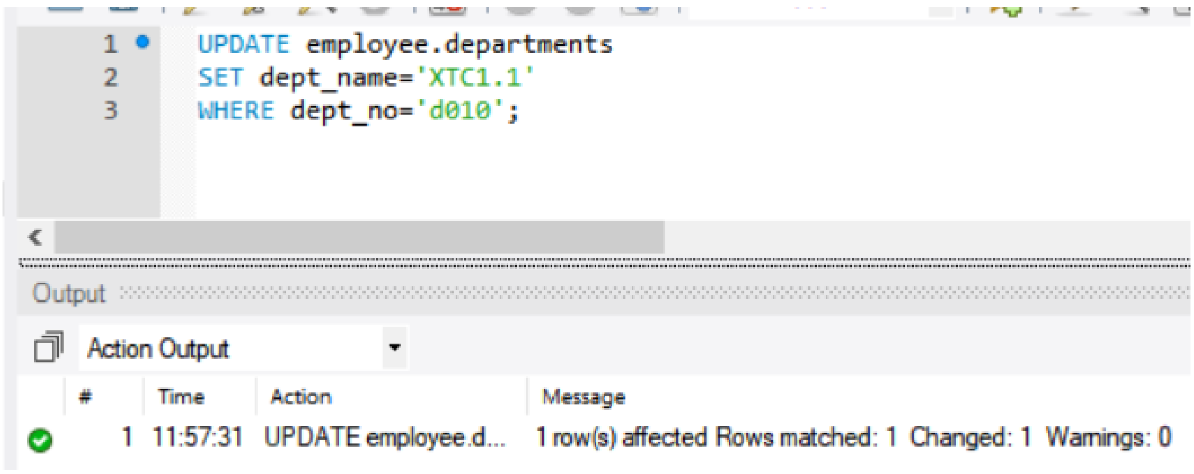
```
#  Time      Action               Message
1  11:55:35  DELETE FROM empl...  1 row(s) affected
```

(13)在 departments 表中修改步骤 11 新增的记录;

代码:

```
UPDATE employee.departments
SET dept_name='XTC1.1'
WHERE dept_no='d010';
```

截图:



```
UPDATE employee.departments
SET dept_name='XTC1.1'
WHERE dept_no='d010';
```

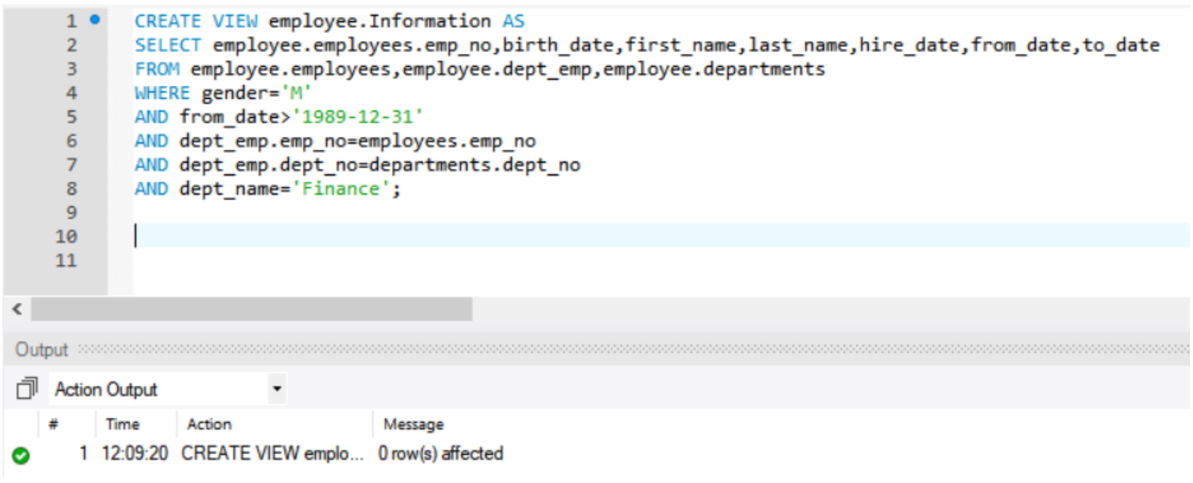
```
Output
Action Output
#  Time      Action                Message
1  11:57:31  UPDATE employee.d...  1 row(s) affected Rows matched: 1  Changed: 1  Warnings: 0
```

