

操作说明

Part5. 沙盒 APP 的开发与启动

目 录

操作说明.....	1
Part5. 沙盒 APP 的开发与启动.....	1
第一步：数据消费方将沙盒 APP 分配给开发者与生产者.....	4
第二步：开发者在线配置/开发与“保存”沙盒 APP.....	7
1.1 配置可用资源及虚拟表.....	8
1.1.1 查看开放资源及策略.....	8
设备层访问策略.....	9
沙盒 APP 层使用策略.....	9
1) 落地策略.....	9
2) 字段读取策略.....	10
3) 使用前的加工处理策略.....	10
1.1.2 查看元数据与开放资源详情.....	11
1.2 查看虚拟表结构.....	12
1.3 通用配置.....	13
1.4 请求连接器配置.....	13
1.4.1 例 1: mysql 请求连接器.....	13
1.4.2 例 2: kafka 请求连接器.....	15
1.5 落地连接器配置.....	16
1.6 数据加工处理的逻辑.....	16
1.6.1 对于“数据加工处理的逻辑”，系统要进行几个方面的校验.....	17
1.6.2 sql 语句的指引.....	20
1) 直接或间接使用开放资源.....	20
2) 子查询中的嵌套查询.....	20
1.6.3 select 需要满足的原则.....	21
1.6.4 对于 OpenAPI 类型开放资源.....	23
第三步：开发者“提交”沙盒 APP.....	23

第四步：开发者“启动”已提交校验通过的沙盒 APP	25
第五步：开发者“发布”沙盒 APP 到生产环境	25
第六步：开发者“停止运行”已启动的沙盒 APP	26
第七步：生产者修改并“保存”沙盒 APP 的运行环境配置	26
第八步：生产者在生产环境中“提交”沙盒 APP	27
第九步：生产者在生产环境“启动”沙盒 APP	28
第十步：生产者“停止运行”已启动的沙盒 APP	29
第十一步：数据提供方可查看资源的使用情况（DPE）	30

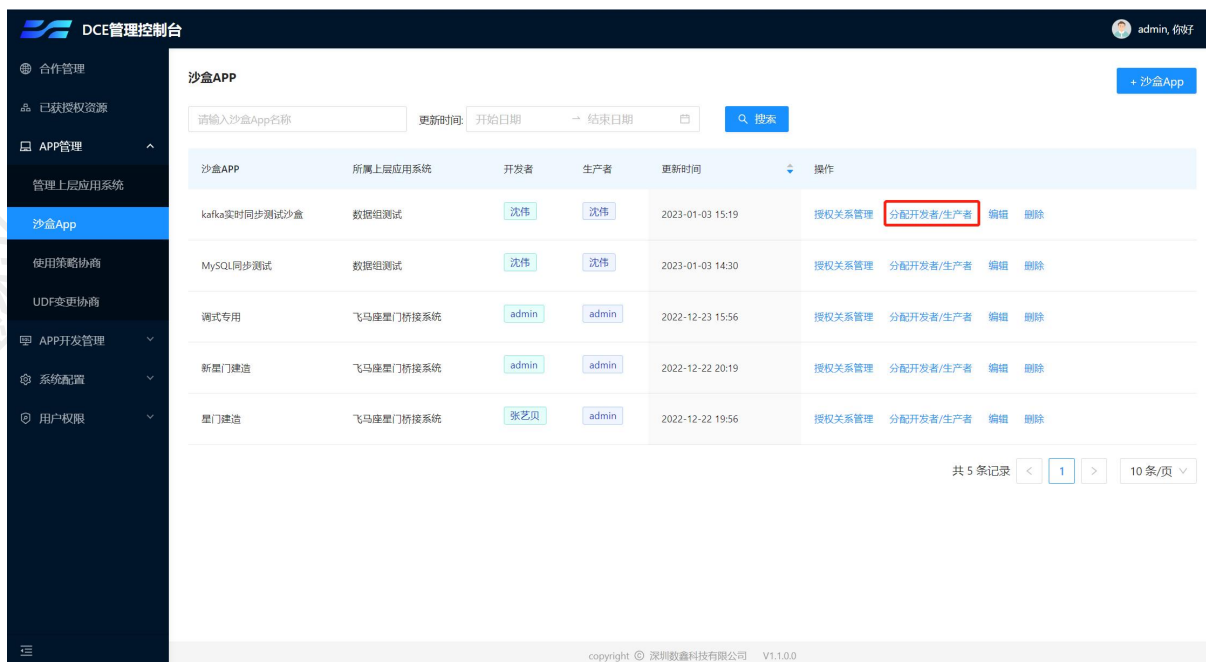
前置：数据消费方与数据提供方，已完成沙盒 APP 层使用策略的在线协商，详见 Part 4

本部分的全部操作，都由数据消费方，登录 [DCE 管理控制台](#) 进行

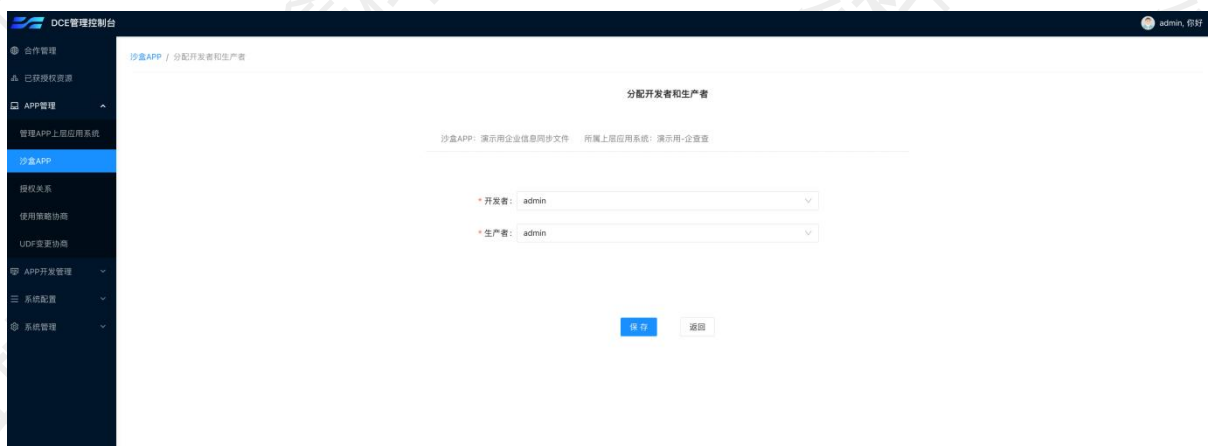
第一步：数据消费方将沙盒 APP 分配给开发者与生产者

操作步骤分解：

1. 数据消费方的 APP 管理者，登录 [DCE 管理控制台](#)，通过侧导航访问：APP 管理>沙盒 APP



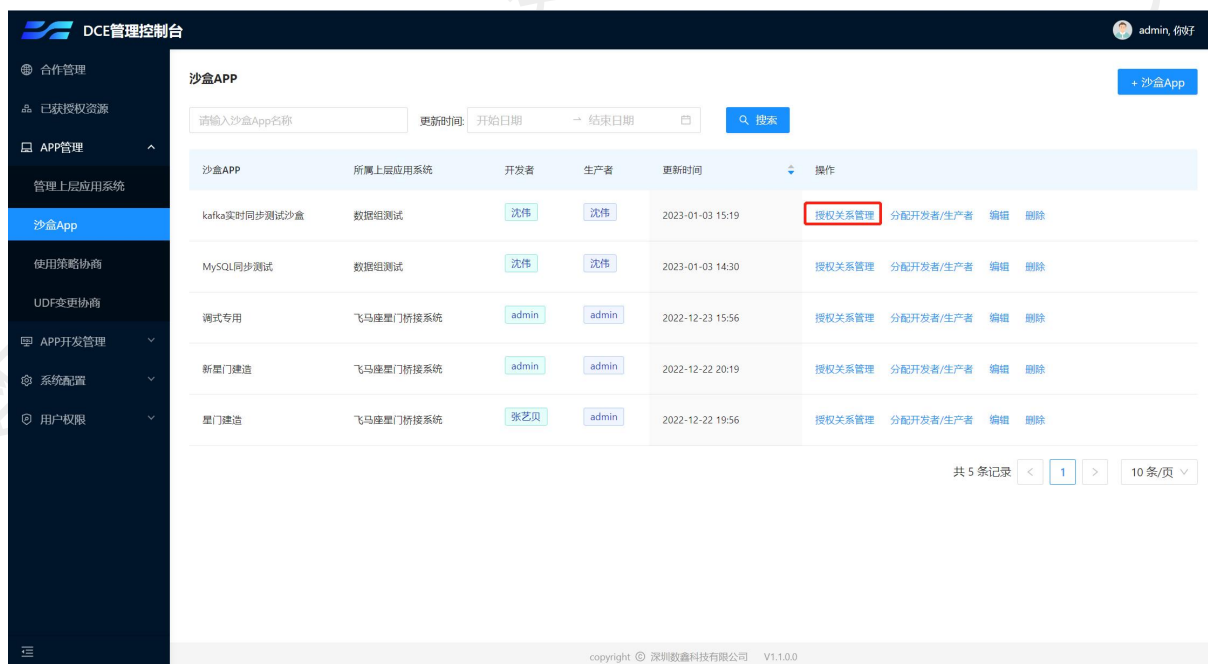
找到需要分配给开发者和生产者的沙盒 APP，点击该沙盒 APP“操作”列的【分配开发者/生产者】，打开“分配开发者和生产者”界面，如下图



