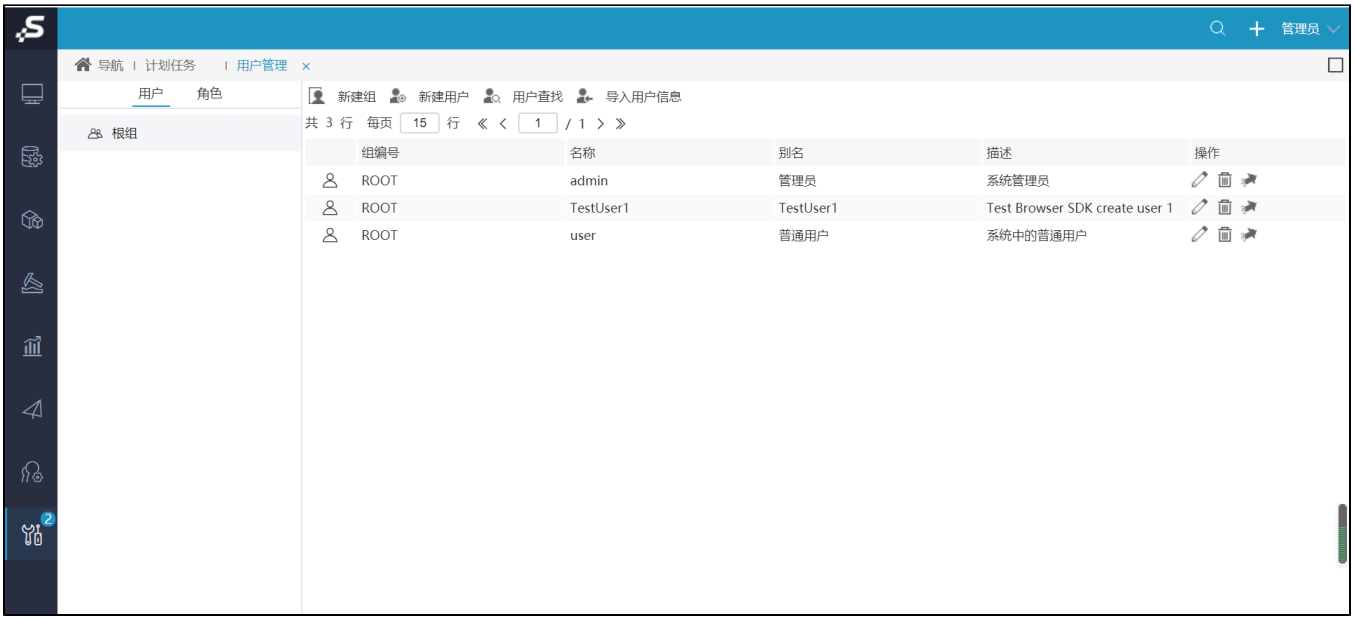
通过计划任务同步用户及机构

> 注意：该文档中的更新和覆盖机制，默认覆盖全部用户所属组。

1. 说明

Smartbi提供创建自定义计划任务，调用 SDK 接口方法，进行用户和机构信息同步的功能。



2. 具体操作

1、在系统运维-》计划任务-》任务-》新建任务，**任务类型** 选择“**定制**”，然后将如下代码粘贴到代码编辑区，根据实际业务逻辑修改。

以下自定义计划任务代码是基于已设置好的数据源ID，将第三方数据库中的用户，机构，角色等信息同步到Smartbi的知识库中。

```
importPackage(Packages.java.io);
importPackage(Packages.java.lang);
importPackage(Packages.java.util);
importPackage(Packages.smartbi.usermanager);
importPackage(Packages.smartbi.sdk);
importPackage(Packages.smartbi.sdk.service.user);
importPackage(Packages.smartbi.sdk.service.datasource);

/**
 *
 *
 * needTopGroup""
 *       needTopGrouptruetopGroupName
 *       needTopGroupfalse""
 */
var needTopGroup = true;
var topGroupName = "YourTopGroupName";
/**
 * IDsqlGroupsqlRolesqlUserSQL
 */
var datasrcId = "DS.test";
/**
 * SQL<br/>
 * SQL
 *
 *       orgName
 *       orgAlias
 *       orgParentName
```

```
 *      isUse
 *      orgCode:
 *      orgParentCode
 */
var sqlGroup = "select orgName, orgAlias, orgParentName, isUse, orgCode, orgParentCode from org";
/**
 * SQL<br/>
 * SQL
 *      roleName
 *      roleAlias
 *      orgName
 *      orgCode
 */
var sqlRole = "select roleName, roleAlias, orgName from roles";
/**
 * SQL <br/>
 * SQL
 *      userId : id
 *      userName
 *      userAlias
 *      orgName orgName#org1#org2#org3
 *      isUse
 *      userPwd
 *      roleName roleName#role1#role2#role3
 *      orgCode orgCode#org1#org2#org3
 */
var sqlUser = "select userId, userName, userAlias, orgName, isUse, userPwd, roleName, orgCode from orgUser
where  userName not in ('admin','scheduleAdmin','service')"; //smartbi"'admin','scheduleAdmin','service'"
/**
 *
 *      true
 *      false
 */
var removeAllAssignedRoles = true;

/**
 *
 *
 * synchronizeRoleFromOtherDb
 *
 */
var topGroup = getTopGroup(needTopGroup, topGroupName);
synchronizeGroupFromOtherDb(datasrcId, sqlGroup, topGroup);
synchronizeRoleFromOtherDb(datasrcId, sqlRole, topGroup);
synchronizeUserFromOtherDb(datasrcId, sqlUser, topGroup, removeAllAssignedRoles);
logger.info("\n\n============================");

/**
 * ""
 *
 * @param needTopGroup
 *            ""false""
 * @param topGroupName
 *            needTopGrouptruetopGroupName
 * @returns
 */
function getTopGroup(needTopGroup, topGroupName) {
    var usrManagerService = new UserManagerService(connector);

    var topGroup = usrManagerService.getDepartmentById("DEPARTMENT"); //
    if (needTopGroup && topGroupName) {
        topGroup = usrManagerService.getDepartmentByName(topGroupName);
        if (!topGroup) {
