窗体顶端

**关键字: 用户权限管理**

B/S系统中的权限比C/S中的更显的重要，C/S系统因为具有特殊的客户端，所以访问用户的权限检测可以通过客户端实现或通过客户端+服务器检测实现，而B/S中，浏览器是每一台计算机都已具备的，如果不建立一个完整的权限检测，那么一个“非法用户”很可能就能通过浏览器轻易访问到B/S系统中的所有功能。因此B/S业务系统都需要有一个或多个权限系统来实现访问权限检测，让经过授权的用户可以正常合法的使用已授权功能，而对那些未经授权的“非法用户”将会将他们彻底的“拒之门外”。下面就让我们一起了解一下如何设计可以满足大部分B/S系统中对用户功能权限控制的权限系统。

**需求陈述**

* **不同职责的人员，对于系统操作的权限应该是不同的。**优秀的业务系统，这是最基本的功能。

* **可以对“组”进行权限分配**。对于一个大企业的业务系统来说，如果要求管理员为其下员工逐一分配系统操作权限的话，是件耗时且不够方便的事情。所以，系统中就提出了对“组”进行操作的概念，将权限一致的人员编入同一组，然后对该组进行权限分配。

* **权限管理系统应该是可扩展的**。它应该可以加入到任何带有权限管理功能的系统中。就像是组件一样的可以被不断的重用，而不是每开发一套管理系统，就要针对权限管理部分进行重新开发。

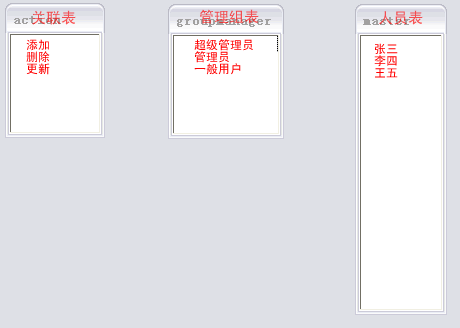
* **满足业务系统中的功能权限。**传统业务系统中，存在着两种权限管理，其一是功能权限的管理，而另外一种则是资源权限的管理，在不同系统之间，功能权限是可以重用的，而资源权限则不能。

**关于设计**

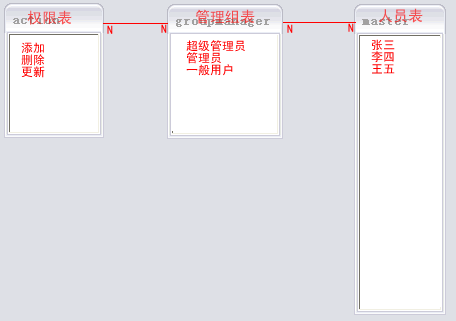
借助NoahWeb的动作编程理念，在设计阶段，系统设计人员无须考虑程序结构的设计，而是从程序流程以及数据库结构开始入手。为了实现需求，数据库的设计可谓及其重要，无论是“组”操作的概念，还是整套权限管理系统的重用性，都在于数据库的设计。

我们先来分析一下数据库结构：

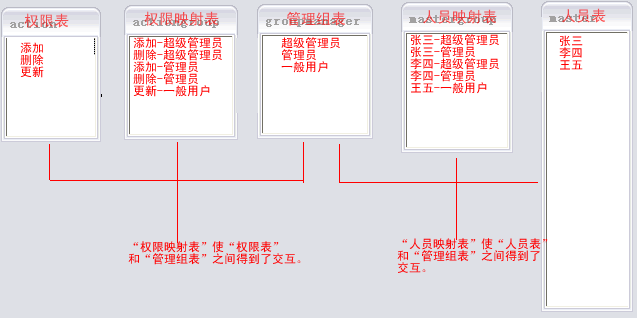
首先，action表（**以下简称为“权限表”**），gorupmanager表（**以下简称为“管理组表”**），以及master表（**以下简称为“人员表”**），是三张实体表，它们依次记录着“权限”的信息，“管理组”的信息和“人员”的信息。如下图：



