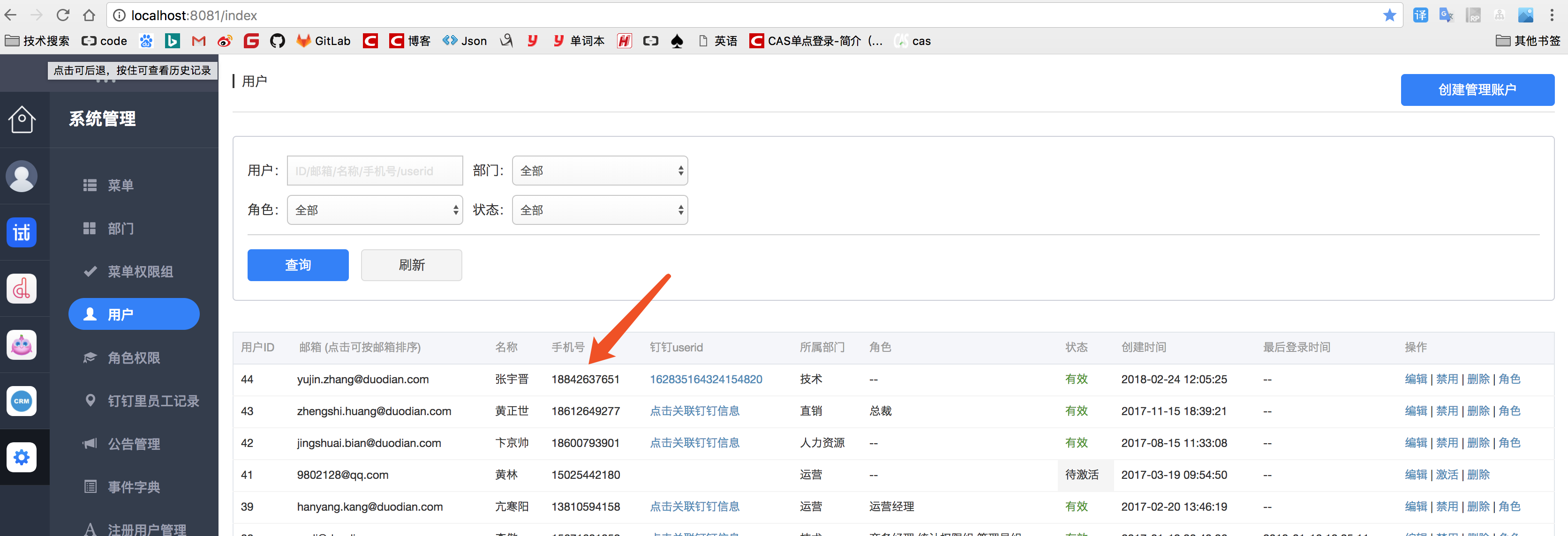
# 1、创建张宇晋账户

## 1、前端展示



## 2、后端代码操作

### 1、contrler SysUserController.java

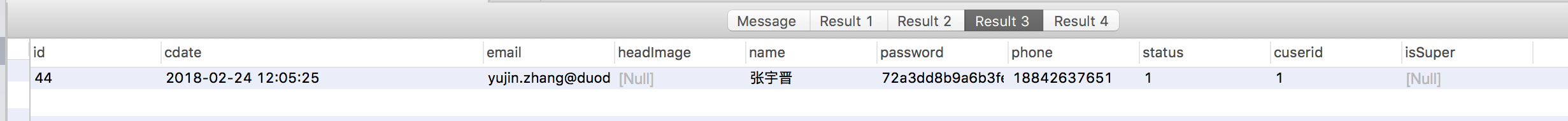
@RequestMapping("save")  
@ResponseBody  
public ResponseBean save(SysAdminUser user){  
 SysAdminUser sysAdminUser = AppSessionHelper.getSessionUser();  
 try {  
 sysUserService.saveSysAdminUser(sysAdminUser.getId(),user);  
 return ResponseBean.buildSuccess();  
 } catch (AppException e) {  
 return ResponseBean.buildFailure(e.getMessage());  
 } catch (Exception e){  
 logger.error(e.getMessage(),e);  
 return ResponseBean.buildFailure(ErrorCodeEnum.***系统错误***);  
 }  
}