            var groupId = usrManagerService.createDepartment("DEPARTMENT", topGroupName, topGroupName, "",
""); //
            topGroup = usrManagerService.getDepartmentById(groupId);
        }
    }
    return topGroup;
}
```

```
/**
 *
 *
 * @param dsId
 *            ID
 * @param sqlGroup
 *            SQL
 * @param topGroup
 *
 */
function synchronizeGroupFromOtherDb(dsId, sqlGroup, topGroup) {
    var usrManagerService = new UserManagerService(connector);
    var datasrcService = new DataSourceService(connector);

    var gridDataGroup = datasrcService.executeNoCacheable(dsId, sqlGroup);
    for (var i = 0; i < gridDataGroup.getRowsCount(); i++) {
        var orgName = gridDataGroup.get(i, 0).getValue();
        var orgAlias = gridDataGroup.get(i, 1).getValue();
        var orgParentName = gridDataGroup.get(i, 2).getValue();
        var isUse = gridDataGroup.get(i, 3).getValue();
        var orgCode = gridDataGroup.get(i, 4).getValue();
        var orgParentCode = gridDataGroup.get(i, 5).getValue();
        logger.info("=====///" + orgName + "/" + orgAlias + "/" + orgParentName + "/"
                + isUse);
        //
        var parentGroup = usrManagerService.getDepartmentByCode(orgParentCode);
        if (!parentGroup) {
            parentGroup = topGroup;
        }

        //
        var group = usrManagerService.getDepartmentByCode(orgCode);
        if (!group) {
            //
            usrManagerService.createDepartment(parentGroup.getId(), orgName, orgAlias, "", orgCode);
        } else {
            //
            usrManagerService.updateDepartment(group.getId(), orgAlias,"", orgCode);
            //
            if (parentGroup.getId() != usrManagerService.getParentDepartment(group.getId()).getId()) {
                usrManagerService.moveDepartment(group.getId(), parentGroup.getId());
            }
        }
    }
}

/**
 *
 *
 * @param dsId
 *            ID
 * @param sqlRole
 *            SQL
 * @param topGroup
 *
 */
function synchronizeRoleFromOtherDb(dsId, sqlRole, topGroup) {
    var usrManagerService = new UserManagerService(connector);
    var datasrcService = new DataSourceService(connector);

    var gridDataRole = datasrcService.executeNoCacheable(dsId, sqlRole);
    for (var i = 0; i < gridDataRole.getRowsCount(); i++) {
        var roleName = gridDataRole.get(i, 0).getValue();
        var roleAlias = gridDataRole.get(i, 1).getValue();
        var orgName = gridDataRole.get(i, 2).getValue();
        logger.info("=====//" + roleName + "/" + roleAlias + "/" + orgName);

        //
        var group = usrManagerService.getDepartmentByName(orgName || "NonEmptyOrgName");
