[sql语法：inner join on, left join on, right join on详细使用方法](http://blog.csdn.net/steryzone/article/details/4997060)

标签： [join](http://www.csdn.net/tag/join)[sql](http://www.csdn.net/tag/sql)[table](http://www.csdn.net/tag/table)[null](http://www.csdn.net/tag/null)

2009-12-13 12:03 185263人阅读 [评论](http://blog.csdn.net/steryzone/article/details/4997060/#comments)(42) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/steryzone/article/details/4997060/#report)

版权声明：本文为博主原创文章，未经博主允许不得转载。

inner join(等值连接) 只返回两个表中联结字段相等的行

left join(左联接) 返回包括左表中的所有记录和右表中联结字段相等的记录

right join(右联接) 返回包括右表中的所有记录和左表中联结字段相等的记录

INNER JOIN 语法：

INNER JOIN 连接两个数据表的用法：

SELECT \* FROM 表1 INNER JOIN 表2 ON 表1.字段号=表2.字段号

INNER JOIN 连接三个数据表的用法：

SELECT \* FROM (表1 INNER JOIN 表2 ON 表1.字段号=表2.字段号) INNER JOIN 表3 ON 表1.字段号=表3.字段号

INNER JOIN 连接四个数据表的用法：

SELECT \* FROM ((表1 INNER JOIN 表2 ON 表1.字段号=表2.字段号) INNER JOIN 表3 ON 表1.字段号=表3.字段号) INNER JOIN

表4 ON Member.字段号=表4.字段号

INNER JOIN 连接五个数据表的用法：

SELECT \* FROM (((表1 INNER JOIN 表2 ON 表1.字段号=表2.字段号) INNER JOIN 表3 ON 表1.字段号=表3.字段号) INNER JOIN 表4 ON Member.字段号=表4.字段号) INNER JOIN 表5 ON Member.字段号=表5.字段号

连接六个数据表的用法：略，与上述联接方法类似，大家举一反三吧：）

注意事项：

在输入字母过程中，一定要用英文半角标点符号，单词之间留一半角空格；   
在建立数据表时，如果一个表与多个表联接，那么这一个表中的字段必须是“数字”数据类型，而多个表中的相同字段必须是主键，而且是“自动编号”数据类型。否则，很难联接成功。   
代码嵌套快速方法：如，想连接五个表，则只要在连接四个表的代码上加一个前后括号（前括号加在FROM的后面，后括号加在代码的末尾即可），然后在后括号后面继续添加“INNER JOIN 表名X ON 表1.字段号=表X.字段号”代码即可，这样就可以无限联接数据表了：）

**1.理论**

只要两个表的公共字段有匹配值，就将这两个表中的记录组合起来。

个人理解：以一个共同的字段求两个表中符合要求的交集，并将每个表符合要求的记录以共同的字段为牵引合并起来。

**语法**

select \* FROM table1 INNER JOIN table2 ON table1 . field1 compopr table2 . field2

INNER JOIN 操作包含以下部分：

|  |  |
| --- | --- |
| 部分 | 说明 |
| table1, table2 | 要组合其中的记录的表的名称。 |
| field1，field2 | 要联接的字段的名称。如果它们不是数字，则这些字段的数据类型必须相同，并且包含同类数据，但是，它们不必具有相同的名称。 |
| compopr | 任何关系比较运算符：“=”、“<”、“>”、“<=”、“>=”或者“<>”。 |

**说明**

可以在任何 FROM 子句中使用 INNER JOIN 操作。这是最常用的联接类型。只要两个表的公共字段上存在相匹配的值，Inner 联接就会组合这些表中的记录。

可以将 INNER JOIN 用于 Departments 及 Employees 表，以选择出每个部门的所有雇员。而要选择所有部分（即使某些部门中并没有被分配雇员）或者所有雇员（即使某些雇员没有分配到任何部门），则可以通过 LEFT JOIN 或者 RIGHT JOIN 操作来创建外部联接。

