1. RBAC模型

本章是主要介绍权限模型RBAC，基于RBAC模型设计出简单的数据库，为后续的Shiro安全框架的授权控制奠定基础。关于本章，我们需要把握如下内容：

* 理解RBAC模型
* 基于RBAC模型，设计出数据库模型
* 理解RBAC数据模型对应的表和表结构

目录

[5 RBAC模型 1](#_Toc493778526)

[5.1 BRAC模型 2](#_Toc493778527)

[5.1.1 RBAC0 2](#_Toc493778528)

[5.1.2 RBAC1 3](#_Toc493778529)

[5.1.3 RBAC2 3](#_Toc493778530)

[5.1.4 RBAC3 4](#_Toc493778531)

[5.2 权限模型 4](#_Toc493778532)

[5.3 权限表 5](#_Toc493778533)

[5.4 权限实体类 7](#_Toc493778534)

[5.5 总结 7](#_Toc493778535)

前面几章，我们主要研究了基于Shiro安全框架实现身份认证，接下来我们继续研究安全控制的另外一个方面—授权，也就是访问控制，也叫权限控制。

为了很好的理解并且运用Shiro安全框架的授权，我们先研究一下通用的权限模型，理解项目中权限相关的有哪些对象、关系是怎样的、以及对于的数据库建模，为我们后面章节中对Shiro安全框架的授权运用打下基础，同时后面web集成、框架集成的案例，也是基于本章设计的数据库进行的，请大家务必对本章内容有个熟悉的把握。如果读者对这块有自己的体会和理解，可以跳过本章，继续后面章节。

* 1. RBAC模型

授权，也是权限，就是控制访问，对资源的访问进行控制。当一个Subject操作某个资源时，需要判断一下是否有对应的权限，如果有，可以正常的操作，否则就阻止操作，提示没有权限。对于web应用来说，一般就是对url的控制，也有控制到方法或某个数据的，不管哪种，都是控制访问操作。关于权限，可以不可置疑的说，任何一个系统都有，只是控制的细粒度不同，复杂度不同，再怎么样都要控制到是否登录。

RBAC是Role-BasedAccess Control的英文缩写，意思是基于角色的访问控制。RBAC认为权限授权实际上是Who、What、How的问题。在RBAC模型中，who、what、how构成了访问权限三元组，也就是“Who对What(Which)进行How的操作，也就是“主体”对“客体”的操作，其中who——是权限的拥有者或主体（如：User、Role），what——是资源或对象（Resource、Class)，how—是操作(新增、修改、删除、查询等)

RBAC其实是一种分析模型，主要分为：基本模型RBAC0（Core RBAC）、角色分层模型RBAC1（Hierarchal RBAC）、角色限制模型RBAC2（Constraint RBAC）和统一模型RBAC3（Combines RBAC）。

* + 1. RBAC0

RBAC0，它是RBAC0的核心，RBAC1、RBAC2、RBAC3都是先后在RBAC0上的扩展。RBAC0定义了能构成RBAC控制系统的最小的元素集合，RBAC0由四部分构成：

* 用户(User)
* 角色(Role)
* 会话(Session)
* 许可(Permission)

在RBAC0模型中，我们将对资源的各种操作，叫许可，也叫权限，包含操作和操作的资源。它们的关系是将许可赋给角色，再将角色赋给用户。动态运行的时候，根据前面用户同角色的关系，建立每个用户同各自角色之间的对应，形成会话。它们关系如下图5-1RBAC0模型所描述：

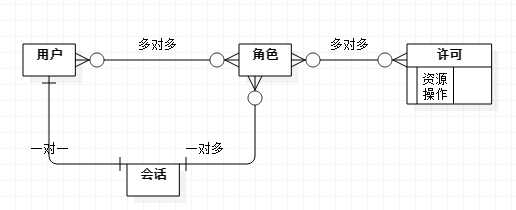


图5-1 RBAC0模型

用户与角色是多对多的关系；角色和许可也是多对多的关系；用户与会话是一对一关系；会话与角色是一对多关系；

* + 1. RBAC1

RBAC1，它是RBAC角色的分层模型，RBAC1建立在RBAC0基础之上，在角色中引入了继承的概念，有了继承那么角色就有了上下级或者等级关系。如图5-2RBAC1模型：

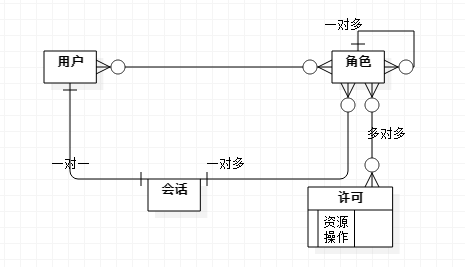


图5-2 RBAC1模型

* + 1. RBAC2

RBAC2，它是RBAC的约束模型，RBAC2也是建立的RBAC0的基础之上的，在RBAC0基础上加上了约束的概念，主要引入了静态职责分离SSD(Static Separation of Duty)和动态职责分离DSD(Dynamic Separation of Duty)。