## 3、表

### 1.1、用户表 sys\_admin\_user

@Id  
@GeneratedValue(strategy = GenerationType.***IDENTITY***)  
private Long id;  
private String email;  
private String password;  
private String name; //名称  
private String phone; //手机号码  
private Integer status; //状态  
private Integer isSuper; //是否是超级管理员  
  
private Long cuserId; //创建人id

### 1.2、数据库数据 sys\_admin\_user



### 2.1 部门表 sys\_dept\_admin\_user

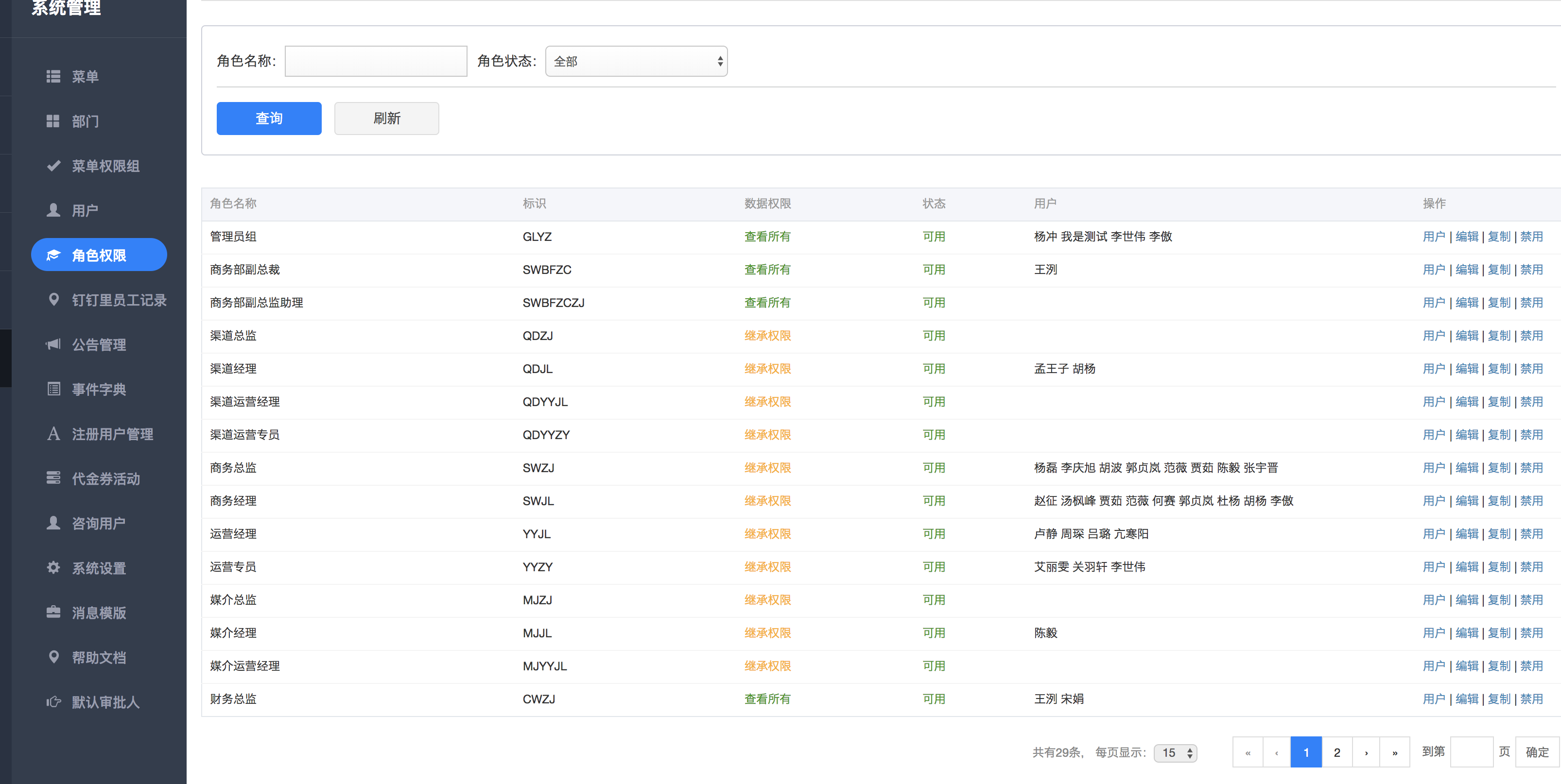
\*/  
@Entity  
@Table(name = "sys\_dept\_admin\_user")  
public class SysDepartmentAdminUser implements Serializable {  
  
 private static final long ***serialVersionUID*** = -3792256317744904398L;  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.***IDENTITY***)  
 private Long id;  
 private Long departmentId;  
 private Long admId;



