

可重复读简介



```
COUNT((SELECT sec_id
      FROM teaches t
      WHERE i.ID = t.ID
      ))AS number_of_section
```

我SELECT只选择了一个属性，返回的不是单一的列吗？



是的，你的子查询确实只返回了一个属性列 `sec_id`，所以从表面上看，你返回的是**单一列**，但问题出在 SQL 的**子查询上下文**和 `COUNT()` 函数的用法。

虽然子查询只返回了一列（`sec_id`），但它**可能返回多行数据**。而 `COUNT()` 函数在 SQL 中并不直接处理多行的子查询结果，因为它期望的参数是单个列的值或一个表达式，而不是子查询返回的多行数据。

具体原因：

- 子查询中的多行问题：

sql

```
(SELECT sec_id FROM teaches t WHERE i.ID = t.ID)
```

对于每个教师 `i.ID`，这个子查询可能返回多个 `sec_id`，即教师教授的多个课程。例如，如果教师 `John` 教了 3 门课程，这个子查询会返回多个 `sec_id`，类似于：

```
101
102
103
```

尽管你只选择了一列，但返回的是多行数据。`COUNT()` 无法直接处理这种多行结果，因为它期望的是某一列中的单个值。

- **COUNT() 的行为：** `COUNT()` 函数期望一个列的值或单个表达式来计算结果集中的行数。它不能处理一个返回多行的子查询。你的子查询返回的是多个 `sec_id`，而不是单个值或一个简单的列表表达式。

如何修正：

要修正这个问题，你需要在子查询中**直接统计**与每个教师相关的课程数，而不是返回多个 `sec_id`。可以通过在子查询中使用 `COUNT()` 函数来实现。

改正后的查询：

你可以将子查询改成直接统计行数，而不是返回多个 `sec_id`：

```
sql

SELECT
  i.ID AS instructor,
  i.name,
  (SELECT COUNT(t.sec_id)
   FROM teaches t
   WHERE i.ID = t.ID) AS number_of_section
FROM instructor i;
```

解释：

1. **子查询中的 `COUNT(t.sec_id)`**：这个子查询不会返回多个 `sec_id`，而是直接返回教师教授的课程数量。对于每个教师 `i.ID`，子查询计算与之相关的 `sec_id` 数量，即教师教授的课程数量。
2. **外部查询**：外部查询通过 `SELECT` 返回教师的 `ID` 和 `name`，以及每个教师教授的课程数量。

示例：

假设我们有如下数据：

`instructor` 表：

ID	name
1	John
2	Alice

`teaches` 表：

ID	sec_id
1	101
1	102
2	201

在这个例子中，`John` 教了 2 门课程，`Alice` 教了 1 门课程。上述查询会返回：

instructor	name	number_of_section
1	John	2
2	Alice	1

总结：

虽然你的子查询只返回了 `sec_id` 这一列，但由于它返回的是**多行**，而 `COUNT()` 函数无法直接处理多行的子查询。因此，必须在子查询内部使用 `COUNT()` 来统计教师教授的课程数量，然后将结果返回给外部查询。





单个值或一个简单的列表表达式

举例说明



为了更好地说明 `COUNT()` 期望的是单个值或简单的列表表达式，而不是一个子查询返回的多行数据，让我们通过一个实际例子来演示问题。

1. 表结构和数据

假设我们有以下两张表：`instructor` 表和 `teaches` 表。

`instructor` 表（教师表）

ID	name
1	John
2	Alice
3	Bob

`teaches` 表（课程表）

ID	sec_id
1	101
1	102
2	201
2	202
3	301

2. 子查询问题的例子

现在假设我们要统计每位教师教授的课程数，并使用子查询的方式来实现。最初我们可能会写出如下查询：

sql

```
SELECT
  i.ID AS instructor,
  i.name,
  COUNT((SELECT t.sec_id
          FROM teaches t
          WHERE i.ID = t.ID)) AS number_of_section
FROM instructor i;
```

错误原因：

- 对于教师 `John` (`ID = 1`)，子查询 `SELECT t.sec_id FROM teaches t WHERE i.ID = t.ID` 返回了两个值：`101` 和 `102`，因为 `John` 教授了两门课程。
- 这个子查询返回了**多行数据**，而 `COUNT()` 函数无法直接处理多行结果。
- `COUNT()` 期望的是一个**单列或单值**，而不是子查询返回的多行结果。这会导致 SQL 引擎无法理解如何统计这些多行的 `sec_id`，从而报错。

3. 正确做法：在子查询中使用 `COUNT()`

为了解决这个问题，我们需要在子查询中直接统计与每位教师相关的课程数，而不是返回多个 `sec_id`。

改正后的 SQL 语句：

```
sql

SELECT
  i.ID AS instructor,
  i.name,
  (SELECT COUNT(t.sec_id)
   FROM teaches t
   WHERE t.ID = i.ID) AS number_of_section
FROM instructor i;
```