SSD是用户和角色的指派阶段加入的，主要是对用户和角色有如下约束:

* 互斥角色

同一个用户在两个互斥角色中只能选择一个

* 基数约束

一个用户拥有的角色是有限的，一个角色拥有的许可也是有限的

* 先决条件约束

用户想要获得高级角色，首先必须拥有低级角色

DSD是会话和角色之间的约束，可以动态的约束用户拥有的角色，如一个用户可以拥有两个角色，但是运行时只能激活一个角色。

如图5-3 RBAC2模型：

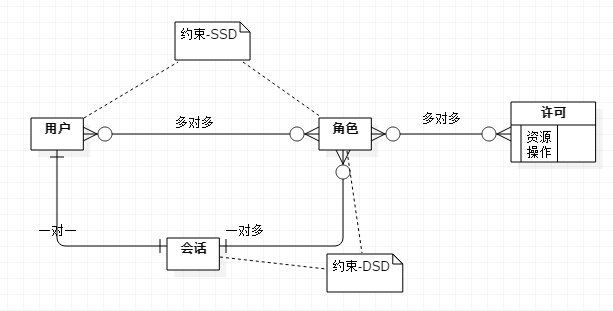


图5-3 RBAC2模型

* + 1. RBAC3

RBAC3，它是RBAC1与RBAC2合集，所以RBAC3是既有角色分层又有约束的一种模型，如图5-4 RBAC3模型。

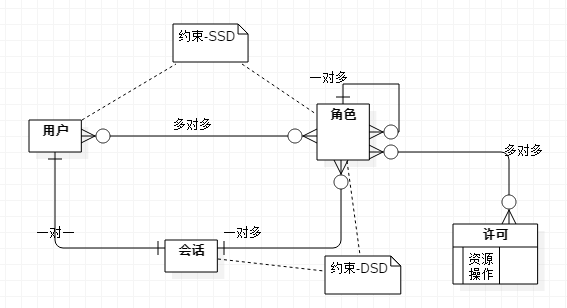


图5-4 RBAC3模型

* 1. 权限模型

前面我们介绍了四个权限模型，在实际项目中，往往需要这四个模型的基础上，加上适合实际项目匹配的设计。比如实际项目中，每个用户会分部门，每个部门的员工除了需要按自己在所属的角色分配权限外，也可以按部门分配权限。比如一个开发部的张三，他是开发部的经理。他就既拥有开发部部门的权限，比如查看代码服务器上的代码权限，也拥有作为一个经理的权限，比如对部门员工的请假申请进行审核。

考虑到各种项目需要，实现项目中的权限模型是比较复杂的，我们需要根据具体用户的需求，在我们基本RBAC模型基础上进行设计扩展。本书案例用到的权限模型，不追求全面，主要体现基本的RBAC模型的应用和基于Shiro安全框架实现，实体模型图如图5-5 权限模型。

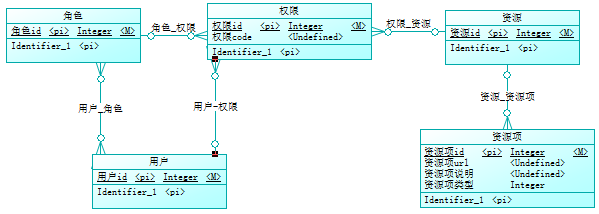


图5-5 权限模型

用户的具体操作权限，可以通过两个途径分配：分配角色给用户；分配权限给用户。用户和角色是多对多的关系，用户和权限之间是一对多的关系，角色同权限是一对多的关系。

权限描述对资源的各种操作。一般对一个资源至少有四种操作（增、删、改、查），为了简便数据存储，权限里面包含三个信息，一个是操作的资源对象，一个是操作code，一个是角色或用户，整个意思就是谁(角色)对那个资源可以进行什么操作。比如经理可以对请假进行审核，张三可以对文件进行查看等等。操作code是个二进制数字，至少四为，每位对应增、删、改、查操作的许可。比如0表示不允许，1表示允许。0000表示四个操作都不允许；1000表示只允许增操作；1100表示允许增、删操作；1001表示允许增、查操作等等。

每个资源有至少四种操作，每种操作又有不同的请求url，所以我们把资源对象设计成2个，资源和资源项。

* 1. 权限表

针对5.2的数据模型，设计出对应的表结构，详细内容请查看随书代码[code/ShiroTP04/demo\_shiro.sql](../code/ShiroTP04/demo_shiro.sql)：

1. 资源表(shr\_resource)

CREATE TABLE `shr\_resource` (

`rsc\_id` int(11) NOT NULL AUTO\_INCREMENT,

`rsc\_name` varchar(50) DEFAULT NULL,

`rsc\_remark` varchar(200) DEFAULT NULL COMMENT '权限操作码\r\n1 创建\r\n2 修改\r\n4 删除\r\n8 查看\r\n16 打印\r\n32 其他\r\n64 其他',

