



OCA Oracle 11g: 巧用子查询

OCA Oracle 11g: 巧用子查询

子查询指的是将一个 SELECT 的语句的输出作为另一个 SELECT 语句（或一个 DML 语句）的输入进行查询。本技术专题主要介绍了如何定义子查询、阐述了子查询的类型以及如何在 SQL 中使用子查询、编写单行或多行子查询等。

定义 SQL 子查询

子查询是一个嵌套在 SELECT、INSERT、UPDATE 或 DELETE 语句，或者另一个子查询里的查询。一个子查询可以返回一行或多行结果给父查询。标量子查询是只返回一个值的查询：只有一个字段的一行记录。标量子查询可以在 SQL 语句的大多数位置，其中你可以使用一个表达式或一个实际值。

❖ SQL 基础一考试指南：定义 SQL 子查询

在 SQL 中使用子查询

SQL 中有很多地方可以用到子查询的。本章选自《OCA Oracle 数据库 11g: SQL 基础一考试指南》，将详细阐述如何在 SQL 中使用子查询，以及子查询所能解决的问题，如星型转换、比较、从 SELECT 生成表，等等。

❖ SQL 基础一考试指南：在 SQL 中使用子查询（一）

❖ SQL 基础一考试指南：在 SQL 中使用子查询（二）

SQL 子查询的类型

本节介绍了单行和多行查询、关联子查询的定义以及执行查询的顺序和步骤。在第二篇文章中还列出一些更加复杂的子查询的相关练习。

❖ [SQL 基础一考试指南：SQL 子查询类型（一）](#)

❖ [SQL 基础一考试指南：SQL 子查询类型（二）](#)

编写单行和多行的子查询

子查询是一个嵌套在 SELECT、INSERT、UPDATE 或 DELETE 语句，或者另一个子查询里的查询。本文选自《OCA Oracle 数据库 11g：SQL 基础一考试指南 Guide》，本节将对在查询中哪些地方可以使用子查询和哪些子句是必须的以及哪些又是可以选择的进行阐述。阅读本文，以便更好的了解子查询的定义。

❖ [SQL 基础一考试指南：编写单行或多行子查询（一）](#)

❖ [SQL 基础一考试指南：编写单行或多行子查询（二）](#)

SQL 基础一考试指南：定义 SQL 子查询

子查询是一个嵌套在 SELECT、INSERT、UPDATE 或 DELETE 语句，或者另一个子查询里的查询。一个子查询可以返回一行或多行结果给父查询。标量子查询是只返回一个值的查询：只有一个字段的一行记录。标量子查询可以在 SQL 语句的大多数位置，其中你可以使用一个表达式或一个实际值。

查询中可能会用到子查询的有如下几个位置：

- 用于字段映射的 SELECT 列
- 在 FROM 子句中
- 在 WHERE 子句中
- 在 HAVING 子句中

子查询经常是作为内查询使用，而它所在的查询语句则称为外查询。这个专业术语没有任何的问题，只是它可能让人误解为只能有内和外两个级别的查询。事实上，Oracle 实现的子查询并没有硬性限制查询嵌套级数：FROM 子句中的语句中的嵌套深度没有限制，而 WHERE 子句上的嵌套上限为 255。

子查询可以使用任何一种常用的选择和映射子句。而下面就是必须的子句：

- SELECT
- FROM

下面是可选的子句：

- WHERE
- GROUP BY
- HAVING

SQL 语句中的子查询必须在调用它的父查询之前执行，这样子查询的结果才可以传递给父查询。

练习 8-1 子查询的类型

在这个练习中，你将要编写代码来演示可以使用子查询的位置。你可以使用 SQL*Plus 或 SQL Developer 来编写代码。所有查询在连接到 HR Schema 后都应该能执行通过。

1、作为 HR 登陆你的数据库

2、作为 HR 登陆你的数据库

```
select sysdate Today,  
(select count(*) from departments) Dept_count,  
(select count(*) from employees) Emp_count from dual;
```

3、编写一个查询，确定所有是经理的员工。这需要在 WHERE 子句中使用子查询来选择所有以 MANAGER_ID 字段出现在 EMPLOYEE_ID 字段中的员工：

```
select last_name from employees where  
(employee_id in (select manager_id from employees));  
ch08.indd 359 4/28/08 10:48:49 AM
```

