https://blog.csdn.net/u013630349/article/details/51733749

权限

在系统中，权限通过模块+动作来产生，模块就是整个系统中的一个子模块，可能对应一个菜单，动作也就是整个模块中（在B/S系统中也就是一个页面的所有操作，比如“浏览、添加、修改、删除”等）。将模块与之组合可以产生此模块下的所有权限。

权限组

为了更方便的权限的管理，另将一个模块下的所有权限组合一起，组成一个“权限组”，也就是一个模块管理权限，包括所有基本权限操作。比如一个权限组（用户管理），包括用户的浏览、添加、删除、修改、审核等操作权限，一个权限组也是一个权限。

角色

权限的集合，角色与角色之间属于平级关系，可以将基本权限或权限组添加到一个角色中，用于方便权限的分配。

用户组

将某一类型的人、具有相同特征人组合一起的集合体。通过对组授予权限（角色），快速使一类人具有相同的权限，来简化对用户授予权限的繁琐性、耗时性。用户组的划分，可以按 职位、项目 或其它来实现。用户可以属于某一个组或多个组。

通过给某个人赋予权限，有4种方式

A.       通过职位

a)        在职位中，职位成员的权限继承当前所在职位的权限，对于下级职位拥有的权限不可继承。

b)       实例中：如前台这个职位，对于考勤查询有权限，则可以通过对前台这个职位设置考勤查询的浏览权，使他们有使用这个对象的权限，然后再设置个，考勤查询权（当然也可以不设置，默认能进此模块的就能查询），则所有前台人员都拥有考勤查询的权利。

B.       通过项目

a)        在项目中，项目成员的权限来自于所在项目的权限，他们同样不能继承下级项目的权限，而对于项目组长，他对项目有全权，对下级项目也一样。

b)       实例中：在项目中，项目成员可以对项目中上传文档，查看本项目的文档,可以通过对项目设置一个对于本项目的浏览权来实现进口，这样每个成员能访问这个项目了，再加上项目文档的上传权和查看文档权即可。

c)       对于组长，因为可以赋予组长一个组长权（组长权是个特殊的权限，它包含其他各种权限的一个权限包），所有组长对于本项目有全权，则项目组长可以对于项目文档查看，审批，删除，恢复等，这些权限对于本项目的下级项目依然有效。

C.      通过角色

a)        角色中的成员继承角色的权限，角色与角色没有上下级关系，他们是平行的。通过角色赋予权限，是指没办法按职位或项目的分类来赋予权限的另一种方式，如：系统管理员，资料备份员…

b)       实例中：对于本系统中，全体人员应该默认都有的模块，如我的邮件，我的文档，我的日志，我的考勤……，这些模块系统成员都应该有的，我们建立一个角色为系统默认角色，把所有默认访问的模块的浏览权加入到里面去，则系统成员都能访问这些模块。

D.      直接指定

a)        直接指定是通过对某个人具体指定一项权限，使其有使用这个权限的能力。直接指定是角色指定的一个简化版，为了是在建立像某个项目的组长这种角色时，省略创建角色这一个步骤，使角色不至于过多。

b)       实例中：指定某个项目的组长，把组长权指定给某个人。

针对职位、项目组：

如果用添加新员工，员工调换职位、项目组，满足了员工会自动继承所在职位、项目组的权限，不需要重新分配权限的功能。

用户管理

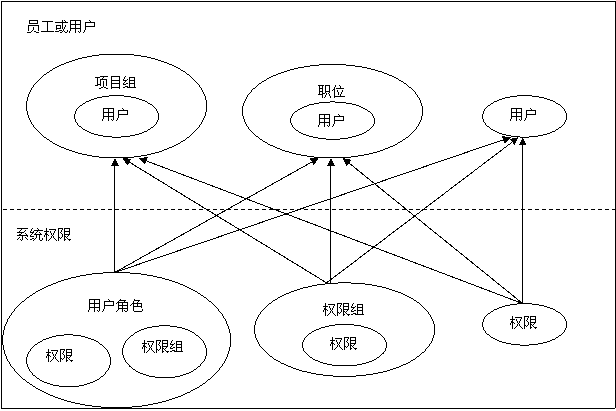
用户可以属于某一个或多个用户组，可以通过对用户组授权，来对组中的所有用户进行权限的授予。一个用户可以属于多个项目组，或担任多个职位。