在“分配开发者和生产者”界面中，选择相应的“开发者”和“生产者”账号，并点击界面下方的【保存】，即完成该沙盒 APP 的分配操作。

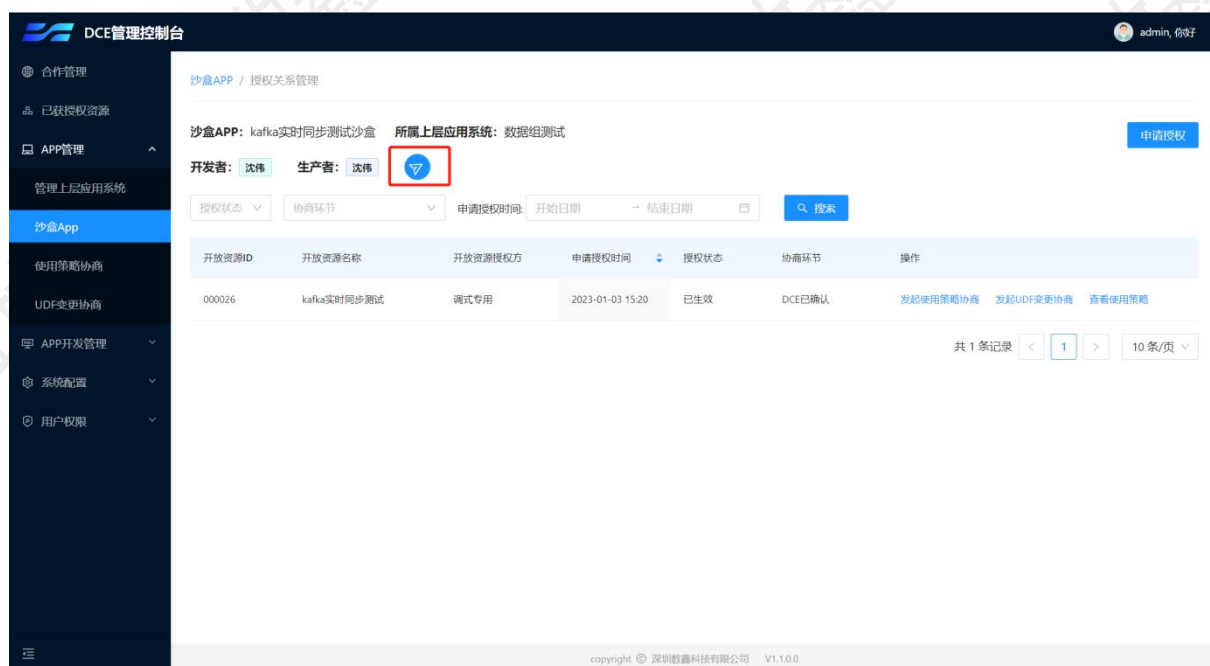
2. 为了及时通知开发者与生产者跟进后续的工作，数据消费方的管理员，可以在 DCE 管理控制台中，发送相应的通知邮件给对应的开发者与生产者，让开发人员了解该沙盒 APP 需使用的开放资源的设备层访问策略以及沙盒 APP 使用策略的详情，该内容，是开发人员进行沙盒 APP 在线开发的重要参照信息，操作如下：

DCE 管理控制台>APP 管理>沙盒 APP：

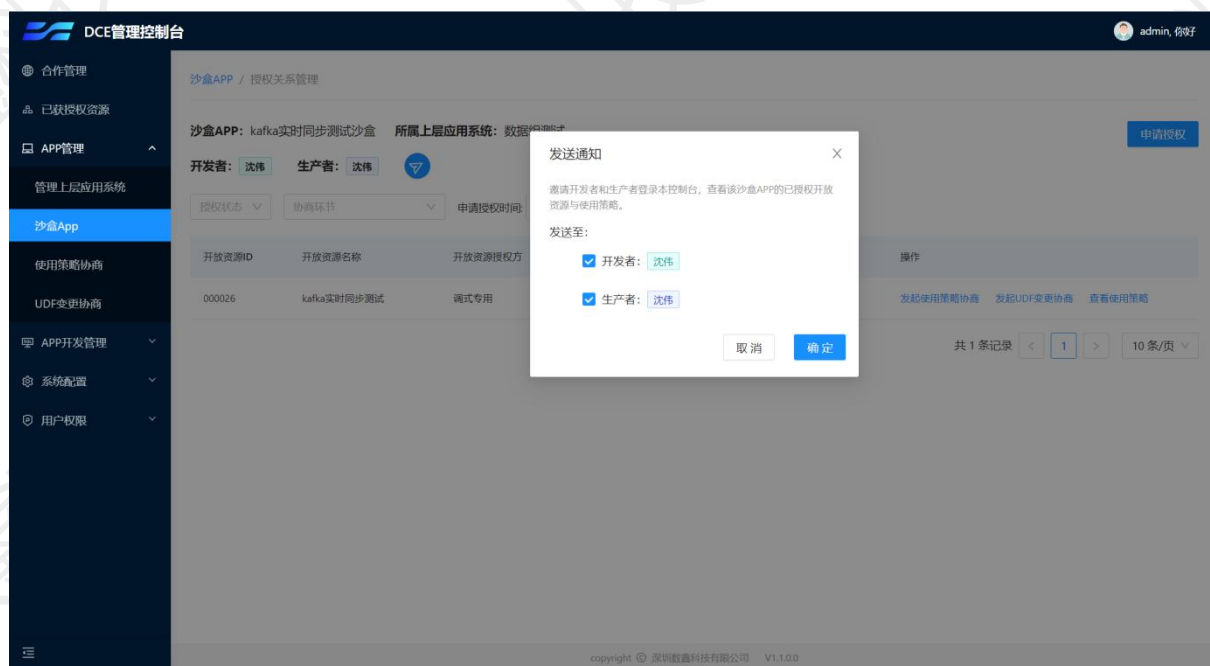


在沙盒 APP 记录行的“操作”，点击【授权关系管理】，进入该沙盒 APP 的授权关系管理页，

如下图



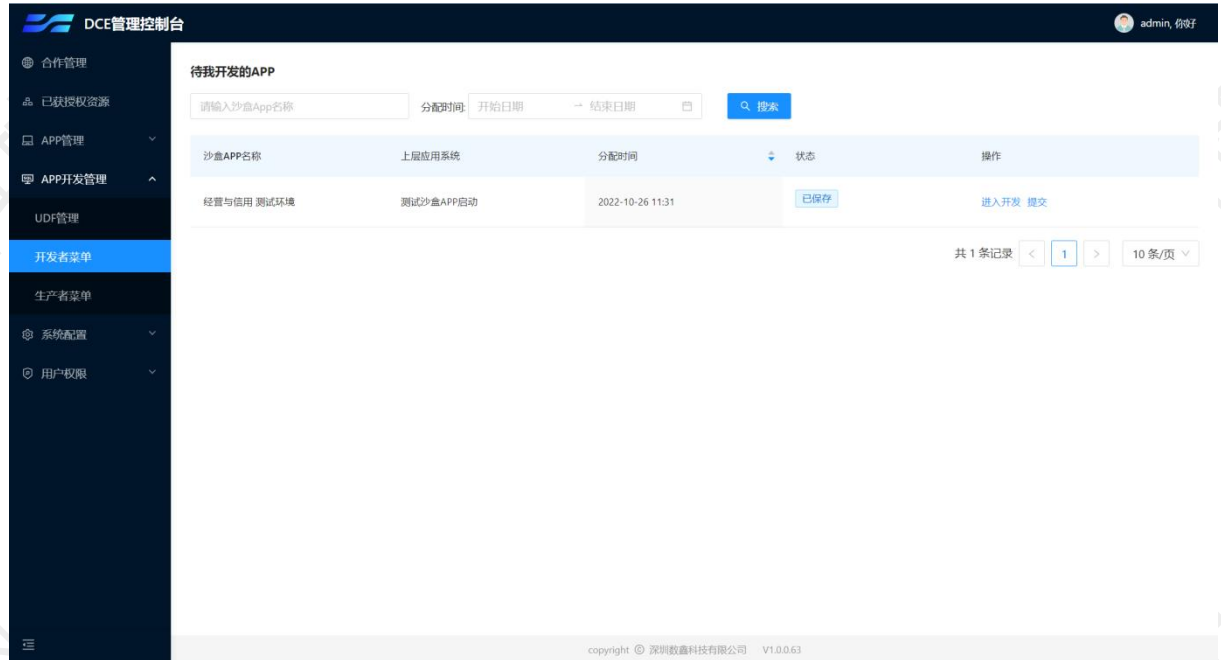
点击页面中的【发送通知】按钮，当前即出现“发送通知”的弹窗



勾选弹窗内的“开发者”或“生产者”，点击【确定】，平台即发送通知邮件至相应开发者/生产者账号的邮箱中。

第二步：开发者在线配置/开发与“保存”沙盒 APP

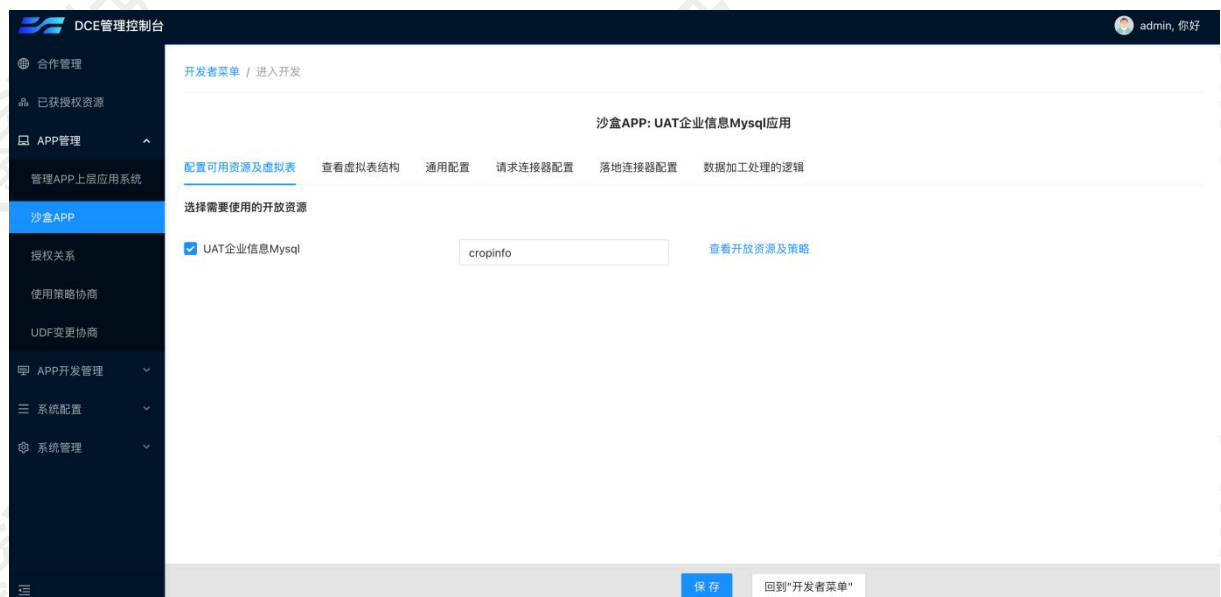
开发者登录 DCE 管理控制台 > APP 开发管理>开发者菜单：当前开发者可见在第一步中，已分配给自己的沙盒 APP 列表：



