

多租户管理系统 V1.0



多租户管理系统

源码文档

版本：V1.0

雷山碧阅科技有限公司

2024-7-11

行李托运平台 V1.0

```
1 File: src/pages/adminList/index.tsx
2 ```tsx
3 import {PlusOutlined} from '@ant-design/icons';
4 import {Button, Select, message} from 'antd';
5 import React, {useEffect, useRef, useState} from 'react';
6 import {PageContainer} from '@ant-design/pro-layout';
7 import type {ActionType, ProColumns} from '@ant-design/pro-table';
8 import ProTable from '@ant-design/pro-table';
9 import ProForm, {ModalForm, ProFormSelect, ProFormText} from '@ant-design/pro-form';
10 import {
11   addAdmin,
12   adminList,
13   roleSelect,
14   updataPassword,
15   updataStatus,
16   updateAdmin,
17 } from './service';
18 import type {TableListItem, TableListPagination} from './data';
19 import {formatTime} from '@utils';
20 import {checkPassword} from '@utils/rolue';
21 import ImgUpload from '@components/ossUpload';
22 import styles from './index.less';
23 import {history} from 'umi';
24 import {ProCard} from '@ant-design/pro-components';
25 import {listTenant} from '@api/tenant';
26
27 /**
28  * 添加节点
29  *
30  * @param fields
31  */
32 const handleAdd = async (fields: any) => {
33   const hide = message.loading('正在添加');
34   try {
35     await addAdmin({...fields});
36     hide();
37     message.success('添加成功');
38     return true;
39   } catch (error) {
40     hide();
41     return false;
42   }
43 };
44
45 /**
46  * 更新节点
47  *
48  * @param fields
49  */
50 const handleUpdate = async (fields: any, currentRow?: TableListItem) => {
51   const hide = message.loading('正在修改');
52   try {
53     await updateAdmin({
54       guid: currentRow?.guid,
```

行李托运平台 V1.0

```
1      avatar: currentRow?.currentRow,
2      ...fields,
3    });
4    hide();
5    message.success('修改成功');
6    return true;
7  } catch (error) {
8    hide();
9    return false;
10  }
11 };
12 /**
13  * 修改密码
14  * @param fields
15  * @param currentRow
16  * @returns
17  */
18 const handlePasswordUpdate = async (fields: any, currentRow?: TableListItem) => {
19   const hide = message.loading('正在修改');
20   try {
21     await updataPassword({
22       guid: currentRow?.guid,
23       ...fields,
24     });
25     hide();
26     message.success('修改成功');
27     return true;
28   } catch (error) {
29     hide();
30     return false;
31   }
32 };
33 // 添加或者修改密码表单
34 const passWordBOX = () => (
35   <>
36     <ProFormText.Password
37       // @ts-ignore
38       rules={[
39         {
40           required: true,
41           message: '用户密码为必填项',
42         },
43         ({getFieldValue}: any) => ({
44           validator(role: any, value: string) {
45             const password_value = getFieldValue('passwords');
46             const data = checkPassword(value);
47             if (!data.status) {
48               return Promise.reject(data.message);
49             }
50             if (password_value && password_value !== value) {
51               return Promise.reject('两次密码不一致');
52             }
53             return Promise.resolve();
```

行李托运平台 V1.0

```
1      },
2    },
3  ]}
4  width="md"
5  name="password"
6  label="用户密码"
7  />
8  <ProFormText.Password
9    rules={[
10      {
11        required: true,
12        message: '请确认密码',
13      },
14      ({getFieldValue}: any) => ({
15        validator(role: any, value: string) {
16          const password_value = getFieldValue('password');
17          if (password_value && password_value !== value) {
18            return Promise.reject('两次密码不一致');
19          }
20          return Promise.resolve();
21        },
22      }),
23    ]}
24    width="md"
25    name="passwords"
26    label="确认密码"
27  />
28 </>
29 );
30 // 修改或添加表单 name role avatar
31 // @ts-ignore
32 const EditBox = (tenant = 'master') => (
33   <>
34     <ProFormText
35       rules={[
36         {
37           required: true,
38           message: '用户名称为必填项',
39         },
40       ]}
41       width="md"
42       name="nick_name"
43       label="用户昵称"
44     />
45     <ProForm.Item
46       name="avatar"
47       label="头像"
48       rules={[
49         {
50           required: true,
51           message: '请上传头像',
52         },
53       ]}
```

行李托运平台 V1.0

```
1      >
2      <ImgUpload
3        needCrop={true}
4        // @ts-ignore
5        cropProps={{aspect: 1, rotate: true}}
6        maxSize={0.5}
7        pictureQuality={0.7}
8      />
9    </ProForm.Item>
10   <ProFormSelect
11     name="role_guid"
12     label="角色名称"
13     showSearch
14     width="md"
15     debounceTime={300}
16     request={async ({keyWords}) => {
17       try {
18         const data = await roleSelect({name: keyWords, tenant_id: tenant});
19         return data.map((item: { guid: string; name: string }) => {
20           return {
21             value: item.guid,
22             label: item.name,
23           };
24         });
25       } catch (error) {
26         return [];
27       }
28     }}
29     placeholder="请选择角色,输入可搜索"
30     rules={[{required: true, message: '请选择角色'}]}
31   />
32 </>
33 );
34 const TableList: React.FC = () => {
35   const [query] = useState<any>(history.location.query);
36   /** 新建窗口的弹窗 */
37   const [createModalVisible, handleModalVisible] = useState<boolean>(false);
38   const [tablePage, handleTablePage] = useState<string>('saas');
39   const [updateModalVisible, handleUpdateModalVisible] = useState<boolean>(false);
40   const [updatePasswordModalVisible, handleupdatePasswordModalVisible] = useState<boolean>(false);
41   const actionRef = useRef<ActionType>();
42   const [currentRow, setCurrentRow] = useState<TableListItem>();
43   const [tenantList, setTenantList] = useState<[]>();
44   const [currentTenant, setCurrentTenant] = useState<string>('');
45   const columns: ProColumns<TableListItem>[] = [
46     {
47       title: 'guid',
48       width: 150,
49       dataIndex: 'guid',
50       ellipsis: true,
51       copyable: true,
52     },
53   ];
```

行李托运平台 V1.0

```
1      {
2        title: '账号',
3        dataIndex: 'admin_name',
4        ellipsis: true,
5      },
6      {
7        title: '用户昵称',
8        dataIndex: 'nick_name',
9        ellipsis: true,
10     },
11     {
12       title: '用户头像',
13       dataIndex: 'avatar',
14       search: false,
15       render: (_, any, record: TableListItem) => [
16         <img key="avatar" className={styles.avatarImg} src={record.avatar} alt="头像"/>,
17       ],
18     },
19     {
20       title: '角色',
21       dataIndex: 'role_name',
22       ellipsis: true,
23     },
24     {
25       title: '状态',
26       dataIndex: 'status',
27       hideInForm: true,
28       initialValue: '1',
29       valueEnum: {
30         1: {
31           text: '正常',
32           status: 'Processing',
33         },
34         2: {
35           text: '关闭',
36           status: 'Default',
37         },
38       },
39     },
40     {
41       title: '创建时间',
42       dataIndex: 'created_at',
43       search: false,
44       renderText: (val: number) => formatTime(val, ''),
45     },
46     {
47       title: '最后登录时间',
48       dataIndex: 'last_time',
49       search: false,
50       renderText: (val: number) => formatTime(val, ''),
51     },
52     {
53       title: '操作',
```

行李托运平台 V1.0

```
1      dataIndex: 'option',
2      valueType: 'option',
3      width: 250,
4      render: (_, record: TableListItem) => [
5        <Button
6          key="config"
7          type="link"
8          onClick={() => {
9            setCurrentRow(record);
10            handleUpdateModalVisible(true);
11          }}
12        >
13          编辑
14        </Button>,
15        <Button
16          key="password"
17          type="link"
18          onClick={() => {
19            setCurrentRow(record);
20            handleupdatePasswordModalVisible(true);
21          }}
22        >
23          修改密码
24        </Button>,
25        <Button
26          key="status"
27          type="link"
28          danger={record.status === 1}
29          onClick={() => {
30            updateStatus({tenant_id: currentTenant, guid: record.guid, status: record.status === 1 ? 2 : 1})
31              .then(() => {
32                message.success('修改成功');
33                if (actionRef.current) {
34                  actionRef.current.reload();
35                }
36              })
37              .catch(() => {
38              });
39            }}
40        >
41          {record.status === 1 ? '禁止' : '启用'}
42        </Button>,
43      ],
44    },
45  ];
46  const getTenantList = async () => {
47    let data = await listTenant({});
48    if (data) {
49      data = data.data?.map((item: any) => {
50        return {label: item.title, value: item.id}
51      })
52      setTenantList(data);
53    }
```

行李托运平台 V1.0

```
1      };
2      useEffect(() => {
3          getTenantList();
4      }, []);
5
6      return (
7          <PageContainer
8              tabList={[
9                  {
10                      tab: 'saas 管理员管理',
11                      key: 'saas',
12                  },
13                  {
14                      tab: '租户管理员管理',
15                      key: 'tenant',
16                  },
17              ]}
18              onTabChange={value => {
19                  handleTablePage(value)
20              }}
21          >
22              {tablePage == 'saas' ? null : <ProCard bordered>
23                  <span
24                      style={{marginLeft: 48}}>
25                      当前租户 :
26                  </span>
27                  <Select
28                      value={currentTenant}
29                      style={{width: 280, marginLeft: 8}}
30                      onChange={async value => {
31                          setCurrentTenant(value);
32                          actionRef.current?.reload()
33                      }}
34                      options={tenantList}
35                  />
36              </ProCard>
37              <ProTable<TableListItem, TableListPagination>
38                  key={tablePage}
39                  headerTitle="查询表格"
40                  actionRef={actionRef}
41                  // style={{marginTop: 2}}
42                  rowKey="guid"
43                  search={{
44                      labelWidth: 120,
45                  }}
46                  toolBarRender={() => [
47                      <Button
48                          type="primary"
49                          key="primary"
50                          onClick={() => {
51                              handleModalVisible(true);
52                          }}
53                      />
```


行李托运平台 V1.0

```
1      }}
2      >
3      <PlusOutlined/> 新建
4    </Button>,
5  ]}
6  request={async (params) => {
7    if (tablePage === 'saas') {
8      params.tenant_id = 'master';
9    } else {
10      if (currentTenant === '') {
11        //@ts-ignore
12        setCurrentTenant(tenantList[0].value)
13        //@ts-ignore
14        params.tenant_id = tenantList[0].value;
15      } else {
16        params.tenant_id = currentTenant;
17      }
18    }
19
20    if (query.guid) {
21      params['a.guid'] = query.guid;
22    }
23
24    const msg = await adminList(params);
25    return {
26      data: msg.list,
27      success: true,
28      total: msg.count,
29    };
30  }}
31  columns={columns}
32 />
33
34 {/* 新建 modal */}
35 <ModalForm
36   title="新建"
37   width="600px"
38   visible={createModalVisible}
39   onVisibleChange={handleModalVisible}
40   modalProps={{
41     destroyOnClose: true,
42   }}
43   onFinish={async (value) => {
44     const success = await handleAdd({
45       tenant_id: tablePage === 'saas' ? 'master' : currentTenant,
46       avatar: "", ...value
47     });
48     if (success) {
49       handleModalVisible(false);
50       if (actionRef.current) {
51         actionRef.current.reload();
52       }
53     }
54   }
55 }
```

行李托运平台 V1.0

```
1      }}
2    >
3      <ProFormText
4        rules={[
5          {
6            required: true,
7            message: '账号为必填项',
8          },
9        ]}
10       width="md"
11       name="admin_name"
12       label="账号"
13     />
14     { /* 与编辑组件重复抽离 */ }
15     {EditBox(tablePage == 'saas' ? 'master' : currentTenant)}
16     { /* 与密码组件重复抽离 */ }
17     {passWordBOX()}
18   </ModalForm>
19   { /* 编辑 */ }
20   <ModalForm
21     title="编辑"
22     width="600px"
23     visible={updateModalVisible}
24     onVisibleChange={handleUpdateModalVisible}
25     modalProps={{
26       destroyOnClose: true,
27     }}
28     initialValues={{
29       ...currentRow,
30     }}
31     onFinish={async (value) => {
32       value.tenant_id = currentTenant;
33       const success = await handleUpdate(value, currentRow);
34       if (success) {
35         handleUpdateModalVisible(false);
36         setCurrentRow(undefined);
37         if (actionRef.current) {
38           await actionRef.current.reload();
39         }
40       }
41     }}
42   >
43     {EditBox(tablePage == 'saas' ? 'master' : currentTenant)}
44   </ModalForm>
45   { /* 修改密码 */ }
46   <ModalForm
47     title="编辑"
48     width="600px"
49     visible={updatePasswordModalVisible}
50     onVisibleChange={handleupdatePasswordModalVisible}
51     initialValues={{
52       password: "",
53       passwords: "",
```

行李托运平台 V1.0

```
1      }}
2      modalProps={{
3          destroyOnClose: true,
4      }}
5      onFinish={async (value) => {
6          value.tenant_id = currentTenant;
7          const success = await handlePasswordUpdate(value, currentRow);
8
9          if (success) {
10              handleupdatePasswordModalVisible(false);
11              setCurrentRow(undefined);
12              if (actionRef.current) {
13                  actionRef.current.reload();
14              }
15          }
16      }}
17  >
18      {passWordBOX()}
19  </ModalForm>
20 </PageContainer>
21 );
22 };
23 export default TableList;
24
25 ```
26 File: src/pages/case/index.tsx
27 ```tsx
28 import {ModalForm, ProFormDependency, ProFormSelect, ProFormText} from "@ant-design/pro-form"
29 import {PageContainer} from "@ant-design/pro-layout";
30 import type { InputRef, TableColumnsType, TableColumnType} from "antd";
31 import {Button, Form, Input, message, Space, Table} from "antd";
32 import {useRef, useState} from "react";
33 import Highlighter from 'react-highlight-words';
34 import type {FilterDropdownProps} from "antd/es/table/interface";
35 import {SearchOutlined} from "@ant-design/icons";
36 import {deleteRedis} from "@pages/case/service";
37
38 // 类别产品模版
39 export const productTemplate = [
40     {
41         label: '快递零售',
42         value: 'entity_retail',
43         status: {30: '待确认', 31: '待发货', 32: '待收货'},
44     },
45     {label: '酒店住宿', value: 'hotel_lodging', status: {41: '待确认', 42: '已确认'}},
46     {label: '各类门票', value: 'travel_ticket', status: {51: '待出行'}},
47     {label: '线下核销', value: 'catering', status: {61: '待使用'}},
48     {label: '剧本/道具/大礼包', value: 'props', status: {71: '待发放'}},
49     {label: '旅游线路', value: 'tourism', status: {81: '待出行'}},
50 ];
51 const caseTypeList = [
52     {
53         value: 'productInfo',
```