4、编写一个查询，确定每一个国家的最高薪水。这需要在 FROM 子句中使用子查询：

```
select max(salary), country_id from
```

```
(select salary, department_id, location_id, country_id from  
employees natural join departments natural join locations) group by country_id  
;
```

(作者: John Watson, Roopesh Ramklass 译者: 曾少宁/陈柳 来源: TT 中国)

SQL 基础一考试指南：在 SQL 中使用子查询（一）

SQL 中有很多地方可以用到子查询的。本章选自《OCA Oracle Database 11g: SQL Fundamentals I Exam Guide》，将详细阐述如何在 SQL 中使用子查询，以及子查询所能解决的问题，如星型转换、比较、从 SELECT 生成表，等等。

下面是一些你可能需要将一个查询的结果作为另一个查询的输入的情况：

用子查询结果集比较

哪些员工的薪水是少于平均薪水值？这问题可以通过两个语句来回答，或者使用一个带子查询的语句。下面是这两种方法的例子：

```
select avg(salary) from employees;  
select last_name from employees where salary < result_of_previous_query ;
```

另一个就是使用一个带子查询的语句：

```
select last_name from employees  
where salary < (select avg(salary)from employees);
```

在这个例子中，一个子查询用于替代父查询的 WHERE 子句的取值：它返回一个值，然后这个值与父查询接收的记录行进行比较。

这个子查询可能返回多行记录。比如，你可以使用下面的语句查找所有拥有一个或多个员工的部门：

```
select department_name from departments where department_id in
(select distinct(department_id) from employees);
```

在上一个例子中，子查询替代了连接的使用。相同的结果可以使用下面的语句实现：

```
select department_name from departments inner join employees
on employees.department_id = departments.department_id
group by department_name;
```

如果子查询能返回多于一行的记录，那么比较操作符必须是可以接受多个值的。这些操作符包括 IN、NOT IN、ANY 和 ALL。如果比较操作符是 EQUAL、GREATER_THAN 或 LESS_THAN（它们只接受单个值），父查询就会出错。

星型转换

用子查询来替代连接操作的一个扩展是使用星型转换，这常用在数据仓库应用中。设想在一个很大的记录销售的表中，每一个卖单都标记为一个买家通过一个特定渠道购买的一个特定的产品。这些属性都通过标记码区分，它们是作为一个相应表的外键使用的，这些表的每一行记录描述了产品、买家和销售渠道。为了确定在德国通过 Internet 销售一空的所有书本记录，我们可以执行以下的查询：

```
select ... from sales s, products p, buyers b, channels c
where s.prod_code=p.prod_code
and s.buy_code=b.buy_code
and s.chan_code=c.chan_code
and p.product='Books'
```



```
and b.country='Germany'  
and c.channel='Internet';
```

这个查询使用 WHERE 子句来连接这些表，然后再过滤结果。下面是一个可以产生相同结果的替代查询：

```
select ... from sales  
where prod_code in (select prod_code from products where product='Books')  
and buy_code in (select buy_code from buyers where country='Germany')  
and chan_code in (select chan_code from channels where channel='Internet');
```

这里第一个语句使用了星型转换进行重写。这样写除了使代码变得更有美感（大多数 SQL 开发者都会有想同审美观点）之外，还有技术原因来说明为什么数据会更有效率地执行修改源查询代码。此外，星型查询也更容易维护；我们可以非常容易地增加一些查询分支或用一系列值来替换一些字面值（'Books', 'Germany', and 'Internet'）。

(作者: John Watson, Roopesh Ramklass 译者: 曾少宁 / 陈柳 来源: TT 中国)

SQL 基础一考试指南：在 SQL 中使用子查询（二）

生成一个表给 SELECT 子句

子查询也可以用在 FROM 子句中，这里它们有时是被作为内嵌视图使用。我们可以设想 HR Schema 上的另一个问题：员工被分配到一个部门，而部门有一个位置。每一个位置是在一个国家中。即便他们为不同的部门工作，你又如何找出一个国家中员工的平均工资呢？我们可以这样做：

```
select avg(salary), country_id from
(select salary, department_id, location_id, country_id from
employees natural join departments natural join locations)
group by country_id;
```

这个子查询构造了一个表，表里有每一个员工的工资和他的部门所在国家。然后父查询接收这个表，计算出 SALARY 的平均值，并将结果按 COUNTRY_ID 分组。

生成映射值

子查询的第三个使用位置是查询的 SELECT 列表。你如何确定最高工资和最高销售提成，以及根据最高工资员工应获取最高销售提成的原则的最大提成比例是多少？我们可以用两个子查询来解决：

```
select
(select max(salary) from employees) *
```

```
(select max(commission_pct) from employees)
/ 100
from dual;
```