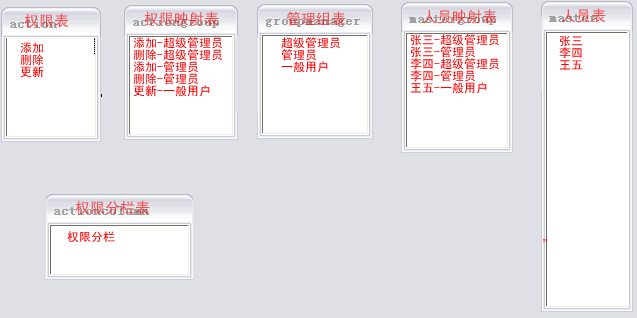
这三个表之间的关系是多对多的，一个权限可能同时属于多个管理组，一个管理组中也可能同时包含多个权限。同样的道理，一个人员可能同时属于多个管理组，而一个管理组中也可能同时包含多个人员。如下图：



由于这三张表之间存在着多对多的关系，那么它们之间的交互，最好使用另外两张表来完成。而这两张表起着映射的作用，分别是“actiongroup”表**（以下简称“权限映射表”）**和“mastergroup”表**（以下简称“人员映射表”）**，前者映射了权限表与管理组表之间的交互。后者映射了人员表与管理组表之间的交互。如下图：

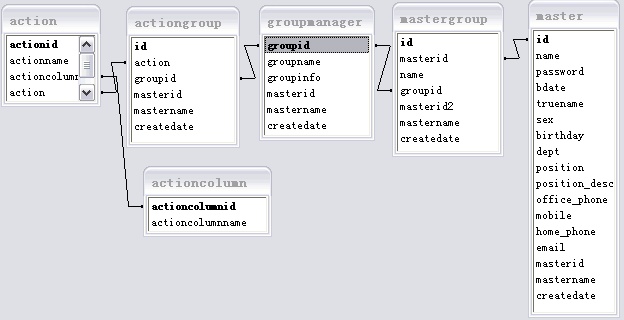


另外，还需要一张表来控制系统运行时左侧菜单中的权限分栏，也就是“权限分栏表”，如下图：



根据上面的分析，我们进行数据库结构设计，如下图：

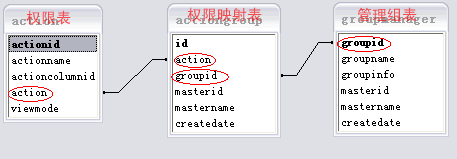
[点击这里查看权限管理系统数据表字段设计](http://www.noahweb.net/mail/2/Project.htm#biao)



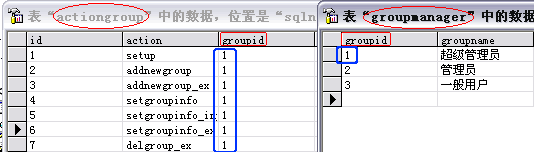
为了能够进行良好的分析，我们将数据库结构图拆分开来，三张实体表的作用已经很清晰，现在我们来看一下两张映射表的作用。

**一 权限映射表** 如下图：

首先，我们来了解一下**权限映射表**与**管理组表**以及**权限表**之间的字段关联。



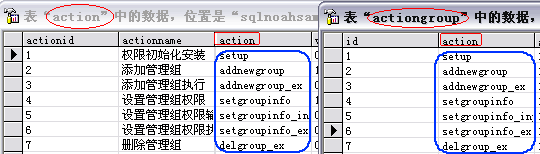
看图中的红圈，先看gorupid字段相关联，这种关联方式在实际数据库中的表现如下图：