PRIMARY KEY (`rsc\_id`),

UNIQUE KEY `pms\_resource\_name` (`rsc\_name`)

) ENGINE=InnoDB AUTO\_INCREMENT=16 DEFAULT CHARSET=utf8;

1. 资源项表(shr\_resource\_item)

CREATE TABLE `shr\_resource\_item` (

`itm\_id` int(11) NOT NULL AUTO\_INCREMENT,

`itm\_type` int(11) NOT NULL COMMENT '1 创建 2修改 4删除 8查看 16打印 32其他 64其他',

`itm\_url` varchar(200) DEFAULT NULL,

`itm\_remark` varchar(500) DEFAULT NULL,

`itm\_rsc\_id` int(11) NOT NULL DEFAULT '1',

PRIMARY KEY (`itm\_id`),

KEY `itm\_rsc\_id` (`itm\_rsc\_id`),

CONSTRAINT `shr\_resource\_item\_ibfk\_1` FOREIGN KEY (`itm\_rsc\_id`) REFERENCES `shr\_resource` (`rsc\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=13 DEFAULT CHARSET=utf8;

1. 角色表(shr\_role)

CREATE TABLE `shr\_role` (

`rl\_id` int(11) NOT NULL AUTO\_INCREMENT,

`rl\_name` varchar(50) NOT NULL,

`rl\_remark` varchar(200) DEFAULT NULL,

PRIMARY KEY (`rl\_id`),

UNIQUE KEY `rl\_name` (`rl\_name`)

) ENGINE=InnoDB AUTO\_INCREMENT=9 DEFAULT CHARSET=utf8;

1. 角色权限表(shr\_role\_permission)

CREATE TABLE `shr\_role\_permission` (

`prm\_rl\_id` int(11) NOT NULL,

`prm\_rsc\_id` int(11) NOT NULL,

`prm\_operation\_code` int(11) NOT NULL,

`prm\_id` int(11) NOT NULL AUTO\_INCREMENT,

PRIMARY KEY (`prm\_id`),

KEY `prm\_rl\_id` (`prm\_rl\_id`),

KEY `prm\_rsc\_id` (`prm\_rsc\_id`),

CONSTRAINT `shr\_role\_permission\_ibfk\_1` FOREIGN KEY (`prm\_rl\_id`) REFERENCES `shr\_role` (`rl\_id`),

CONSTRAINT `shr\_role\_permission\_ibfk\_2` FOREIGN KEY (`prm\_rsc\_id`) REFERENCES `shr\_resource` (`rsc\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=9 DEFAULT CHARSET=utf8;

1. 角色用户关系表(shr\_ur\_rl\_map)

CREATE TABLE `shr\_ur\_rl\_map` (

`ur\_id` int(11) NOT NULL,

`rl\_id` int(11) NOT NULL,

PRIMARY KEY (`ur\_id`,`rl\_id`),

KEY `rl\_id` (`rl\_id`),

CONSTRAINT `shr\_ur\_rl\_map\_ibfk\_1` FOREIGN KEY (`ur\_id`) REFERENCES `shr\_user` (`ur\_id`),

CONSTRAINT `shr\_ur\_rl\_map\_ibfk\_2` FOREIGN KEY (`rl\_id`) REFERENCES `shr\_role` (`rl\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8;

1. 用户表(shr\_user)

CREATE TABLE `shr\_user` (

`ur\_id` int(11) NOT NULL AUTO\_INCREMENT,

`ur\_user\_name` varchar(20) NOT NULL,

`ur\_password` varchar(100) DEFAULT NULL,

`ur\_salt` varchar(50) DEFAULT NULL,

PRIMARY KEY (`ur\_id`),

UNIQUE KEY `ur\_user\_name` (`ur\_user\_name`)

) ENGINE=InnoDB AUTO\_INCREMENT=43 DEFAULT CHARSET=utf8;

1. 用户权限表(shr\_user\_permission)

CREATE TABLE `shr\_user\_permission` (

`prm\_ur\_id` int(11) NOT NULL DEFAULT '0',

`prm\_rsc\_id` int(11) NOT NULL DEFAULT '0',

`prm\_operation\_code` int(11) DEFAULT NULL,

`prm\_id` int(11) NOT NULL AUTO\_INCREMENT,

PRIMARY KEY (`prm\_id`),

KEY `prm\_ur\_id` (`prm\_ur\_id`),

KEY `prm\_rsc\_id` (`prm\_rsc\_id`),

CONSTRAINT `shr\_user\_permission\_ibfk\_1` FOREIGN KEY (`prm\_ur\_id`) REFERENCES `shr\_user` (`ur\_id`),

CONSTRAINT `shr\_user\_permission\_ibfk\_2` FOREIGN KEY (`prm\_rsc\_id`) REFERENCES `shr\_resource` (`rsc\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=9 DEFAULT CHARSET=utf8;

* 1. 权限实体类
  2. 总结