## 4、登录查看 只有一个主页 数据显示

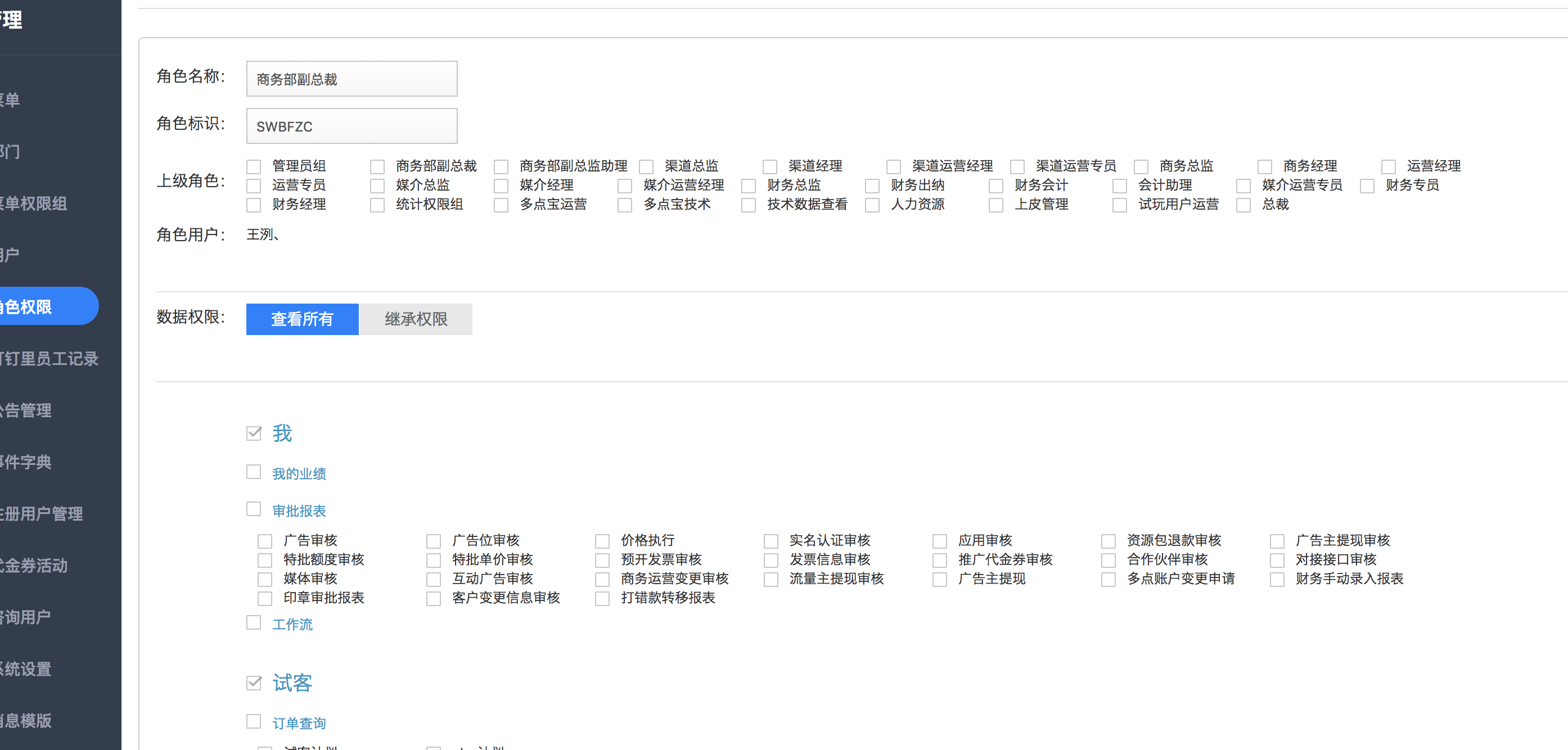
# 2、创建公司的角色

## 1、前端展示

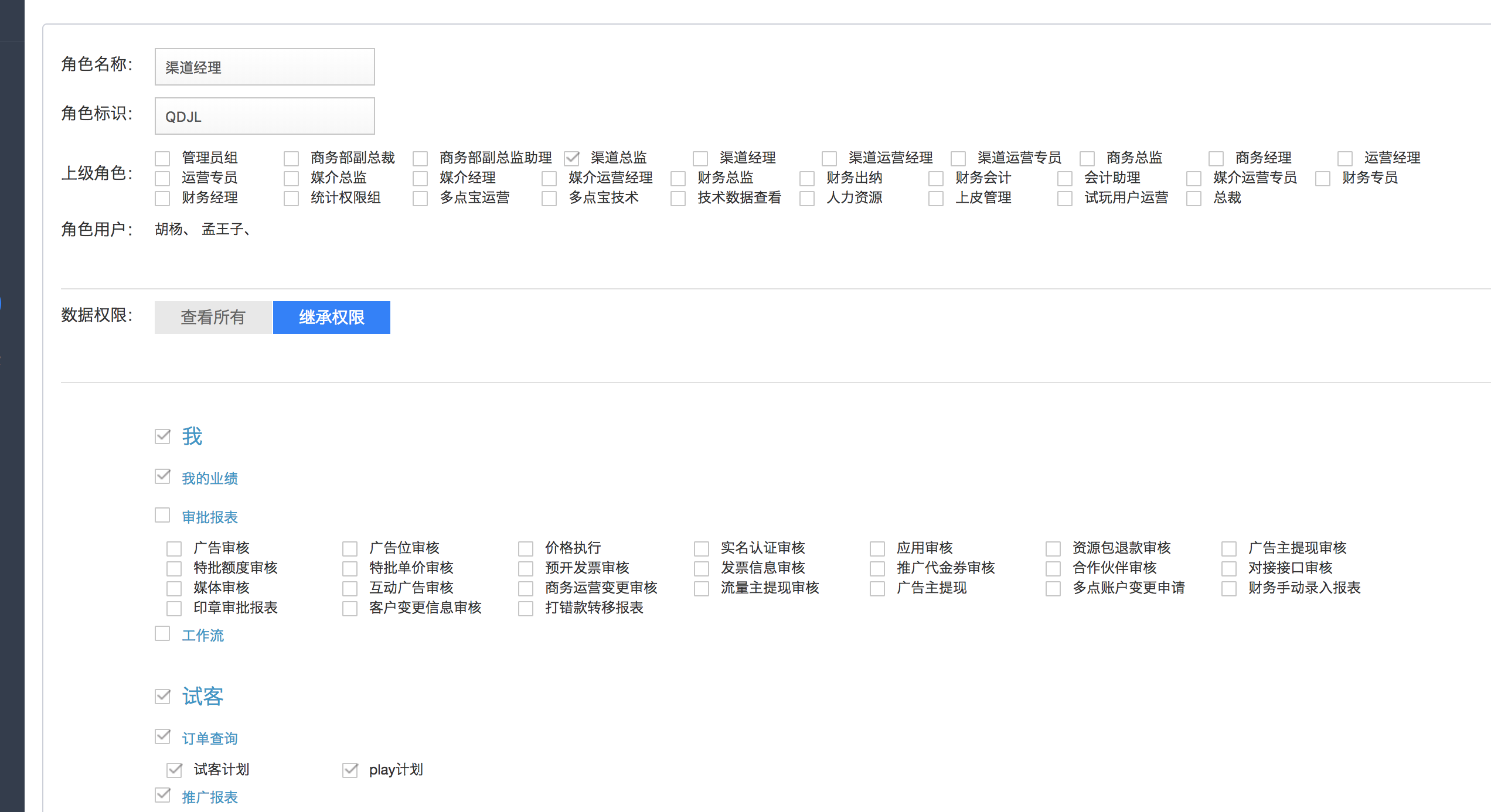


## 2、角色是否独自拥有权限或者是继承权限

### 1、从查看所有权限，没有上级角色



### 2、继承权限，选中一个上级角色，下面的不需要选中，在执行逻辑的时候，通过判断是否具有上级角色（将来这里的菜单参数会从前台传入），然后给他分配上级角色所具备的数据权限即可，如果上级角色功能比较少，还可以继续选则下面的数据菜单权限



@Override  
public void saveAuthRole(SysAuthRole sysAuthRole) {  
 if (sysAuthRole == null) {  
 throw new AppException("参数缺失，请刷新页面重试");  
 }  
 if (StringUtils.isBlank(sysAuthRole.getName())) {  
 throw new AppException("请填写角色名称！");  
 }  
 if (StringUtils.isBlank(sysAuthRole.getCode())) {  
 throw new AppException("请填写角色标识！");  
 }  
 //监测code的唯一性  
 SysAuthRole checkCodeRole = null;  
 if(sysAuthRole.getId() != null){  
 checkCodeRole = sysAuthRoleDAO.findTopByCodeAndStatusAndIdNot(sysAuthRole.getCode(),SysAuthRole.***STATUS\_ENABLE***,sysAuthRole.getId());  
 }else {  
 checkCodeRole = sysAuthRoleDAO.findTopByCodeAndStatus(sysAuthRole.getCode(),SysAuthRole.***STATUS\_ENABLE***);  
 }  
 if (checkCodeRole != null){  