这个用法中，用于映射字段的 SELECT 列表是由子查询的结果生成的。这种方式的子查询必须是标量的，否则父查询会执行出错。

生成传递给 DML 语句的记录行

DML 语句会在第 10 章中详细介绍，下面是一些例子：

```
insert into sales_hist select * from sales where date > sysdate-1;
update employees set salary = (select avg(salary) from employees);
delete from departments
where department_id not in (select department_id from employees);
```

第一个例子使用一个子查询来确定一个表中将要插入到另一个表中的记录行。第二个例子使用一个子查询来计算所有员工的平均工资并将这个值（标量数）传递给一个更新语句。第三个例子使用一个子查询来获取所有正在使用的 DEPARTMENT_ID 并将这个列表传递给 DELETE 命令，这将会删除所有没有被使用的部门。

需要注意的是在 VALUES 子句使用子查询是不合法的；而下面是合法的：

```
insert into dates select sysdate from dual;
```

但这个是不合法的：

```
insert into dates (date_col) values (select sysdate from dual);
```

练习 8-2 一些更复杂的子查询

在这个练习中，你将编写一些更复杂一些的子查询。你可以使用 SQL*Plus 或 SQL Developer 进行代码编写。所有这些查询都应该能在 HR Schema 上执行。

1、以用户 HR 登录到你的数据库。

2、编写一个查询，找出所有工作部门在 United Kingdom 的员工。这将需要三个层次的嵌套子查询：

```
select last_name from employees where department_id in  
(select department_id from departments  
where location_id in  
(select location_id from locations where country_id =  
(select country_id from countries  
where country_name='United Kingdom')  
));
```

3、通过单独运行子查询来检查第 2 步中得到的结果是正确的。首先，找出 United Kingdom 的 COUNTRY_ID：

```
select country_id from countries where country_name='United Kingdom';
```

结果将会是 UK。然后找出相应的位置：

```
select location_id from locations where country_id = 'UK';
```

返回的 LOCATION_ID 将会是 2400, 2500 和 2600。然后找出在这些位置的部门的 DEPARTMENT_ID:

```
select department_id from departments where location_id in (2400, 2500, 2600);
```

结果将会是两个部门: 40 和 80。最后, 找出相应的员工:

```
select last_name from employees where department_id in (40, 80);
```

4、编写一个查询, 找出所有工资超过平均水平并且工作在 IT 部门的员工。这将需要 2 个子查询, 但不是嵌套的:

```
select last_name from employees
where department_id in
(select department_id from departments where department_name like 'IT%')
and salary > (select avg(salary) from employees);
```

(作者: John Watson, Roopesh Ramklass 译者: 曾少宁 / 陈柳 来源: TT 中国)

SQL 基础一考试指南：SQL 子查询类型（一）

单行和多行子查询

单行子查询只返回一行记录。其中的一个特例是标量子查询，它返回只有一个字段的一行记录。事实上标量子查询可用于任意形式的查询中（并且往往是非常有用的），可以是常量值、常量或者表达式。多行子查询返回多行记录集。这些查询产生的结果集通常将传递给 DML 或 SELECT 语句进行进一步的处理。但单行子查询和多行子查询都会在父查询执行前进行一次值比较。单行和多行子查询可以用在父查询的 WHERE 和 HAVING 子句中，但对于合法的比较操作符有一些限制。如果比较操作符是下表中的任意一个，子查询就必须有一个单行子查询：

Symbol	Meaning
=	equal
>	greater than
>=	greater than or equal
<	less than
<=	less than or equal
<>	not equal
!=	not equal

如果子查询使用上表中的任意一个操作符而又返回多行结果记录，该查询会出错。下表中的操作符则可用于多行子查询：

Symbol	Meaning
--------	---------

IN	equal to any member in a list
NOT IN	not equal to any member in a list
ANY	returns rows that match any value on a list
ALL	returns rows that match all the values in a list

关联子查询

关联子查询是一个比单行和多行子查询更复杂的执行方法，它的功能也更加强大。如果一个子查询引用了父查询的字段，这样它的结果也将依赖于父查询结果。这样就不可能在父查询值比较前进行子查询值比较。考虑下面这个语句，它列出所有薪水在平均薪水值以下的员工：

```
select last_name from employees
where salary < (select avg(salary) from employees);
```

