细说 Spring Security 安全框架（二）

第三章 基于角色的权限

3.1 认证和授权

authentication：认证， 认证访问者是谁。 一个用户或者一个其他系统是不是当前要访问的系统中的有效用户。

authorization：授权， 访问者能做什么

比如说张三用户要访问一个公司 oa 系统。 首先系统要判断张三是不是公司中的有效用户。

认证张三是不是有效的用户，是不是公司的职员

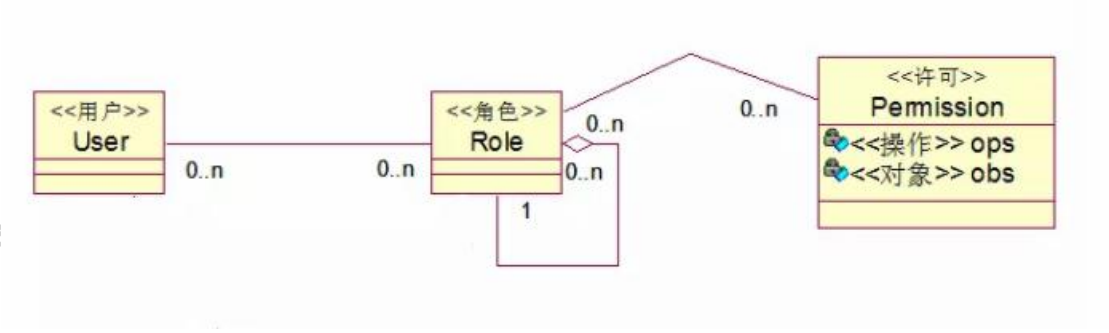
授权： 判断张三能否做某些操作， 如果张三是个领导可以批准下级的请假， 其他的操作。

如果张三只是一个普通用户，只能看自己的相关数据， 只能提交请假申请等等。

3.2 RBAC 是什么？

RBAC 是基于角色的访问控制（Role-Based Access Control ）在 RBAC 中，权限与角色相关联，用户通过成为适当角色的成员而得到这些角色的权限。这就极大地简化了权限的管理。这样管理都是层级相互依赖的，权限赋予给角色，而把角色又赋予用户，这样的权限设计很清楚，管理起来很方便。

其基本思想是，对系统操作的各种权限不是直接授予具体的用户，而是在用户集合与权限集合之间建立一个角色集合。每一种角色对应一组相应的权限。一旦用户被分配了适当的角色后，该用户就拥有此角色的所有操作权限。这样做的好处是，不必在每次创建用户时都进行分配权限的操作，只要分配用户相应的角色即可，而且角色的权限变更比用户的权限变更要少得多，这样将简化用户的权限管理，减少系统的开销。