点击沙盒 APP“操作”列的 【进入开发】，进入该沙盒 APP 的低代码在线开发界面。

一个沙盒 APP 的开发，分成六个界面，开发者在开发过程中，可以来回切换进入这几个界面：

配置可用资源及虚拟表、查看虚拟表结构、通用配置、请求连接器配置、落地连接器配置、数据加工处理的逻辑



1.1 配置可用资源及虚拟表

在本界面中，呈现了当前沙盒 APP 可以使用的开放资源，一个沙盒 APP 可以使用多个开放资源，开发人员在本界面中，可进行如下操作：

- 勾选需要使用的开放资源

开发人员可以根据实际业务需要，从当前沙盒 APP 已被授权使用的开放资源中，勾选需要使用的开放资源

- 为开放资源输入虚拟表名称

并为当前已勾选的开放资源设定对应的虚拟表名称，虚拟表的名称，仅可以由英文大小写字母组成。



1.1.1 查看开放资源及策略

点击开放资源虚拟表输入框右侧的【查看开放资源及策略】，当前立即打开新窗口，显示当前开放资源对应当前沙盒 APP 的访问策略+沙盒 APP 使用策略界面，该界面中的全部内容，都是沙盒 APP 访问被该沙盒 APP 被授权使用的开放资源时的鉴权内容，任意一条不符合策略，都会导致该沙盒 APP 无法访问/使用该数据。

DCE管理控制台

查看开放资源

授权方: UAT专用公司
授权方DPE设备号: 0f1e6928c43d

开放资源名称: UAT企业信息Mysql
开放资源ID: 000475

访问策略

授权关系时间限制: 授权关系长期有效

限制开放资源访问的总次数: 关

DCE安全级别: 可信+

需要上报数据使用日志信息: 无

沙盒APP使用策略

授权使用开放资源的沙盒APP: UAT企业信息Mysql应用
沙盒APP服务的上层应用系统: UAT企业上层应用

落地策略

读取数据后是否落地: 是, 并承诺不转存转发

数据库类型: JDBC
接受定期删除: 每 1小时 删除一次

限定落地存储的操作次数: 1 次

对落地存储后的数据继续进行加工处理: 打开

接受落地存储的数据携带原始使用策略: 关闭

字段读取策略

抽取/推送模式: 抽取

申请使用的字段: phone corp founding_time uniform_credit_code an_enterprise_name

使用前的加工处理策略

UDF函数: ReplacePhNum

选择的字段: phone

使用数据的事件

无

本界面内容, 是开发人员进行当前沙盒 APP 开发的重要参考, 在开发过程中需要详细参考。

在本界面, 可见当前开放资源对于当前 DCE 设备层的访问策略&当前沙盒 APP 对当前开放资源的使用策略, 如:

设备层访问策略

- 1) 开放资源的有效授权访问时间
- 2) 开放资源的访问次数 (包含 API 调用次数、数据读取次数等);

沙盒 APP 层使用策略

- 1) 落地策略

深圳数鑫科技有限公司

9

如选择“是，承诺不转存转发”，则表示数据需要落地，将需要做如下设置：

- 数据库类型：可选择落地的数据库类型，可以选择多种数据库类型，约定数据可落地到哪几个指定类型的数据库；
- 选择数据库类型后，如果打开了“接受定期删除”的按钮，则需要填写每隔多长时间删除一次，约定落地到指定数据库中的删除时间限制。
- 限定落地存储的操作次数：如果打开了该设置按钮，则需要填写次数，可根据实际读取数据的需要，填写合理的落地次数，该次数，表示该沙盒 APP 每次读取该开放资源的数据时，所使用的 Insert 语句的数量上限。

落地策略

读取数据后是否落地：	是，并承诺不转存转发
数据库类型：	JDBC
	接受定期删除： 每 24小时 删除一次
限定落地存储的操作次数：	9999 次

2) 字段读取策略

目前仅支持“拉取”方式获得数据，

申请使用的字段：数据消费方为该沙盒 APP 所授权使用的开放资源，申请使用的字段

字段读取策略

拉取/推送模式

拉取

申请使用的字段

startdate

starttime

color

enddate

endtime

3) 使用前的加工处理策略

- 如果是一个 scala 文件，则点击 UDF 函数的链接，可以打开新窗口，查看到该 UDF 函数的具体内容，如下图所示：

UDF 函数可以定义对选定的字段的处理方法，如下图的 UDF 函数的处理方法，是将“手机

号码”字段的中间五位数字替换为*****

ReplacePhNum

类型：用户自定义标量函数

函数名：ReplacePhNum

函数处理逻辑说明：将联系方式中的五位替换成*

函数返回值说明：脱敏后的字符串

更新时间：2022-08-23 11:37:11

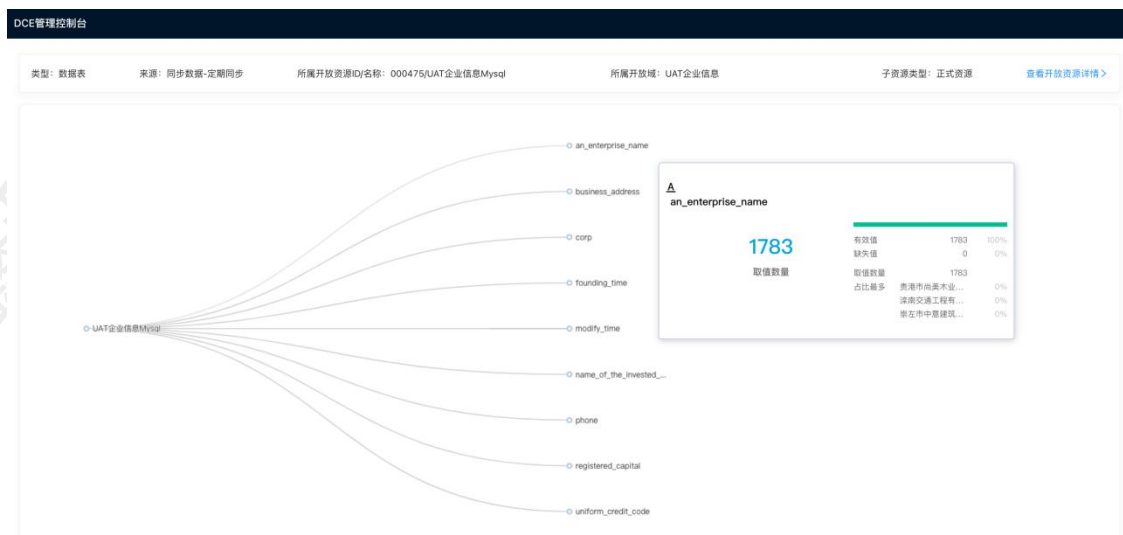
UDF文件：

```
1 package com.dce.customer.udf
2
3 import org.apache.flink.table.functions.ScalarFunction
4
5 class ReplacePhNum extends ScalarFunction {
6     def eval(str: String): String = {
7         str match {
8             case null => "*****"
9             //替换中间5位为*
10            case x => x.replace(x.drop(3).dropRight(3), "*****")
11        }
12    }
13 }
```