        if (!group) {
            group = topGroup;
```

```
        }

        //
        var role = usrManagerService.getRoleByName2(roleName);
        if (!role) {
            //
            usrManagerService.createRole(roleName, roleAlias, "", group.getId());
        } else {
            //
            usrManagerService.updateRole(role.getId(), roleAlias, "");
        }
    }
}

/**
 *
 *
 * @param dsId
 *            ID
 * @param sqlUser
 *            SQL
 * @param topGroup
 *
 * @param removeAllAssignedRoles
 *            false
 */
function synchronizeUserFromOtherDb(dsId, sqlUser, topGroup, removeAllAssignedRoles) {
    var usrManagerService = new UserManagerService(connector);
    var datasrcService = new DataSourceService(connector);

    var gridDataUser = datasrcService.executeNoCacheable(dsId, sqlUser);
    for (var i = 0; i < gridDataUser.getRowsCount(); i++) {
        var userId = gridDataUser.get(i, 0).getValue();
        var userName = gridDataUser.get(i, 1).getValue();
        var userAlias = gridDataUser.get(i, 2).getValue();
        var orgName = gridDataUser.get(i, 3).getValue();
        var isUse = gridDataUser.get(i, 4).getValue();
        var userPwd = gridDataUser.get(i, 5).getValue();
        var roleName = gridDataUser.get(i, 6).getValue();
        var orgCode = gridDataUser.get(i, 7).getValue();
        logger.info("=====/////" + userName + "/" + userAlias + "/" + orgName + "/"
                + isUse + "/" + userPwd + "/" + roleName);

        //
        isUse = ("" + isUse).toLowerCase();
        isUse = (isUse === "0" || isUse === "false" || !isUse) ? false : true;

        // ID
        var usrOrgIdList = [];
        var usrOrgNameList = (orgName || "").split("#");
        var usrOrgCodeList = (orgCode || "").split("#");
        for (var j = 0; j < usrOrgCodeList.length; j++) {
            var parentGroup = usrManagerService.getDepartmentByCode(usrOrgCodeList[j] || "NonEmptyOrgCode");
            if (parentGroup) {
                usrOrgIdList.push(parentGroup.getId());
            }
        }
        // topGroup
        if (usrOrgIdList.length < 1) {
            usrOrgIdList.push(topGroup.getId());
        }

        //
        var user = usrManagerService.getUserByName(userName);
        if (!user) {
            usrManagerService.createUserById(usrOrgIdList[0], userId, userName, userAlias, "", userPwd,
isUse);
            user = usrManagerService.getUserByName(userName);
        }
        //
        usrManagerService.assignDepartmentsToUser(user.getId(), usrOrgIdList);
```