行李托运平台 V1.0

```
1      param: '产品 id',
2      example: '25f2e9c4f1844f1faa154dd41ca40122',
3      label: '商品信息缓存',
4      page: "展示该商品相关的全部页面",
5      effect: '存储商品的信息',
6      textInput: true
7    },
8    {
9      value: 'product_status_search',
10     label: '已上架商品缓存列表',
11     page: "全部商品相关的页面",
12     effect: '存储全部已经上架的商品 id',
13   },
14   {
15     value: 'product_category_search',
16     param: '类别 id',
17     example: '25f2e9c4f1844f1faa154dd41ca40122',
18     label: '类别商品缓存',
19     page: "类别相关的页面，例如集市",
20     effect: '存储该类别下全部商品 id',
21     textInput: true
22   },
23   {
24     value: 'product_type_search',
25     param: '开发设置的类型',
26     example: 'entity_retail',
27     label: '类型商品缓存',
28     page: "产品类型的页面，例如集市、店铺详情中的商品列表",
29     effect: '存储该类型下的全部商品 id',
30     selectInput: productTemplate
31   },
32   {
33     value: 'product_integral_search',
34     label: '积分商品缓存',
35     page: "积分商品列表页面",
36     effect: '存储全部的积分商品 id',
37   },
38   {
39     value: 'product_hide_search',
40     label: '隐藏商品缓存',
41     page: "涉及集市、店铺详情中的商品列表",
42     effect: '存储全部的隐藏商品 id',
43   },
44   },
45   {
46     value: 'product_store_search',
47     param: '店铺 id',
48     example: '25f2e9c4f1844f1faa154dd41ca40122',
49     label: '商品店铺缓存',
50     page: "涉及店铺详情中的商品列表",
51     effect: '存储该店铺下的全部商品 id',
52     textInput: true
53   },
```

行李托运平台 V1.0

```
1  {
2      value: 'product_name_search',
3      param: '搜索名称',
4      example: '千户苗寨',
5      label: '商品搜索信息缓存',
6      page: " 涉及到产品搜索功能的页面，例如集市，产品搜索页面等",
7      effect: '存储该搜索内容下全部相关的商品 id',
8      textInput: true
9  },
10
11  {
12      value: 'product_weight_search',
13      label: '商品权重缓存有序集合',
14      page: " 涉及商品列表排序",
15      effect: '存储全部商品的权重',
16  },
17  {
18      value: 'gameUserInfo',
19      param: '用户 id',
20      example: '25f2e9c4f1844f1faa154dd41ca40122',
21      label: '用户游戏相关的展示缓存（卡券数量，等级，经验，剧本等）',
22      page: " 涉及我的和令牌页面",
23      effect: '存储用户游戏相关数据',
24      textInput: true
25  },
26  {
27      value: 'guidePropList',
28      label: '引游人首页卡券列表缓存',
29      page: " 涉及引游人推荐卡券列表",
30      effect: '存储引游人卡券推荐列表页面数据',
31  },
32  {
33      value: 'guideGrantPropList',
34      param: '引游人 id 或者合伙人 id',
35      example: '25f2e9c4f1844f1faa154dd41ca40122',
36      label: '引游人或合伙人发放道具列表',
37      page: " 涉及引游人和合伙人的已发放卡券列表",
38      effect: '存储引游人或合伙人发放道具列表数据',
39      textInput: true
40  },
41  {
42      value: 'hotelDayStock',
43      param: 'yyyy-mm-dd 格式的日期',
44      example: '2024-09-01',
45      label: '指定日期当天有库存的酒店的缓存',
46      page: " 涉及全部酒店相关产品",
47      effect: '存储当天有酒店库存的全部店铺 id',
48      textInput: true
49  },
50  {
51      value: 'orderDetails',
52      param: '订单 id',
53      example: '298417635459072000',
```

行李托运平台 V1.0

```
1      label: '订单信息缓存',
2      page: " 涉及订单详情页面",
3      effect: '存储订单详情',
4      textInput: true
5    },
6    {
7      value: 'orderList',
8      param: '用户 id',
9      example: '25f2e9c4f1844f1faa154dd41ca40122',
10     label: '用户订单列表缓存',
11     page: " 涉及订单列表页面",
12     effect: '存储用户订单列表',
13     textInput: true
14   },
15   {
16     value: 'userAccount',
17     param: '用户 id',
18     example: '25f2e9c4f1844f1faa154dd41ca40122',
19     label: '用户帐户金币缓存',
20     page: " 涉及我的页面",
21     effect: '存储用户金币信息',
22     textInput: true
23   },
24
25   {
26     value: 'followList',
27     param: '类型(guide_prop、product、store):用户 id',
28     example: 'guide_prop:25f2e9c4f1844f1faa154dd41ca40122',
29     label: '收藏列表缓存',
30     page: " 涉及收藏相关的列表页面",
31     effect: '存储用户收藏列表',
32     textInput: true
33   },
34   {
35     value: 'followUser',
36     param: '用户 id',
37     example: '25f2e9c4f1844f1faa154dd41ca40122',
38     label: '用户收藏缓存',
39     page: " 涉及用户是否收藏了某项内容",
40     effect: '存储用户收藏的全部内容',
41     textInput: true
42   },
43   {
44     value: 'storeInfo',
45     param: '店铺 id',
46     example: '25f2e9c4f1844f1faa154dd41ca40122',
47     label: '店铺信息缓存',
48     page: " 涉及店铺详情页面",
49     effect: '存储店铺详情',
50     textInput: true
51   },
52   {
53     value: 'contentVillage',
```

行李托运平台 V1.0

```
1      label: '首页村寨缓存',
2      page: " 涉及首页雷山地图",
3      effect: '存储雷山地图信息',
4    },
5    {
6      value: 'marketCategory',
7      label: '集市类别缓存',
8      page: " 涉及集市类别数据",
9      effect: '存储集市类别下的类别树',
10    },
11  ]
12
13  interface DataType {
14    value: string;
15    label: string;
16    param?: string,
17    example?: string,
18    page: string;
19    effect: string;
20  }
21  type DataIndex = keyof DataType;
22  const CaseForm = () => {
23    const [searchText, setSearchText] = useState("");
24    const [searchedColumn, setSearchedColumn] = useState("");
25    const searchInput = useRef<InputRef>(null);
26    const [open, setOpen] = useState(false)
27    const [form] = Form.useForm<{ type: string; param?: string }>();
28    const handleSearch = (
29      selectedKeys: string[],
30      confirm: FilterDropdownProps['confirm'],
31      dataIndex: DataIndex,
32    ) => {
33      confirm();
34      setSearchText(selectedKeys[0]);
35      setSearchedColumn(dataIndex);
36    };
37
38    const handleReset = (clearFilters: () => void) => {
39      clearFilters();
40      setSearchText("");
41    };
42
43    const getColumnSearchProps = (dataIndex: DataIndex): TableColumnType<DataType> => ({
44      filterDropdown: ({setSelectedKeys, selectedKeys, confirm, clearFilters, close}) => (
45        <div style={{padding: 8}} onKeyDown={(e) => e.stopPropagation()}>
46          <Input
47            ref={searchInput}
48            placeholder={`Search ${dataIndex}`}
49            value={selectedKeys[0]}
50            onChange={(e) => setSelectedKeys(e.target.value ? [e.target.value] : [])}
51            onPressEnter={() => handleSearch(selectedKeys as string[], confirm, dataIndex)}
52            style={{marginBottom: 8, display: 'block'}}
53          />
          <Space>
```

行李托运平台 V1.0

```
1      <Button
2          type="primary"
3          onClick={() => handleSearch(selectedKeys as string[], confirm, dataIndex)}
4          icon={<SearchOutlined/>}
5          size="small"
6          style={{width: 90}}
7      >
8          搜索
9      </Button>
10     <Button
11         onClick={() => clearFilters && handleReset(clearFilters)}
12         size="small"
13         style={{width: 90}}
14     >
15         重置
16     </Button>
17     <Button
18         type="link"
19         size="small"
20         onClick={() => {
21             confirm({closeDropdown: false});
22             setSearchText((selectedKeys as string[][0]));
23             setSearchedColumn(dataIndex);
24         }}
25     >
26         过滤
27     </Button>
28     <Button
29         type="link"
30         size="small"
31         onClick={() => {
32             close();
33         }}
34     >
35         关闭
36     </Button>
37 </Space>
38 </div>
39 ),
40 filterIcon: (filtered: boolean) => (
41     <SearchOutlined style={{color: filtered ? '#1677ff' : undefined}}/>
42 ),
43 onFilter: (value, record) => {
44     // @ts-ignore
45     return record[dataIndex]
46         .toString()
47         .toLowerCase()
48         .includes((value as string).toLowerCase())
49 },
50 onFilterDropdownOpenChange: (visible) => {
51     if (visible) {
52         setTimeout(() => searchInput.current?.select(), 100);
53     }
```


行李托运平台 V1.0

```
1      },
2      render: (text) =>
3          searchedColumn === dataIndex ? (
4
5              <Highlighter
6                  highlightStyle={{backgroundColor: '#ffc069', padding: 0}}
7                  searchWords={['searchText']}
8                  autoEscape
9                  textToHighlight={text ? text.toString() : ''}
10             />
11          ) : (
12              text
13          ),
14      });
15
16      const columns: TableColumnsType<DataType> = [
17          {
18              title: '名称',
19              dataIndex: 'label',
20              key: 'label',
21              ...getColumnSearchProps('label'),
22          },
23          {
24              title: '作用',
25              dataIndex: 'effect',
26              key: 'effect',
27          },
28          {
29              title: '涉及页面',
30              dataIndex: 'page',
31              key: 'page',
32          },
33          {
34              title: '所需参数',
35              dataIndex: 'param',
36              key: 'param',
37              render: (_, record) => record?.param ?? '-'
38          },
39          {
40              title: '例子',
41              dataIndex: 'example',
42              key: 'example',
43              render: (_, record) => record?.example ?? '-'
44          },
45          {
46              title: '操作',
47              key: 'action',
48              width: 150,
49              render: (_, record) => (
50                  <Space size="middle">
51                      <a onClick={() => {
52                          // @ts-ignore
53                          form.setFieldValue('type', record.value)
```

行李托运平台 V1.0

```
1      setOpen(true)
2
3    >>
4    更新缓存
5    </a>
6    </Space>
7  )
8  }
9  ],
10 return <>
11   <PageContainer title="缓存更新">
12     <Table columns={columns} dataSource={caseTypeList}/>
13     <ModalForm
14       visible={open}
15       onVisibleChange={setOpen}
16       form={form}
17       onFinish={async (values) => {
18         // eslint-disable-next-line @typescript-eslint/no-shadow
19         const data: string = await deleteRedis(values)
20         if (data.includes('成功')) {
21           message.success(data);
22         }
23
24         return true;
25       }}
26     >
27       { /* eslint-disable-next-line react/jsx-no-undef */ }
28       <ProFormSelect
29         options={caseTypeList}
30         disabled
31         required
32         showSearch
33         name="type"
34         label="缓存类型"
35       />
36       { /* eslint-disable-next-line react/jsx-no-undef */ }
37       <ProFormDependency name={['type']>
38         {{{type}}} => {
39           // eslint-disable-next-line @typescript-eslint/no-shadow
40           const item = caseTypeList.find((item: any) => item.value === type);
41           if (!item) {
42             return null
43           }
44           if (item.textInput) {
45             return (<>
46               <p>参数要求: {item.param}</p>
47               <p>示例: {item.example}</p>
48               <ProFormText
49                 name='param'
50                 required
51                 label="参数"
52                 placeholder={item.param}
53               /></>)
```

行李托运平台 V1.0

```
1      }
2      if (item.selectInput) {
3          return <ProFormSelect
4              options={item.selectInput}
5              width="md"
6              required
7              name="param"
8              label="产品类型"
9          />
10     }
11     return null
12 }
13 </ProFormDependency>
14
15 </ModalForm>
16 </PageContainer>
17 </>
18
19 }
20 export default CaseForm
21 ```
22 File: src/pages/customer/user/list/index.tsx
23 ```tsx
24 import {Avatar, Button, message, Modal, Select, Switch} from 'antd';
25 import {PageContainer} from '@ant-design/pro-layout';
26 import type {ActionType, ProColumns} from '@ant-design/pro-table';
27 import ProTable from '@ant-design/pro-table';
28 import {exportUserInfoExcel, getAccountAndPassword, updateUserStatus, userList} from '../service';
29 import type {TableListItem, TableListPagination} from '../data';
30 import styles from './index.less';
31 import React, {useEffect, useRef, useState} from 'react';
32 import {FileExcelOutlined} from '@ant-design/icons';
33 import TagNames from '@components/TagNames';
34 import DateFormat from '@components/DateFormat';
35 import moment from 'moment';
36 import UserDetailForm from '../components/UserDetailForm';
37 import {history} from 'umi';
38 import UpdateUserInfoModal from '@pages/customer/user/components/UpdateUserInfoModal';
39 import UpdatePasswordModal from '@pages/customer/user/components/UpdatePasswordModal';
40 import {listTenant} from '@api/tenant';
41 import {ProCard} from '@ant-design/pro-components';
42
43 const query: any = history.location.query;
44 const {confirm} = Modal;
45 const TableList: React.FC = () => {
46     /** 新建窗口的弹窗 */
47     const actionRef = useRef<ActionType>();
48     const [currentRow, setCurrentRow] = useState<TableListItem>();
49     const [updateUserVisible, handleUpdateUserVisible] = useState<boolean>(false);
50     const [updatePasswordVisible, handleUpdatePasswordVisible] = useState<boolean>(false);
51
52     //用户禁用状态
53     const [switchDisabled, setSwitchDisabled] = useState<boolean>(false);
```

