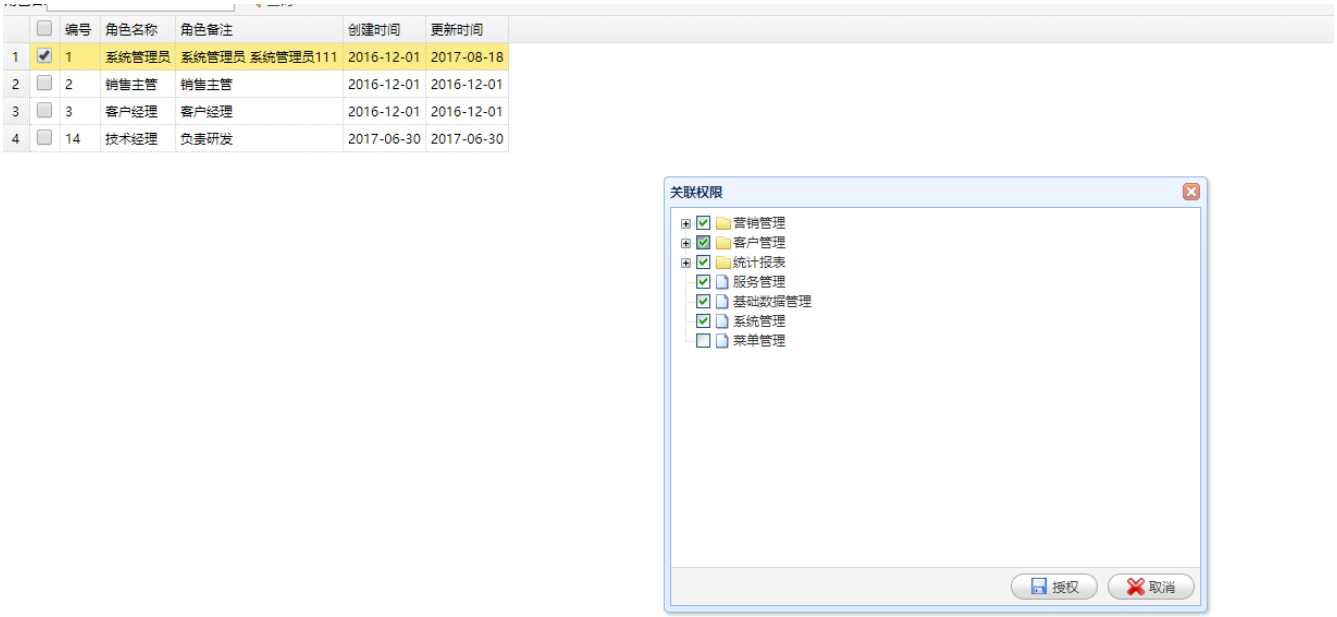
一 关联权限

1 需求分析

针对于现有的角色进行权限赋值

2 页面原型



3 编写Crntroller

```
@RequestMapping("queryAllsModuleDtos")
    @ResponseBody
    public List<ModuleDto> queryAllsModuleDtos(Integer rid) {

        return moduleService.queryAllsModuleDtos(rid);
    }
@RequestMapping("addPermission")
    @ResponseBody
    public MessageModel addPermission(Integer rid,Integer[] moduleIds){
        roleService.addPermission(rid,moduleIds);
        return success("角色授权成功");
    }
```

4 编写Service

```
public List<ModuleDto> queryAllsModuleDtos(Integer rid){

        List<ModuleDto> moduleDtos = moduleDao.queryAllsModuleDtos();

        List<Integer> moduleIds = permissionDao.queryPermissionModuleIdsByRid(rid);

        if (null != moduleDtos && moduleDtos.size() > 0) {

            for (ModuleDto moduleDto : moduleDtos) {

                Integer mid = moduleDto.getId();

                for (int i = 0; i <moduleIds.size() ; i++) {

                    if(moduleIds.get(i)==mid){
                        moduleDto.setChecked(true);// 节点选中！！！
                    }
                }

//                if (moduleIds.contains(moduleDto.getId())) {
//                    moduleDto.setChecked(true);// 节点选中！！！
//                }
            }
        }

        return moduleDtos;
    }
public void addPermission(Integer rid, Integer[] moduleIds) {
        /**
         * 1.参数合法性校验
         *    rid 角色记录必须存在
         *    moduleIds  可空
         * 2.删除原始权限
         *     查询原始权限
         *         原始权限存在  执行删除操作
         * 3. 执行批量添加
         *     根据moduleId  查询每个模块  权限值
         *     List<Permission>
         */
        AssertUtil.isTrue(null==rid||null==roleDao.queryRoleById(rid), "待授权的角色不存在!");
        int count=permissionDao.queryPermissionCountByRid(rid);
        if(count>0){
            AssertUtil.isTrue(permissionDao.deletePermissionByRid(rid)<count,
CrmConstant.OPS_FAILED_MSG);
        }
        List<Permission> permissions=null;
        if(null!=moduleIds&&moduleIds.length>0){
            /**
             * 执行批量添加
             */
            permissions=new ArrayList<>();
            Module module=null;
            for(Integer moduleId:moduleIds){
```

```
                module=moduleDao.queryModuleById(moduleId);
                module.getOptValue();
                Permission permission=new Permission();
                if(null !=module){
                    permission.setAclValue(module.getOptValue());
                }
                permission.setRoleId(rid);
                permission.setModuleId(moduleId);
                permission.setCreateDate(new Date());
                permission.setUpdateDate(new Date());
                permissions.add(permission);
            }
            AssertUtil.isTrue(permissionDao.insertBatch(permissions)<moduleIds.length,
CrmConstant.OPS_FAILED_MSG);
        }
    }
```