单行子查询只需要执行一次，而且其结果会代入父查询中。但是，目前要解决的是的语句是列出所有部门中薪水在平均薪水值以下的员工。因此，子查询必须对每个员工进行探究以便确定部门中的平均薪水值，这样，员工部门的代码就必须传到子查询中。可以操作如下：

```
select p.last_name, p.department_id from employees p
where p.salary < (select avg(s.salary) from employees s
where s.department_id=p.department_id);
```

在这个例子中，子查询引用了一个字段：p.department_id，它是定义在父查询的select 列表中的。这表示，不是只对子查询进行一次值比较，而是对父查询中的每一行记

录在子查询中进行比较。在执行这个查询时，Oracle 将会检查 EMPLOYEES 表中的每一行记录，并使用当前员工记录行的 DEPARTMENT_ID 来执行子查询。

该查询的执行顺序是这样的：

1. 从 EMPLOYEES 表的第一行记录开始查询。
2. 读取当前记录行的 DEPARTMENT_ID 和 SALARY 字段值。
3. 对第 2 步得到的 DEPARTMENT_ID 执行子查询。
4. 比较第 3 步子查询结果与第 2 步得到的 SALARY 值，如果 SALARY 小于子查询结果，则返回该行记录。
5. 继续查询 EMPLOYEES 表的下一行记录。
6. 重复第 2 步。

单行或多行子查询都会在外查询值比较前进行一次值比较；而关联子查询则必须在外查询的每一行记录上进行一次值比较。如果使用合适的操作符，一个关联子查询可能是单行或者多行的。

(作者: John Watson, Roopesh Ramklass 译者: 曾少宁/陈柳 来源: TT 中国)

SQL 基础一考试指南：SQL 子查询类型（二）

练习 8-3 研究不同类型的子查询

在这个练习中，我们将演示可能发生在不同类型的子查询中的问题。我们使用 SQL*Plus 或 SQL Developer 来编写代码。所有的查询都应该能在 HR Schema 上执行：假定 EMPLOYEES 已经有一些标准记录集。

1、以用户 HR 登录到数据库。

2、编写一个查询，判断谁的薪水比 Mr. Tobias 高：

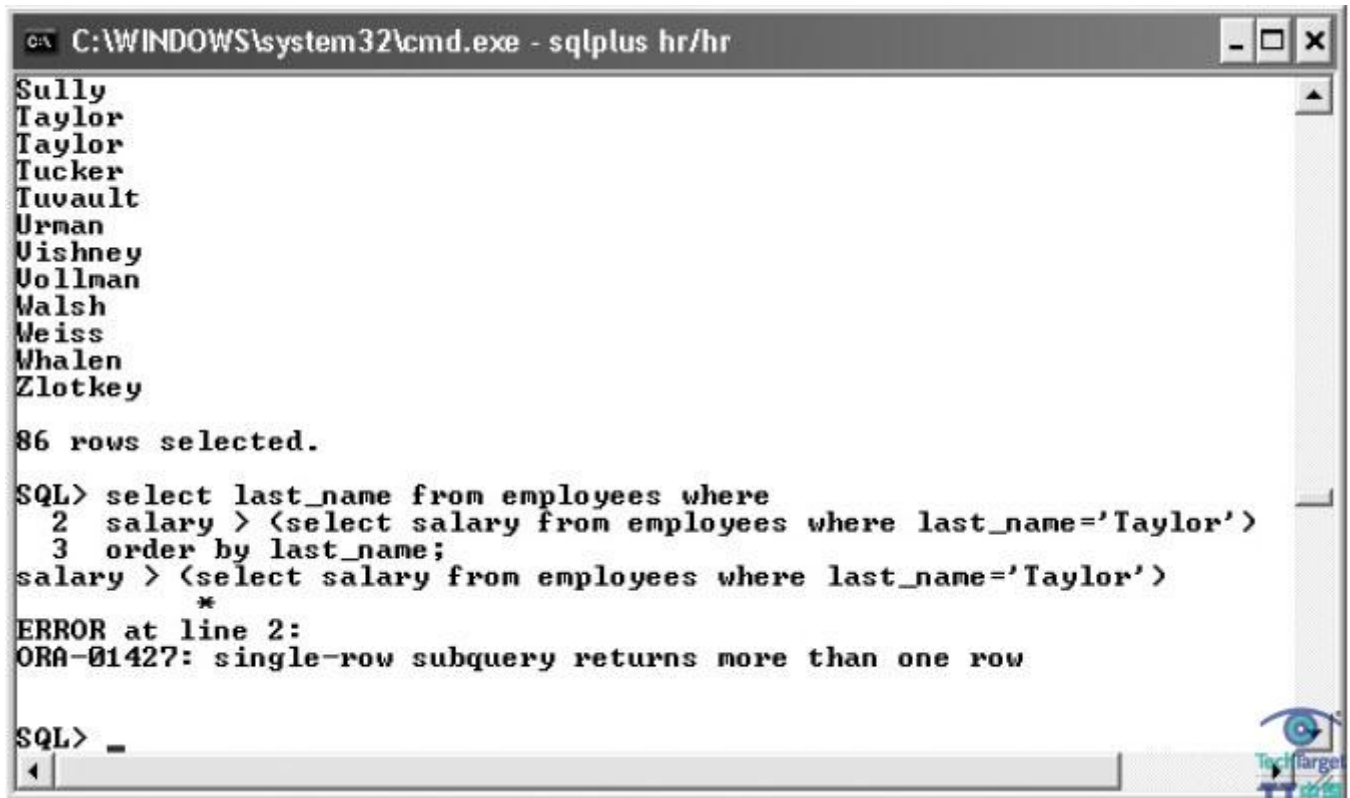
```
select last_name from employees where
salary > (select salary from employees where last_name='Tobias')
order by last_name;
```