行李托运平台 V1.0

```
1      const [accountAndPassword, setAccountAndPassword] = useState<{
2          username: undefined | string;
3          password: undefined | string;
4      }>({
5          username: undefined,
6          password: undefined,
7      });
8      const [userModalVisible, updateUserModalVisible] = useState(false);
9      const [userGuid, updateUserGuid] = useState('');
10     const [selectedRowKeyList, setSelectedRowKeyList] = useState<any[]>([]);
11     const [tenantList, setTenantList] = useState<any[]>([]);
12     const [tabPageKey, handlerTabPageKey] = useState<string>('user');
13     const [currentTenant, setCurrentTenant] = useState<string>('');
14     const getTenantList = async () => {
15         let data = await listTenant({});
16         if (data) {
17             data = data.data?.map((item: any) => {
18                 return {label: item.title, value: item.id}
19             })
20             setTenantList(data);
21         }
22     };
23     useEffect(() => {
24         if (query.guid) {
25             updateUserGuid(query.guid);
26             updateUserModalVisible(true);
27         }
28
29         getTenantList();
30
31         return () => {
32             updateUserGuid('');
33         };
34     }, []);
35     const onSelectChange: any = (rowKeys: any[]) => {
36         // 拿到之前选中的数据
37         const temp: any = [...selectedRowKeyList];
38         // 将当前选中的数据与之前选中的数据合并,去重
39         const tempArr = [...temp, ...rowKeys];
40         const tempSet = new Set(tempArr);
41         const newArr = Array.from(tempSet);
42         // 将合并后的数据赋值给选中的数据
43         setSelectedRowKeyList(newArr);
44     };
45     const onEditClick = (record: any) => {
46         getAccountAndPassword({guid: record.guid}).then((res) => {
47             if (res.username) {
48                 setAccountAndPassword(res);
49                 setCurrentRow(record);
50                 handleUpdatePasswordVisible(true);
51             } else {
52                 message.error('该用户没有账号,请先创建账户').then((r) => r);
53             }
54         });
55     };
56 }
```

行李托运平台 V1.0

```
1      });
2    };
3    const columns: ProColumns<TableListItem>[] = [
4      {
5        title: 'guid',
6        dataIndex: 'guid',
7        // width: 100,
8        copyable: true,
9        ellipsis: true,
10     },
11     {
12       title: '用户头像',
13       dataIndex: 'headImg',
14       search: false,
15       render: (_, any, record: TableListItem) => (
16         <Avatar key="avatar" className={styles.avatarImg} src={record.headImg} alt="头像"/>
17       ),
18     },
19     {
20       title: '昵称',
21       dataIndex: 'nickName',
22       ellipsis: true,
23     },
24     {
25       title: '性别',
26       dataIndex: 'gender',
27       valueType: 'select',
28       valueEnum: {
29         1: {
30           text: '男',
31         },
32         2: {
33           text: '女',
34         },
35       },
36       search: false,
37     },
38     {
39       title: '标签',
40       // width: 150,
41       search: false,
42       dataIndex: 'tagNames',
43       render: (_, row) => <TagNames tagNames={row.tagNames}/>,
44     },
45     {
46       title: '手机号',
47       dataIndex: 'mobilePhone',
48       ellipsis: true,
49       copyable: true,
50     },
51     {
52       title: '是否是商户',
53       dataIndex: 'isMerchant',
```

行李托运平台 V1.0

```

1      valueType: 'select',
2      valueEnum: {
3        1: {
4          text: '否',
5        },
6        0: {
7          text: '是',
8        },
9      },
10    },
11    {
12      title: '注册时间',
13      dataIndex: 'createTime',
14      search: false,
15      render: (text: any) => <DateFormat date={text}/>,
16    },
17    {
18      title: '注册时间',
19      dataIndex: 'createTime',
20      valueType: 'dateRange',
21      hideInTable: true,
22      search: {
23        transform: (value) => {
24          return {
25            createStart: moment(value[0]).startOf('day').unix(),
26            createEnd: moment(value[1]).endOf('day').unix(),
27          };
28        },
29      },
30    },
31    {
32      title: '状态',
33      dataIndex: 'status',
34      valueType: 'select',
35      valueEnum: {
36        '0': {text: '正常'},
37        '1': {text: '禁用'},
38      },
39      render: (_, record) => (
40        <Switch
41          disabled={switchDisabled}
42          onClick={() => {
43            confirm({
44              title: record.status === 1 ? '是否要启用该用户?' : '是否要禁用该用户?',
45              onOk() {
46                setSwitchDisabled(true);
47                updateUserStatus({
48                  guid: record.guid,
49                  status: record.status === 1 ? 0 : 1,
50                })
51              }
52            }) => {
53              setSwitchDisabled(false);
54              message.success('修改成功').then(() => r);
55            }
56          }
57        >
58      )
59    }
60  ],
61  [
62    {
63      title: '注册时间',
64      dataIndex: 'createTime',
65      valueType: 'dateRange',
66      hideInTable: true,
67      search: {
68        transform: (value) => {
69          return {
70            createStart: moment(value[0]).startOf('day').unix(),
71            createEnd: moment(value[1]).endOf('day').unix(),
72          };
73        },
74      },
75    },
76    {
77      title: '状态',
78      dataIndex: 'status',
79      valueType: 'select',
80      valueEnum: {
81        '0': {text: '正常'},
82        '1': {text: '禁用'},
83      },
84      render: (_, record) => (
85        <Switch
86          disabled={switchDisabled}
87          onClick={() => {
88            confirm({
89              title: record.status === 1 ? '是否要启用该用户?' : '是否要禁用该用户?',
90              onOk() {
91                setSwitchDisabled(true);
92                updateUserStatus({
93                  guid: record.guid,
94                  status: record.status === 1 ? 0 : 1,
95                })
96              }
97            }) => {
98              setSwitchDisabled(false);
99              message.success('修改成功').then(() => r);
100            }
101          }
102        >
103      )
104    }
105  ],
106  [
107    {
108      title: '注册时间',
109      dataIndex: 'createTime',
110      valueType: 'dateRange',
111      hideInTable: true,
112      search: {
113        transform: (value) => {
114          return {
115            createStart: moment(value[0]).startOf('day').unix(),
116            createEnd: moment(value[1]).endOf('day').unix(),
117          };
118        },
119      },
120    },
121    {
122      title: '状态',
123      dataIndex: 'status',
124      valueType: 'select',
125      valueEnum: {
126        '0': {text: '正常'},
127        '1': {text: '禁用'},
128      },
129      render: (_, record) => (
130        <Switch
131          disabled={switchDisabled}
132          onClick={() => {
133            confirm({
134              title: record.status === 1 ? '是否要启用该用户?' : '是否要禁用该用户?',
135              onOk() {
136                setSwitchDisabled(true);
137                updateUserStatus({
138                  guid: record.guid,
139                  status: record.status === 1 ? 0 : 1,
140                })
141              }
142            }) => {
143              setSwitchDisabled(false);
144              message.success('修改成功').then(() => r);
145            }
146          }
147        >
148      )
149    }
150  ],
151  [
152    {
153      title: '注册时间',
154      dataIndex: 'createTime',
155      valueType: 'dateRange',
156      hideInTable: true,
157      search: {
158        transform: (value) => {
159          return {
160            createStart: moment(value[0]).startOf('day').unix(),
161            createEnd: moment(value[1]).endOf('day').unix(),
162          };
163        },
164      },
165    },
166    {
167      title: '状态',
168      dataIndex: 'status',
169      valueType: 'select',
170      valueEnum: {
171        '0': {text: '正常'},
172        '1': {text: '禁用'},
173      },
174      render: (_, record) => (
175        <Switch
176          disabled={switchDisabled}
177          onClick={() => {
178            confirm({
179              title: record.status === 1 ? '是否要启用该用户?' : '是否要禁用该用户?',
180              onOk() {
181                setSwitchDisabled(true);
182                updateUserStatus({
183                  guid: record.guid,
184                  status: record.status === 1 ? 0 : 1,
185                })
186              }
187            }) => {
188              setSwitchDisabled(false);
189              message.success('修改成功').then(() => r);
190            }
191          }
192        >
193      )
194    }
195  ],
196  [
197    {
198      title: '注册时间',
199      dataIndex: 'createTime',
200      valueType: 'dateRange',
201      hideInTable: true,
202      search: {
203        transform: (value) => {
204          return {
205            createStart: moment(value[0]).startOf('day').unix(),
206            createEnd: moment(value[1]).endOf('day').unix(),
207          };
208        },
209      },
210    },
211    {
212      title: '状态',
213      dataIndex: 'status',
214      valueType: 'select',
215      valueEnum: {
216        '0': {text: '正常'},
217        '1': {text: '禁用'},
218      },
219      render: (_, record) => (
220        <Switch
221          disabled={switchDisabled}
222          onClick={() => {
223            confirm({
224              title: record.status === 1 ? '是否要启用该用户?' : '是否要禁用该用户?',
225              onOk() {
226                setSwitchDisabled(true);
227                updateUserStatus({
228                  guid: record.guid,
229                  status: record.status === 1 ? 0 : 1,
230                })
231              }
232            }) => {
233              setSwitchDisabled(false);
234              message.success('修改成功').then(() => r);
235            }
236          }
237        >
238      )
239    }
240  ],
241  [
242    {
243      title: '注册时间',
244      dataIndex: 'createTime',
245      valueType: 'dateRange',
246      hideInTable: true,
247      search: {
248        transform: (value) => {
249          return {
250            createStart: moment(value[0]).startOf('day').unix(),
251            createEnd: moment(value[1]).endOf('day').unix(),
252          };
253        },
254      },
255    },
256    {
257      title: '状态',
258      dataIndex: 'status',
259      valueType: 'select',
260      valueEnum: {
261        '0': {text: '正常'},
262        '1': {text: '禁用'},
263      },
264      render: (_, record) => (
265        <Switch
266          disabled={switchDisabled}
267          onClick={() => {
268            confirm({
269              title: record.status === 1 ? '是否要启用该用户?' : '是否要禁用该用户?',
270              onOk() {
271                setSwitchDisabled(true);
272                updateUserStatus({
273                  guid: record.guid,
274                  status: record.status === 1 ? 0 : 1,
275                })
276              }
277            }) => {
278              setSwitchDisabled(false);
279              message.success('修改成功').then(() => r);
280            }
281          }
282        >
283      )
284    }
285  ],
286  [
287    {
288      title: '注册时间',
289      dataIndex: 'createTime',
290      valueType: 'dateRange',
291      hideInTable: true,
292      search: {
293        transform: (value) => {
294          return {
295            createStart: moment(value[0]).startOf('day').unix(),
296            createEnd: moment(value[1]).endOf('day').unix(),
297          };
298        },
299      },
300    },
301    {
302      title: '状态',
303      dataIndex: 'status',
304      valueType: 'select',
305      valueEnum: {
306        '0': {text: '正常'},
307        '1': {text: '禁用'},
308      },
309      render: (_, record) => (
310        <Switch
311          disabled={switchDisabled}
312          onClick={() => {
313            confirm({
314              title: record.status === 1 ? '是否要启用该用户?' : '是否要禁用该用户?',
315              onOk() {
316                setSwitchDisabled(true);
317                updateUserStatus({
318                  guid: record.guid,
319                  status: record.status === 1 ? 0 : 1,
320                })
321              }
322            }) => {
323              setSwitchDisabled(false);
324              message.success('修改成功').then(() => r);
325            }
326          }
327        >
328      )
329    }
330  ],
331  [
332    {
333      title: '注册时间',
334      dataIndex: 'createTime',
335      valueType: 'dateRange',
336      hideInTable: true,
337      search: {
338        transform: (value) => {
339          return {
340            createStart: moment(value[0]).startOf('day').unix(),
341            createEnd: moment(value[1]).endOf('day').unix(),
342          };
343        },
344      },
345    },
346    {
347      title: '状态',
348      dataIndex: 'status',
349      valueType: 'select',
350      valueEnum: {
351        '0': {text: '正常'},
352        '1': {text: '禁用'},
353      },
354      render: (_, record) => (
355        <Switch
356          disabled={switchDisabled}
357          onClick={() => {
358            confirm({
359              title: record.status === 1 ? '是否要启用该用户?' : '是否要禁用该用户?',
360              onOk() {
361                setSwitchDisabled(true);
362                updateUserStatus({
363                  guid: record.guid,
364                  status: record.status === 1 ? 0 : 1,
365                })
366              }
367            }) => {
368              setSwitchDisabled(false);
369              message.success('修改成功').then(() => r);
370            }
371          }
372        >
373      )
374    }
375  ],
376  [
377    {
378      title: '注册时间',
379      dataIndex: 'createTime',
380      valueType: 'dateRange',
381      hideInTable: true,
382      search: {
383        transform: (value) => {
384          return {
385            createStart: moment(value[0]).startOf('day').unix(),
386            createEnd: moment(value[1]).endOf('day').unix(),
387          };
388        },
389      },
390    },
391    {
392      title: '状态',
393      dataIndex: 'status',
394      valueType: 'select',
395      valueEnum: {
396        '0': {text: '正常'},
397        '1': {text: '禁用'},
398      },
399      render: (_, record) => (
400        <Switch
401          disabled={switchDisabled}
402          onClick={() => {
403            confirm({
404              title: record.status === 1 ? '是否要启用该用户?' : '是否要禁用该用户?',
405              onOk() {
406                setSwitchDisabled(true);
407                updateUserStatus({
408                  guid: record.guid,
409                  status: record.status === 1 ? 0 : 1,
410                })
411              }
412            }) => {
413              setSwitchDisabled(false);
414              message.success('修改成功').then(() => r);
415            }
416          }
417        >
418      )
419    }
420  ],
421  [
422    {
423      title:
```

行李托运平台 V1.0

```
1      Modal.destroyAll();
2
3      if (actionRef.current) {
4          actionRef.current.reload();
5      }
6      })
7      .catch(() => {
8          setSwitchDisabled(false);
9          message.error('修改失败').then((r) => r);
10         return false;
11     });
12     },
13     });
14     }}
15     checked={record.status === 0}
16     />
17 },
18 },
19 {
20     title: '操作',
21     dataIndex: 'option',
22     valueType: 'option',
23     width: 250,
24     render: (_, any, record: TableListItem) => [
25         // <Access
26         //     accessible={
27         //         menuKeys.includes(path + userListAction.updatePwd) &&
28         //         currentUser?.menu[path + userListAction.updatePwd]?.type === 2
29         //     }
30         // >
31         <Button
32             style={{padding: 0}}
33             key="update"
34             type="link"
35             onClick={() => {
36                 // const formValue = {
37                 //     ...record
38                 // }
39                 setCurrentRow(record);
40                 handleUpdateUserVisible(true);
41             }}
42         >
43             编辑
44         </Button>,
45         // </Access>,
46         // <Access
47         //     accessible={
48         //         menuKeys.includes(path + userListAction.edit) &&
49         //         currentUser?.menu[path + userListAction.edit]?.type === 2
50         //     }
51         // >
52         <Button
53             style={{padding: 0}}
```

