Alibaba Cloud

大数据计算服务 最佳实践

文档版本: 20220630



法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。 如果您阅读或使用本文档,您的阅读或使用行为将被视为对本声明全部内容的认可。

- 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档,且仅能用 于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息,您应当严格 遵守保密义务;未经阿里云事先书面同意,您不得向任何第三方披露本手册内容或 提供给任何第三方使用。
- 未经阿里云事先书面许可,任何单位、公司或个人不得擅自摘抄、翻译、复制本文 档内容的部分或全部,不得以任何方式或途径进行传播和宣传。
- 由于产品版本升级、调整或其他原因,本文档内容有可能变更。阿里云保留在没有 任何通知或者提示下对本文档的内容进行修改的权利,并在阿里云授权通道中不时 发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠 道下载、获取最新版的用户文档。
- 4. 本文档仅作为用户使用阿里云产品及服务的参考性指引,阿里云以产品及服务的"现状"、"有缺陷"和"当前功能"的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引,但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的,阿里云不承担任何法律责任。在任何情况下,阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害,包括用户使用或信赖本文档而遭受的利润损失,承担责任(即使阿里云已被告知该等损失的可能性)。
- 5. 阿里云网站上所有内容,包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计,均由阿里云和/或其关联公司依法拥有其知识产权,包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意,任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外,未经阿里云事先书面同意,任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称(包括但不限于单独为或以组合形式包含"阿里云"、"Aliyun"、"万网"等阿里云和/或其关联公司品牌,上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司)。
- 6. 如若发现本文档存在任何错误,请与阿里云取得直接联系。

通用约定

格式	说明	样例		
⚠ 危险	该类警示信息将导致系统重大变更甚至故 障,或者导致人身伤害等结果。	⚠ 危险 重置操作将丢失用户配置数据。		
▲ 警告	该类警示信息可能会导致系统重大变更甚 至故障,或者导致人身伤害等结果。	警告 重启操作将导致业务中断,恢复业务 时间约十分钟。		
〔〕) 注意	用于警示信息、补充说明等,是用户必须 了解的内容。	大意 权重设置为0,该服务器不会再接受新 请求。		
? 说明	用于补充说明、最佳实践、窍门等 <i>,</i> 不是 用户必须了解的内容。	⑦ 说明 您也可以通过按Ctrl+A选中全部文件。		
>	多级菜单递进。	单击设置> 网络> 设置网络类型。		
粗体	表示按键、菜单、页面名称等UI元素。	在 结果确认 页面,单击 确定 。		
Courier字体	命令或代码。	执行 cd /d C:/window 命令,进入 Windows系统文件夹。		
斜体	表示参数、变量。	bae log listinstanceid		
[] 或者 [alb]	表示可选项,至多选择一个。	ipconfig [-all -t]		
{} 或者 {a b}	表示必选项,至多选择一个。	switch {act ive st and}		

最佳实践·目录

目录

1.	.SQL	07
	1.1. MaxCompute SQL示例解析	07
	1.2. 不兼容SQL重写	08
	1.3. 导出SQL的运行结果	16
	1.4. 分区剪裁合理性评估	19
	1.5. 分组取出每组数据的前N条	21
	1.6. 多行数据合并为一行数据	22
	1.7. 行转列及列转行最佳实践	23
	1.8. MaxCompute SQL中的关联操作	25
2	.数据迁移	32
	2.1. 数据迁移	32
	2.2. MaxCompute跨项目迁移	33
	2.3. Hadoop数据迁移MaxCompute最佳实践	38
	2.4. Oracle数据迁移MaxCompute最佳实践	45
	2.5. Kafka数据迁移MaxCompute最佳实践	47
	2.6. Elasticsearch数据迁移至MaxCompute	51
	2.7. JSON数据从MongoDB迁移至MaxCompute	54
	2.8. RDS迁移至MaxCompute实现动态分区	57
	2.9. JSON数据从OSS迁移至MaxCompute	60
	2.10. MaxCompute数据迁移至OTS	63
	2.11. MaxCompute数据迁移至OSS	66
	2.12. 迁移ECS自建MySQL数据库至MaxCompute	67
	2.13. Amazon Redshift数据迁移至MaxCompute	75
	2.14. BigQuery数据迁移至MaxCompute	91
	2.15. 日志数据迁移至MaxCompute	99
	2.15.1. 概述	99