- 如果是一个 ZIP 包，则可点击【下载】，将 UDF 函数文件包下载到本地查看详情。

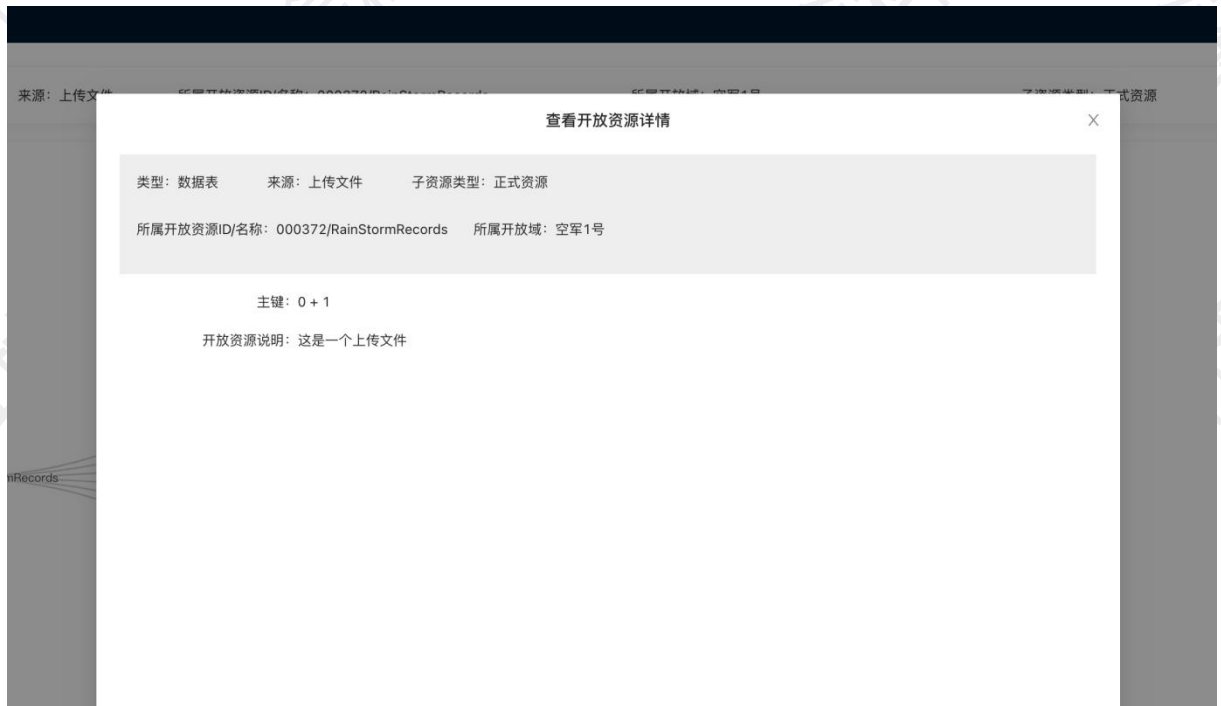
1.1.2 查看元数据与开放资源详情

点击“查看开放资源”界面右上方的【查看元数据】，可查看该开放资源的元数据结构，鼠标移动到逐个字段上，可以看到该字段下的数据取值分布情况



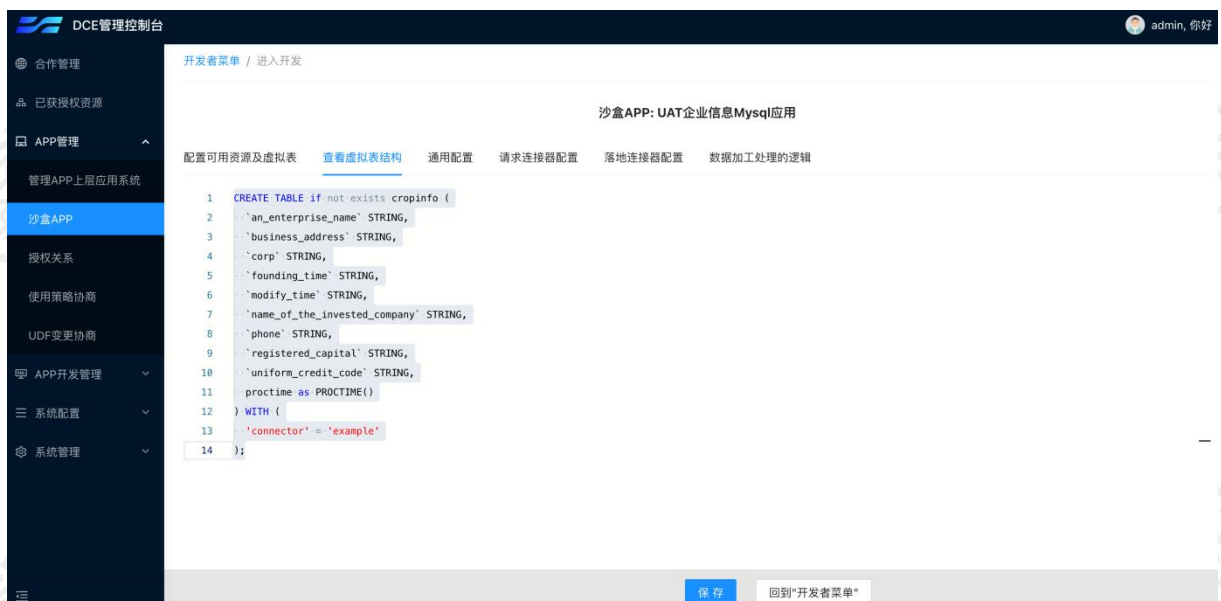
点击“查看元数据”界面右上方的【查看开放资源详情】，可以查看到该开放资源的详细信息，

如下图：



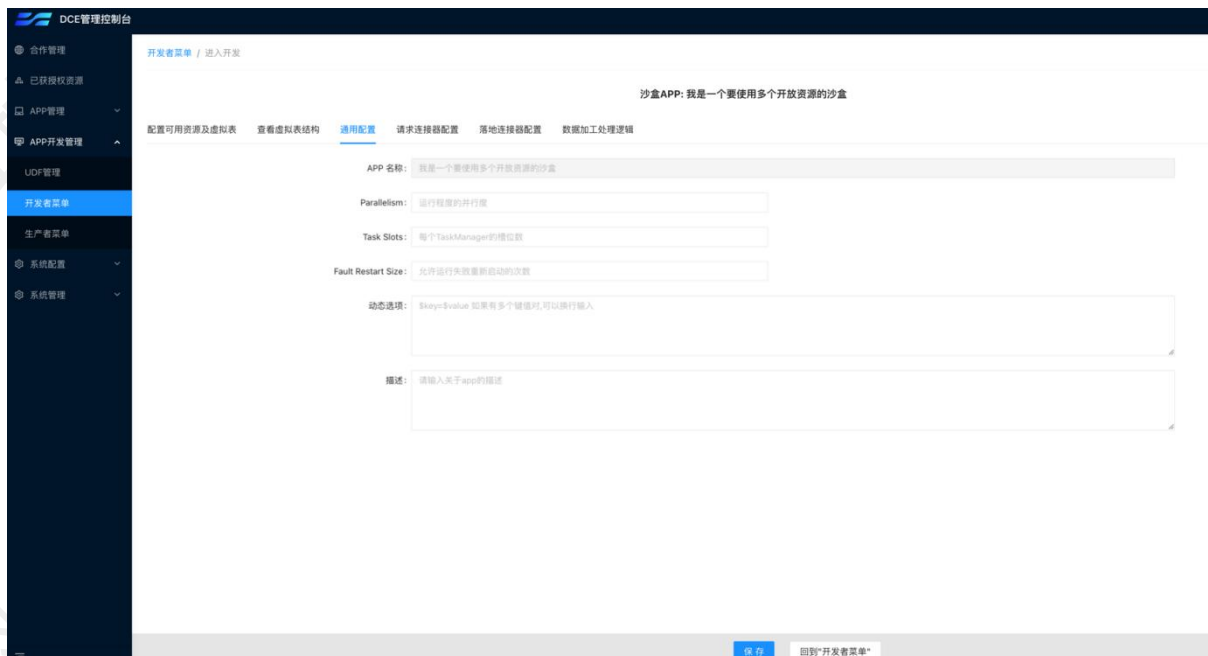
1.2 查看虚拟表结构

在“配置可用资源及虚拟表”界面中，完成开放资源的勾选和虚拟表名称的配置后，点击“查看虚拟表结构”进入本界面，即可查看到当前沙盒 APP 已选的开放资源虚拟表的名称以及相应的表结构