行李托运平台 V1.0

```
1      disabled={record.hasAccount === '1'}
2      key='updatePassword'
3      type='link'
4      onClick={() => onEditClick(record)}
5    >
6      修改密码
7    </Button>,
8    // </Access>,
9    <Button
10      style={{padding: 0}}
11      key="detail"
12      type="link"
13      onClick={() => {
14        // history.push("/customer/user/detail?guid=" + record.guid);
15        // 打开用户详情弹窗
16        updateUserGuid(record.guid);
17        updateUserModalVisible(true);
18      }}
19    >
20      详情
21    </Button>,
22  ],
23 },
24 ],
25
26 /**
27  * 导出用户数据
28  * @param selectedRowKeys
29  */
30 const exportUserData = (selectedRowKeys: (number | string)[]) => {
31   return () => {
32     const params = {
33       guids: selectedRowKeys,
34     };
35     if (selectedRowKeys.length > 0) {
36       // 请求接口,获取 excel 文件
37       exportUserInfoExcel(params)
38         .then((file: any) => {
39           const url = window.URL.createObjectURL(file);
40           const link = document.createElement('a');
41           link.href = url;
42           const filename = `用户信息_${moment().format('YYYY-MM-DD')}`;
43           link.setAttribute('download', filename);
44           link.click();
45
46           // It's important to revoke the object URL to avoid memory leaks
47           window.URL.revokeObjectURL(url);
48           message.success('导出成功!');
49         })
50         .catch((err) => {
51           message.error('导出失败');
52           console.error(err);
53         });
54     }
```


行李托运平台 V1.0

```
1      } else {
2          message.error('请先选择用户');
3      }
4  };
5  };
6
7  return (
8      <PageContainer
9          tabList=[
10              {
11                  key: 'user',
12                  tab: '用户列表',
13              },
14              {
15                  key: 'saas',
16                  tab: 'SaaS 用户列表',
17              },
18          ]
19          onTabChange={key: string => {
20              if (key === 'saas') {
21                  setCurrentTenant(tenantList[0].value);
22              } else {
23                  setCurrentTenant('');
24              }
25              handlerTabPageKey(key);
26              actionRef.current?.reload();
27          }}
28      >
29
30      {tabPageKey === 'user' ? null : <ProCard bordered>
31      <span
32          style={{marginLeft: 48}}>
33          当前租户 :
34      </span>
35          <Select
36              value={currentTenant}
37              style={{width: 280, marginLeft: 8}}
38              onChange={async (value) => {
39                  setCurrentTenant(value);
40                  actionRef.current?.reload();
41              }}
42              options={tenantList}
43          />
44
45      </ProCard>
46
47      <ProTable<TableListItem, TableListPagination>
48          headerTitle="用户列表"
49          actionRef={actionRef}
50          rowKey="guid"
51          search={{
52              labelWidth: 120,
53          }}
```

行李托运平台 V1.0

```
1 pagination={{
2   showSizeChanger: true,
3   showTotal: (total) => `共 ${total} 条`,
4 }}
5 request={async (params: any) => {
6   const msg = await userList({...params, tenantId: currentTenant});
7   msg.data = msg.data.map((item: any) => {
8     if (item.isPay) {
9       item.isPay = item.isPay.toString();
10    }
11    if (item.isMerchant) {
12      item.isMerchant = item.isMerchant.toString();
13    }
14    if (item.gender) {
15      item.gender = item.gender.toString();
16    }
17    return item;
18  });
19  return {
20    data: msg.data,
21    success: true,
22    total: msg.total,
23  };
24 }}
25 columns={columns}
26 // 选择框
27 rowSelection={{
28   defaultSelectedRowKeys: selectedRowKeyList,
29   onChange: onSelectChange,
30   preserveSelectedRowKeys: true,
31   onSelectNone: () => {
32     setSelectedRowKeyList([]);
33   },
34 }}
35 tableAlertRender=(({selectedRowKeys, onCleanSelected}) => {
36   return (
37     <div>
38       已选择 <a style={{fontWeight: 600}}>{selectedRowKeys.length}</a>{' '}
39       项    
40       {selectedRowKeys.length > 0 && (
41         <a onClick={onCleanSelected} style={{marginLeft: 24}}>
42           取消选择
43         </a>
44       )}
45     </div>
46   );
47 }}
48 // 下拉选项
49 tableAlertOptionRender=(({selectedRowKeys}) => {
50   return [
51     <Button
52       key={'export'}
53       icon={<FileExcelOutlined/>}
```

行李托运平台 V1.0

```
1         type="primary"
2         style={{marginLeft: '20px'}}
3         onClick={exportUserData(selectedRowKeys)}
4     >
5         导出选中的用户
6     </Button>,
7     ],
8     }}
9 </>
10
11 <UpdateUserInfoModal
12     updateUserVisible={updateUserVisible}
13     handleUpdateUserVisible={handleUpdateUserVisible}
14     currentRow={currentRow}
15     actionRef={actionRef}
16 </>
17
18 <UpdatePasswordModal
19     updatePasswordVisible={updatePasswordVisible}
20     handleUpdatePasswordVisible={handleUpdatePasswordVisible}
21     currentRow={currentRow}
22     setCurrentRow={setCurrentRow}
23     accountAndPassword={accountAndPassword}
24     setAccountAndPassword={setAccountAndPassword}
25     actionRef={actionRef}
26 </>
27
28 <UserDetailForm
29     modalVisible={userModalVisible}
30     updateModalVisible={updateUserModalVisible}
31     guid={userGuid}
32     updateGuid={updateUserGuid}
33 </>
34 </PageContainer>
35 );
36 };
37 export default TableList;
38 ```
39 File: src/pages/database/fields/index.tsx
40 ```tsx
41 import React, { useEffect, useRef, useState } from 'react';
42 import { PageContainer } from "@ant-design/pro-layout";
43 import type { ProColumns } from "@ant-design/pro-table";
44 import ProTable from "@ant-design/pro-table";
45 import { Button, Input, Modal, Select, message } from "antd";
46 import { PlusOutlined } from "@ant-design/icons";
47 import { history } from "umi";
48 import type { ActionType } from "@ant-design/pro-table/lib/typing";
49 import { createFields, deleteFields, fieldsList, updateFields } from '../service';
50 import { ModalForm, ProFormDigit, ProFormGroup, ProFormItem, ProFormSelect, ProFormSwitch, ProFormText } from
51 '@ant-design/pro-form';
52 import type { FieldsList } from '../data';
53 import { FieldsTypeEnum } from '../fieldsTypeEnum';
```

行李托运平台 V1.0

```
1  import { schemaList } from '../schema/service';
2  import type { SchemaList } from '../schema/data';
3  import { tableList } from '../table/service';
4  import type { TableList } from '../table/data';
5
6  /**
7   * 添加和编辑
8   */
9  const handleUpdate = async (fields: FieldsList, currentRow?: any) => {
10     const hide = message.loading('正在修改');
11     if (currentRow) {
12         if (fields.isAutoIncrementTemp) {
13             if (fields.fieldType !== 'bigint' && fields.fieldType !== 'integer') {
14                 hide();
15                 message.error("禁止将非 int4、int8 类型的字段设为自增");
16                 return;
17             }
18         }
19         try {
20             await updateFields({
21                 ...fields,
22                 isNullable: fields.isNullableTemp,
23                 isPrimaryKey: fields.isPrimaryKeyTemp,
24                 isAutoIncrement: fields.isAutoIncrementTemp,
25             });
26             hide();
27             message.success('修改成功');
28             return true;
29         } catch (error) {
30             hide();
31             return false;
32         }
33     } else {
34         if (fields.isAutoIncrement) {
35             if (fields.fieldType !== 'bigint' && fields.fieldType !== 'integer') {
36                 hide();
37                 message.error("禁止将非 int4、int8 类型的字段设为自增");
38                 return;
39             }
40         }
41         try {
42             await createFields({
43                 ...fields,
44             });
45             hide();
46             message.success('新建成功');
47             return true;
48         } catch (error) {
49             console.log('error');
50             console.log(error);
51             hide();
52             return false;
53         }
54     }
55 }
```

行李托运平台 V1.0

```
1      }
2    };
3
4    const TenantList: React.FC = () => {
5      const schemaName = history.location.query?.schemaName;
6      const tableName = history.location.query?.tableName;
7      const actionRef = useRef<ActionType>(null);
8      const { confirm } = Modal;
9      let deleteValue = "";
10     const [modalVisible, handleModalVisible] = useState<boolean>(false);
11     const [currentRow, setCurrentRow] = useState<any>();
12     const [schemaOptions, setSchemaOptions] = useState([]);
13     const [tableOptions, setTableOptions] = useState([]);
14     const handleModelDestroy = (field: boolean) => {
15       if (!field) {
16         setCurrentRow(undefined);
17       }
18       handleModalVisible(field);
19     }
20     const getSchemaList = async () => {
21       const res = await schemaList({ current: 1, pageSize: 1000 });
22       if (res) {
23         const options = res.data.map((item: SchemaList) => {
24           return {
25             label: item.schemaName,
26             value: item.schemaName
27           }
28         });
29         setSchemaOptions(options);
30       }
31     }
32     const getTableList: any = async (value: any) => {
33       if (value || schemaName) {
34         const res = await tableList({ current: 1, pageSize: 1000, schemaName: (value ? value : schemaName) });
35         if (res) {
36           const options = res.data.map((item: TableList) => {
37             return {
38               label: item.tableName,
39               value: item.tableName
40             }
41           });
42           setTableOptions(options);
43           if (actionRef.current) {
44             actionRef.current.reload();
45           }
46         }
47       }
48     }
49
50     useEffect(() => {
51       getSchemaList();
52       getTableList();
53     }, []);
```

行李托运平台 V1.0

```
1
2  /**
3   * 删除确认
4   */
5  const showDeleteConfirm = (record: FieldsList) => {
6    confirm({
7      title: '请确认删除',
8      content: (
9        <div>
10          <div style={{ marginTop: 40 }} />
11          <Input
12            onChange={e => {
13              deleteValue = e.target.value;
14            }}
15            placeholder="请输入字段名加密钥"
16          />
17          <div style={{ marginTop: 6 }} />
18          <p style={{ color: 'red', fontSize: 12 }}> 请输入字段名加密钥以确认删除操作</p>
19        </div>
20      ),
21      onOk() {
22        if (deleteValue === (record.fieldName + 'lsyzw')) {
23          if (schemaName && schemaName !== null && schemaName !== "" && tableName && tableName !== null
24            && tableName !== "") {
25            deleteFields({ schemaName: schemaName, tableName: tableName, fieldName: record.fieldName }).then(res
26              => {
27                if (res) {
28                  message.success('删除成功');
29                }
30                actionRef.current?.reload();
31              }).catch(err => {
32                console.log("删除失败:", err);
33                message.error('删除失败');
34                actionRef.current?.reload();
35              });
36            } else {
37              message.error('模式名称或表名称不存在，请重试');
38            }
39          } else {
40            message.error('输入错误，请确认输入正确租户表名');
41          }
42        },
43      onCancel() {
44        console.log('取消删除');
45      },
46    });
47  };
48
49  const columns: ProColumns<FieldsList>[] = [
50    {
51      title: '序号',
52      dataIndex: 'index',
53      valueType: 'index',
```

行李托运平台 V1.0

```
1      width: 80,
2    },
3    {
4      title: '模式选择',
5      hideInTable: true,
6      renderFormItem: () => {
7        return (
8          <Select
9            showSearch
10             onChange={(value) => {
11               history.push('/database/fields?schemaName=' + value);
12               getTableList(value);
13             }}
14             defaultValue={schemaName}
15             options={schemaOptions}
16          />
17        );
18      }
19    },
20    {
21      title: '表选择',
22      hideInTable: true,
23      renderFormItem: () => {
24        return (
25          <Select
26            showSearch
27             onChange={(value) => {
28               history.push('/database/fields?schemaName=${schemaName}&tableName=${value}');
29               if (actionRef.current) {
30                 actionRef.current.reload();
31               }
32             }}
33             placeholder={'请先选择模式'}
34             defaultValue={tableName}
35             options={tableOptions}
36          />
37        );
38      }
39    },
40    {
41      dataIndex: 'fieldName',
42      title: '字段名称',
43      key: 'fieldName',
44      copyable: true
45    },
46    {
47      dataIndex: 'fieldType',
48      title: '类型',
49      align: 'center',
50      search: false,
51    },
52    {
53      dataIndex: 'length',
```

行李托运平台 V1.0

```
1      title: '长度',
2      align: 'center',
3      search: false,
4    },
5    {
6      dataIndex: 'scale',
7      title: '小数点',
8      align: 'center',
9      search: false,
10   },
11   {
12     dataIndex: 'precision',
13     title: '精度',
14     align: 'center',
15     search: false,
16   },
17   {
18     dataIndex: 'isAutoIncrement',
19     title: '自增',
20     align: 'center',
21     search: false,
22   },
23   {
24     dataIndex: 'fieldDefault',
25     title: '默认值',
26     align: 'center',
27     search: false,
28   },
29   {
30     dataIndex: 'isNullable',
31     title: '允许为空',
32     align: 'center',
33     search: false,
34   },
35   {
36     dataIndex: 'isPrimaryKey',
37     title: '主键',
38     align: 'center',
39     search: false,
40   },
41   {
42     dataIndex: 'fieldComment',
43     title: '注释',
44     search: false,
45   },
46   {
47     title: '操作',
48     key: 'options',
49     width: 300,
50     search: false,
51     align: 'center',
52     valueType: 'option',
53     render: (_, any, record: FieldsList) => [
```