 throw new AppException("角色标识【"+sysAuthRole.getCode()+"】已被占用，请修改！");  
 }  
 if (sysAuthRole.getDataType() == null) {  
 throw new AppException("请配置数据权限！");  
 }  
 if (StringUtils.isBlank(sysAuthRole.getMenuIds())) {  
 throw new AppException("请配置菜单权限！");  
 }  
 SysAuthRole authRole = new SysAuthRole();  
 if (sysAuthRole.getId() != null) {  
 //修改  
 authRole = sysAuthRoleDAO.findOne(sysAuthRole.getId());  
  
 if (authRole == null){  
 authRole = new SysAuthRole();  
 }  
 //清除该角色对应的菜单记录 后面重新添加菜单记录  
 sysAuthRoleMenuDAO.deleteByRoleId(sysAuthRole.getId());  
 } else {  
 //新建  
 SysAuthRole lastSortRole = sysAuthRoleDAO.findTopByStatusOrderBySortsDesc(SysAuthRole.***STATUS\_ENABLE***);  
 authRole.setStatus(SysAuthRole.***STATUS\_ENABLE***);  
 authRole.setSorts(lastSortRole.getSorts() + 1);  
 }  
 authRole.setName(sysAuthRole.getName());  
 authRole.setCode(sysAuthRole.getCode());  
 authRole.setPid(sysAuthRole.getPid());  
 authRole.setDataType(sysAuthRole.getDataType());  
  
 sysAuthRoleDAO.save(authRole);  
  