如果试图联接包含备注或 OLE 对象数据的字段，将发生错误。

可以联接任何两个相似类型的数字字段。例如，可以联接自动编号和长整型字段，因为它们均是相似类型。然而，不能联接单精度型和双精度型类型字段。

下例展示了如何通过 CategoryID 字段联接 Categories 和 Products 表：

SELECT CategoryName, ProductName

FROM Categories INNER JOIN Products

ON Categories.CategoryID = Products.CategoryID;

在前面的示例中，CategoryID 是被联接字段，但是它不包含在查询输出中，因为它不包含在 SELECT 语句中。若要包含被联接字段，请在 SELECT 语句中包含该字段名，在本例中是指 Categories.CategoryID。

也可以在 JOIN 语句中链接多个 ON 子句，请使用如下语法：

SELECT fields  
FROM table1 INNER JOIN table2  
ON table1.field1 compopr table2.field1 AND  
ON table1.field2 compopr table2.field2 OR  
ON table1.field3 compopr table2.field3;

也可以通过如下语法嵌套 JOIN 语句：

SELECT fields  
FROM table1 INNER JOIN  
(table2 INNER JOIN [( ]table3  
[INNER JOIN [( ]tablex [INNER JOIN ...)]   
ON table3.field3 compopr tablex.fieldx)]  
ON table2.field2 compopr table3.field3)   
ON table1.field1 compopr table2.field2;

LEFT JOIN 或 RIGHT JOIN 可以嵌套在 INNER JOIN 之中，但是 INNER JOIN 不能嵌套于 LEFT JOIN 或 RIGHT JOIN 之中。

**2.操作实例**

表A记录如下：  
aID               aNum  
1                  a20050111  
2                  a20050112  
3                  a20050113  
4                  a20050114  
5                  a20050115

表B记录如下:  
bID               bName  
1                   2006032401  
2                  2006032402  
3                  2006032403  
4                  2006032404  
8                  2006032408

**实验如下:  
1.left join**

sql语句如下:   
select \* from A  
left join B   
on A.aID = B.bID

结果如下:  
aID               aNum                          bID                  bName  
1                   a20050111                1                      2006032401  
2                   a20050112                2                     2006032402  
3                   a20050113                3                     2006032403  
4                   a20050114                4                     2006032404  
5                   a20050115                NULL              NULL  
（所影响的行数为 5 行）

**结果说明**:  
               left join是以A表的记录为基础的,A可以看成左表,B可以看成右表,left join是以左表为准的.  
换句话说,左表(A)的记录将会全部表示出来,而右表(B)只会显示符合搜索条件的记录(例子中为: A.aID = B.bID).  
B表记录不足的地方均为NULL.

**2.right join**sql语句如下:   
select \* from A  
right join B   
on A.aID = B.bID  
结果如下:  
aID               aNum                          bID                  bName  
1                   a20050111                1                      2006032401  
2                   a20050112                2                     2006032402  
3                   a20050113                3                     2006032403  
4                   a20050114                4                     2006032404  
NULL           NULL                          8                     2006032408  
（所影响的行数为 5 行）  
**结果说明**:  
        仔细观察一下,就会发现,和left join的结果刚好相反,这次是以右表(B)为基础的,A表不足的地方用NULL填充.

**3.inner join**sql语句如下:   
select \* from A  
innerjoin B   
on A.aID = B.bID

结果如下:  
aID               aNum                          bID                  bName  
1                   a20050111                1                      2006032401  
2                   a20050112                2                     2006032402  
3                   a20050113                3                     2006032403  
4                   a20050114                4                     2006032404

**结果说明**:  
        很明显,这里只显示出了 A.aID = B.bID的记录.这说明inner join并不以谁为基础,它只显示符合条件的记录.  还有就是inner join 可以结合where语句来使用 如：   select \* from A innerjoin B on A.aID = B.bID where b.bname='2006032401' 这样的话 就只会放回一条数据了