(14)新建视图，查询所有在 1990 年后入职过"Finance"部门的男员工信息，包括：emp_no, birth_date, first_name, last_name, hire_date, from_date, to_date。

代码:

```
CREATE VIEW employee.Information AS
SELECT
employee.employees.emp_no,birth_date,first_name,last_name,hire_date,from_date,to
_date
FROM employee.employees,employee.dept_emp,employee.departments
WHERE gender='M'
AND from_date>'1989-12-31'
AND dept_emp.emp_no=employees.emp_no
AND dept_emp.dept_no=departments.dept_no
AND dept_name='Finance';
```

截图:



```
CREATE VIEW employee.Information AS
SELECT employee.employees.emp_no,birth_date,first_name,last_name,hire_date,from_date,to_date
FROM employee.employees,employee.dept_emp,employee.departments
WHERE gender='M'
AND from_date>'1989-12-31'
AND dept_emp.emp_no=employees.emp_no
AND dept_emp.dept_no=departments.dept_no
AND dept_name='Finance';
```

```
Output
Action Output
#  Time      Action                Message
1  12:09:20  CREATE VIEW emplo...  0 row(s) affected
```

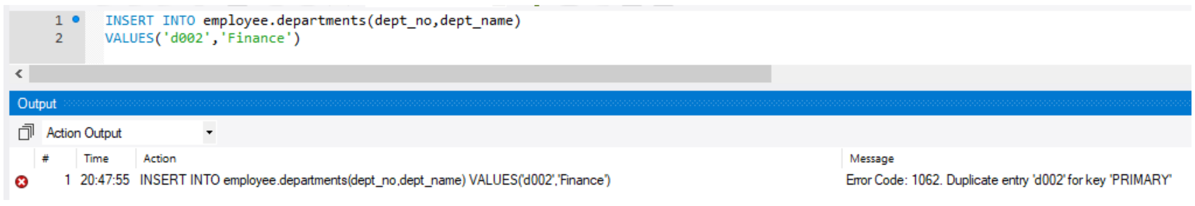
二、思考题

1. 如果 insert 一条数据到Departments，但dept_no 或 dept_name 和已有数据重复，会发生什么？同学们请自己尝试一下。

   如果对Dept_emp表新增数据，数据需满足哪些条件？有什么机制可以保证数据的正确性？

答：（1）

dept_no和dept_name不能与已有数据重复。两者一个是primary key, 一个是unique key，都具有唯一性。



```
INSERT INTO employee.departments(dept_no,dept_name)
VALUES('d002','Finance')
```

Output

Action Output

```
#   Time      Action                                                                   Message
1   20:47:55  INSERT INTO employee.departments(dept_no,dept_name) VALUES('d002','Finance')   Error Code: 1062. Duplicate entry 'd002' for key 'PRIMARY'
```

（2）对dept_emp表新增数据时，数据需要满足：

- emp_no, dept_no, from_date, to_date 四列的值非空，其中，emp_no的值应在employees的表中存在，dept_no的值应在departments表中存在。