2.15.2. 通过Tunnel迁移日志数据至MaxCompute	100
2.15.3. 通过DataHub迁移日志数据至MaxCompute	101
2.15.4. 通过DataWorks数据集成迁移日志数据至MaxCompute	102
3.数据开发	107
3.1. 日期数据格式转换: STRING、TIMESTAMP、DATETIME互相转换	107
3.2. 基于MaxCompute UDF将IPv4或IPv6地址转换为归属地	109
3.3. IntelliJ IDEA Java UDF开发最佳实践	120
3.4. 使用MaxCompute分析IP来源最佳实践	123
3.5. 解决DataWorks 10 MB文件限制问题最佳实践	127
3.6. 实现指定用户访问特定UDF最佳实践	127
3.7. PyODPS节点实现结巴中文分词	131
3.8. PyODPS节点实现避免将数据下载到本地	135
4.计算优化	137
4.1. SQL调优	137
4.2. JOIN长尾优化	138
4.3. 其它计算长尾调优	144
4.4. 长周期指标的计算优化方案	146
5.作业诊断	148
5.1. 通过Logview诊断慢作业	148
6.成本优化	152
6.1. 成本优化概述	152
6.2. 选择付费方式	152
6.3. 计算成本优化	153
6.4. 存储成本优化	156
6.5. 数据上传下载成本优化	156
6.6. 成本追踪	157
6.7. 计费命令参考	157
6.8. MaxCompute账单分析最佳实践	159

7.安全管理	164
7.1. MaxCompute项目设置RAM子账号为超级管理员	164
7.2. 基于Policy对具备内置角色的用户进行权限管理	166

1.SQL 1.1. MaxCompute SQL示例解析

本文为您介绍MaxCompute SQL常见使用场景,让您快速掌握SQL的写法。

准备数据集

本文以emp表和dept表为示例数据集。您可以自行在MaxCompute项目上创建表并上传数据。数据导入请参见概述。

下载emp表数据文件和dept表数据文件。

创建emp表。

```
CREATE TABLE IF NOT EXISTS emp (
EMPNO STRING,
JOB STRING,
MGR BIGINT,
HIREDATE DATETIME,
SAL DOUBLE,
COMM DOUBLE,
DEPTNO BIGINT);
```

创建dept表。

```
CREATE TABLE IF NOT EXISTS dept (
DEPTNO BIGINT,
DNAME STRING,
LOC STRING);
```

SQL示例

• 示例1: 查询员工人数大于零的所有部门。

为了避免数据量太大,此场景下建议您使用JOIN子句。

```
SELECT d.*
FROM dept d
JOIN (
    SELECT DISTINCT deptno AS no
    FROM emp
) e
ON d.deptno = e.no;
```

• 示例2: 查询薪金比SMITH高的所有员工。

此场景为MAPJOIN的典型场景。

```
SELECT /*+ MapJoin(a) */ e.empno
  , e.ename
  , e.sal
FROM emp e
JOIN (
    SELECT MAX(sal) AS sal
    FROM `emp`
    WHERE `ENAME` = 'SMITH'
) a
ON e.sal > a.sal;
```

• 示例3: 查询所有员工的姓名及其直接上级的姓名。

此场景为等值连接。

```
SELECT a.ename
, b.ename
FROM emp a
LEFT OUTER JOIN emp b
ON b.empno = a.mgr;
```

● 示例4: 查询基本薪金大于1500的所有工作。

此场景下需要使用HAVING子句。

SELECT emp.`JOB` , MIN(emp.sal) AS sal FROM `emp` GROUP BY emp.`JOB` HAVING MIN(emp.sal) > 1500;

• 示例5: 查询在每个部门工作的员工数量、平均工资和平均服务期限。

此场景为使用内建函数的典型场景。

```
SELECT COUNT(empno) AS cnt_emp
, ROUND(AVG(sal), 2) AS avg_sal
, ROUND(AVG(datediff(getdate(), hiredate, 'dd')), 2) AS avg_hire
```

FROM `emp`

GROUP BY `DEPTNO`;

• 示例6: 查询每个部门的薪水前3名的人员的姓名以及其排序。

此场景为典型的Top N场景。

```
SELECT *
FROM (
   SELECT deptno
   , ename
   , sal
   , ROW_NUMBER() OVER (PARTITION BY deptno ORDER BY sal DESC) AS nums
   FROM emp
) emp1
WHERE emp1.nums < 4;</pre>
```

• 示例7: 查询每个部门的人数以及该部门中办事员(CLERK)人数的占比。

```
SELECT deptno
, COUNT(empno) AS cnt
, ROUND(SUM(CASE
    WHEN job = 'CLERK' THEN 1
    ELSE 0
    END) / COUNT(empno), 2) AS rate
FROM `EMP`
GROUP BY deptno;
```

注意事项

- 使用GROUP BY时, SELECT部分必须是分组项或聚合函数。
- ORDER BY后面必须加LIMIT N。
- SELECT表达式中不能用子查询,可以改写为JOIN。
- JOIN不支持笛卡尔积,可以使用MAPJOIN替代。
- UNION All需要改成子查询的格式。
- IN/NOT IN语句对应的子查询只能有一列,而且返回的行数不能超过1000,否则也需要改成JOIN操作执行。

1.2. 不兼容SQL重写

本文为您介绍如何修改不兼容SQL。

背景信息

MaxCompute 2.0完全拥抱开源生态,支持更多的语言功能,拥有更快的运行速度。但是MaxCompute 2.0会执行更严格的语法检测,一些在旧版本编译器下正常执行的不严谨的语法在MaxCompute 2.0下执行会报错。

group.by.with.star

说明:即 select * ...group by... 语句。

- MaxCompute 2.0版本中,要求Group By列表是源表中所有的列,否则执行报错。
- 旧版MaxCompute中,即使Group By列表不覆盖源表中所有的列,也支持 select * from group by key 语法。