1.3 通用配置

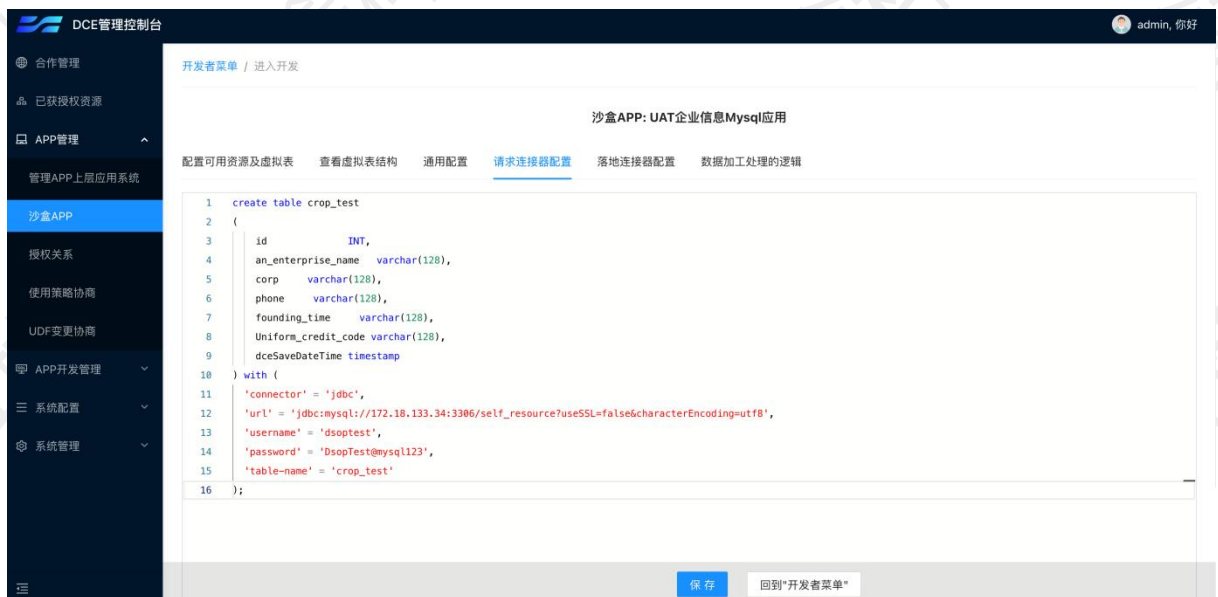
通用配置的配置项，若不配置，则系统会自动采用 Flink 集群中对应的配置参数



1.4 请求连接器配置

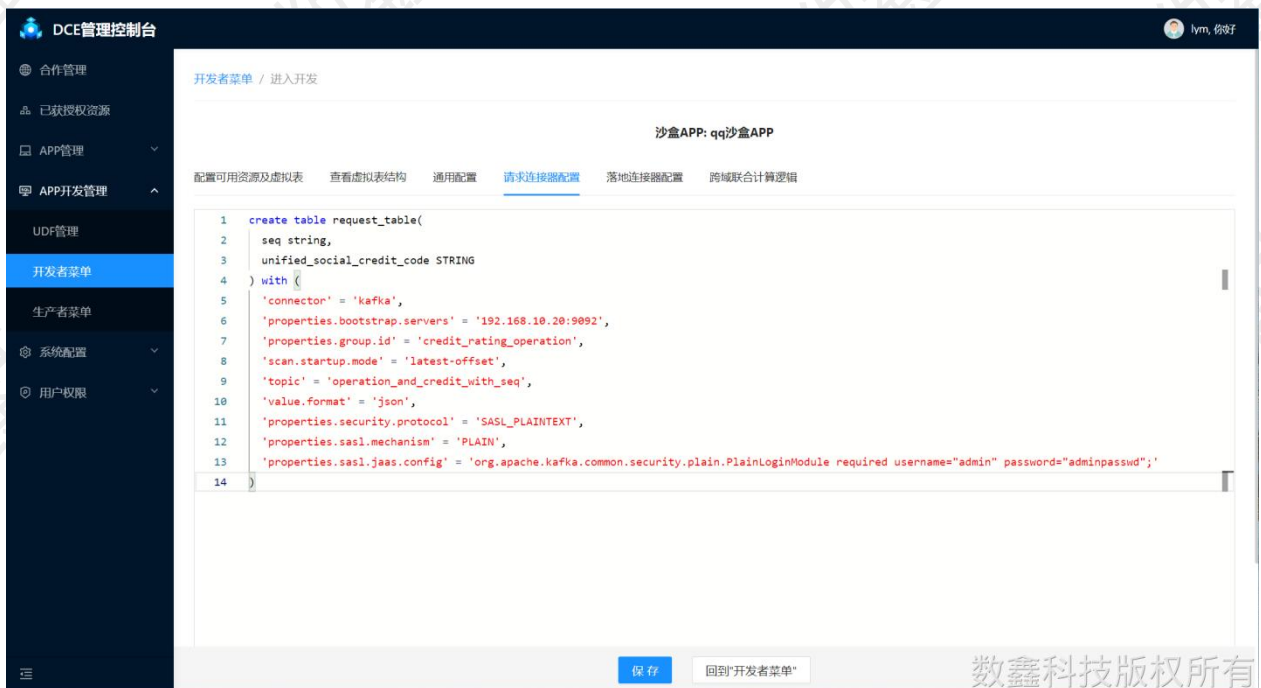
若沙盒 APP 需要用消费方域内数据联合 JOIN 来自 DPE 的域外数据，那么就需要配置相应的 Flink SQL Source Connector 以及对应的虚拟表结构。沙盒 APP 请求连接器支持 Flink 官方 SQL Connector，比如 MySQL、Kafka 等连接器。

1.4.1 例 1：mysql 请求连接器



```
create table crop_test
(
id INT,
an_enterprise_name varchar(128),
corp varchar(128),
phone varchar(128),
founding_time varchar(128),
Uniform_credit_code varchar(128),
dceSaveDateTime timestamp
) with (
'connector' = 'jdbc',
'url' =
'jdbc:mysql://172.18.133.34:3306/self_resource?useSSL=false&characterEncoding=utf8',
'username' = 'dsopetest',
'password' = 'DsopTest@mysql123',
'table-name' = 'crop_test'
);
```

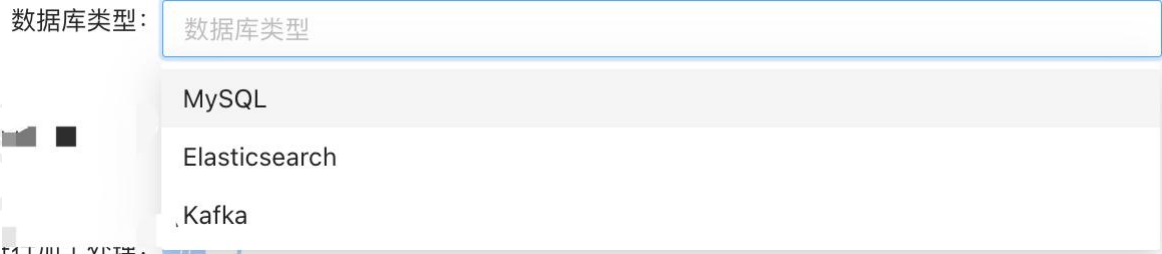
1.4.2 例 2: kafka 请求连接器



```
create table request_table(  
  seq string,  
  unified_social_credit_code STRING  
) with (  
  'connector' = 'kafka',  
  'properties.bootstrap.servers' = '192.168.10.20:9092',  
  'properties.group.id' = 'credit_rating_operation',  
  'scan.startup.mode' = 'latest-offset',  
  'topic' = 'operation_and_credit_with_seq',  
  'value.format' = 'json',  
  'properties.security.protocol' = 'SASL_PLAINTEXT',  
  'properties.sasl.mechanism' = 'PLAIN',  
  'properties.sasl.jaas.config' =  
'org.apache.kafka.common.security.plain.PlainLoginModule required  
username="admin" password="adminpasswd";'  
)
```

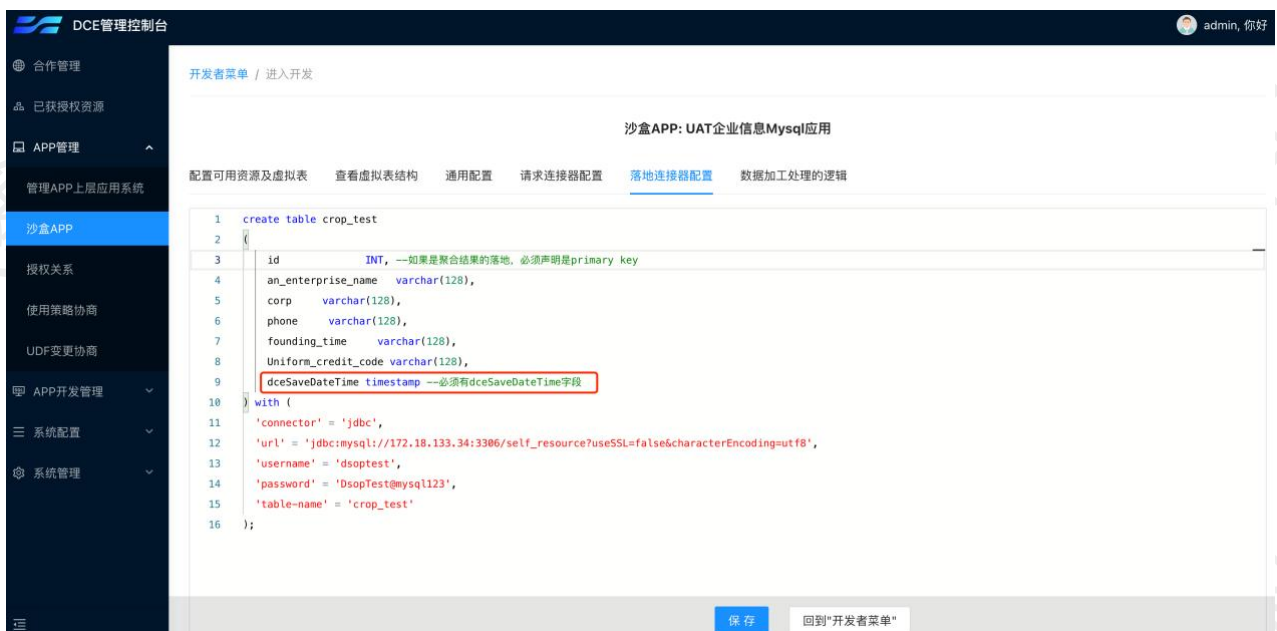
1.5 落地连接器配置

能支持落地连接器的类型，需要与应用策略协商时允许的落地数据库对应，目前支持的落地数据库类型有三种，如下图所示：



注意使用开放资源如要落地必须在落地连接器中包含 dceSaveDateTime。

Mysql 落地连接器示例如图



1.6 数据加工处理的逻辑



此栏目必填，本栏目的 sql，需要严格遵从当前沙盒 APP 的使用策略约定的条款内容，从以下几个方面，进行在线校验，如不满足任意一条约定，该沙盒 APP 都将无法成功运行：

1.6.1 对于“数据加工处理的逻辑”，系统要进行几个方面的校验

1) 至少选择一个开放资源

并为已选择的开放资源，填写其虚拟表的名称，如下图：



2) 对落地策略的校验

insert 语句与开放资源有关联，与落地策略中的落地次数要相符

3) 对字段读取策略的校验

使用的字段，必须符合沙盒 APP 使用策略中“字段读取策略”的约定

例如：

a.使用了开放资源，使用字段为 msg，没有使用数组型字段 rows，符合使用策略

```
1  INSERT INTO
2  |   fire_trend(countryName, cityName)
3  SELECT
4  |   c.countryName,
5  |   CONCAT(c.cityName, m.msg) AS cityName
6  FROM
7  |   country_info c
8  |   JOIN (
9  |       select
10 |         `date`,
11 |         msg
12 |       FROM
13 |         fireList FOR SYSTEM_TIME AS OF proctime
14 |     ) AS m ON c.`dateTime` = m.`date`
```