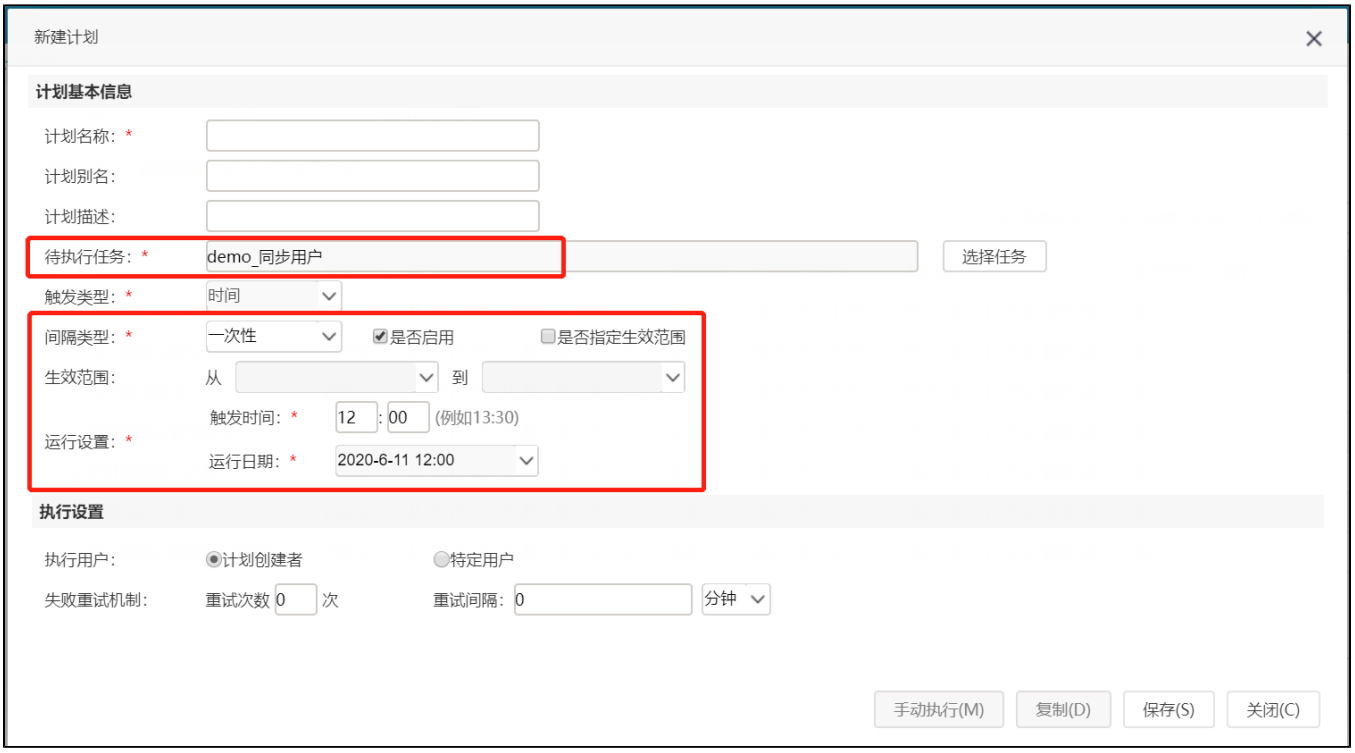
```
        //
        var usrRoleIdList = [];
        var usrRoleNameList = (roleName || "").split("#");
        for (var k = 0; k < usrRoleNameList.length; k++) {
            var userRole = usrManagerService.getRoleByName2(usrRoleNameList[k]);
            if (userRole) {
                usrRoleIdList.push(userRole.getId());
            }
        }
        if (!removeAllAssignedRoles) {
            //
            var oldAssignedRoles = usrManagerService.getAssignedRolesOfUser(user.getId());
            for (var m = 0; m < oldAssignedRoles.size(); m++) {
                var oldRoleId = oldAssignedRoles.get(m).getId();
                if (usrRoleIdList.join("###").indexOf(oldRoleId) < 0)
                    usrRoleIdList.push(oldRoleId);
            }
        }
        usrManagerService.assignRolesToUser(user.getId(), usrRoleIdList);

        // updateUserMD5smartbi
        // usrManagerService.updateUser(user.getId(), userAlias, "", userPwd, isUse);
        //
        // updateUserByEncryptedPassword
        // "0" + MD5; "1" + DES; "2" +
        // MD5updateUserByEncryptedPassword"0"Smartbi
        // DESupdateUserByEncryptedPassword"1"Smartbi
        // updateUserByEncryptedPassword"2"Smartbi2
        var encryptedPwd = "0" + userPwd;
        usrManagerService.updateUserByEncryptedPassword(user.getId(), userAlias, "", encryptedPwd, isUse);
    }
}
```

注意：请参考代码中的注释，修改其中的必要信息，比如下面这些。

- var needTopGroup ： 是否需要顶级机构。
- var topGroupName ： 顶级机构名称。
- var datasrcId ： 数据源的ID值。需要在“定制管理 -> 数据管理 -> 数据源”中新建关系数据源，所有的SQL需要在该数据源上执行。
- var sqlGroup ： 用来同步机构的SQL语句。
- var sqlRole ： 用来同步角色的SQL语句。
- var sqlUser ： 原来同步用户的SQL语句。
- var removeAllAssignedRoles ： 同步时是否删除已有角色。

2、保存创建的任务，参考以下计划 创建一个执行计划，设置定时执行同步任务。



3、gif演示示例的源码请参考： 通过计划任务自动同步用户机构和角色.zip