 //保存菜单权限 关联菜单  
 List<String> menuIdStrList = Arrays.asList(sysAuthRole.getMenuIds().split(","));  
 if (CollectionUtils.isNotEmpty(menuIdStrList)){  
 for (String menuIdStr : menuIdStrList){  
 SysAuthRoleMenu roleMenu = new SysAuthRoleMenu(authRole.getId(),Long.valueOf(menuIdStr));  
 sysAuthRoleMenuDAO.save(roleMenu);  
 }  
 }  
}

# 3、数据库表

### 1、表 sys\_auth\_role

@Id  
@GeneratedValue(strategy = GenerationType.***IDENTITY***)  
private Long id;  
private Long pid; //父级id  
private String code; //code  
private String name; //显示名称  
private Integer sorts; //排序  
private Integer status;  
private Integer dataType; //数据类型 1 查看所有 0 非查看所有  
private Integer rank; //角色的层级

### 2、角色表sys\_auth\_role



# 4、给用户创建角色

## 1、前端展示



## 2、后端代码操作

## 3、contrler SysAuth3RoleController.java .//checked 字段表示 是否选中，上面的操作其实是不需要确定，只要选中复选框就可以选中

@RequestMapping("checkUser")  
@ResponseBody  
public ResponseBean checkUser(Long roleId, Long admId, Boolean checked){  
 try {  
 sysAuth3RoleService.checkUser(roleId,admId,checked);  
 return ResponseBean.buildSuccess();  
 } catch (AppException e) {  
 return ResponseBean.buildFailure(e.getMessage());  
 }  
}

@Override  
 public void checkUser(Long roleId,Long admId,Boolean checked){  
 //先删除原有数据  
 sysAuthRoleAdminUserDAO.deleteByRoleIdAndAdmId(roleId,admId);  
 if (checked){  
 SysAuthRoleAdminUser adminUser = new SysAuthRoleAdminUser(roleId,admId);  
 sysAuthRoleAdminUserDAO.save(adminUser);  
 }

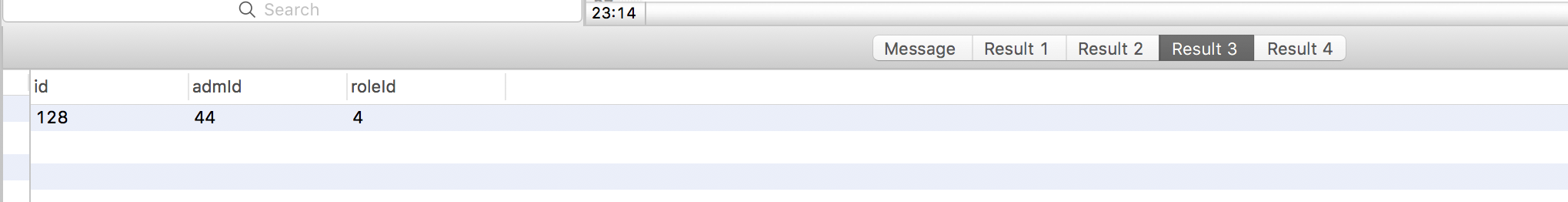
### 3、表 sys\_auth\_role\_admin\_user

@Entity  
@Table(name = "sys\_auth\_role\_admin\_user")  
public class SysAuthRoleAdminUser implements Serializable{  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.***IDENTITY***)  
 private Long id;  
 private Long roleId; //角色 管理员组，商务总监等  
 private Long admId;