以下机制可以保证数据的正确性：

- 创建表单时的not null申明可以保证所填值非空
- 将emp_no和dept_no设置为外键，使其与employees和departments表相关联。

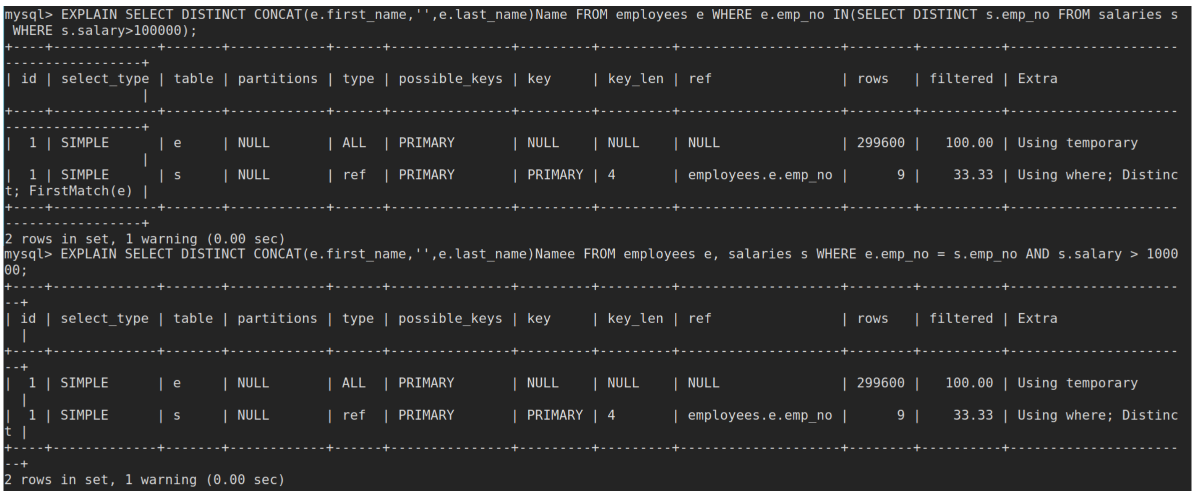
2. 使用Query Profiler、Explain进行SQL语句性能分析

导入Employees数据库：

Employees Sample Database 是MySQL官方提供的测试数据库。该测试库含有6个表，总计4百万数据记录。

答：

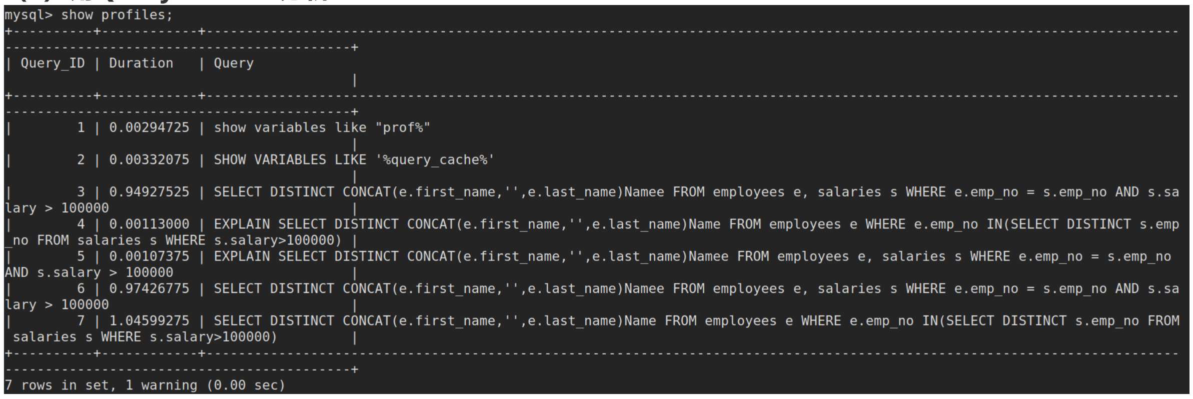
（1）用Explain分析：



```
mysql> EXPLAIN SELECT DISTINCT CONCAT(e.first_name,'',e.last_name)Name FROM employees e WHERE e.emp_no IN(SELECT DISTINCT s.emp_no FROM salaries s
 WHERE s.salary>100000);
+----+-------------+-------+------------+------+---------------+---------+---------+--------------------+--------+----------+----------------------
-----------------+
| id | select_type | table | partitions | type | possible_keys | key     | key_len | ref                | rows   | filtered | Extra
                 |
+----+-------------+-------+------------+------+---------------+---------+---------+--------------------+--------+----------+----------------------
-----------------+
|  1 | SIMPLE      | e     | NULL       | ALL  | PRIMARY       | NULL    | NULL    | NULL               | 299600 |   100.00 | Using temporary
                 |
|  1 | SIMPLE      | s     | NULL       | ref  | PRIMARY       | PRIMARY | 4       | employees.e.emp_no |      9 |    33.33 | Using where; Distinc
t; FirstMatch(e) |
+----+-------------+-------+------------+------+---------------+---------+---------+--------------------+--------+----------+----------------------
-----------------+
2 rows in set, 1 warning (0.00 sec)
mysql> EXPLAIN SELECT DISTINCT CONCAT(e.first_name,'',e.last_name)Namee FROM employees e, salaries s WHERE e.emp_no = s.emp_no AND s.salary > 1000
00;
+----+-------------+-------+------------+------+---------------+---------+---------+--------------------+--------+----------+----------------------
--+
| id | select_type | table | partitions | type | possible_keys | key     | key_len | ref                | rows   | filtered | Extra
  |
+----+-------------+-------+------------+------+---------------+---------+---------+--------------------+--------+----------+----------------------
--+
|  1 | SIMPLE      | e     | NULL       | ALL  | PRIMARY       | NULL    | NULL    | NULL               | 299600 |   100.00 | Using temporary
  |
|  1 | SIMPLE      | s     | NULL       | ref  | PRIMARY       | PRIMARY | 4       | employees.e.emp_no |      9 |    33.33 | Using where; Distinc
t |
+----+-------------+-------+------------+------+---------------+---------+---------+--------------------+--------+----------+----------------------
--+
2 rows in set, 1 warning (0.00 sec)
```