4. 运行过程：示例解释

让我们一步一步看这个查询的执行过程：

- 对于教师 `John` (`ID = 1`)：
 - 子查询 `SELECT COUNT(t.sec_id) FROM teaches t WHERE t.ID = 1` 会返回 `2`，因为 `John` 教授了两门课程，`sec_id` 为 `101` 和 `102`。
 - 最终结果为 `number_of_section = 2`。
- 对于教师 `Alice` (`ID = 2`)：
 - 子查询 `SELECT COUNT(t.sec_id) FROM teaches t WHERE t.ID = 2` 会返回 `2`，因为 `Alice` 教授了两门课程，`sec_id` 为 `201` 和 `202`。
 - 最终结果为 `number_of_section = 2`。
- 对于教师 `Bob` (`ID = 3`)：
 - 子查询 `SELECT COUNT(t.sec_id) FROM teaches t WHERE t.ID = 3` 会返回 `1`，因为 `Bob` 教授了一个课程，`sec_id` 为 `301`。
 - 最终结果为 `number_of_section = 1`。

5. 最终结果

该查询会输出如下结果：

instructor	name	number_of_section
1	John	2
2	Alice	2

instructor	name	number_of_section
3	Bob	1

总结：

- **错误的例子：**使用 `COUNT((SELECT sec_id ...))` 会导致错误，因为子查询返回了多行数据，而 `COUNT()` 期望的是单个值。
- **正确的做法：**在子查询中直接使用 `COUNT()` 来统计每个教师的课程数，从而返回单个结果值给外部查询。

这个例子清晰展示了 SQL 中 `COUNT()` 如何期望处理单列或单值，而不是多行的子查询返回结果。



* **对于教师 `John` (`ID = 1`) **：* 子查询 `SELECT COUNT(t.sec_id) FROM teaches t WHERE t.ID = 1` 会返回 `2`，因为 `John` 教授了两个课程，`sec_id` 为 `101` 和 `102`。* 最终结果为 `number_of_section = 2`。
在这个例子中COUNT处理的不也是101和102两个结果吗？和我的做法有什么区别？



你提到的核心问题是：在子查询中，`COUNT()` 处理的也是 `sec_id`，比如 `101` 和 `102` 这两个结果，为什么子查询的 `COUNT()` 可以工作，而你原本的写法不行？这是一个很好的问题，涉及到 SQL 处理**多行结果**与**多行子查询**的区别。

关键区别：

1. 子查询返回多行 vs. 子查询返回单个值

- **你的做法：**

```
sql
```

```
COUNT((SELECT sec_id FROM teaches t WHERE i.ID = t.ID))
```

这里的子查询返回的是多个 `sec_id`（如 `101`、`102`），这是一组**多行数据**。`COUNT()` 不能直接统计一个返回**多行结果**的子查询，因为 `COUNT()` 期望一个**单一的表达式**或列值（即，它不能在外层直接统计整个子查询的结果）。换句话说，`COUNT()` 需要统计的是某个范围内的单个列或值的数量，而不能处理子查询返回的多行多列数据。

- **改进后的做法：**

```
sql
```

```
(SELECT COUNT(t.sec_id) FROM teaches t WHERE i.ID = t.ID)
```

这里的子查询内部已经使用了 `COUNT()`，并且返回的是**单一数值**（例如，对于 `John` 来说是 `2`）。这个值代表教师 `John` 教授的课程数量。这样，外部查询就能够使用这个**单个值**，而不是试图处理多行数据。

2. `COUNT()` 的工作方式

- `COUNT()` 是一个**聚合函数**，它的作用是在某个**指定的列或表达式**上统计不为 `NULL` 的行数。
- 在你的例子中，如果你想统计教师教授的课程数量，你需要将 `COUNT()` 的操作应用在每个教师教授的课程数量上，而不是试图让 `COUNT()` 去处理子查询返回的多行数据。

示例：

假设我们有如下数据：

`teaches` 表：

ID	sec_id
1	101
1	102
2	201
3	301

- 你的做法：
 - 子查询 `SELECT sec_id FROM teaches t WHERE i.ID = t.ID` 返回了 `sec_id` 为 `101` 和 `102`。
 - `COUNT((SELECT sec_id FROM teaches t WHERE i.ID = t.ID))` 试图让 `COUNT()` 处理这两个值，但 `COUNT()` 不知道如何处理一个返回多行数据的子查询，因为它期望的是一列或一个值。
- 改进后的做法：
 - `SELECT COUNT(t.sec_id) FROM teaches t WHERE t.ID = i.ID` 会直接返回教师教授课程的数量，而不是试图返回多个 `sec_id`。在这里，`COUNT()` 在子查询内部完成了统计工作，直接返回一个数值（例如 `2`），这个数值表示教师 `John` 教了两门课。
 - 这样，外层查询可以将这个单个数值（比如 `2`）作为 `number_of_section` 返回给用户。

总结：

1. 你的做法的问题是：`COUNT()` 不能直接应用在返回多行的子查询上，因为 `COUNT()` 期望一个单列或单值。
2. 正确的做法是在子查询中直接使用 `COUNT()`，让子查询返回一个单一的值，然后外层查询可以使用这个结果。

这就是为什么在你的改进版本中，`COUNT()` 可以正常工作，而在原始版本中会导致错误。