## 4、数据库数据 （roleId为1 表示最高级管理员）

### 2、角色用户表 sys\_auth\_role\_admin\_user

### select \* from sys\_auth\_role\_admin\_user s where s.admid = 44;

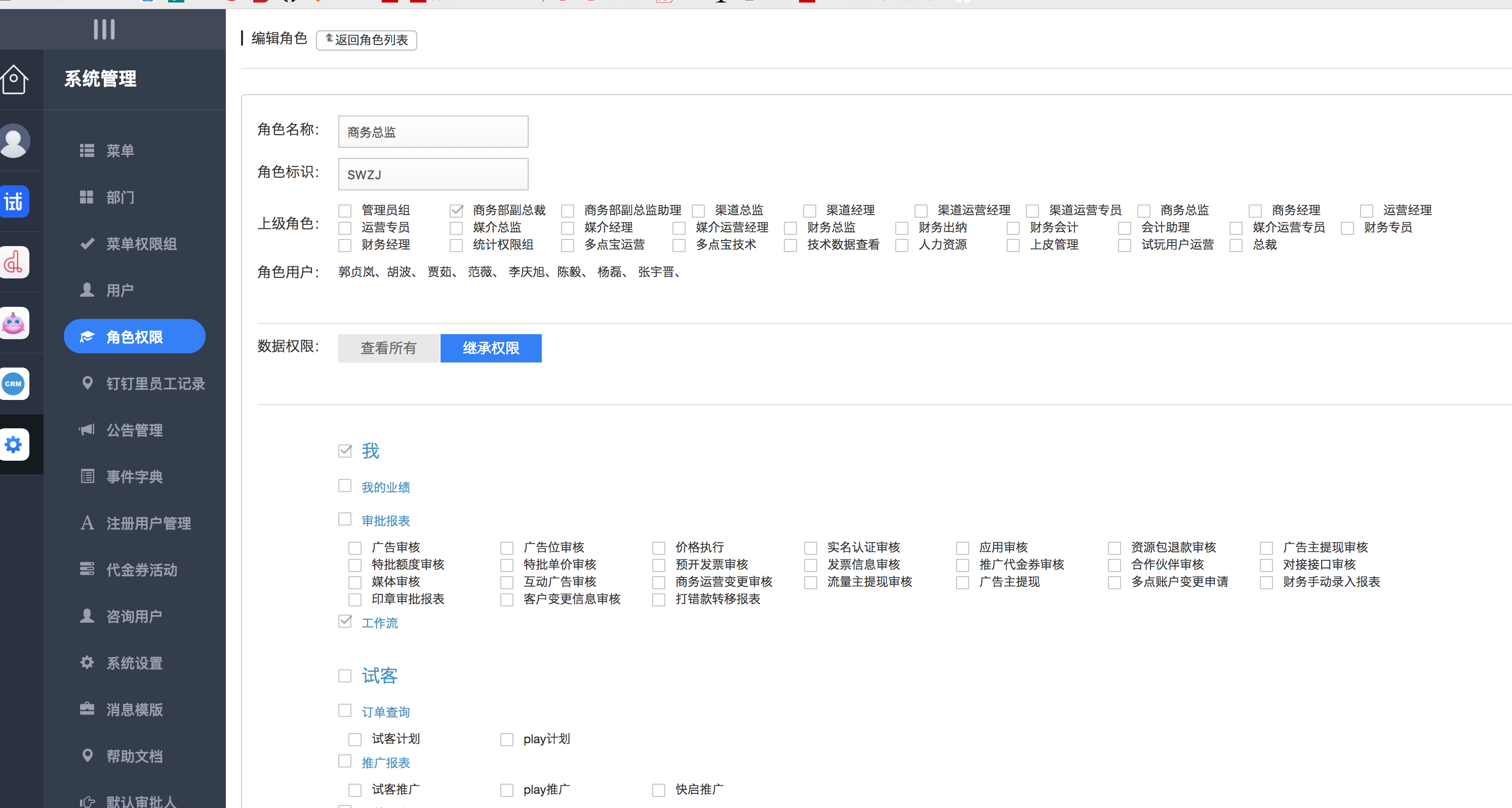


## 5、登录张宇晋查看 发现有一个工作流

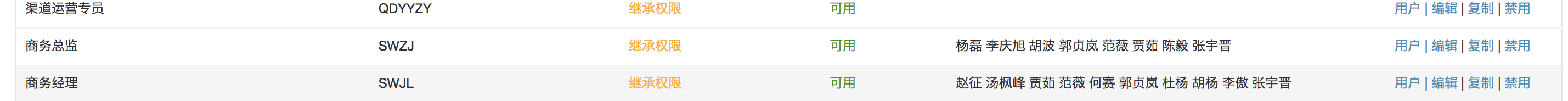


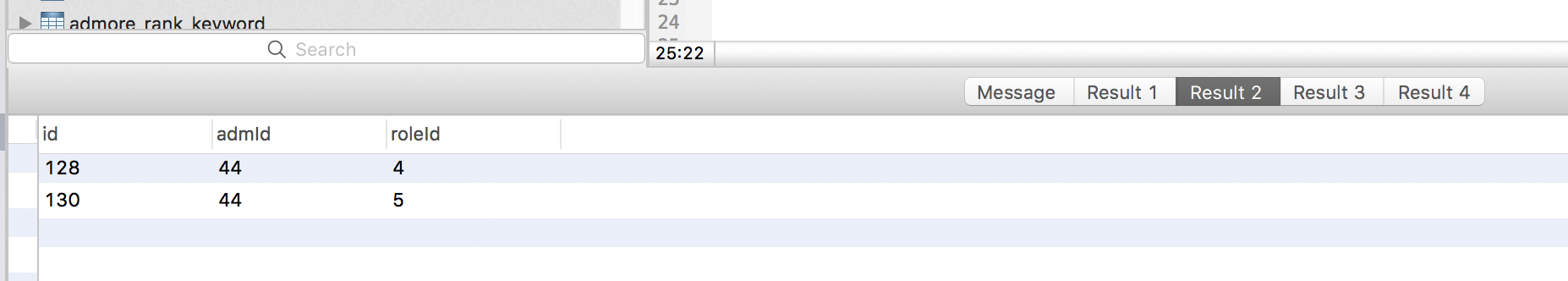
# 5、给用户张宇晋分发权限菜单

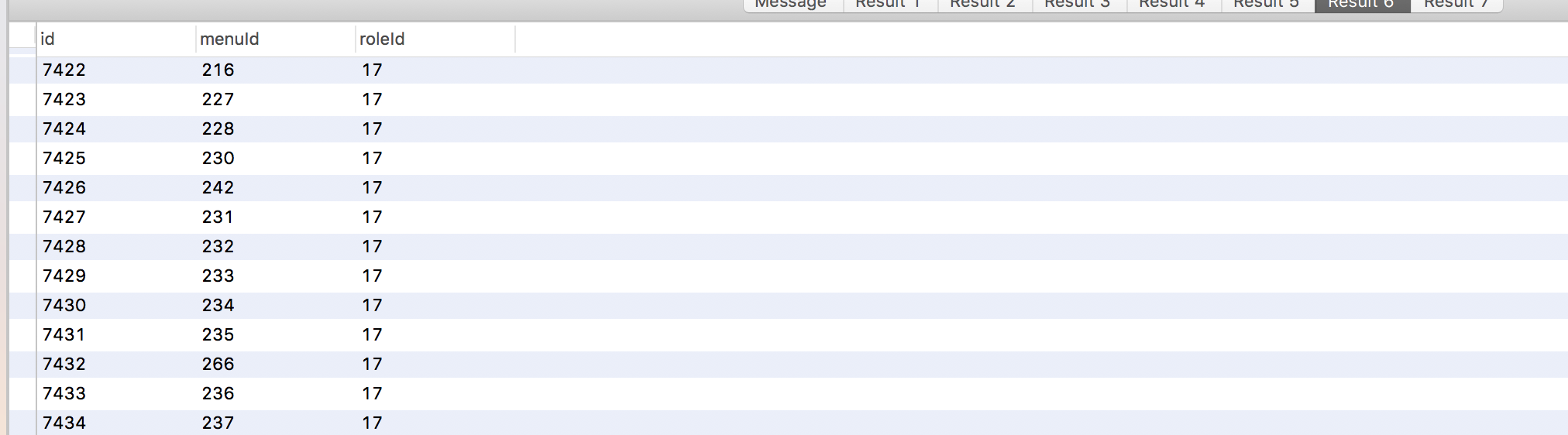