可以看到两条语句的执行计划相同。

（2）用Query Profiler分析：



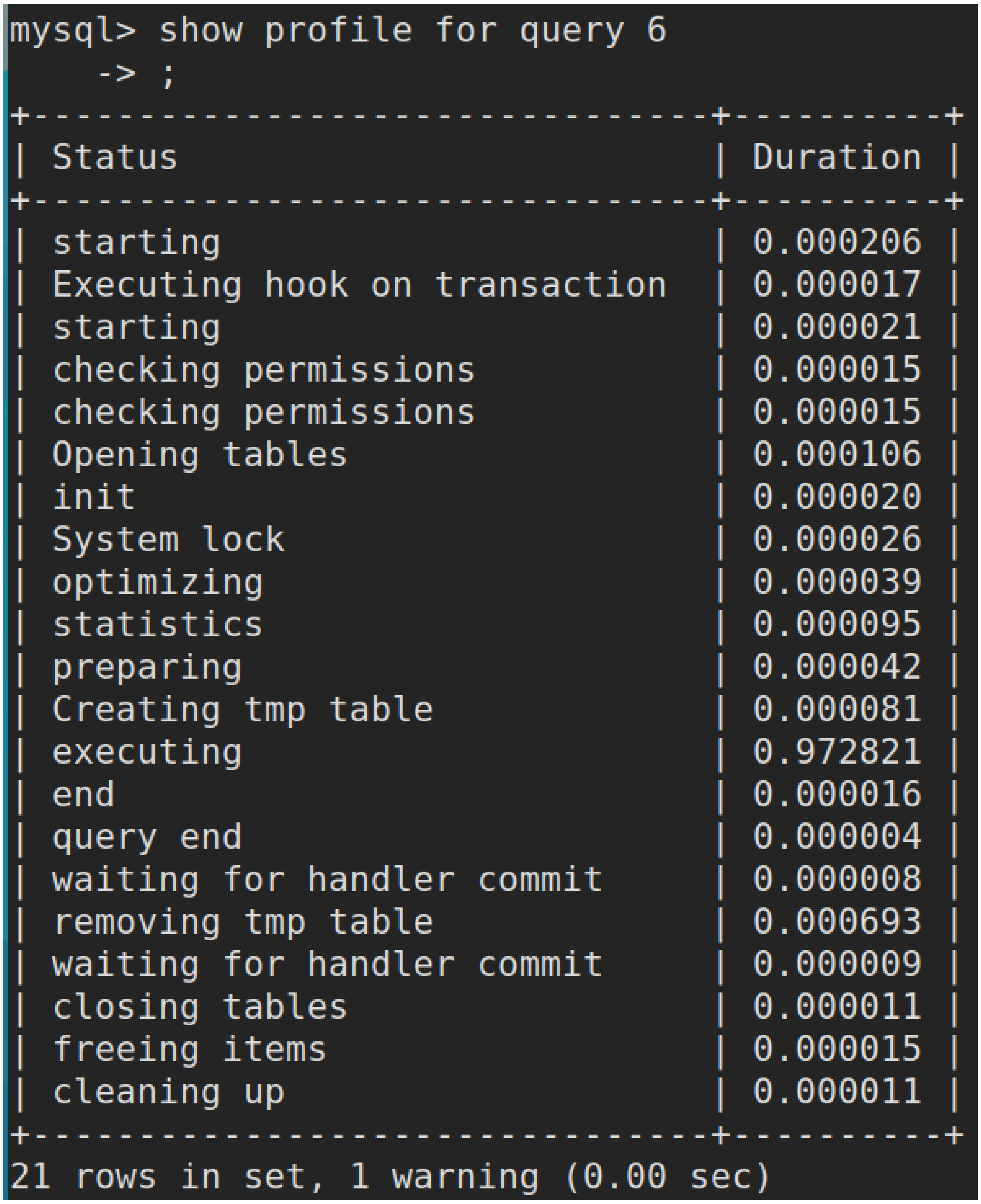
```
mysql> show profiles;
+----------+------------+----------------------------------------------------------------------------------------------------------------------
-------------------------------------------+
| Query_ID | Duration   | Query
                                           |
+----------+------------+----------------------------------------------------------------------------------------------------------------------
-------------------------------------------+
|        1 | 0.00294725 | show variables like "prof%"
                                           |
|        2 | 0.00332075 | SHOW VARIABLES LIKE '%query_cache%'
                                           |
|        3 | 0.94927525 | SELECT DISTINCT CONCAT(e.first_name,'',e.last_name)Namee FROM employees e, salaries s WHERE e.emp_no = s.emp_no AND s.sa
lary > 100000                              |
|        4 | 0.00113000 | EXPLAIN SELECT DISTINCT CONCAT(e.first_name,'',e.last_name)Name FROM employees e WHERE e.emp_no IN(SELECT DISTINCT s.emp
_no FROM salaries s WHERE s.salary>100000) |
|        5 | 0.00107375 | EXPLAIN SELECT DISTINCT CONCAT(e.first_name,'',e.last_name)Namee FROM employees e, salaries s WHERE e.emp_no = s.emp_no
AND s.salary > 100000                      |
|        6 | 0.97426775 | SELECT DISTINCT CONCAT(e.first_name,'',e.last_name)Namee FROM employees e, salaries s WHERE e.emp_no = s.emp_no AND s.sa
lary > 100000                              |
|        7 | 1.04599275 | SELECT DISTINCT CONCAT(e.first_name,'',e.last_name)Name FROM employees e WHERE e.emp_no IN(SELECT DISTINCT s.emp_no FROM
 salaries s WHERE s.salary>100000)         |
+----------+------------+----------------------------------------------------------------------------------------------------------------------
-------------------------------------------+
7 rows in set, 1 warning (0.00 sec)
```