使用了开放资源， rows 为数组 t 后面括号内容可以不写，注意 tmp_result 中的字段如与 t 中的字段重名要加以区分；

在使用开放资源创建 view 时 rows 数组型字段不允许使用别名， 其他字段要在 select 时符合策略

```
1  CREATE VIEW tmp_not_allow AS
2  SELECT
3  |   m.myrows,
4  |   m.`date`
5  FROM
6  |   country_info c
7  |   JOIN (
8  |       select
9  |         `date`,
10 |         `rows` AS myrows
11 |       FROM
12 |         fireList FOR SYSTEM_TIME AS OF proctime
13 |     ) AS m ON c.`dateTime` = m.`date`
```

```
1 CREATE VIEW tmp_not_allow1 AS
2 SELECT
3     m.`rows` AS myrows,
4     m.`date`
5 FROM
6     country_info c
7 JOIN (
8     select
9         `date`,
10        `rows`
11 FROM
12     fireList FOR SYSTEM_TIME AS OF proctime
13 ) AS m ON c.`dateTime` = m.`date`
```

b.其他情况使用开放资源的数组型字段 rows 时，必须使用 UNNEST 方法，并且数组内字段要符合策略

```
1 CREATE VIEW tmp_result1 AS
2 SELECT
3     `rows`,
4     `date`
5 FROM
6     tmp_result
```

c.使用的字段与使用策略不符

```
1 INSERT INTO
2     fire_trend(countryName, cityName, energyEstimate, lon, lat)
3 SELECT
4     CONCAT(province, countryName),
5     cityName,
6     energyEstimate,
7     CAST(CEIL(t.lon) AS INT),
8     CAST(CEIL(t.lat) AS INT)
9 FROM
10     tmp_result,
11     UNNEST(`rows`) AS t
```

4) 对使用前的加工处理策略的校验

使用的字段与策略相符，但字段处理函数不相符


```
1  INSERT INTO
2      fire_trend(countryName, cityName, energyEstimate, lon, lat)
3  SELECT
4      countryName,
5      cityName,
6      energyEstimate,
7      CAST(t.lon AS INT),
8      CAST(t.lat AS INT)
9  FROM
10     tmp_result,
11     UNNEST(`rows`) AS t
```

1.6.2 sql 语句的指引

1) 直接或间接使用开放资源

假设开放资源对应的虚拟表名为 fireList。

开放资源的 DDL 语句由框架生成并执行, 加入了 proctime as PROCTIME() 产生处理时间列;

对于嵌套字段的 select 示例, 有以下 user_info 表

```
CREATE TABLE print_table (
    user ROW<
        name STRING,
        age INT,
        books ARRAY<ROW<bookName STRING, editor STRING>>,
        disc MAP<STRING, MAP<STRING, INT>>>
    ) WITH (
        'connector' = 'print'
    );
```

ROW 类`user`.`name`, ARRAY 类`user`.`books`[1] 表示第一本书,MAP 类`user`.`disc`['key_out']['key_in']

2) 子查询中的嵌套查询

嵌套结构加了飘号`""` (Esc 按键)后子查询外部不能选到该嵌套字段, 可用 AS 方法修正如


```
SELECT u.user_id, u.item_id, u.cat_id, u.action, u.province, p.province_id, u.proctime,
p.province_name, p.age
FROM
user_behavior AS u
Left JOIN
(SELECT province_name, province_id, `user`.`age` AS age FROM dim_province FOR
SYSTEM_TIME AS OF proctime where `user`.`age` > 29) AS p
ON u.province > p.province_id and u.action = p.action
```

1.6.3 select 需要满足的原则

- 1.不允许使用*: select * from fireList;
- 2.不允许使用存在于字段列表但未授权的字段:select notAllowField from fireList;
- 3.如允许使用的字段有 flink 函数约束, 必须在 select 时立即使用:select SUM(lon) from fireList;
- 4.如允许使用的字段有 UDF 函数约束(假设函数名 Len), 必须在 select 时立即使用:select Len(cityName) from fireList;
- 5.查询筛选:对开放资源的直接查询只能使用 ODRL 中的关系及逻辑运算(= <> > >= < <= AND), 要做进一步筛选可在查询之外处理
- 6.CREATE VIEW 时 openAPI 中数组型字段不允许使用别名, 在使用 VIEW 中的数组型字段时, 必须使用 UNNEST 方法, 并且数组内字段要符合策略

7.1-6 条适用于子查询; 假设有两张虚拟表 user_behavior, dim_province, sql 语句 join 如下:

```
1  SELECT
2      u.user_id,
3      u.item_id,
4      u.cat_id,
5      u.action,
6      p.province_name
7  FROM
8      user_behavior u
9      JOIN dim_province p ON u.province = p.province_id
10     and u.province > 15
```

等价于

```

1  SELECT
2      u.user_id,
3      u.item_id,
4      u.cat_id,
5      u.action,
6      p.province_name
7  FROM
8      (
9          select
10             *
11          From
12             user_behavior
13      ) u
14  JOIN (
15      select
16          *
17      From
18          dim_province
19      ) p ON u.province = p.province_id
20      and u.province > 15

```

两者优化后执行图一致，不符合第 1 条使用规则；修改如下并把过滤条件下推：

```

1  SELECT
2      u.user_id,
3      u.item_id,
4      u.cat_id,
5      u.action,
6      p.province_name
7  FROM
8      (
9          select
10             user_id,
11             item_id,
12             cat_id,
13             action,
14             province
15          from
16             user_behavior
17          where
18             province > 15
19      ) u
20  JOIN (
21      select
22          province_name,
23          province_id
24      from
25          dim_province
26      ) p ON u.province = p.province_id

```

1.6.4 对于 OpenAPI 类型开放资源

只能用于 lookup join, 即 Temporal Join; OpenAPI 接口的请求参数在 SQL 的 on 中用=传递, 其中如有分页参数则为每一页创建视图, 分页参数放在 on 条件中传递, 如:

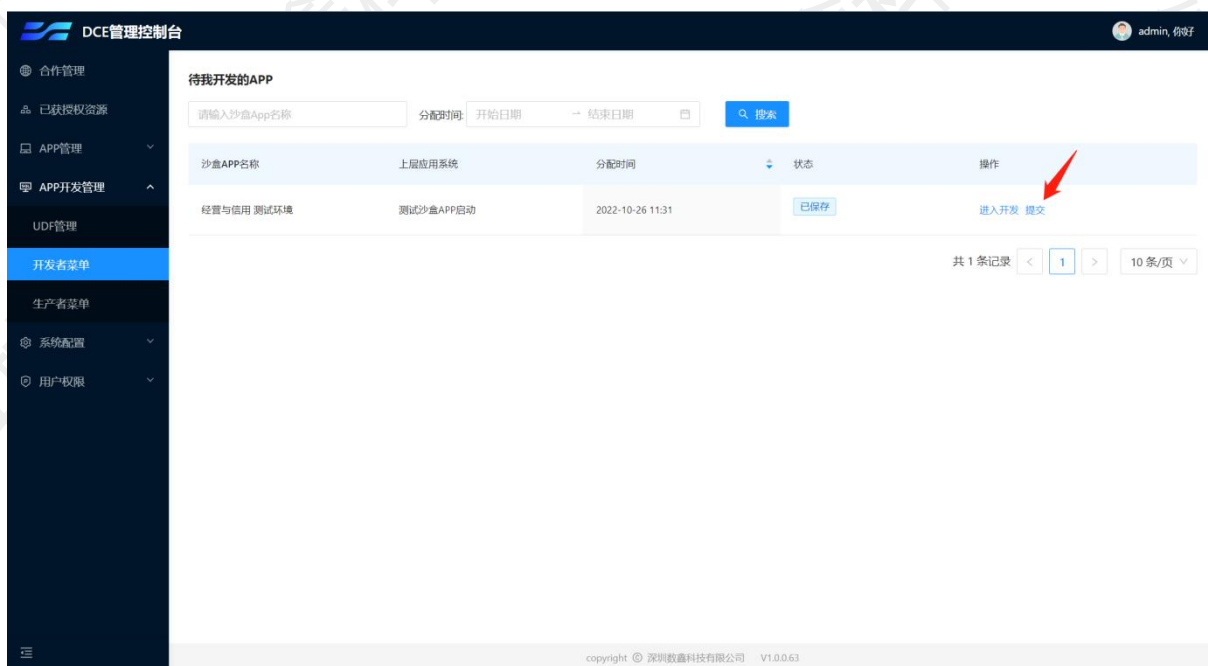
```
1  SELECT
2      c.countryName,
3      c.cityName,
4      r.code,
5      r.msg
6  FROM
7      country_info c
8  JOIN (
9      select
10         `date`,
11         code,
12         msg
13     FROM
14         fireList FOR SYSTEM_TIME AS OF proctime
15 ) AS r ON c.`dateTime` = r.`date`
```

完成以上 6 个部分的配置和填写, 且每个部分都【校验】通过之后, 点击界面底部的【保存】, 可将当前 6 个部分的内容一并保存。



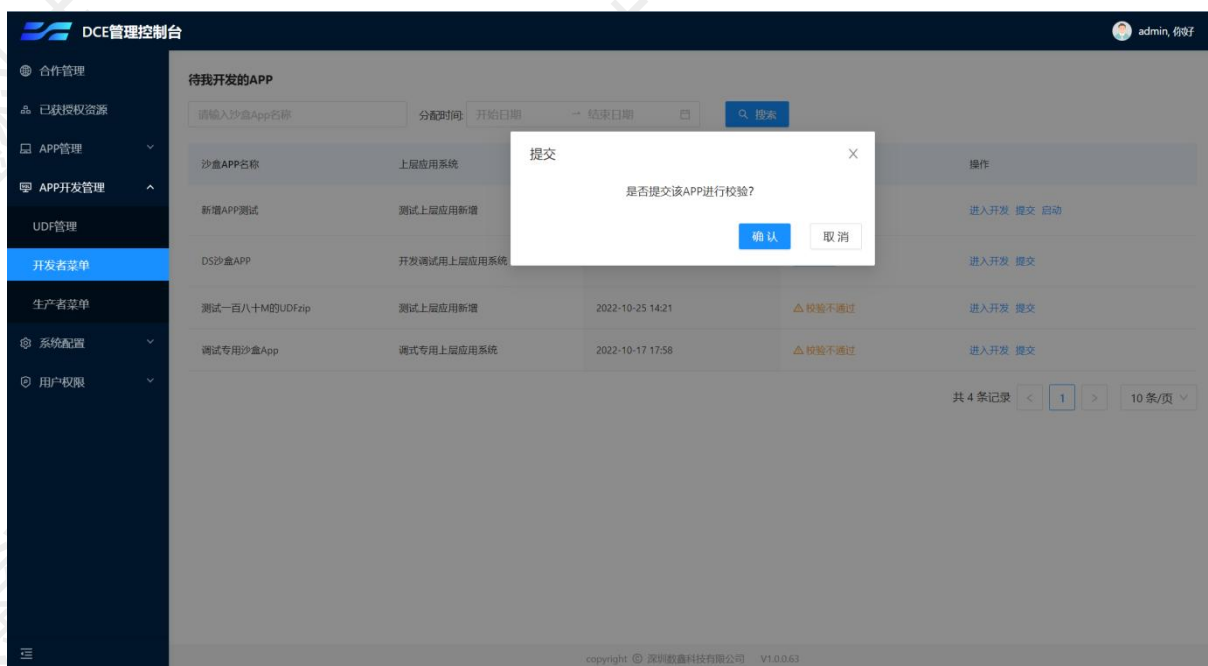
第三步：开发者“提交”沙盒 APP

在“开发者菜单”, 可见状态=“已保存”的沙盒 APP



点击【提交】

当前显示二次确认窗口，如下图，点击【确认】，即将沙盒 APP 提交到开发环境中：



确认提交到开发环境时，系统会对 sql 语句是否符合使用策略进行校验，（在提交时，不对使用策略中“使用数据的条件”进行校验）。校验不通过会有提示，校验通过出现启动按钮：