## 1、观察商务总监会发现只有一个工作流



## 2、其实这里就是给用户添加多个角色，来控制的

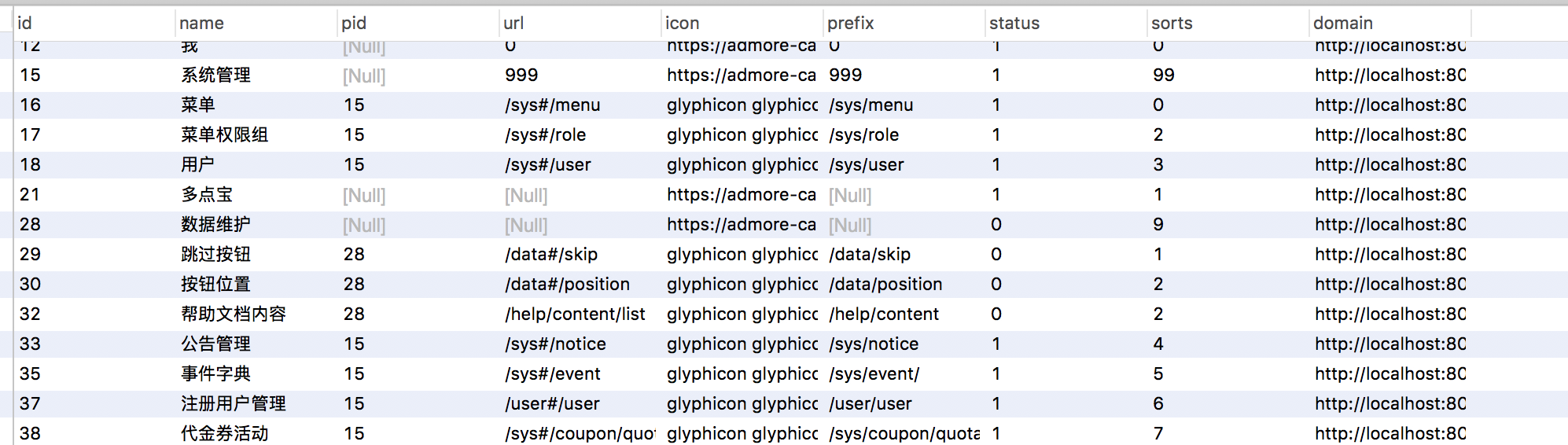






## 3、菜单表

@Id  
@GeneratedValue(strategy = GenerationType.***IDENTITY***)  
private Long id;  
private Long pid; //父级id  
private String name; //显示名称  
private String icon; //图标  
private String url; //跳转地址  
private String prefix; //权限前缀  
private Integer sorts; //排序  
private Integer status;  
private String domain; //菜单域名前缀