授权管理

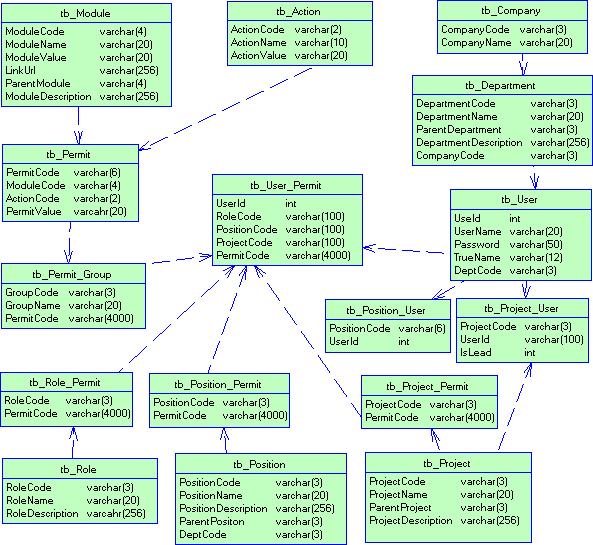
将一个基本权限或角色授予用户或用户组，使用户或用户组拥有授予权限的字符串，如果角色、职位、项目中存在相同的基本权限，则取其中的一个；如脱离角色、职位、项目组，只是取消用户或用户组的中此角色、职位、项目组所授予的权限。用户所拥有的权限是所有途径授予权限的集合。管理员用户可以查看每个用户的最终权限列表。

权限管理

基本操作权限与权限组（基本操作权限的集合）的管理。



物理数据模型图如下：



根据以上设计思想,权限管理总共需要以下基本表：

tb\_User：用户信息基本表；

tb\_Department：部门表；

tb\_Company：公司表；

tb\_Module：系统模块表；

tb\_Action：系统中所有操作的动作表；

tb\_Permit：由tb\_Module与tb\_Action两表结合产生的系统基本权限表；

tb\_Permit\_Group：权限组表，将一模块的中的所有权限划分一个权限组中，可以通过权限组授予用户权限；

tb\_Role：角色表，基本权限的集合。无上级与下级之分；

tb\_Position：职位表，有上级与下级之分；

tb\_Project：项目组表，

tb\_Role\_Permit：角色授权表；

tb\_Postion\_Permit：职位授权表；

tb\_Project\_Permit：项目授权表；

tb\_Project\_User：项目成员表，IsLead字段代表此成员为项目组长；

tb\_Postion\_User：职位成员表；

tb\_User\_Permit：用户授权表，用户ID与角色、职位、项目及直接授予的权限串表；

权限的产生：

由tb\_Module中的ModuleCode与tb\_Action中的ActionCode组成

权限代码PermitCode=ModuleCode+ActionCode。

实例：ModuleCode=0101，ActionCode=01,则PermitCode=010101。

权限值则有ModuleValue与ActionCode组合而成，采用下划线来连接。

实例：ModuleValue=Sys\_User,ActionValue=AdD，PermitValue= Sys\_User\_Add

权限组：

       包括一组同一模块下的权限的组合，如管理用户包括基本的权限：添加、删除、修改、查看等，将这些组合起来构成一个用户组——“用户管理”权限组。其它类似。只是为了更方便的查看系统权限与权限的分配。

       实例：如管理用户的权限代码为010101à查看用户，010102à添加用户,010103à删除用户，010104à修改用户，010105à审核用户等，将这些基本权限组合起来一个集合而构成了“用户管理”权限组。

角色、职位、项目：

       也就是按特定的需要划分一种权限的集合。使用角色授权表、职位授权表、项目授权表来实现。授权表中存放的是权限代码PermitCode,而不是权限组的GroupCode代码。

用户授权：

       由用户授权表来实现，用户授权表中的RoleCode、PositionCode、ProjectCode分别是角色表中RoleCode组成的串、职位表PositionCode组成的串、ProjectCode组成的串。与角色授权表中的角色代码RoleCode、职位授权表中PositionCode、项目授权表中的ProjectCode不对应（不是主表与从表之间外键关系）。

       从而能够实现了一个用户可以拥有多个角色、多个职位、多个项目的情况。

用户授权表中的PermitCode为直接授权的权限代码串，直接给用户分配权限。

实例：

用户ID为UserId=1的用户权限授权表的记录为：

RoleCode=001,003

PostionCode = 001,002

ProjectCode=001,005

PermitCode = 010101,020102

表明此用户拥有两个角色，代码为001和003，并继承这两个角色的权限；

担任两个职位，代码为001与002，并继承两个职位的权限；

属于两个项目组中的成员，项目代码为001与005，并继承两个项目中的权限。

直接指定给用户的权限为010101与010102这两个权限代码的权限

用户权限字符串：

根据用户授权表的角色代码、职位代码、项目代码得到权限字符串及表中直接分配的权限字符串组合成一个用户的所有权限字符串集合。

--------------------- ！