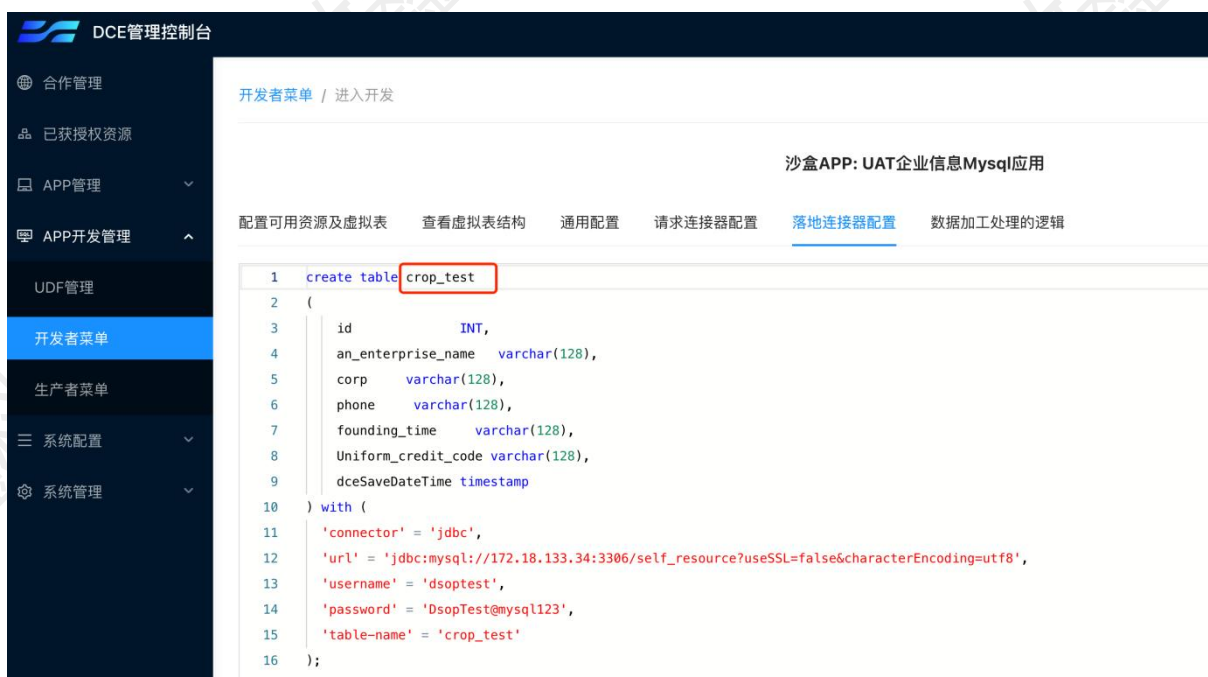
如果该沙盒 app 相关的开放资源发起了新的使用策略协商或新的 UDF 变更协商，都需要重新提交，此时在“开发者菜单”，该沙盒 APP 状态列，将出现图标 **NEW**，提示开发者注意有新的使用策略，需要重新配置和提交该沙盒 APP。

第四步：开发者“启动”已提交校验通过的沙盒 APP

在“开发者菜单”中，点击【启动】已校验通过的沙盒 APP，如果启动成功，则沙盒 APP 将在开发环境中运行读取该沙盒 APP 可用的开放资源所对应的试用资源的数据。

开发人员可以在本沙盒 APP 配置的落地连接器对应的落地数据表中，验证沙盒 APP 在开发环境运行后，是否获取到了试用资源的数据。

如：在该例中，落地连接器配置中，落地表名为 crop_test



开发者可以使用数据库管理工具 Navicat，查看落地表“crop_test”，dceSaveTime 处于【启动】运行的时间区间内，如已获取到数据，则表示沙盒 APP 已在开发环境运行并获取到了试用资源的数据

第五步：开发者“发布”沙盒 APP 到生产环境

开发者可以将已成功运行的沙盒 APP，发布到生产环境中



沙盒 APP 被发布到生产环境后，在开发者菜单中，该沙盒 APP 的状态栏，将出现“已发布”的状态图标。此时，在该沙盒 APP 的“生产者”的菜单中，将可以看到已被发布到生产环境中的沙盒 APP。

第六步：开发者“停止运行”已启动的沙盒 APP



- 在沙盒 APP 运行时，开发者可以点击【停止运行】，中止沙盒 APP 的运行；
- 沙盒 APP 在执行完任务后，也会自动“停止运行”，这都是正常的停止状态。
- 运行中的沙盒 APP，也会发生异常停止，开发者可以点击“已停止”状态前面的△图标，