# 1、系统管理员用户表，包括用户名和密码 ，创建管理员用户就会到这里

select \* from sys\_admin\_user s

where s.email='yujin.zhang@duodian.com';

-- 部门表

SELECT \* from sys\_department ;

-- 管理员部门对应表

select \* from sys\_dept\_admin\_user s

WHERE s.admid = '44';

# 2、系统角色表

select \* from sys\_auth\_role;

# 3、系统角色 和 用户id表

-- private Long id;

-- private Long roleId; 角色id

-- private Long admId; 管理员 id 如果为1为则为系统管理员

select \* from sys\_auth\_role\_admin\_user s

where s.admid = 44;

# 4、角色菜单对应表

-- private Long id;

-- private Long roleId; //角色id

-- private Long menuId; //菜单id

select \* from sys\_auth\_role\_menu s

where s.roleId in(4,17) ;

-- 5、系统菜单 pid为上级

select \* from sys\_menu ;

# 6、权限拦截器

@Component  
public class PermissionInterceptor implements HandlerInterceptor {  
  
 private Logger logger = LoggerFactory.getLogger(PermissionInterceptor.class);  
  
  
 @Resource  
 private LoginService loginService;  
  
 @Override  
 public boolean preHandle(HttpServletRequest httpServletRequest, HttpServletResponse httpServletResponse, Object o) throws Exception {  
 String uri = httpServletRequest.getRequestURI();  
  
 SysAdminUser adminUser = AppSessionHelper.getSessionUser();  
 if(adminUser.getIsSuper() != null && adminUser.getIsSuper().intValue() == 1){  
 return true;  
 }  
  
 Set<String> sysProtects = loginService.getSystemProtectUrl();//选择系统的所有菜单  
 Set<String> admProtects = loginService.getAdmUserProtectUrl(adminUser.getId());//选择用户所具有的菜单权限  
  
 for (String c : sysProtects){  
 if(StringUtils.isNotBlank(c)){  
 if (uri.startsWith(c)){  
 if (admProtects.contains(c)){  
 return true;  
 } else {  
 logger.info("prefix:" + uri + ":" + c);  
 nopermission(httpServletRequest,httpServletResponse);  
 return false;  
 }  
 }  
 }  
 }  
 return true;  
 }  
  
 @Override  
 public void postHandle(HttpServletRequest httpServletRequest, HttpServletResponse httpServletResponse, Object o, ModelAndView modelAndView) throws Exception {  
  
 }  
  
 @Override  
 public void afterCompletion(HttpServletRequest httpServletRequest, HttpServletResponse httpServletResponse, Object o, Exception e) throws Exception {  
  
 }  
  
  
 private void invalidate(HttpServletRequest request,HttpServletResponse response) throws IOException {  
 request.getSession().invalidate();  
 String ajax = request.getHeader("X-Requested-With");  
 if (StringUtils.equals(ajax,"XMLHttpRequest")) {  
 //设置登陆超时header  
 response.addHeader("sessionout","true");  
 } else {  
 response.sendRedirect("/public/sessionout");  
 }  
 }  
  
 private void nopermission(HttpServletRequest request,HttpServletResponse response) throws IOException{  
 String ajax = request.getHeader("X-Requested-With");  
 if (StringUtils.equals(ajax,"XMLHttpRequest")){  
 response.addHeader("nopermission","true");  
 } else {  
 response.sendRedirect("/public/nopermission");  
 }  
 }  
}

@Override  
@Cacheable(value = CacheConstants.***CACHE\_PUBLIC***, key = "'getAdmUserProtectUrl' + #admId")  
public Set<String> getAdmUserProtectUrl(Long admId) {

Set<Long> menuIds = new HashSet<>(sysAuthRoleMenuDAO.findAdmUserMenuIds(admId));  
 Set<String> result = new HashSet<>();  
  
 for (Long menuId : menuIds){  
 SysMenu sysMenu = sysMenuDAO.findOne(menuId);  
 if (sysMenu != null && sysMenu.getStatus() != null  
 && sysMenu.getStatus().compareTo(SysMenu.***STATUS\_ENABLE***) == 0  
 && StringUtils.isNotBlank(sysMenu.getPrefix())){  
 result.add(sysMenu.getPrefix());  
 }  
 }  
 return result;  
}

@Query("select menuId from SysAuthRoleMenu where roleId in (select roleId from SysAuthRoleAdminUser where admId = :admId)")  
public List<Long> findAdmUserMenuIds(@Param("admId") Long admId);