<http://www.cnblogs.com/xiaoqiangzhou/p/3919270.html>

想必大家都用过sql中的in语句吧，我这里描述下我遇到的一种in语句问题，并总结一些给大家分享下，不对的地方还希望大虾指点下。

问题描述：**IN子查询时，子查询中字段在表中不存在时语句却不报错**

平常工作中我们会经常用到in或者not in语句来处理一些问题，比如通过in子查询语句检索符合或者不符合条件的集合结果、批量删除、修改一些符合条件或者不符合条件的集合。但大家是否注意到当子查询中字段名在表中不存在时语句不会报错（会返父查询中所有的结果），如果大家不注意这点，在使用in语句进行批量删除时就可能悲剧了。下面用实例来说明

1 --一个简单的in查询语句

2 select \* from tuser where userno in(select userno from filter\_barcode)

上面这条语句子查询里，userno 并不存在filter\_barcode表中，但是整个语句确能正常执行（执行子查询的话会报字段不存在的提示），而且返回的是tuser表中所有的结果集。如果大家不注意这种情况，一旦不是用来查询，是用来删除的，那整个表数据就被不知不觉给删除了。

但是当将子查询中userno字段改成一个即不再tuser也不再filter\_barcode表中的字段，那语句就会报错

select \* from tuser where userno in(select useno from filter\_barcode)

Msg **207**, Level **16**, State **1**, Line **1**

列名 'useno' 无效。

原因：**原来是在不使用表别名的前提下如果in子查询里字段在内表找不到就会去引用外表的。**

现实情况下子查询引用外层查询的列是正常的，只不过一般不在in子查询中引用外层查询的列。  
但是在exists,not exists子查询中用得比较多，

select a.\* from tuser a where exists

(select top **1** \* from filter\_barcode b where a.userno=b.userno)

--执行上面这语句就会提示

Msg **207**, Level **16**, State **1**, Line **1**

列名 'userno' 无效。

以下四条是我从其他地方看到的，贴出来给大家参考

1.子表引用父表列，而自己没有，在子表有数据的情况下，返回所有非空键的父表记录，子表为空，则结果无  
2.子表引用父表属性，只有最外层子查询才能引用  
3.有前缀标识，按前缀，如果子表父表前缀一样，按4的规则  
4.如果无前缀标识唯一性，子查询表也有此字段，那么以局部子查询为准  
    如果前缀一样，子查询存在此字段，则以子查询表为准，否则以父表的为准

总结；为了避免这种问题有几个方法供大家参考

1、当需要用到in子查询时，先执行下in里面的子查询语句是否有误，如果误则进行相应修改

2、使用表前缀（别名）才是硬道理，例如

select \* from tuser a where a.userno in(select b.userno from filter\_barcode b)

Msg **207**, Level **16**, State **1**, Line **1**

列名 'userno' 无效。

--这样就会进行报错，而不会返回tuser所有的数据

3、使用exists语句来代替in语句

select a.\* from tuser a where exists

(select top **1** \* from filter\_barcode b where a.userno=b.userno)

Msg **207**, Level **16**, State **1**, Line **2**

列名 'userno' 无效。

关于exists和in的区别用法这里就不在讲述，大家可以查询相关资料

对于in 和 exists的区别: 如果子查询得出的结果集记录较少，主查询中的表较大且又有索引时应该用in, 反之如果外层的主查询记录较少，子查询中的表大，又有索引时使用exists。其实我们区分in和exists主要是造成了驱动顺序的改变（这是性能变化的关键），如果是exists，那么以外层表为驱动表，先被访问，如果是IN，那么先执行子查询，所以我们会以驱动表的快速返回为目标，那么就会考虑到索引及结果集的关系了 ，另外IN是不对NULL进行处理。