RBAC： 用户是属于角色的， 角色拥有权限的集合。 用户属于某个角色， 他就具有角色对应的权限。

系统中有张三，李四，他们是普通员工，只能查看数据。

系统中经理，副经理他们能修改数据。

设计有权限的集合，角色: 经理角色，具有修改数据的权限，删除，查看等等。

普通用户角色： 只读角色，只能看数据 ，不能修改，删除。

让张三，李四是只读的，普通用户角色。 让经理，副经理他们都是经理角色。

公司以后增加新的普通员工，加入到“普通用户角色”就可以了，不需要在增加新的角色。

公司增加经理了， 只要加入到“经理角色”就可以了。

权限：能对资源的操作， 比如增加，修改，删除，查看等等。

角色：自定义的， 表示权限的集合。一个角色可以有多个权限。

RBAC 设计中的表：

1. 用户表： 用户认证（登录用到的表）

   用户名，密码，是否启用，是否锁定等信息。

2. 角色表：定义角色信息

角色名称， 角色的描述。

3.用户和角色的关系表： 用户和角色是多对多的关系。

一个用户可以有多个角色， 一个角色可以有多个用户。

4.权限表， 角色和权限的关系表

角色可以有哪些权限。

3.3 spring specurity 中认证的接口和类

1） UserDetails：接口，表示用户信息的。

boolean isAccountNonExpired(); 账号是否过期

boolean isAccountNonLocked();账号是否锁定

boolean isCredentialsNonExpired();证书是否过期

boolean isEnabled();账号是否启用

Collection<? extends GrantedAuthority> getAuthorities(); 权限集合

User 实现类

org.springframework.security.core.userdetails.User

可以：自定义类实现 UserDetails 接口，作为你的系统中的用户类。这个类可以交给 spring security 使用。

2） UserDetailsService 接口：

主要作用：获取用户信息，得到是 UserDetails 对象。一般项目中都需要自定义类实现这个接口，从数据库中获取数据。

一个方法需要实现：

UserDetails loadUserByUsername(String var1) ：根据用户名称，获取用户信息（用户名称，密码，角色结合，是否可用，是否锁定等信息）

UserDetailsService 接口的实现类：

1. InMemoryUserDetailsManager：在内存中维护用户信息。

   优点：使用方便。

   缺点：数据不是持久的。系统重启后数据恢复原样。

2.JdbcUserDetailsManager ：用户信息存放在数据库中，底层使用 jdbcTemplate 操作数据库。 可以 JdbcUserDetailsManager 中的方法完成用户的管理

createUser ： 创建用户

updateUser：更新用户

deleteUser：删除用户

userExists：判断用户是否存在

数据库文件：

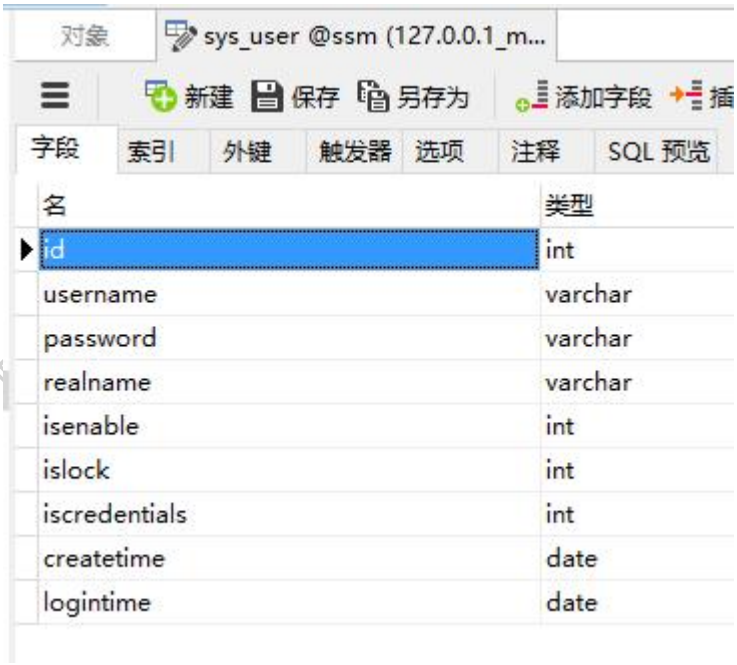
org.springframework.security.core.userdetails.jdbc

users.ddl 文件

```
create table users(username varchar_ignorecase(50) not null primary key,password varchar_ignorecase(500) not null,enabled boolean not null);
create table authorities (username varchar_ignorecase(50) not null,authority varchar_ignorecase(50) not null,constraint fk_authorities_users foreign key(username) references users(username));
create unique index ix_auth_username on authorities (username,authority);
```

3.4 定义用户，角色，角色关系表

用户信息表 sys_user：



sys_role：角色表

字段 | 索引 | 外键 | 触发器 | 选项 | 注释 | SQL 预览

名	类型
id	int
rolename	varchar
rolememo	varchar

sys_user_role: 用户-角色关系表

名	类型
userid	int
roleid	int