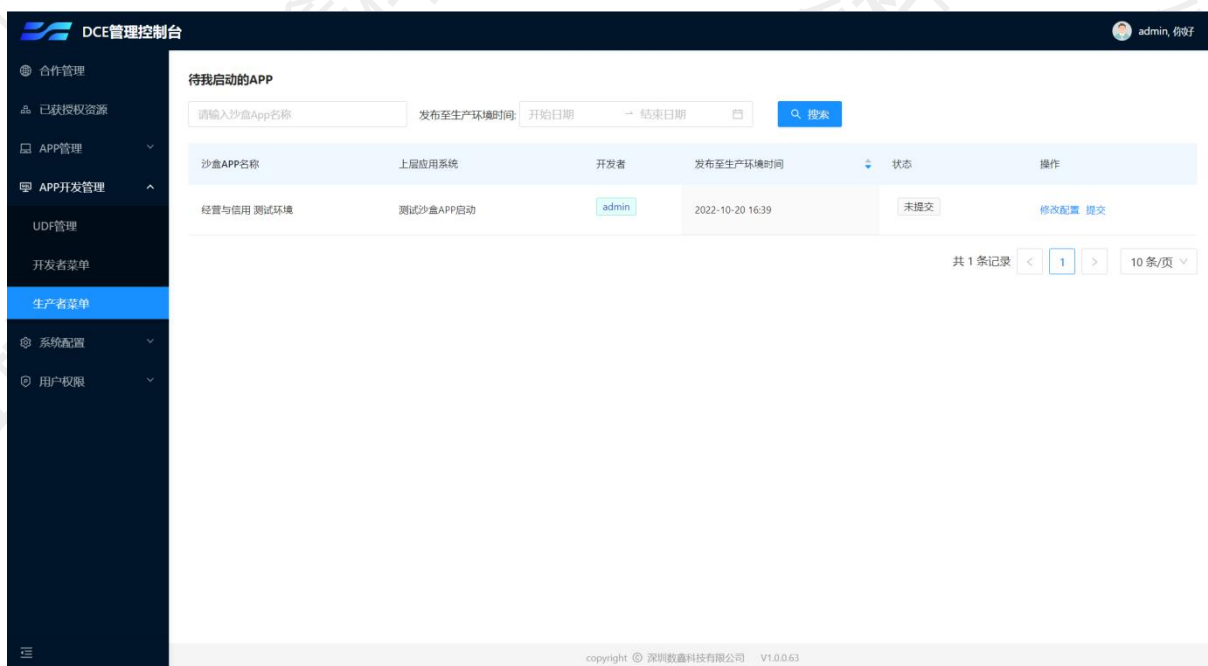
查看系统异常停止的原因。



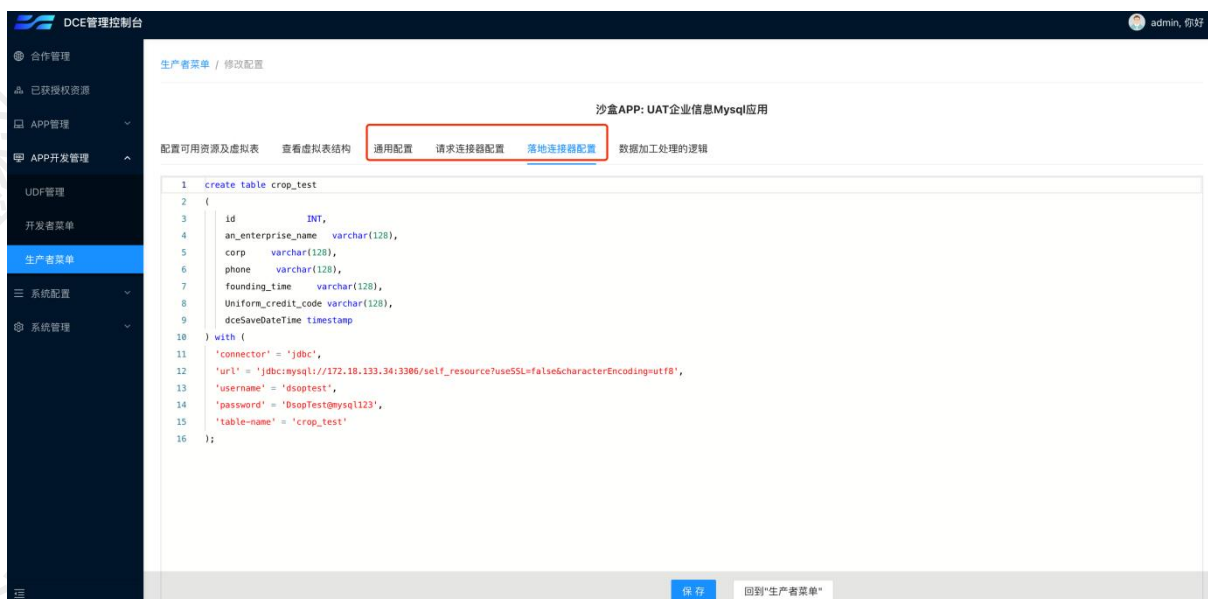
第七步：生产者修改并“保存”沙盒 APP 的运行环境配置

生产者账号登录至 DCE 管理控制台，在 APP 开发管理>生产者菜单，可以看到被分配给自己，且已经被开发者【发布】到生产环境的沙盒 APP。

生产者需要通过【修改配置】，将沙盒 APP 的运行环境的配置，修改为生产环境的配置，然后重新提交和启动该沙盒 APP，让沙盒 APP 在正式环境中运行。



点击沙盒 APP 操作列的【修改配置】，进入该沙盒 APP 的修改配置操作界面：

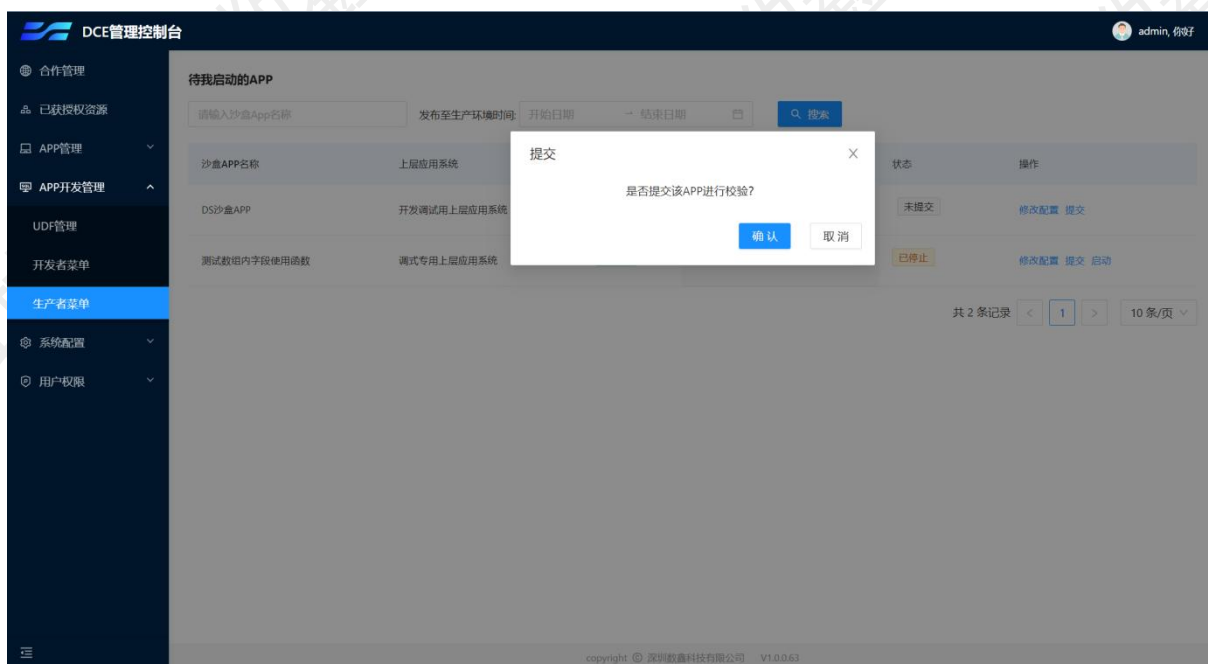


生产者在修改配置的界面中，需要将“通用配置”“请求连接器配置”、“落地连接器配置”修改为生产环境的配置，修改完毕后，点击界面底部的【保存】，保存当前最新的配置信息，并回到“生产者菜单”，

第八步：生产者在生产环境中“提交”沙盒 APP

在“生产者菜单”列表，点击该沙盒 APP 操作列的【提交】，当前显示二次确认窗口，如下图所示，点击【确认】，将提交该沙盒 APP 至生产环境中进行校验（在提交时，不对使用策略中“使

用数据的条件”进行校验)：



稍等片刻，当前会显示沙盒 APP 提交校验的结果。

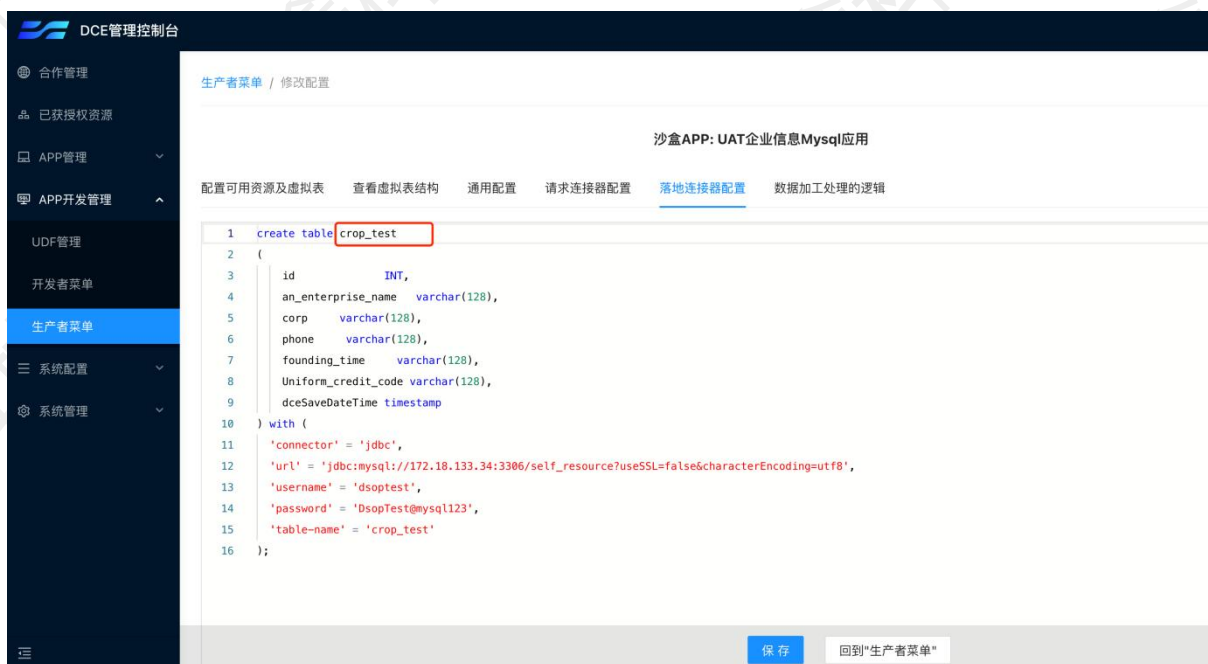
对于开发者提交的新版本，生产者需要与开发者确认是否有修改或者新增“请求连接器”、“落地连接器”的配置信息，再根据变更的内容进行【修改配置】，并保存修改后的配置，回到“生产者菜单”，点击该沙盒 APP 操作列的【提交】，重新提交新版本进行校验，校验通过后，才可在生产环境中启动运行新版本的沙盒 APP。

第九步：生产者在生产环境“启动”沙盒 APP

当沙盒 APP 当前已有版本提交至生产环境中“校验通过”，那么生产者即可在“生产者菜单”，该沙盒 APP 的操作列，见到【启动】的操作入口，此时启动运行的，是已提交并校验通过的沙盒 APP。

生产者可以在本沙盒 APP 配置的落地连接器对应的落地数据表中，验证沙盒 APP 在开发环境运行后，是否获取到了正式资源的数据。

如：在该例中，落地连接器配置中，落地表名为 crop_test



生产者可以使用数据库管理工具 Navicat，查看落地表“crop_test”，dceSaveTime 处于【启动】运行的时间区间内，如已获取到数据，则表示沙盒 APP 已在生产环境运行并获取到了正式资源的数据。

沙盒 APP 在正式（生产）环境成功启动运行后，即刻建立会话，开始访问沙盒 APP 被授权访问的开放资源。

在沙盒 APP 访问开放资源时，平台将会对访问开放资源的 DCE 设备进行 **设备访问策略的鉴权** 与 **沙盒 APP 使用策略的鉴权**。

设备访问策略的鉴权包括：开放资源是否在有效的授权时间范围内，开放资源的访问次数是否在访问次数限制次数内，访问开放资源的 DCE 设备，是否低于设备访问授权约定的安全级别。

如果以上任意一条未通过鉴权，数据消费方都无法访问/使用相应的开放资源的数据。

沙盒 APP 在拉取数据时，沙盒 APP 以及 DPE 都会根据该沙盒 APP 所对应开放资源的使用策略中的“使用数据的条件”，对拉取的数据进行使用数据的条件过滤。

第十步：生产者“停止运行”已启动的沙盒 APP

- 在沙盒 APP 运行时，生产者可以点击【停止运行】，中止沙盒 APP 的运行；
- 沙盒 APP 在执行完任务后，也会自动“停止运行”，这都是正常的停止状态。

- 运行中的沙盒 APP，也会发生异常停止，生产者可以点击“已停止”状态前面的△图标，查看系统异常停止的原因。

第十一步：数据提供方可查看资源的使用情况（DPE）

数据提供方，可登录 [DPE 管理控制台](#)，看到已授权资源的使用情况：