找到：

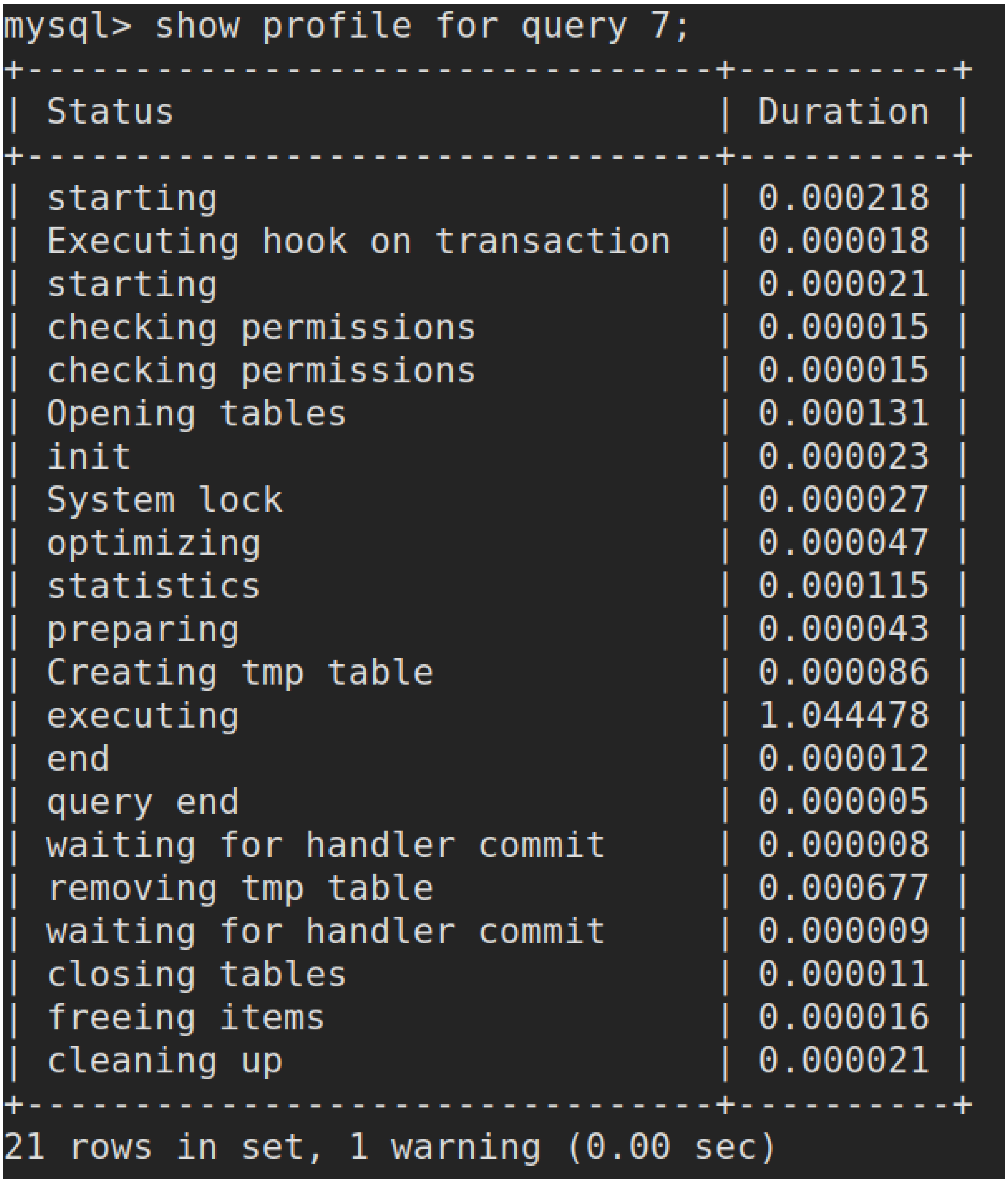
- 第一条语句的Query_ID为6
- 第二条语句的Query_ID为7

可以看到第二条语句的耗时更少。

①`show profile for query X`



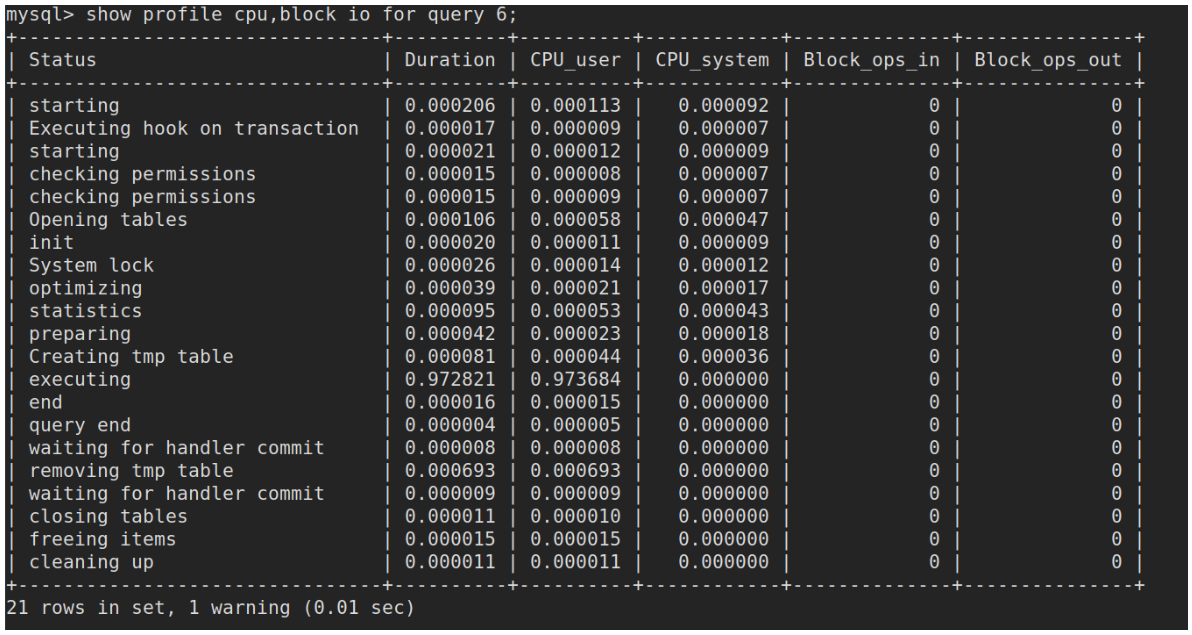
```
mysql> show profile for query 6
    -> ;
+--------------------------------+----------+
| Status                         | Duration |
+--------------------------------+----------+
| starting                       | 0.000206 |
| Executing hook on transaction  | 0.000017 |
| starting                       | 0.000021 |
| checking permissions           | 0.000015 |
| checking permissions           | 0.000015 |
| Opening tables                 | 0.000106 |
| init                           | 0.000020 |
| System lock                    | 0.000026 |
| optimizing                     | 0.000039 |
| statistics                     | 0.000095 |
| preparing                      | 0.000042 |
| Creating tmp table             | 0.000081 |
| executing                      | 0.972821 |
| end                            | 0.000016 |
| query end                      | 0.000004 |
| waiting for handler commit     | 0.000008 |
| removing tmp table             | 0.000693 |
| waiting for handler commit     | 0.000009 |
| closing tables                 | 0.000011 |
| freeing items                  | 0.000015 |
| cleaning up                    | 0.000011 |
+--------------------------------+----------+
21 rows in set, 1 warning (0.00 sec)
```