如图中所示，**管理组表**中“超级管理员”的groupid为1，那么**权限映射表**中groupid为1的权限也就是“超级管理员”所拥有的权限。

使用groupid字段关联，是为了查到一个管理组能够执行的权限有哪些。但这些权限的详细信息却是action字段关联所查询到的。

action字段相关联在数据库中的表现如下图：



通过这种关联，才查询到**权限映射表**之中那些权限的详细信息。综合起来，我们就知道了一个管理组可以执行的权限有哪些，以及这些权限的详细信息是什么。

或许你会问，为什么不使用actionid字段相关联呢？因为：

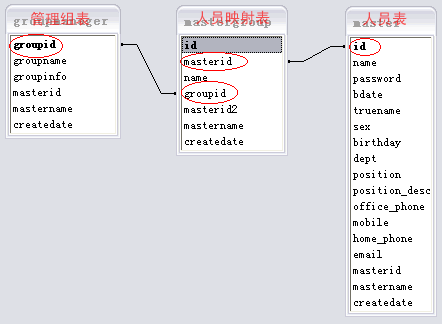
* **权限表**中的id字段在经过多次的数据库操作之后可能会发生更改。
* **权限映射表**中仅仅记录着一个管理组可以执行的权限。
* 一旦**权限表**中的id更改，那么**权限映射表**中的记录也就更改了。
* 一个管理组可以执行的权限势必将出错，这是非常不希望的。

考虑到上面的情况，所以应该使用action字段相关联，因为：

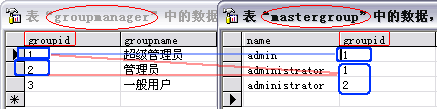
* 在**权限表**中，id可能发生变化，而action字段却是在任何情况下也不可能发生变化的。
* **权限映射表**中记录的action字段也就不会变。
* 一个管理组可以执行的权限就不会出错了。

**二 人员映射表** 如下图：

我们来了解一下**人员映射表**与**管理组表**以及**人员表**之间的字段关联，如下图：



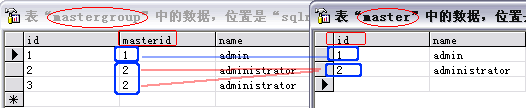
看图中的红圈部分，先看groupid字段关联，这种关联方式在数据库中的表现如下图：



如图，“超级管理员”组的groupid为1，我们再看**人员映射表**，admin属于超级管理员组，而administrator属于超级管理员组，同时也属于管理员组。

使用这种关联方式，是为了查到一个管理组中的人员有谁。和上面一样，人员的详细信息是靠id字段（**人员映射表**中是masterid字段）关联查询到的。

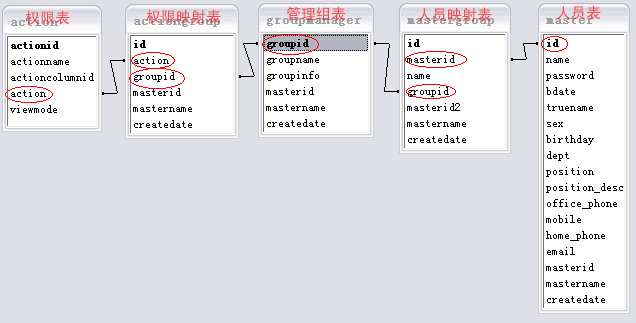
id字段（**人员映射表**中是masterid字段）关联表现在数据库中的形式如下图：



一个人员可能同时属于多个“管理组”，如图中，administrator就同时属于两个“管理组”。所以，在**人员映射表**中关于administrator的记录就会是两条。

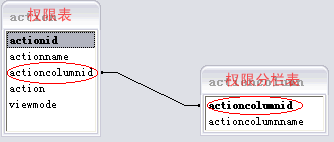
这种关联方式才查询到管理组中人员的详细信息有哪些。综合起来，才可以知道一个管理组中的人员有谁，以及这个人员的详细信息。