这将会返回 86 个名字，并以字母顺序排序。

3、编写一个查询，判断谁的薪水比 Mr. Taylor 高：

```
select last_name from employees where
salary > (select salary from employees where last_name='Taylor')
order by last_name;
```

这次查询会失败并抛出错误：“ORA-01427: single-row subquery returns more than one row”。下图是在 SQL*Plus 中显示的第 2 步操作的最后几行输出，以及第 3 步出现的错误。



我们要判断为什么第 2 步成功了而第 3 步却失败了。答案就在数据的状态中：

```
select count(last_name) from employees where last_name='Tobias';
select count(last_name) from employees where last_name='Taylor';
```

第 2 和第 3 步查询中使用的“大于 (>)”操作符要求使用单行子查询，但是第 3 步使用的子查询可能会根据使用的不同查询规则而返回多行记录。

4、我们要修改第 2 步和第 3 步中的代码，使它能在任何 LAST_NAME 值查询中都执行成功。下面是两种可选的方法：第一个是使用一个能处理多行子查询的操作符；另一个是使用一个总是返回单行记录的子查询。

方法一：

```
select last_name from employees where  
salary > all (select salary from employees where last_name='Taylor')  
order by last_name;
```

方法二：

```
select last_name from employees where  
salary > (select max(salary) from employees where last_name='Taylor')  
order by last_name;
```

SCENARIO & SOLUTION	
如何设计最好的子查询，使它们不会执行失败并出现错误： “ORA-01427: single-row subquery returns more than one row” ?	可以有 2 种常用方法：一是使用聚集（Aggregation），它可以将你得到的多行记录减为一行；二是使用 IN、ANY 或 ALL 操作符，这些操作符允许与返回的多行记录进行比较。但它们都不是最好的方法；最好的方法是要用主键来判断

	所返回的记录行，而不是用非唯一键。
有时我们会需要考虑选择使用子查询还是使用其它替代技术，如星号（*）通配符。那么哪一种才是更好的选择呢？	这个选择与条件有关。在数据库中，通常不同的技术会有不同的执行方法。根据它所配置的实例、数据库和数据结构，各自的性能也会有所不同。如果面对这样的选择，查询语句应该要进行调优分析。DBA 可以对此提供有用的建议。

(作者: John Watson, Roopesh Ramklass 译者: 曾少宁/陈柳 来源: TT 中国)

SQL 基础一考试指南：编写单行或多行子查询（一）

子查询是一个嵌套在 SELECT、INSERT、UPDATE 或 DELETE 语句，或者另一个子查询里的查询。本文选自《OCA Oracle Database 11g: SQL Fundamentals I Exam Guide》，本文将对在查询中哪些地方可以使用子查询和哪些子句是必须的以及哪些又是可以选择的进行阐述。阅读本文，以便更好的了解子查询的定义。