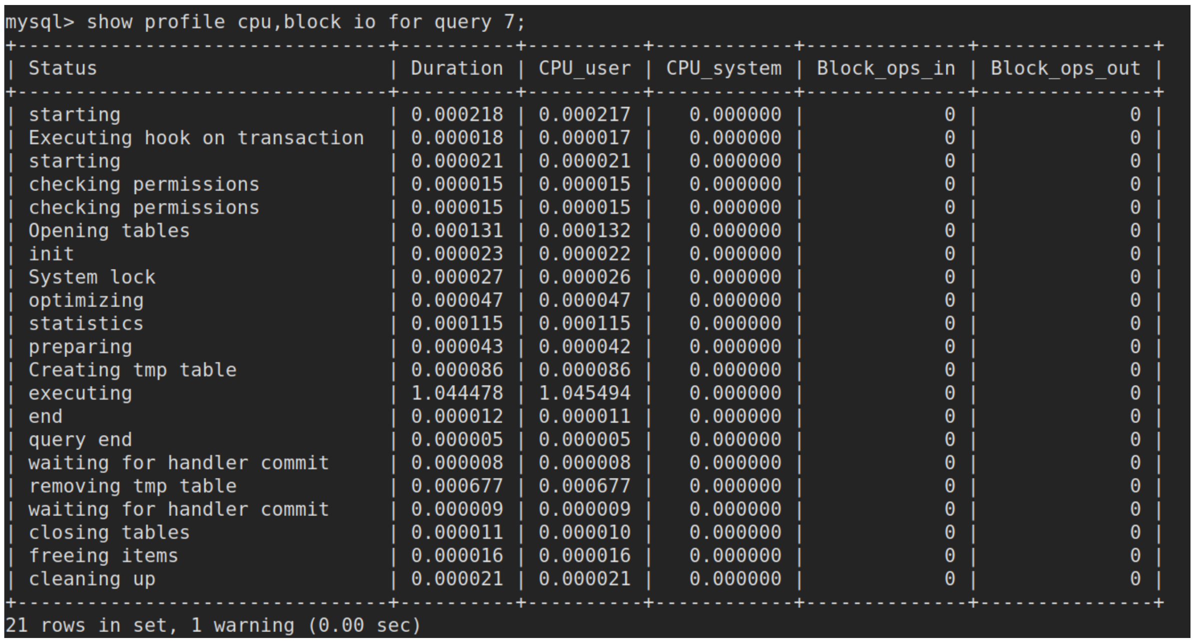
```
mysql> show profile for query 7;
+----------------------------------+----------+
| Status                           | Duration |
+----------------------------------+----------+
| starting                         | 0.000218 |
| Executing hook on transaction    | 0.000018 |
| starting                         | 0.000021 |
| checking permissions             | 0.000015 |
| checking permissions             | 0.000015 |
| Opening tables                   | 0.000131 |
| init                             | 0.000023 |
| System lock                      | 0.000027 |
| optimizing                       | 0.000047 |
| statistics                       | 0.000115 |
| preparing                        | 0.000043 |
| Creating tmp table               | 0.000086 |
| executing                        | 1.044478 |
| end                              | 0.000012 |
| query end                        | 0.000005 |
| waiting for handler commit       | 0.000008 |
| removing tmp table               | 0.000677 |
| waiting for handler commit       | 0.000009 |
| closing tables                   | 0.000011 |
| freeing items                    | 0.000016 |
| cleaning up                      | 0.000021 |
+----------------------------------+----------+
21 rows in set, 1 warning (0.00 sec)
```