行李托运平台 V1.0

```
1      <Button
2        key={'edit'}
3        type={'link'}
4        onClick={() => {
5          //将 record 转为编辑操作参数所需的值
6          record.isAutoIncrementTemp = (record.isAutoIncrement === 'YES');
7          record.isPrimaryKeyTemp = (record.isPrimaryKey === 'YES');
8          record.isNullableTemp = (record.isNullable === 'YES');
9          setCurrentRow(record);
10         handleModalVisible(true);
11       }}
12     >
13       编辑
14     </Button>,
15     <Button
16       key={'delete'}
17       type={'link'}
18       style={{ color: 'red' }}
19       onClick={() => showDeleteConfirm(record)}
20     >
21       删除
22     </Button>
23   ]
24 }
25 ];
26
27 return (
28   <PageContainer>
29     <ProTable
30       actionRef={actionRef}
31       headerTitle={schemaName && tableName ? `模式: ${schemaName} — 表: ${tableName}` : '字段列表'}
32       rowKey="fieldName"
33       search={{
34         labelWidth: 120,
35       }}
36       columns={columns}
37       request={async (param: any) => {
38         if (schemaName && tableName) {
39           param.schemaName = schemaName;
40           param.tableName = tableName;
41           const res = await fieldsList(param);
42           if (res?.total > 0) {
43             res.data.map((item: FieldsList) => {
44               if (item.fieldType === 'character varying') {
45                 item.fieldType = 'varchar';
46               }
47             //将 fieldDefault 去掉多余字符
48             const defaultValue = item.fieldDefault;
49             if (defaultValue) {
50               const temp = defaultValue.split("::");
51               if (temp.length > 1) {
52                 let tempValue = temp[0];
53                 tempValue = tempValue.replaceAll("", "");
```

行李托运平台 V1.0

```
1         item.fieldDefault = tempValue;
2     } else {
3         item.fieldDefault = temp[0];
4     }
5     }
6     return item;
7     });
8     return res;
9     }
10    }
11    return { data: [] };
12  }}
13  pagination={{
14    showSizeChanger: true,
15    pageSizeOptions: [10, 20, 50, 100],
16  }}
17  toolBarRender={() => [
18    <Button
19      type="primary"
20      key="primary"
21      onClick={() => {
22        if (schemaName && tableName) {
23          handleModalVisible(true);
24        } else {
25          message.info("请先选择模式和表");
26        }
27      }}
28    >
29      <PlusOutlined /> 新建
30    </Button>,
31  ]}
32 />
33 <ModalForm
34   title={currentRow ? '编辑' : '新建'}
35   width="600px"
36   visible={modalVisible}
37   onVisibleChange={handleModelDestroy}
38   modalProps={{
39     destroyOnClose: true,
40   }}
41   initialValues={{
42     ...currentRow,
43   }}
44   onFinish={async (value: FieldsList) => {
45     const success = await handleUpdate(value, currentRow);
46     if (success) {
47       handleModalVisible(false);
48       setCurrentRow(undefined);
49       if (actionRef.current) {
50         actionRef.current.reload();
51       }
52     }
53   }}
```

行李托运平台 V1.0

```
1      >
2      <ProFormGroup>
3        <ProFormText
4          rules={[
5            {
6              required: true,
7              message: '模式名称为必填项',
8            },
9          ]}
10         fieldProps={{ maxLength: 20 }}
11         width="sm"
12         name="schemaName"
13         initialValue={schemaName}
14         label="模式名称"
15         disabled
16       />
17       <ProFormText
18         rules={[
19           {
20             required: true,
21             message: '表名称为必填项',
22           },
23         ]}
24         fieldProps={{ maxLength: 20 }}
25         width="sm"
26         name="tableName"
27         initialValue={tableName}
28         label="表名称"
29         disabled
30       />
31     </ProFormGroup>
32     {currentRow ?
33       // 修改
34       <>
35         <ProFormGroup>
36           <ProFormText
37             rules={[
38               {
39                 required: true,
40                 message: '字段名称为必填项',
41               },
42             ]}
43             fieldProps={{ maxLength: 20 }}
44             width="sm"
45             name="fieldName"
46             label="字段名称"
47             disabled
48           />
49           <ProFormText
50             fieldProps={{ maxLength: 20 }}
51             width="sm"
52             name="newFieldName"
53             placeholder="修改字段名时输入"
```

行李托运平台 V1.0

```
1         label="修改名称"
2     />
3 </ProFormGroup>
4 <ProFormGroup>
5     <ProFormSelect
6         rules={[
7             {
8                 required: true,
9                 message: '字段类型为必填项',
10            },
11        ]}
12         width="sm"
13         name="fieldType"
14         label="类型"
15         showSearch
16         options={FieldsTypeEnum}
17     />
18     <ProFormSwitch
19         rules={[
20             {
21                 required: true,
22            },
23        ]}
24         name={'isPrimaryKeyTemp'}
25         label={'主键'}
26         width={'sm'}
27         fieldProps={{
28             checkedChildren: '是',
29             unCheckedChildren: '否',
30        }}
31         // disabled
32     />
33 </ProFormGroup>
34 <ProFormGroup>
35     <ProFormDigit
36         label="长度"
37         name="fieldLength"
38         min={0}
39         initialValue={currentRow.length}
40     />
41     <ProFormDigit
42         label="精度"
43         name="fieldPrecision"
44         min={0}
45         initialValue={currentRow.precision}
46     />
47
48 </ProFormGroup>
49 <ProFormGroup>
50     <ProFormSwitch
51         rules={[
52             {
53                 required: true,
```

行李托运平台 V1.0

```
1      },
2    ]}
3    name={'isRequiredTemp'}
4    label={'允许为空'}
5    width={'sm'}
6    fieldProps={{
7      checkedChildren: '是',
8      unCheckedChildren: '否',
9    }}
10  />
11  <ProFormSwitch
12    rules={[
13      {
14        required: true,
15      },
16    ]}
17    name={'isAutoIncrementTemp'}
18    label={'是否自增'}
19    tooltip={"禁止将非 int4、int8 类型字段设为自增，存在数据的表禁止取消自增"}
20    width={'sm'}
21    fieldProps={{
22      checkedChildren: '是',
23      unCheckedChildren: '否',
24    }}
25    // disabled
26  />
27  <ProFormText
28    fieldProps={{ { maxLength: 10 } }}
29    width={190}
30    name="fieldDefault"
31    label="默认值"
32  />
33  </ProFormGroup>
34  <ProFormText
35    fieldProps={{ { maxLength: 100 } }}
36    width={480}
37    name="fieldComment"
38    label="描述"
39  />
40  </>
41  :
42  // 新建
43  <>
44    <ProFormGroup>
45      <ProFormText
46        rules={[
47          {
48            required: true,
49            message: '字段名为必填项',
50          },
51        ]}
52        fieldProps={{ { maxLength: 20 } }}
53        width="sm"
```

行李托运平台 V1.0

```
1         name="fieldName"
2         label="名"
3     />
4     <ProFormSelect
5         rules={[
6             {
7                 required: true,
8                 message: '字段类型为必填项',
9             },
10        ]}
11        width="sm"
12        name="fieldType"
13        label="类型"
14        showSearch
15        options={FieldsTypeEnum}
16    />
17 </ProFormGroup>
18 <ProFormGroup>
19     <ProFormDigit
20         label="长度"
21         name="fieldLength"
22         min={0}
23         initialValue={0}
24     />
25     <ProFormDigit
26         label="精度"
27         name="fieldPrecision"
28         min={0}
29         initialValue={0}
30     />
31     <ProFormSwitch
32         rules={[
33             {
34                 required: true,
35             },
36        ]}
37         name={'isPrimaryKey'}
38         label={'主键'}
39         width={'sm'}
40         initialValue={false}
41         fieldProps={{
42             checkedChildren: '是',
43             unCheckedChildren: '否',
44         }}
45     />
46 </ProFormGroup>
47 <ProFormGroup>
48     <ProFormSwitch
49         rules={[
50             {
51                 required: true,
52             },
53        ]}
```

行李托运平台 V1.0

```
1      name={'isNullable'}
2      label={'允许为空'}
3      width={'sm'}
4      initialValue={true}
5      fieldProps={{
6        checkedChildren: '是',
7        unCheckedChildren: '否',
8      }}
9    />
10   <ProFormSwitch
11     rules={[
12       {
13         required: true,
14       }],
15     name={'isAutoIncrement'}
16     label={'是否自增'}
17     width={'sm'}
18     tooltip={'禁止将非 int4、int8 类型字段设为自增'}
19     initialValue={false}
20     fieldProps={{
21       checkedChildren: '是',
22       unCheckedChildren: '否',
23     }}
24   />
25   <ProFormText
26     style={{ marginLeft: 100 }}
27     fieldProps={{ maxLength: 10 }}
28     width={190}
29     name="fieldDefault"
30     label="默认值"
31   />
32 </ProFormGroup>
33 <ProFormText
34   fieldProps={{ maxLength: 100 }}
35   width={480}
36   name="fieldComment"
37   label="描述"
38 />
39 </>
40 }
41 </ModalForm>
42 </PageContainer>
43 );
44 };
45
46
47 export default TenantList;
48
49 File: src/pages/auth/menu/index.tsx
50
51 import {Button, message, Tree, Spin, Popconfirm} from 'antd';
52 import React, {useState, useEffect} from 'react';
53 import {listMenu, addMenu, updateMenu, deleteMenu} from '../service';
```

行李托运平台 V1.0

```
1 import {PageContainer} from '@ant-design/pro-layout';
2 import {ModalForm, ProFormText, ProFormSelect} from '@ant-design/pro-form';
3 import type {RootObject, addFile} from './data';
4 import Styles from './index.less';
5 import TreeTag from '@components/TreeTag';
6 /**
7  * 添加节点
8  *
9  * @param fields
10  */
11 const handleAdd = async (fields: addFile) => {
12   const hide = message.loading('正在添加');
13   try {
14     await addMenu({...fields});
15     hide();
16     message.success('添加成功');
17     return true;
18   } catch (error) {
19     hide();
20     return false;
21   }
22 };
23 /**
24  * 更新节点
25  *
26  * @param fields
27  */
28 const handleUpdate = async (fields: any, currentRow?: RootObject) => {
29   const hide = message.loading('正在修改');
30
31   try {
32     await updateMenu({
33       ...currentRow,
34       ...fields,
35     });
36     hide();
37     message.success('修改成功');
38     return true;
39   } catch (error) {
40     hide();
41     message.error('修改失败请重试! ');
42     return false;
43   }
44 };
45 const TableList: React.FC = () => {
46   /** 新建窗口的弹窗 */
47   const [createModalVisible, handleModalVisible] = useState<boolean>(false);
48
49   const [updateModalVisible, handleUpdateModalVisible] = useState<boolean>(false);
50   const [loading, setLoading] = useState<boolean>(false);
51   const [info, setInfo] = useState<boolean>(false);
52   const [currentRow, setCurrentRow] = useState<RootObject | null>();
53   const [tablePage, handleTablePage] = useState<string>('saas');
```


行李托运平台 V1.0

```
1  const [treeData, setTreeData] = useState<RootObject[]>([]);
2  const [tenantTreeData, setTenantTreeData] = useState<RootObject[]>([]);
3  // const [selectTabKey, setSelectTabKey] = useState<TabEnum>(TabEnum.master);
4
5  // 添加和编辑相同部分表单
6  const editAndAddFrom = () => {
7    <>
8      { /* <ProFormRadio.Group
9        label={'菜单'}
10        name={'master'}
11        tooltip={'总台菜单指租户总台,租户菜单指租户的系统的菜单'}
12        rules={[{ required: true }]}
13        initialValue={selectTabKey === TabEnum.master ? 1 : 2}
14        options=[
15          { label: '总台菜单', value: 1 },
16          { label: '租户菜单', value: 2 },
17        ]}
18      /> */}
19    <ProFormText
20      rules={[
21        {
22          required: true,
23          message: '该项为必填项',
24        },
25      ]}
26      width="md"
27      name="name"
28      disabled={info}
29      label={`名称`}
30    />
31    <ProFormSelect
32      name="type"
33      label="类型"
34      width="md"
35      disabled={info}
36      valueEnum={{
37        1: '页面',
38        2: '动作',
39        3: '平台',
40        4: '模块',
41      }}
42      rules={[{required: true, message: '该选项为必填'}]}
43    />
44    <ProFormText
45      rules={[
46        {
47          required: true,
48          message: '该项为必填项',
49        },
50      ]}
51      width="md"
52      name="url"
53      disabled={info}
```

行李托运平台 V1.0

```
1         label={'地址/操作'}
2     />
3 </>
4 );
5 /**
6  * 获取数据
7  */
8 const getData = async () => {
9     setLoading(true);
10
11     const param = {
12         tenant_id: 'master',
13     }
14     const res = await listMenu(param);
15     setTreeData(res);
16     setLoading(false);
17 };
18 /**
19  * 获取数据
20  */
21 const getTenatData = async () => {
22     setLoading(true);
23
24     const param = {
25         tenant_id: 'tenant',
26     }
27     const res = await listMenu(param);
28     setTenantTreeData(res);
29     setLoading(false);
30 };
31
32 useEffect(() => {
33     getTenatData()
34     getData();
35 }, []);
36
37 /**
38  * 添加点击
39  */
40 const addClickHandle = () => {
41     setCurrentRow(null);
42     handleModalVisible(true);
43 };
44
45 // const onTabChange = (key: string) => {
46 //     setSelectTabKey(string2Enum(key));
47 //     getData(string2Enum(key));
48 // };
49
50 return (
51     <PageContainer
52         header={{
53             extra: [
```

行李托运平台 V1.0

```
1      <Button key="add" type="primary" onClick={addClickHandle}>
2          新建根节点
3      </Button>,
4
5  ],
6  }}
7  tabList=[
8      {
9          tab: 'saas 菜单管理',
10         key: 'saas',
11     },
12     {
13         tab: '租户菜单管理',
14         key: 'tenant',
15     }
16 ]
17 onTabChange={(value) => {
18     handleTablePage(value)
19 }}
20
21 >
22 <Spin spinning={loading}>
23     {treeData.length > 0 && (
24         <Tree
25             blockNode
26             defaultExpandAll
27             selectable={false}
28             fieldNames={{key: 'id', title: 'name'}}
29             titleRender={(data: RootObject) => {
30                 return (
31                     <div key={data.id} className={Styles.treeltem}>
32                         <TreeTag title={data?.name} type={data?.type}/>
33
34                         <div>
35                             <Button
36                                 type="text"
37                                 onClick={(event) => {
38                                     event.stopPropagation();
39                                     setCurrentRow(data);
40                                     setInfo(true);
41                                     handleUpdateModalVisible(true);
42                                 }}
43                             >
44                                 详情
45                             </Button>
46                             <Button
47                                 type="link"
48                                 onClick={(event) => {
49                                     event.stopPropagation();
50                                     setCurrentRow(data);
51                                     handleModalVisible(true);
52                                 }}
53                             >
```