再结合上面谈到的**权限表**和**权限映射表**，就实现了需求中的“组”操作，如下图：

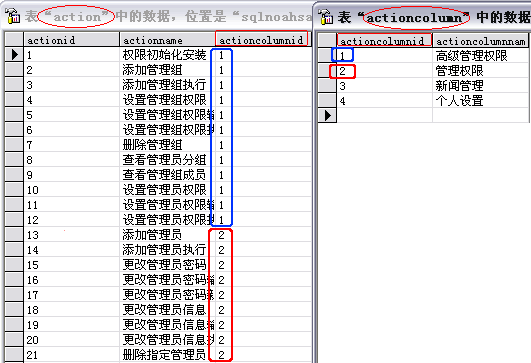


其实，**管理组表**中仅仅记录着组的基本信息，如名称，组id等等。至于一个组中人员的详细信息，以及该组能够执行的权限的详细信息，都记录在**人员表**和**权限表**中。两张**映射表**才真正记录着一个组有哪些人员，能够执行哪些权限。通过两张映射表的衔接，三张实体表之间的交互才得以实现，**从而完成了需求中提到的“组”操作**。

我们再来看一下**权限分栏表**与**权限表**之间的交互。这两张表之间的字段关联如下图：



两张表使用了actioncolumnid字段相关联，这种关联方式在数据库中的表现如下图：



如图所示，通过这种关联方式，我们可以非常清晰的看到**权限表**中的权限属于哪个分栏。

现在，数据库结构已经很清晰了，分配权限的功能以及“组”操作都已经实现。下面我们再来分析一下需求中提到的关于权限管理系统的重用性问题。

为什么使用这种数据库设计方式搭建起来的系统可以重用呢？

* **三张实体表中记录着系统中的三个决定性元素。**“权限”，“组”和“人”。而这三种元素可以任意添加，彼此之间不受影响。无论是那种类型的业务系统，这三个决定性元素是不会变的，也就意味着结构上不会变，而变的仅仅是数据。
* **两张映射表中记录着三个元素之间的关系。**但这些关系完全是人为创建的，需要变化的时候，只是对数据库中的记录进行操作，无需改动结构。
* **权限分栏表中记录着系统使用时显示的分栏**。无论是要添加分栏，修改分栏还是减少分栏，也只不过是操作记录而已。

综上所述，这样设计数据库，系统是完全可以重用的，并且经受得住“变更”考验的。

**总结：**

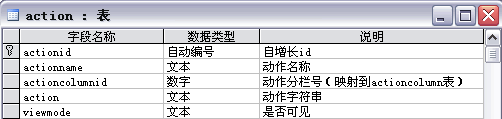
此套系统的重点在于，三张**实体表**牢牢地抓住了系统的核心成分，而两张映射表完美地映射出三张实体表之间的交互。其难点在于，理解映射表的工作，它记录着关系，并且实现了“组”操作的概念。而系统总体的设计是本着可以在不同的MIS系统中“重用”来满足不同系统的功能权限设置。

**附录：**

**权限管理系统数据表的字段设计**

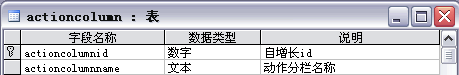
下面我们来看看权限管理系统的数据库表设计，共分为六张表，如下图：

action表：

**

action表中记录着系统中所有的动作，以及动作相关描述。

actioncolumn表：



actioncolumn表中记录着动作的分栏，系统运行时，左侧菜单栏提供了几块不同的功能，每一块就是一个分栏，每添加一个分栏，该表中的记录就会增加一条,相对应的，左侧菜单栏中也会新增机一个栏。

actiongroup表：



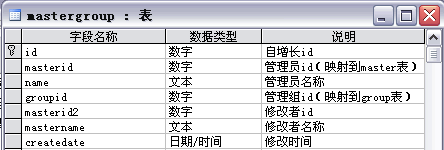
actiongroup表记录着动作所在的组。

groupmanager表：



groupmanager表记录着管理组的相关信息，每添加一个管理组，这里的记录就会增加一条。

mastergroup表：



mastergroup表记录着管理员所在的管理组，由于一名管理员可能同同时属于多个组，所以该表中关于某一名管理员的记录可能有多条。

master表：



master表记录着所有管理员的信息，每添加一个管理员，该表就会增加一条记录。窗体底端