可以看到，第二条语句在执行时准备的时间比第一条稍长，但在执行时间上却少很多。

②`show profile cpu,block io for query X`



```
mysql> show profile cpu,block io for query 6;
+--------------------------------+----------+----------+------------+--------------+---------------+
| Status                         | Duration | CPU_user | CPU_system | Block_ops_in | Block_ops_out |
+--------------------------------+----------+----------+------------+--------------+---------------+
| starting                       | 0.000206 | 0.000113 |   0.000092 |            0 |             0 |
| Executing hook on transaction  | 0.000017 | 0.000009 |   0.000007 |            0 |             0 |
| starting                       | 0.000021 | 0.000012 |   0.000009 |            0 |             0 |
| checking permissions           | 0.000015 | 0.000008 |   0.000007 |            0 |             0 |
| checking permissions           | 0.000015 | 0.000009 |   0.000007 |            0 |             0 |
| Opening tables                 | 0.000106 | 0.000058 |   0.000047 |            0 |             0 |
| init                           | 0.000020 | 0.000011 |   0.000009 |            0 |             0 |
| System lock                    | 0.000026 | 0.000014 |   0.000012 |            0 |             0 |
| optimizing                     | 0.000039 | 0.000021 |   0.000017 |            0 |             0 |
| statistics                     | 0.000095 | 0.000053 |   0.000043 |            0 |             0 |
| preparing                      | 0.000042 | 0.000023 |   0.000018 |            0 |             0 |
| Creating tmp table             | 0.000081 | 0.000044 |   0.000036 |            0 |             0 |
| executing                      | 0.972821 | 0.973684 |   0.000000 |            0 |             0 |
| end                            | 0.000016 | 0.000015 |   0.000000 |            0 |             0 |
| query end                      | 0.000004 | 0.000005 |   0.000000 |            0 |             0 |
| waiting for handler commit     | 0.000008 | 0.000008 |   0.000000 |            0 |             0 |
| removing tmp table             | 0.000693 | 0.000693 |   0.000000 |            0 |             0 |
| waiting for handler commit     | 0.000009 | 0.000009 |   0.000000 |            0 |             0 |
| closing tables                 | 0.000011 | 0.000010 |   0.000000 |            0 |             0 |
| freeing items                  | 0.000015 | 0.000015 |   0.000000 |            0 |             0 |
| cleaning up                    | 0.000011 | 0.000011 |   0.000000 |            0 |             0 |
+--------------------------------+----------+----------+------------+--------------+---------------+
21 rows in set, 1 warning (0.01 sec)
```



```
mysql> show profile cpu,block io for query 7;
+--------------------------------+----------+----------+------------+--------------+---------------+
| Status                         | Duration | CPU_user | CPU_system | Block_ops_in | Block_ops_out |
+--------------------------------+----------+----------+------------+--------------+---------------+
| starting                       | 0.000218 | 0.000217 |   0.000000 |            0 |             0 |
| Executing hook on transaction  | 0.000018 | 0.000017 |   0.000000 |            0 |             0 |
| starting                       | 0.000021 | 0.000021 |   0.000000 |            0 |             0 |
| checking permissions           | 0.000015 | 0.000015 |   0.000000 |            0 |             0 |
| checking permissions           | 0.000015 | 0.000015 |   0.000000 |            0 |             0 |
| Opening tables                 | 0.000131 | 0.000132 |   0.000000 |            0 |             0 |
| init                           | 0.000023 | 0.000022 |   0.000000 |            0 |             0 |
| System lock                    | 0.000027 | 0.000026 |   0.000000 |            0 |             0 |
| optimizing                     | 0.000047 | 0.000047 |   0.000000 |            0 |             0 |
| statistics                     | 0.000115 | 0.000115 |   0.000000 |            0 |             0 |
| preparing                      | 0.000043 | 0.000042 |   0.000000 |            0 |             0 |
| Creating tmp table             | 0.000086 | 0.000086 |   0.000000 |            0 |             0 |
| executing                      | 1.044478 | 1.045494 |   0.000000 |            0 |             0 |
| end                            | 0.000012 | 0.000011 |   0.000000 |            0 |             0 |
| query end                      | 0.000005 | 0.000005 |   0.000000 |            0 |             0 |
| waiting for handler commit     | 0.000008 | 0.000008 |   0.000000 |            0 |             0 |
| removing tmp table             | 0.000677 | 0.000677 |   0.000000 |            0 |             0 |
| waiting for handler commit     | 0.000009 | 0.000009 |   0.000000 |            0 |             0 |
| closing tables                 | 0.000011 | 0.000010 |   0.000000 |            0 |             0 |
| freeing items                  | 0.000016 | 0.000016 |   0.000000 |            0 |             0 |
| cleaning up                    | 0.000021 | 0.000021 |   0.000000 |            0 |             0 |
+--------------------------------+----------+----------+------------+--------------+---------------+
21 rows in set, 1 warning (0.00 sec)
```

可以看到，在CPU的使用情况上，第二条语句也比第一条好。