行李托运平台 V1.0

```
1      添加子节点
2      </Button>
3      <Button
4          type="link"
5          onClick={ (event) => {
6              event.stopPropagation();
7              setCurrentRow(data);
8              handleUpdateModalVisible(true);
9          }}
10     >
11         编辑
12     </Button>
13     <Popconfirm
14         title="是否确认删除? "
15         onConfirm={async () => {
16             const hide = message.loading('正在删除');
17             try {
18                 await deleteMenu({
19                     id: data.id,
20                 });
21                 hide();
22                 message.success('删除成功');
23                 if (tablePage == 'saas') {
24                     getData()
25                 } else {
26                     getTenatData()
27                 }
28                 return true;
29             } catch (error) {
30                 hide();
31                 message.error('删除失败请重试! ');
32                 return false;
33             }
34         }}
35         okText="确认"
36         cancelText="取消"
37     >
38         <Button type="link" danger>
39             删除
40         </Button>
41     </Popconfirm>
42 </div>
43 </div>
44     );
45 }}
46     treeData={tablePage == 'saas' ? treeData : tenantTreeData}
47 </>
48 }}
49 </Spin>
50 <ModalForm
51     title="新建"
52     width="600px"
53     visible={createModalVisible}
```

行李托运平台 V1.0

```
1      onVisibleChange={handleModalVisible}
2      onFinish={async (value: { name: string; type: number; url: string }) => {
3          let pid: number = 0;
4          if (currentRow) {
5              pid = currentRow.id || 0;
6          }
7          const success = await handleAdd({ master: tablePage == 'saas' ? 1 : 2, pid, ...value});
8          if (success) {
9              handleModalVisible(false);
10             setCurrentRow(null);
11             if (tablePage == 'saas') {
12                 getData()
13             } else {
14                 getTenatData()
15             }
16         }
17     }}
18     modalProps={{
19         destroyOnClose: true,
20     }}
21     initialValues={{
22         pname: currentRow?.name,
23         purl: currentRow?.url,
24     }}
25 >
26     {currentRow && (
27         <>
28             <ProFormText disabled width="md" name="pname" label={`父菜单名称`} />
29             <ProFormText disabled width="md" name="purl" label={`父地址`} />
30         </>
31     )}
32     {editAndAddFrom()}
33 </ModalForm>
34
35 <ModalForm
36     title={info ? '详情' : '编辑'}
37     width="600px"
38     visible={updateModalVisible}
39     onVisibleChange={(value) => {
40         if (!value) {
41             setInfo(false);
42         }
43         handleUpdateModalVisible(value);
44     }}
45     onFinish={async (value) => {
46         const success = await handleUpdate({
47             id: currentRow?.id, ...value
48         });
49         if (success) {
50             handleUpdateModalVisible(false);
51             setCurrentRow(null);
52             if (tablePage == 'saas') {
53                 getData()
```

行李托运平台 V1.0

```
1         } else {
2             getTenatData()
3         }
4     }
5 }}
6 modalProps={{
7     destroyOnClose: true,
8 }}
9 initialValues={{
10     name: currentRow?.name,
11     type: currentRow?.type + '', // 转为字符串
12     url: currentRow?.url,
13 }}
14 >
15     {editAndAddFrom()}
16 </ModalForm>
17 </PageContainer>
18 );
19 };
20
21 export default TableList;
22 ```
23 File: src/pages/auth/role/index.tsx
24 ```tsx
25 import {PlusOutlined} from '@ant-design/icons';
26 import {Button, message, Modal, Tree, Spin, Select} from 'antd';
27 import React, {useState, useRef, useEffect} from 'react';
28 import {PageContainer} from '@ant-design/pro-layout';
29 import type {ProColumns, ActionType} from '@ant-design/pro-table';
30 import type {TreeProps} from 'antd/es/tree';
31 import ProTable from '@ant-design/pro-table';
32 import {ModalForm, ProFormText, ProFormSelect} from '@ant-design/pro-form';
33 import {roleList, addRole, updateRule, treeList, treeUpdate} from '../service';
34 import {listMenu} from '../menu/service';
35 import type {TableListItem, TableListPagination} from '../data';
36 import type {RootObject} from '../menu/data';
37 import TreeTag from '@components/TreeTag';
38 import {ProCard} from "@ant-design/pro-components";
39 import {listTenant} from "@api/tenant";
40 /**
41  * 添加节点
42  *
43  * @param fields
44  */
45 const handleAdd = async (fields: {
46     tenant_id: string;
47     updated_at: string;
48     name: string;
49     guid: string;
50     created_at: string;
51     id: number;
52     deleted_at?: boolean;
53     app_id: string;
```

行李托运平台 V1.0

```
1      platform: string;
2      status: number
3  }) => {
4      const hide = message.loading('正在添加');
5
6      try {
7          await addRole({...fields});
8          hide();
9          message.success('添加成功');
10         return true;
11     } catch (error) {
12         hide();
13         return false;
14     }
15 };
16 /**
17  * 更新节点
18  *
19  * @param fields
20  */
21 const handleUpdate = async (fields: any, currentRow?: TableListItem) => {
22     const hide = message.loading('正在配置');
23
24     try {
25         await updateRule({
26             ...currentRow,
27             ...fields,
28         });
29         hide();
30         message.success('配置成功');
31         return true;
32     } catch (error) {
33         hide();
34         message.error('配置失败请重试! ');
35         return false;
36     }
37 };
38 /**
39  * 删除节点
40  *
41  * @param selectedRows
42  */
43 const TableList: React.FC = () => {
44     /** 新建窗口的弹窗 */
45     const [createModalVisible, handleModalVisible] = useState<boolean>(false);
46     /** 分布更新窗口的弹窗 */
47     const [updateModalVisible, handleUpdateModalVisible] = useState<boolean>(false);
48     const [tablePage, handleTablePage] = useState<string>('saas');
49     const [authVisible, handleAuthVisible] = useState<boolean>(false);
50     const actionRef = useRef<ActionType>();
51     const [currentRow, setCurrentRow] = useState<TableListItem>();
52     const [treeData, setTreeData] = useState<RootObject[]>([]);
53     const [tenantTreeData, setTenantTreeData] = useState<RootObject[]>([]);
```

行李托运平台 V1.0

```
1  const [checkedKeys, setCheckedKeys] = useState<number[]>([]);
2  const [tenantList, setTenantList] = useState([]);
3  const [currentTenant, setCurrentTenant] = useState<string>('');
4  // 配置权限树控件加载
5  const [loading, setLoading] = useState<boolean>(false);
6
7  useEffect(() => {
8    const getTreeData = async () => {
9      const data: RootObject[] = await listMenu({});
10     setTreeData(data);
11   };
12   const getTeantTreeData = async () => {
13     const data: RootObject[] = await listMenu({tenantId: 2});
14     setTenantTreeData(data);
15   };
16   getTreeData();
17   getTeantTreeData()
18 }, []);
19 const getTenantList = async () => {
20   let data = await listTenant({});
21   if (data) {
22     data = data.data?.map((item: any) => {
23       return {label: item.title, value: item.id}
24     })
25     setTenantList(data);
26   }
27 };
28
29 useEffect(() => {
30   getTenantList();
31 }, []);
32
33 const columns: ProColumns<TableListItem>[] = [
34   {
35     title: 'ID',
36     dataIndex: 'id',
37     search: false,
38   },
39   {
40     title: '名称',
41     dataIndex: 'name',
42     valueType: 'textarea',
43   },
44   {
45     title: '状态',
46     dataIndex: 'status',
47     hideInForm: true,
48     valueEnum: {
49       1: {
50         text: '正常',
51         status: 'Processing',
52       },
53       2: {
```


行李托运平台 V1.0

```
1         text: '禁用',
2         status: 'Error',
3     },
4 },
5 },
6 {
7     title: '创建时间',
8     dataIndex: 'created_at',
9     search: false,
10 },
11 {
12     title: '操作',
13     dataIndex: 'option',
14     valueType: 'option',
15     render: (_, record) => [
16         <a
17             key="config"
18             onClick={() => {
19                 handleUpdateModalVisible(true);
20                 setCurrentRow(record);
21             }}
22         >
23             编辑
24         </a>,
25         <a
26             key="auth"
27             onClick={async () => {
28                 setCurrentRow(record);
29                 setLoading(true);
30                 handleAuthVisible(true);
31                 const data: number[] = await treeList({guid: record?.guid});
32                 setCheckedKeys(data);
33                 setLoading(false);
34             }}
35         >
36             配置权限
37         </a>,
38     ],
39 },
40 ];
41 const onSelect: TreeProps['onSelect'] = (selectedKeys, info) => {
42     console.log('selected', selectedKeys, info);
43 };
44
45 const onCheck: TreeProps['onCheck'] = (keys: any, info: any) => {
46     const now = info.node;
47     if (info.checked && now.pid !== 0 && !checkedKeys.includes(now.pid)) {
48         message.error('请保证其父级已被选中，不然选择无效');
49         return;
50     }
51     treeUpdate({guid: currentRow?.guid, menu_id: now?.id, edit: info.checked ? 1 : 2});
52
53     setCheckedKeys(keys.checked);
```

行李托运平台 V1.0

```
1      };
2      return (
3        <PageContainer
4          tabList=[
5            {
6              tab: 'saas 角色管理',
7              key: 'saas',
8            },
9            {
10             tab: '租户角色管理',
11             key: 'tenant',
12           },
13          ]
14          onTabChange={(value) => {
15            handleTablePage(value)
16          }}
17        >
18          {tablePage === 'saas' ? null : <ProCard bordered>
19            <span
20              style={{marginLeft: 48}}>
21              当前租户 :
22            </span>
23            <Select
24              value={currentTenant}
25              style={{width: 280, marginLeft: 8}}
26              onChange={async (value) => {
27                setCurrentTenant(value);
28                actionRef.current?.reload()
29              }}
30              options={tenantList}
31            />
32          </ProCard>
33          <ProTable<TableListItem, TableListPagination>
34            headerTitle="查询表格"
35            key={tablePage}
36            actionRef={actionRef}
37            rowKey="id"
38            search={{
39              labelWidth: 120,
40            }}
41            toolBarRender={() => [
42              <Button
43                type="primary"
44                key="primary"
45                onClick={() => {
46                  handleModalVisible(true);
47                }}
48              />
49            ]
50            <PlusOutlined/> 新建
51          </Button>,
52        ]}
53      );
```

行李托运平台 V1.0

```
1      request={async (params) => {
2          if (tablePage === 'saas') {
3              // @ts-ignore
4              params.tenant_id = 'master';
5          } else {
6              if (currentTenant === '') {
7                  // @ts-ignore
8                  setCurrentTenant(tenantList[0].value)
9                  // @ts-ignore
10                 params.tenant_id = tenantList[0].value;
11             } else {
12                 // @ts-ignore
13                 params.tenant_id = currentTenant;
14             }
15         }
16
17         const msg = await roleList(params);
18         return {
19             data: msg.data,
20             success: true,
21             total: msg.count,
22         };
23     }}
24     columns={columns}
25 />
26
27 {/* 新建 modal */}
28 <ModalForm
29     title="新建"
30     width="600px"
31     visible={createModalVisible}
32     onVisibleChange={handleModalVisible}
33     modalProps={{
34         destroyOnClose: true,
35     }}
36     onFinish={async (value) => {
37         const success = await handleAdd({
38             ...value as TableListItem,
39             'tenant_id': tablePage === 'saas' ? 'master' : currentTenant
40         });
41         if (success) {
42             handleModalVisible(false);
43             if (actionRef.current) {
44                 actionRef.current.reload();
45             }
46         }
47     }}
48 >
49 <ProFormText
50     rules={[
51         {
52             required: true,
53             message: '名称为必填项',
```

行李托运平台 V1.0

```
1      },
2    }}
3    width="md"
4    name="name"
5    label="角色名称"
6  />
7 </ModalForm>
8
9 <ModalForm
10   title="编辑"
11   width="600px"
12   visible={updateModalVisible}
13   onVisibleChange={handleUpdateModalVisible}
14   initialValues={{
15     ...currentRow,
16     status: currentRow ? currentRow.status + " : ",
17   }}
18   modalProps={{
19     destroyOnClose: true,
20   }}
21   onFinish={async (value) => {
22     const success = await handleUpdate(value, currentRow);
23
24     if (success) {
25       handleUpdateModalVisible(false);
26       setCurrentRow(undefined);
27
28       if (actionRef.current) {
29         actionRef.current.reload();
30       }
31     }
32   }}
33 >
34   <ProFormText
35     rules={[
36       {
37         required: true,
38         message: '名称为必填项',
39       },
40     ]}
41     width="md"
42     name="name"
43     label="角色名称"
44   />
45   <ProFormSelect
46     name="status"
47     width="md"
48     label="状态"
49     valueEnum={{
50       '1': '正常',
51       '2': '禁用',
52     }}
53   />
```

行李托运平台 V1.0

```
1      </ModalForm>
2
3      <Modal
4        visible={authVisible}
5        title="分配权限"
6        onCancel={() => {
7          handleAuthVisible(false);
8        }}
9        destroyOnClose
10       footer={null}
11     >
12       <Spin spinning={loading}>
13         <Tree
14           checkable
15           blockNode
16           selectable={false}
17           checkStrictly
18           defaultExpandAll
19           checkedKeys={checkedKeys}
20           fieldNames={{key: 'id', title: 'name'}}
21           onSelect={onSelect}
22           onCheck={onCheck}
23           treeData={tablePage === 'saas' ? treeData : tenantTreeData}
24           titleRender={(data) => <TreeTag title={data?.name} type={data.type}/>}
25         />
26       </Spin>
27     </Modal>
28   </PageContainer>
29 );
30 };
31
32 export default TableList;
33
34 File: src/pages/database/table/index.tsx
35 ```tsx
36 import React, { useEffect, useRef, useState } from 'react';
37 import { PageContainer } from "@ant-design/pro-layout";
38 import type { ProColumns } from "@ant-design/pro-table";
39 import ProTable from "@ant-design/pro-table";
40 import { Button, Input, Modal, Select, message } from "antd";
41 import { PlusOutlined } from "@ant-design/icons";
42 import { history } from "umi";
43 import type { ActionType } from "@ant-design/pro-table/lib/typing";
44 import { createTable, deleteTable, tableList, updateTable } from '../service';
45 import { ModalForm, ProFormText } from '@ant-design/pro-form';
46 import type { TableList } from '../data';
47 import { schemaList } from '../schema/service';
48 import type { SchemaList } from '../schema/data';
49 /**
50  * 添加和编辑
51  */
52 const handleUpdate = async (fields: any, currentRow?: any) => {
53   const hide = message.loading('正在修改');
```