下面是单行和多行子查询的例子。它们是基于演示的 HR Schema 编写的。你要怎么找到那些经理办公地点在 United Kingdom 的员工呢？这个查询需求可以用多行子查询实现：

```
select last_name from employees
where manager_id in
(select employee_id from employees where department_id in
(select department_id from departments where location_id in
(select location_id from locations where country_id='UK')));
```

前面的查询例子内嵌了 3 级子查询。我们注意到这些子查询使用了 IN 操作符，因为它对应的子查询可能返回多行数据。

如果你要查找平均薪水最高的工作，可以通过一个单行子查询来实现：

```
select job_title from jobs natural join employees group by job_title
having avg(salary) =
(select max(avg(salary)) from employees group by job_id);
```

这个子查询返回单个值：各部门中的最高平均薪水值。这里是可以使用“=”操作符的，因为 MAX 方法保证了只返回一个值。

这里也可以使用 ANY 和 ALL 操作符，但它们的功能可以被其他一些常用的使用聚集的操作符所替代。比如，下面这两个语句，在部门的 80 位员工中，薪水比其他人高的将被返回，最后得出的是一个均等的结果集。

```
select last_name from employees where salary > all
(select salary from employees where department_id=80);
select last_name from employees where salary >
(select max(salary) from employees where department_id=80);
```

下表总结了 ANY 和 ALL 的用法：

Operator	Meaning
< ANY	less than the highest 小于最大值
> ANY	not equal to any member in a list 不等于列表中任一个值
ANY	more than the lowest 大于最小值
= ANY	equivalent to IN 与 IN 相同
> ALL	more than the highest 大于最大值

< ALL	less than the lowest 小于最小值
-------	-----------------------------------

(作者: John Watson, Roopesh Ramklass 译者: 曾少宁/陈柳 来源: TT 中国)

SQL 基础一考试指南：编写单行或多行子查询（二）

练习 8-4 编写可靠的且用户友好的查询

在本练习中，我们要开发一个多行子查询，它将提示用户输入。我们可以使用 SQL*Plus 或 SQL Developer 进行开发。在连接到 HR Schema 后，所有的查询都必须能成功执行。其中我们假定数据库表中已经有了一些记录集。

- 1、以用户 HR 登录到你的数据库。
- 2、设计一个查询，提示用户输入部门名称，然后列出该部门的所有员工名：

```
select last_name from employees where department_id =  
(select department_id from departments where department_name = '&Department_name');
```

- 3、运行步骤 2 中的查询 3 次，在提示符中分别输入下面的值：

第一次, Executive;

第二次, executive;

第三次, Executiv。

4、从步骤 3 的运行结果我们可以看到：第一次执行成功，因为我们输入了一个准确匹配的值，但其它的都失败了。下面我们修改一下查询，使它更加用户友好，这样它可以处理大小写或拼写不同的问题：


```
select last_name from employees where department_id =  
(select department_id from departments  
where upper(department_name) like upper('%&Department_name%'));
```

5、运行步骤 4 中的查询 3 次，其中仍然使用步骤 3 中的相同数值。这次，所有的查询都能成功执行。

6、再次运行步骤 4 的查询，但这次输入值“Pu”。这次查询会失败，并返回一个错误“ORA-01427: single-row subquery returns more than one row”，这是因为我们为了用户友好所做的修改会接收用户的随意输入，从而无法保证它是单行子查询。这里的“Pu”就匹配到了两个部门。

7、我们继续修改该查询，使它能够处理 ORA-01427 错误，并调整输出结果以使结果更加清晰：

```
select last_name, department_name from employees join departments  
on employees.department_id = departments.department_id  
where departments.department_id in  
(select department_id from departments  
where upper(department_name) like upper('%&Department_name%'));
```

8、下面是最后一步执行的结果：最终的查询代码已经很接近既准确又用户友好的理想结果了：

INSIDE THE EXAM 考点

使用子查询

子查询有 3 种形式：单行、多行和关联的。单行子查询的一个特殊形式是标量子查询，它只返回一个值。这是一种单行单字段子查询。第一次的 SQL OCP 考试只要求标量子查询和单字段多行子查询，关联子查询和多字段子查询在这里不太可能考到，但会测试你对他们的一般理解。

在 WHERE 子句中使用子查询时，你必须知道紧跟单行子查询和多行子查询的操作符用法。

(作者: John Watson, Roopesh Ramklass 译者: 曾少宁/陈柳 来源: TT 中国)