5 编写Dao

```
public List<ModuleDto> queryAllsModuleDtos();
@Select("select module_id from t_permission where role_id=#{rid}")
public List<Integer> queryPermissionModuleIdsByRid(@Param("rid") Integer rid);
public Integer insertBatch(List<Permission> permissions);
```

6 编写Mapper

```
<select id="queryAllsModuleDtos" resultType="moduleDto">
        SELECT
            id,
            parent_id AS pId,
            module_name AS NAME
        FROM
            t_module
        WHERE
            is_valid = 1
    </select>
<insert id="insertBatch" parameterType="list">
      insert into t_permission(role_id,module_id,acl_value,create_date,update_date) values
      <foreach collection="list" item="item" separator=",">
         (#{item.roleId},#{item.moduleId},#{item.aclValue},#{item.createDate},#
{item.updateDate})
      </foreach>
   </insert>
```

7 编写Vo

```
public class Module extends BaseVO {
    private Integer id;
    private String moduleName;
```

```
    private String moduleStyle;
    private String url;
    private Integer parentId;
    private Integer grade;
    private String optValue;
    private Integer orders;

    private String parentModuleName;

    public String getParentModuleName() {
        return parentModuleName;
    }
    public void setParentModuleName(String parentModuleName) {
        this.parentModuleName = parentModuleName;
    }
    public Integer getId() {
        return id;
    }
    public void setId(Integer id) {
        this.id = id;
    }
    public String getModuleName() {
        return moduleName;
    }
    public void setModuleName(String moduleName) {
        this.moduleName = moduleName;
    }
    public String getModuleStyle() {
        return moduleStyle;
    }
    public void setModuleStyle(String moduleStyle) {
        this.moduleStyle = moduleStyle;
    }
    public String getUrl() {
        return url;
    }
    public void setUrl(String url) {
        this.url = url;
    }
    public Integer getParentId() {
        return parentId;
    }
    public void setParentId(Integer parentId) {
        this.parentId = parentId;
    }
    public Integer getGrade() {
        return grade;
    }
    public void setGrade(Integer grade) {
        this.grade = grade;
    }
    public String getOptValue() {
        return optValue;
```

```
    }
    public void setOptValue(String optValue) {
        this.optValue = optValue;
    }
    public Integer getOrders() {
        return orders;
    }
    public void setOrders(Integer orders) {
        this.orders = orders;
    }
}
```

8 前台js

```
/**
 * 打开关联权限对话框
 */
function openRelatePermissionDlg(){

    var rows=$("#dg").datagrid("getSelections");
    if(rows.length==0){
        $.messager.alert("来自crm","请选择角色进行授权!","info");
        return;
    }

    if(rows.length>1){
        $.messager.alert("来自crm","只能选择一条角色进行授权!","info");
        return;
    }

    $("#rid").val(rows[0].id);

    loadModuleData();

    $("#dlg02").dialog("open");
}


function zTreeOnCheck() {
    var znodes=ztreeObj.getCheckedNodes(true);
    var moduleIds="moduleIds=";
    if(znodes.length>0){
        for(var i=0;i<znodes.length;i++){
            if(i<=znodes.length-2){
                moduleIds=moduleIds+znodes[i].id+"&moduleIds=";
            }else{
                moduleIds=moduleIds+znodes[i].id;
            }
        }
    }
    console.log(moduleIds);
    $("#moduleIds").val(moduleIds);
```

```
}

var ztreeObj;
function loadModuleData(){
    $.ajax({
        type:"post",
        url:ctx+"/module/queryAllsModuleDtos",
        data:"rid="+$("#rid").val(),
        dataType:"json",
        success:function(data){
            // zTree 的参数配置，深入使用请参考 API 文档（setting 配置详解）
            var setting = {
                check: {
                    enable: true
                },
                data: {
                    simpleData: {
                        enable: true
                    }
                },
                callback: {
                    onCheck: zTreeOnCheck
                }
            };
            // zTree 的数据属性，深入使用请参考 API 文档（zTreeNode 节点数据详解）
            var zNodes =data;
            ztreeObj= $.fn.zTree.init($("#treeDemo"), setting, zNodes);
        }
    });
}

function closeDialog02() {
    $("#dlg02").dialog("close");
}

/**
 * 添加权限
 */
function addPermission(){
    /**
     * 1.角色id
     * 2.资源id
     */
    $.ajax({
        type:"post",
        url:ctx+"/role/addPermission",
        data:"rid="+$("#rid").val()+"&"+$("#moduleIds").val(),
        dataType:"json",
        success:function(data){
            console.log(data);
            if(data.code==200){
                $.messager.alert("来自crm",data.msg,"info");
                $("#moduleIds").val("");
```

```
                $("#rid").val("");
                closeDialog02();
            }else{
                $.messager.alert("来自crm",data.msg,"info");
            }
        }
    });
}
```