行李托运平台 V1.0

```
1      if (currentRow) {
2          const oldTableName = fields.oldTableName;
3          const newTableName = fields.newTableName;
4          if (oldTableName && newTableName && oldTableName === newTableName) {
5              hide();
6              message.error('修改的名称不能与原名称一致');
7              return false;
8          }
9          try {
10             await updateTable({
11                 ...fields,
12             });
13             hide();
14             message.success('修改成功');
15             return true;
16         } catch (error) {
17             hide();
18             return false;
19         }
20     } else {
21         try {
22             await createTable({
23                 ...fields,
24             });
25             hide();
26             message.success('新建成功');
27             return true;
28         } catch (error) {
29             console.log('error');
30             console.log(error);
31             hide();
32             return false;
33         }
34     }
35 };
36
37 const TenantList: React.FC = () => {
38     const schemaName = history.location.query?.schemaName;
39     const actionRef = useRef<ActionType>(null);
40     const { confirm } = Modal;
41     let deleteValue = '';
42     const [modalVisible, handleModalVisible] = useState<boolean>(false);
43     const [currentRow, setCurrentRow] = useState<any>();
44     const [schemaOptions, setSchemaOptions] = useState([]);
45
46     const handleModelDestroy = (field: boolean) => {
47         if (!field) {
48             setCurrentRow(undefined);
49         }
50         handleModalVisible(field);
51     }
52
53     const getSchemaList = async () => {
```

行李托运平台 V1.0

```
1      const res = await schemaList({ current: 1, pageSize: 1000 });
2      if (res) {
3          const options = res.data.map((item: SchemaList) => {
4              return {
5                  label: item.schemaName,
6                  value: item.schemaName
7              }
8          });
9          setSchemaOptions(options);
10     }
11 }
12
13 useEffect(() => {
14     getSchemaList();
15 }, []);
16
17 /**
18  * 删除确认
19  */
20 const showDeleteConfirm = (record: TableList) => {
21     confirm({
22         title: '请确认删除',
23         content: (
24             <div>
25                 <div style={{ marginTop: 40 }} />
26                 <Input
27                     onChange={(e) => {
28                         deleteValue = e.target.value;
29                     }}
30                     placeholder="请输入表名加密钥"
31                 />
32                 <div style={{ marginTop: 6 }} />
33                 <p style={{ color: 'red', fontSize: 12 }}> 请输入表名加密钥以确认删除操作</p>
34             </div>
35         ),
36         onOk() {
37             if (deleteValue === (record.tableName + 'lsyzw')) {
38                 if (schemaName && schemaName !== null && schemaName !== '') {
39                     deleteTable({ schemaName: schemaName, tableName: record.tableName }).then(res => {
40                         if (res) {
41                             message.success('删除成功');
42                         }
43                         actionRef.current?.reload();
44                     }).catch(err => {
45                         console.log("删除失败:", err);
46                         message.error('删除失败');
47                         actionRef.current?.reload();
48                     });
49                 } else {
50                     message.error('模式名称不存在, 请重试');
51                 }
52             } else {
53                 message.error('输入错误, 请确认输入正确租户表名');
```

行李托运平台 V1.0

```
1      }
2    },
3    onCancel() {
4      console.log('取消删除');
5    },
6  }},
7};
8const columns: ProColumns<TableList>[] = [
9  {
10    title: '序号',
11    dataIndex: 'index',
12    valueType: 'index',
13    width: 80,
14  },
15  {
16    title: '模式选择',
17    hideInTable: true,
18    renderFormItem: () => {
19      return <Select
20        showSearch
21        onChange={(value) => {
22          history.push('/database/table?schemaName=' + value);
23          if (actionRef.current) {
24            actionRef.current.reload();
25          }
26        }}
27        defaultValue={schemaName}
28        options={schemaOptions}
29      />
30    },
31  },
32  {
33    dataIndex: 'tableName',
34    title: '表名称',
35    key: 'tableName',
36    copyable: true
37  },
38  {
39    dataIndex: 'description',
40    title: '描述',
41    search: false,
42  },
43  {
44    title: '操作',
45    key: 'options',
46    width: 300,
47    search: false,
48    align: 'center',
49    valueType: 'option',
50    render: (_, any, record: TableList) => [
51      <Button
52        key={'edit'}
53        type={'link'}

```


行李托运平台 V1.0

```
1      onClick={() => {
2          history.push('/database/fields?schemaName=' + schemaName + "&tableName=" + record.tableName);
3      }}
4  >
5      字段管理
6  </Button>,
7  <Button
8      key={'edit'}
9      type={'link'}
10     onClick={() => {
11         setCurrentRow(record);
12         handleModalVisible(true);
13     }}
14 >
15     编辑
16 </Button>,
17 <Button
18     key={'delete'}
19     type={'link'}
20     style={{ color: 'red' }}
21     onClick={() => showDeleteConfirm(record)}
22 >
23     删除
24 </Button>
25 ]
26 }
27 ];
28 return (
29     <PageContainer>
30         <ProTable
31             actionRef={actionRef}
32             headerTitle={schemaName ? "模式: " + schemaName : "表列表"}
33             rowKey="tableName"
34             search={{
35                 labelWidth: 120,
36             }}
37             columns={columns}
38             request={async (param: any) => {
39                 if (schemaName) {
40                     param.schemaName = schemaName;
41                     const res = await tableList(param);
42                     if (res?.total > 0) {
43                         return res;
44                     }
45                 }
46                 return { data: [] };
47             }}
48             pagination={{
49                 showSizeChanger: true,
50                 pageSizeOptions: [10, 20, 50, 100],
51             }}
52             toolBarRender={() => [
53                 <Button
```

行李托运平台 V1.0

```
1         type="primary"
2         key="primary"
3         onClick={() => {
4             if (schemaName) {
5                 handleModalVisible(true);
6             } else {
7                 message.info("请先选择模式");
8             }
9         }}
10    >
11        <PlusOutlined /> 新建
12    </Button>,
13  ]}
14 />
15 <ModalForm
16   title={currentRow ? '编辑' : '新建'}
17   width="600px"
18   visible={modalVisible}
19   onVisibleChange={handleModelDestroy}
20   modalProps={{
21     destroyOnClose: true,
22   }}
23   onFinish={async (value) => {
24     const success = await handleUpdate(value, currentRow);
25     if (success) {
26       handleModalVisible(false);
27       setCurrentRow(undefined);
28       if (actionRef.current) {
29         actionRef.current.reload();
30       }
31     }
32   }}
33 >
34   <ProFormText
35     rules={[
36       {
37         required: true,
38         message: '模式名称为必填项',
39       },
40     ]}
41     fieldProps={{ maxLength: 20 }}
42     width="md"
43     name="schemaName"
44     initialValue={schemaName}
45     label="模式名称"
46     disabled
47   />
48   {currentRow ?
49     <>
50       <ProFormText
51         rules={[
52           {
53             required: true,
```

行李托运平台 V1.0

```
1      message: '表名称为必填项',
2    },
3  ]}
4  fieldProps={{ maxLength: 20 }}
5  width="md"
6  name="oldTableName"
7  initialValue={currentRow.tableName}
8  label="需要修改的表名称"
9  disabled
10 />
11 <ProFormText
12   fieldProps={{ maxLength: 20 }}
13   width="md"
14   name="newTableName"
15   label="新表名称"
16 />
17 <ProFormText
18   fieldProps={{ maxLength: 30 }}
19   width="md"
20   initialValue={currentRow.description}
21   name="tableComment"
22   label="描述"
23 />
24 </> :
25 <
26   <ProFormText
27     rules={[
28       {
29         required: true,
30         message: '表名称为必填项',
31       },
32     ]}
33     fieldProps={{ maxLength: 20 }}
34     width="md"
35     name="tableName"
36     label="表名称"
37   />
38   <ProFormText
39     fieldProps={{ maxLength: 30 }}
40     width="md"
41     name="tableComment"
42     label="描述"
43   />
44 </>
45 }
46 </ModalForm>
47 </PageContainer>
48 );
49 };
50
51 export default TenantList;
52 ```
53 File: src/pages/tenant/addEdit/index.tsx
```

行李托运平台 V1.0

```
1  ````tsx
2  import React, { useEffect, useRef, useState } from "react";
3  import { PageContainer } from "@ant-design/pro-layout";
4  import { history } from "umi";
5  import { getTenantDetail, saveTenant, updateTenant } from "@pages/tenant/service";
6  import type { TenantListItem } from "@pages/tenant/list/data";
7  import type { ProFormInstance } from "@ant-design/pro-form";
8  import {
9    ProForm, ProFormDependency,
10    ProFormGroup,
11    ProFormItem,
12    ProFormList,
13    ProFormRadio,
14    ProFormSelect,
15    ProFormText
16  } from "@ant-design/pro-form";
17  import ProCard from "@ant-design/pro-card";
18  import { ProFormTextArea } from "@ant-design/pro-components";
19  import type { RuleObject, StoreValue } from "rc-field-form/lib/interface";
20  import { Divider, Form, message, type SelectProps } from "antd";
21  import FileUpload from "@components/ossUploadFile";
22
23  const WxTagOptions: SelectProps<any>['options'] | string[] = [
24    {
25      label: '用户端',
26      type: 1,
27      value: 'wx_web'
28    },
29    {
30      label: '商家端',
31      type: 1,
32      value: 'wx_store'
33    },
34    {
35      label: '引游人端',
36      type: 1,
37      value: 'wx_guide'
38    },
39    {
40      label: '抖音用户端',
41      type: 2,
42      value: 'dy_web'
43    },
44  ];
45
46  const TenantAddEdit: React.FC = () => {
47    const queryId = history.location.query?.id;
48    const [form] = Form.useForm();
49    const [loading, setLoading] = useState<boolean>(false);
50
51    const formRef = useRef<ProFormInstance<any>>()>();
52    useEffect(() => {
53      if (queryId) {
```

行李托运平台 V1.0

```
1      setLoading(true);
2      getTenantDetail({ id: queryId, decrypt: true }).then(res => {
3          form.setFieldsValue(res);
4      });
5      setLoading(false);
6  }
7  }, [queryId]);
8
9  const getWxPayInfoList = (type = 1) => {
10     return <ProFormList label={'支付信息'} name={'pays'}
11         // rules=[
12         //     {
13         //         validator: (_, RuleObject, value: StoreValue) => {
14         //             if (value.length < 1) {
15         //                 message.error('支付信息不能为空').then(r => r);
16         //                 return Promise.reject();
17         //             }
18         //             return Promise.resolve();
19         //         }
20         //     ]
21         // ];
22     >
23     {() => {
24
25         if (type == 1) {
26             return <>
27                 <ProFormGroup>
28                     <ProFormText
29                         label={'微信支付商户号'}
30                         name={'mch_id'}
31                         rules={[
32                             {
33                                 required: false
34                             }
35                         ]}
36                         fieldProps={{
37                             showCount: true,
38                             maxLength: 32
39                         }}
40                     />
41
42                     <ProFormText
43                         label={'微信支付商户名称'}
44                         name={'mch_name'}
45                         rules={[
46                             {
47                                 required: false
48                             }
49                         ]}
50                         fieldProps={{
51                             showCount: true,
52                             maxLength: 20
53                         }}

```

行李托运平台 V1.0

```
1      />
2    </ProFormGroup>
3
4    <ProFormGroup>
5
6      <ProFormText
7        label={'v2 密钥'}
8        name={'v2_key'}
9        rules={[
10          {
11            required: false
12          }
13        ]}
14        fieldProps={{
15          showCount: true,
16          maxLength: 32
17        }}
18      />
19
20      <ProFormItem
21        label={'v2 证书'}
22        name={'cert_path'}
23        rules={[
24          {
25            required: false
26          }
27        ]}
28      >
29        <FileUpload />
30      </ProFormItem>
31
32      <ProFormItem
33        label={'v2 证书私钥'}
34        name={'cert_key_path'}
35        rules={[
36          {
37            required: false
38          }
39        ]}>
40        <FileUpload />
41      </ProFormItem>
42
43    </ProFormGroup>
44  </>
45 }
46 if (type == 2) {
47   return <ProFormGroup>
48
49     <ProFormText
50       label={'抖音 salt'}
51       tooltip={'使用抖音支付必须'}
52       name={'dy_salt'}
53
```

行李托运平台 V1.0

```
1         rules=[
2             {
3                 required: false
4             }
5         ]
6         fieldProps={{
7             showCount: true,
8             maxLength: 32
9         }}
10    />
11
12    <ProFormText
13        label={'抖音 token'}
14        name={'dy_token'}
15        tooltip={'使用抖音支付必须'}
16        rules=[
17            {
18                required: false
19            }
20        ]
21        fieldProps={{
22            showCount: true,
23            maxLength: 32
24        }}
25    />
26
27    </ProFormGroup>
28  }
29  return null
30 }}
31
32
33    <ProFormRadio.Group
34        label={'是否默认支付商户'}
35        name={'default'}
36        rules=[
37            {
38                required: true
39            }
40        ]
41        initialValue={2}
42        options=[
43            { label: '是', value: 1 },
44            { label: '否', value: 2 },
45        ]
46    />
47
48    </ProFormList>;
49  }
50
51  const getWxInfoList = () => {
52    return <ProFormList label={'小程序信息'} name={'applets'} rules=[
53      {
```

行李托运平台 V1.0

```
1      validator: ( _: RuleObject, value: StoreValue) => {
2        if (value.length < 1) {
3          message.error('小程序信息不能为空').then(r => r);
4          return Promise.reject();
5        }
6        const tagsMap = {}
7        value.forEach((item: { tag: any; }) => {
8          tagsMap[item.tag] = 1
9        })
10       if (value.length !== Object.keys(tagsMap).length) {
11         message.error('小程序信息重复').then(r => r);
12         return Promise.reject();
13       }
14       return Promise.resolve();
15     }
16   }
17 ]>
18   {(meta, index, action) =>
19     (<>
20       <ProFormGroup>
21         <ProFormSelect
22           label={'平台'}
23           name={'type'}
24           width={'sm'}
25           rules={[
26             {
27               required: true
28             }
29           ]}
30           options={
31             [
32               {
33                 label: '微信小程序',
34                 value: 1
35               }, {
36                 label: '抖音小程序',
37                 value: 2
38               }
39             ]
40           }
41           fieldProps={{
42             onSelect: () => {
43               action.setCurrentRowData(
44                 {
45                   tag: undefined,
46                   pays: undefined
47                 }
48               )
49             }
50           }}
51         />
52         { /* eslint-disable-next-line react/jsx-no-undef */ }
53         <ProFormDependency name={['type']>
54           ({ type }) => {
```


