## ER图

erDiagram  
 TENANT ||--|| ORG\_CONFIG : ont-to-one  
 TENANT ||--|{ SERVICE\_INFO : ont-to-many  
 ORG\_CONFIG ||--|{ FINE\_AUTH\_PERM\_RELATION : one-to-many  
 SERVICE\_INFO ||--|{ FUNCTION : ont-to-many  
 SERVICE\_INFO }|--|| FINE\_AUTH\_PERM\_RELATION : many-to-one  
 SERVICE\_INFO }|--|| SERVICE\_CODE\_PAC\_MAP : many-to-one  
 FUNCTION }|--|| FINE\_AUTH\_PERM\_RELATION : many-to-one  
 FINE\_AUTH\_ADMIN }|--|{ FINE\_AUTH\_ROLE :many-to-many  
 FINE\_AUTH\_ROLE }|--|{ FINE\_AUTH\_PERM :many-to-many  
 FINE\_AUTH\_ADMIN ||--|| MDM\_USER : one-to-one  
 MDM\_USER ||--|{ UNI\_AUTH\_AUTHORIZATION : ont-to-many  
 UNI\_AUTH\_AUTHORIZATION }|--|| UNI\_AUTH\_APP : many-to-one  
 FINE\_AUTH\_PERM\_RELATION ||--|| FINE\_AUTH\_PERM : one-to-one  
 UNI\_AUTH\_APP }|--|| SERVICE\_CODE\_PAC\_MAP : many-to-one  
 SERVICE\_CODE\_PAC\_MAP ||--|| FINE\_AUTH\_PERM\_RELATION : one-to-one  
 FINE\_AUTH\_ROLE ||--|{ FINE\_AUTH\_RECORD : one-to-many   
   
 FINE\_AUTH\_RECORD {  
 string user\_id  
 string role\_id  
 string manage\_org\_id  
 }  
   
 UNI\_AUTH\_AUTHORIZATION {  
 string user\_id  
 string app\_id  
 }  
 UNI\_AUTH\_AUTHORIZATION\_LOG {  
 string user\_id  
 string app\_id  
 }  
 MDM\_ADMIN {  
 string account  
 }  
 MDM\_USER {  
 string id  
 string account  
 }  
 UNI\_AUTH\_APP {  
 string id  
 string app\_package  
 string def\_code\_real  
 }  
 SERVICE\_CODE\_PAC\_MAP {  
 string code  
 string package\_name  
 }   
 FINE\_AUTH\_ADMIN {  
 string id  
 string user\_account  
 string op\_tenant\_id  
 }   
 ORG\_CONFIG {  
 string id  
 string func\_ids  
 string nop\_tenant\_id  
 }   
 SERVICE\_INFO {  
 string id  
 string code  
 string tenant\_id  
 }  
 FUNCTION {  
 string id  
 string code  
 string tenant\_id  
 }  
 FINE\_AUTH\_PERM {  
 long id  
 string module  
 string category  
 string menu  
 string code  
 string name  
 string type  
 long seq  
 }  
 FINE\_AUTH\_PERM\_RELATION {  
 string id  
 string type  
 string type\_code  
 long FINE\_AUTH\_PERM\_id  
 }  
 FINE\_AUTH\_ROLE {  
 long id  
 string tennant\_id  
 }

## 说明

1. 系统初始化的时候
   1. 初始化FINE\_AUTH\_PERM,FINE\_AUTH\_PERM\_RELATION
2. 后续增加按钮,菜单等权限控制信息的时候
   1. 需要增加对应的FINE\_AUTH\_PERM,FINE\_AUTH\_PERM\_RELATION
   2. 对于老租户, 需要手动触发租户的变更, 才会把新增的权限同步到租户管理员上
   3. 租户下的其它管理员不会拥有新增的权限, 需手动添加
3. 租户创建和修改时
   1. 创建或更新对应的fine\_auth\_admin和fine\_auth\_role(系统角色)
   2. 创建和更新admin和role
   3. 创建和更新role和perm的关系
      1. tenant >> service\_info >> fine\_auth\_perm\_relation >> fine\_auth\_perm
      2. tenant >> service\_info >> function >> fine\_auth\_perm\_relation >> fine\_auth\_perm
      3. tenant >> org\_config >> fine\_auth\_perm\_relation >> fine\_auth\_perm
   4. 创建和更新对应的uni\_auth\_authorization记录
      1. fine\_auth\_perm >> fine\_auth\_perm\_relation >> service\_info >> service\_code\_pac\_map >> uni\_auth\_app
      2. fine\_auth\_perm >> fine\_auth\_perm\_relation >> org\_config(func\_ids) >> uni\_auth\_app
4. 增加服务,功能时
   1. 如果需要权限控制, 参考2
5. 删除服务,功能时
   1. 也需要删除对应的信息
6. 创建系统管理员
   1. uasadmin, opadmin, cdsadmin等系统管理员
   2. 手动创建管理员,角色,及管理员角色关系, 角色权限关系
7. 创建普通角色和管理员
   1. web端返回, 租户管理员对应的权限列表
   2. 选择权限, 设置角色名称, 保存
   3. 选择用户, 选择角色, 生成管理员, 保存
8. 老用户管理员生成
   1. mdm\_user >> uni\_auth\_authorization >> uni\_auth\_app >> service\_code\_pac\_map >> fine\_auth\_perm\_relation >> fine\_auth\_perm
   2. 根据上面找到的权限, 生成角色, 绑定权限
   3. 生成管理员, 绑定角色