行李托运平台 V1.0

```
1      return <ProFormSelect
2          label={'小程序'}
3          name={'tag'}
4          width={"sm"}
5          rules={[
6              {
7                  required: true
8              }
9          ]}
10         options={WxTagOptions?.filter(item => item.type === type)}
11     />
12 }}
13
14 </ProFormDependency>
15 <ProFormText
16     label={'小程序名称'}
17     name={'name'}
18     rules={[
19         {
20             required: true
21         }
22     ]}
23     fieldProps={{
24         showCount: true,
25         maxLength: 20
26     }}
27 />
28
29 </ProFormGroup>
30
31 <ProFormGroup>
32
33     <ProFormText
34         label={'appid'}
35         name={'app_id'}
36         rules={[
37             {
38                 required: true
39             }
40         ]}
41         fieldProps={{
42             showCount: true,
43             maxLength: 32
44         }}
45     />
46
47     <ProFormText
48         label={'secret'}
49         name={'app_secret'}
50         rules={[
51             {
52                 required: true
53             }
54         ]}
55     />
```

行李托运平台 V1.0

```
1         ]}
2         fieldProps={{
3             showCount: true,
4             maxLength: 32
5         }}
6     />
7 </ProFormGroup>
8
9 <Divider />
10
11 <ProFormDependency name={['type']>
12     {{{ type }} => {
13         return getWxPayInfoList(type)
14     }}
15 </ProFormDependency>
16
17
18 </>}}
19 </ProFormList>;
20 }
21 return (
22     <PageContainer>
23         <ProCard title={queryId ? '编辑租户' : '新建租户'} loading={loading}>
24             <ProForm
25                 form={form}
26                 formRef={formRef}
27                 onFinish={async (values: TenantListItem) => {
28                     let res = null;
29                     if (queryId) {
30                         res = await updateTenant({ ...values, id: queryId });
31                     } else {
32                         res = await saveTenant(values);
33                     }
34
35
36                     if (res) {
37                         message.success('保存成功');
38                         history.goBack();
39                     } else {
40                         message.error('保存失败');
41                     }
42                 }}
43             >
44                 <ProFormText
45                     label={'标题'}
46                     name={'title'}
47                     width={'md'}
48                     rules={[
49                         { required: true }
50                     ]}
51                     fieldProps={{
52                         showCount: true,
53                         maxLength: 20
```

行李托运平台 V1.0

```
1      }}
2    />
3
4    <ProFormTextArea
5      label={'描述'}
6      name={'desc'}
7      width={'md'}
8      rules={[
9        {
10          required: false
11        }
12      ]}
13      fieldProps={{
14        showCount: true,
15        maxLength: 150
16      }}
17    />
18
19    <Divider />
20
21    {getWxInfoList()}
22
23  </ProForm>
24 </ProCard>
25 </PageContainer>
26   };
27
28 };
29
30 export default TenantAddEdit;
31 ```
32 File: src/pages/tenant/list/index.tsx
33 ```tsx
34 import React, { useRef, useState } from 'react';
35 import { PageContainer } from "@ant-design/pro-layout";
36 import type { ProColumns } from "@ant-design/pro-table";
37 import ProTable from "@ant-design/pro-table";
38 import { deleteTenant, getTenantList } from "@pages/tenant/service";
39 import DateFormat from "@components/DateFormat";
40 import type { TenantListItem } from "@pages/tenant/list/data";
41 import { Button, Input, Modal, message } from "antd";
42 import { PlusOutlined } from "@ant-design/icons";
43 import { history } from "umi";
44 import type { ActionType } from "@ant-design/pro-table/lib/typing";
45
46 const TenantList: React.FC = () => {
47   const actionRef = useRef<ActionType>(null);
48   const { confirm } = Modal;
49   let deleteValue = "";
50
51   const showDeleteConfirm = (record: any) => {
52     confirm({
53       title: '请确认删除',
```

行李托运平台 V1.0

```
1      content: (
2        <div>
3          <div style={{ marginTop: 40 }} />
4          <Input
5            onChange={e} => {
6              deleteValue = e.target.value;
7            }
8            placeholder="请输入租户 id"
9          />
10         <div style={{ marginTop: 6 }} />
11         <p style={{ color: 'red', fontSize: 12 }}> 请输入租户 id 以确认删除操作</p>
12       </div>
13     ),
14     onOk() {
15       if (deleteValue === record.id) {
16         message.success('删除成功');
17         deleteTenant({ id: record.id }).then(res => {
18           if (res) {
19             message.success('删除成功');
20           }
21           actionRef.current?.reload();
22         }).catch(err => {
23           console.log("删除失败:", err);
24           message.error('删除失败');
25           actionRef.current?.reload();
26         });
27       } else {
28         message.error('输入错误，请确认输入正确租户 id');
29       }
30     },
31     onCancel() {
32       console.log('取消删除');
33     },
34   });
35 };
36
37 const columns: ProColumns<TenantListItem>[] = [
38   {
39     dataIndex: 'id',
40     title: 'id',
41     key: 'id',
42     width: 120,
43     ellipsis: true,
44     copyable: true
45   },
46   {
47     dataIndex: 'title',
48     title: '标题',
49     key: 'title',
50     ellipsis: true,
51   },
52   {
53     dataIndex: 'tenancy_db_name',
```

行李托运平台 V1.0

```
1      title: '数据库名称',
2      key: 'tenancy_db_name',
3      width: 150,
4      ellipsis: true,
5      search: false,
6      copyable: true
7    },
8    {
9      dataIndex: 'created_at',
10     title: '创建时间',
11     key: 'created_at',
12     search: false,
13     render: (dom, entity: TenantListItem) => {
14       return <DateFormat date={entity.created_at} />
15     }
16   },
17   {
18     dataIndex: 'updated_at',
19     title: '修改时间',
20     key: 'updated_at',
21     search: false,
22     render: (dom, entity: TenantListItem) => {
23       return <DateFormat date={entity.updated_at} />
24     }
25   },
26   {
27     title: '操作',
28     key: 'options',
29     width: 250,
30     search: false,
31     valueType: 'option',
32     render: (_, any, record: TenantListItem) => [
33       <Button
34         key={'edit'}
35         type={'link'}
36         onClick={() => {
37           history.push('/tenant/addEdit?id=' + record.id);
38         }}
39       >
40         编辑
41       </Button>,
42       <Button
43         key={'delete'}
44         type={"link"}
45         style={{ color: 'red' }}
46         onClick={() => showDeleteConfirm(record)}
47       >
48         删除
49       </Button>
50     ]
51   }
52 ];
53
```

行李托运平台 V1.0

```
1      return (
2        <PageContainer>
3          <ProTable
4            actionRef={actionRef}
5            headerTitle={'租户列表'}
6            rowKey="id"
7            search={{
8              labelWidth: 120,
9            }}
10           columns={columns}
11           request={getTenantList}
12           pagination={{
13             showSizeChanger: true,
14             pageSizeOptions: [10, 20, 50, 100],
15           }}
16           toolBarRender={() => [
17             <Button
18               type="primary"
19               key="primary"
20               onClick={() => {
21                 history.push('/tenant/addEdit');
22               }}
23             >
24               <PlusOutlined /> 新建
25             </Button>,
26           ]}
27         />
28       </PageContainer>
29     );
30   };
31
32   export default TenantList;
33
34
35   ```
36
37   File: src/pages/database/schema/index.tsx
38   ```tsx
39   import React, { useRef, useState } from 'react';
40   import { PageContainer } from "@ant-design/pro-layout";
41   import type { ProColumns } from "@ant-design/pro-table";
42   import ProTable from "@ant-design/pro-table";
43   import DateFormat from "@components/DateFormat";
44   import { Button, Input, Modal, message } from "antd";
45   import { PlusOutlined } from "@ant-design/icons";
46   import { history } from "umi";
47   import type { ActionType } from "@ant-design/pro-table/lib/typing";
48   import { createSchema, deleteSchema, schemaList, updateSchema } from './service';
49   import { ModalForm, ProFormText } from '@ant-design/pro-form';
50   import type { SchemaList } from './data';
51
52
53   /**
```

行李托运平台 V1.0

```
1  * 添加和编辑
2  */
3  const handleUpdate = async (fields: any, currentRow?: any) => {
4      const hide = message.loading('正在修改');
5      if (currentRow) {
6          try {
7              await updateSchema({
8                  oldSchemaName: fields.schemaName,
9                  newSchemaName: fields.newSchemaName,
10              });
11              hide();
12              message.success('修改成功');
13              return true;
14          } catch (error) {
15              hide();
16              return false;
17          }
18      } else {
19          try {
20              await createSchema({
21                  ...fields,
22              });
23              hide();
24              message.success('新建成功');
25              return true;
26          } catch (error) {
27              console.log('error');
28              console.log(error);
29              hide();
30              return false;
31          }
32      }
33  };
34
35  const TenantList: React.FC = () => {
36      const actionRef = useRef<ActionType>(null);
37      const { confirm } = Modal;
38      let deleteValue = '';
39      const [modalVisible, handleModalVisible] = useState<boolean>(false);
40      const [currentRow, setCurrentRow] = useState<any>({});
41
42      /**
43       * 删除确认
44       */
45      const showDeleteConfirm = (record: SchemaList) => {
46          confirm({
47              title: '请确认删除',
48              content: (
49                  <div>
50                      <div style={{ marginTop: 40 }} />
51                      <Input
52                          onChange={e => {
53                              deleteValue = e.target.value;
```

行李托运平台 V1.0

```
1      }}
2      placeholder="请输入模式名加密钥"
3    />
4    <div style={{ marginTop: 6 }} />
5    <p style={{ color: 'red', fontSize: 12 }}> 请输入模式名加密钥以确认删除操作</p>
6  </div>
7  },
8  onOk() {
9    if (deleteValue === (record.schemaName + 'lsyzw')) {
10      deleteSchema({ schemaName: record.schemaName }).then(res => {
11        if (res) {
12          message.success('删除成功');
13        }
14        actionRef.current?.reload();
15      }).catch(err => {
16        console.log("删除失败:", err);
17        message.error('删除失败');
18        actionRef.current?.reload();
19      });
20    } else {
21      message.error('输入错误，请确认输入正确租户模式名');
22    }
23  },
24  onCancel() {
25    console.log('取消删除');
26  },
27  });
28  };
29
30  const columns: ProColumns<SchemaList>[] = [
31    {
32      title: '序号',
33      dataIndex: 'index',
34      valueType: 'index',
35      width: 80,
36    },
37    {
38      dataIndex: 'schemaName',
39      title: '模式名称',
40      key: 'schemaName',
41      copyable: true
42    },
43    {
44      dataIndex: 'schemaOwner',
45      title: '模式所有者',
46      search: false,
47    },
48    {
49      title: '操作',
50      key: 'options',
51      width: 300,
52      search: false,
53      align: 'center',
```


行李托运平台 V1.0

```
1      valueType: 'option',
2      render: (_, any, record: SchemaList) => [
3        <Button
4          key={'edit'}
5          type={'link'}
6          onClick={() => {
7            history.push('/database/table?schemaName=' + record.schemaName);
8          }}
9        >
10         表管理
11      </Button>,
12      <Button
13        key={'edit'}
14        type={'link'}
15        onClick={() => {
16          setCurrentRow(record.schemaName);
17          handleModalVisible(true);
18        }}
19      >
20        编辑
21      </Button>,
22      <Button
23        key={'delete'}
24        type={"link"}
25        style={{ color: 'red' }}
26        onClick={() => showDeleteConfirm(record)}
27      >
28        删除
29      </Button>
30    ]
31  }
32 ],
33
34 return (
35   <PageContainer>
36     <ProTable
37       actionRef={actionRef}
38       headerTitle={'模式列表'}
39       rowKey="schema_title"
40       search={{
41         labelWidth: 120,
42       }}
43       columns={columns}
44       request={schemaList}
45       pagination={{
46         showSizeChanger: true,
47         pageSizeOptions: [10, 20, 50, 100],
48       }}
49       toolBarRender={() => [
50         <Button
51           type="primary"
52           key="primary"
53           onClick={() => {
```

行李托运平台 V1.0

```
1             handleModalVisible(true);
2         }}
3     >
4         <PlusOutlined /> 新建
5     </Button>,
6   ]}
7 />
8 <ModalForm
9   title={currentRow ? '编辑' : '新建'}
10  width="600px"
11  visible={modalVisible}
12  onVisibleChange={handleModalVisible}
13  modalProps={{
14    destroyOnClose: true,
15  }}
16  initialValues={{
17    ...currentRow,
18  }}
19  onFinish={async (value) => {
20    const success = await handleUpdate(value, currentRow);
21    if (success) {
22      handleModalVisible(false);
23      setCurrentRow(undefined);
24      if (actionRef.current) {
25        actionRef.current.reload();
26      }
27    }
28  }}
29 >
30   {currentRow ?
31     <>
32       <ProFormText
33         fieldProps={{ maxLength: 20 }}
34         width="md"
35         name="schemaName"
36         initialValue={currentRow}
37         label="需要修改的模式名称"
38         disabled
39       />
40       <ProFormText
41         rules={[
42           {
43             required: true,
44             message: '模式名称为必填项',
45           },
46         ]}
47         fieldProps={{ maxLength: 20 }}
48         width="md"
49         name="newSchemaName"
50         label="新模式名称"
51       />
52     </>
53   : <ProFormText
```

行李托运平台 V1.0

```
1         rules=[{
2             {
3                 required: true,
4                 message: '模式名称为必填项',
5             },
6         ]}
7         fieldProps={{ maxLength: 20 }}
8         width="md"
9         name="schemaName"
10        label="模式名称"
11    />}
12
13    </ModalForm>
14    </PageContainer>
15  );
16  };
17
18  export default TenantList;
19
20
21  ```
22
23
24
```