示例

- 场景1: Group By Key中不包含所有列。
- 。 错误写法

select * from t group by key;

◦ 报错信息

FAILED: ODPS-0130071:[1,8] Semantic analysis exception - column reference t.value should appear in GROUP BY key

◦ 正确写法

select distinct key from t;

- 场景2: group by key包含所有列。
 - 如下写法不推荐。

select * from t group by key, value; -- t has columns key and value

○ 虽然MaxCompute2.0不会报错,但推荐改为如下。

select distinct key, value from t;

bad.escape

说明:错误的escape序列问题。

按照MaxCompute规定,在String literal中应该用反斜线加三位8进制数字表示从0到127的ASCII字符。例如:使用"\001"、"\002"表示0、1。但 \01、\0001也被当作\001处理了。

这种方式会给新用户带来困扰,比如需要用"\0001"表示"\000"+"1",便没有办法实现。同时对于从其他系统迁移而来的用户而言,会导致正确 性错误。

⑦ 说明 \000 后面再加数字,如 \0001 - \0009 或 \00001 的写法可能会返回错误。

MaxCompute 2.0会解决此问题,对脚本中错误的序列进行修改。

● 错误写法

select split(key, "\01"), value like "\0001" from t;

● 报错信息

FAILED: ODPS-0130161:[1,19] Parse exception - unexpected escape sequence: 01 ODPS-0130161:[1,38] Parse exception - unexpected escape sequence: 0001

• 正确改法

select split(key, "\001"), value like "\001" from t;

column.repeated.in.creation

说明:如果创建表时列名重复,MaxCompute 2.0将会报错。

示例

● 错误写法

create table t (a BIGINT, b BIGINT, a BIGINT);

报错信息

FAILED: ODPS-0130071:[1,37] Semantic analysis exception - column repeated in creation: a

● 正确写法

create table t (a BIGINT, b BIGINT);

string.join.double

说明:写JOIN条件时,等号的左右两边分别是STRING和DOUBLE类型。

- 旧版MaxCompute会把两边都转成BIGINT类型,会导致严重的精度损失问题,例如:1.1= "1"在连接条件中会被认为是相等的。
- MaxCompute 2.0会与Hive兼容转为DOUBLE类型。

示例

```
• 不推荐写法
```

select * from t1 join t2 on t1.double_value = t2.string_value;

• Warning信息

WARNING: [1,48] implicit conversion from STRING to DOUBLE, potential data loss, use CAST function to suppress

• 推荐改法

select * from t1 join t2 on t.double_value = cast(t2.string_value as double);

window.ref.prev.window.alias

说明: Window Function引用同级select List中的其他Window Function Alias的问题。

示例

• 如果rn在t1中不存在,错误写法如下。

```
select row_number() over (partition by cl order by cl) rn,
row_number() over (partition by cl order by rn) rn2
from tl;
```

• 报错信息

FAILED: ODPS-0130071:[2,45] Semantic analysis exception - column rn cannot be resolved

• 正确改法

```
select row_number() over (partition by cl order by rn) rn2
from
(select cl, row_number() over (partition by cl order by cl) rn
from t1
) tmp;
```

select.invalid.token.after.star

说明:select列表里面允许用户使用星号(*)代表选择某张表的全部列,但星号(*)后面不允许加alias,即使星号(*)展开之后只有一列也不 允许,新一代编译器将会对类似语法进行报错。

- 示例
- 错误写法

select * as alias from table_test;

● 报错信息

FAILED: ODPS-0130161:[1,10] Parse exception - invalid token 'as'

● 正确改法

select * from table_test;

agg.having.ref.prev.agg.alias

说明:有Having子句的情况下, select List可以出现前面Aggregate Function Alias的问题。

- 示例
- 错误写法

```
select count(cl) cnt,
sum(cl) / cnt avg
from t1
group by c2
having cnt > 1;
```

● 报错信息

FAILED: ODPS-0130071:[2,11] Semantic analysis exception - column cnt cannot be resolved ODPS-0130071:[2,11] Semantic analysis exception - column reference cnt should appear in GROUP BY key

其中s、cnt在源表t1中都不存在,但因为有Having子句,旧版MaxCompute并未报错,MaxCompute 2.0则会提示 column cannot be resolve , 并报错。

• 正确改法

```
select cnt, s, s/cnt avg
from
(
select count(c1) cnt,
sum(c1) s
from t1
group by c2
having count(c1) > 1
) tmp;
```

order.by.no.limit

说明: MaxCompute默认 order by 后需要增加 limit 限制数量,因为 order by 是全量排序,没有 limit 时执行性能较低。

示例

● 错误写法

```
select * from (select *
from (select cast(login_user_cnt as int) as uv, '3' as shuzi
from test_login_cnt where type = 'device' and type_name = 'mobile') v
order by v.uv desc) v
order by v.shuzi limit 20;
```

• 报错信息

FAILED: ODPS-0130071:[4,1] Semantic analysis exception - ORDER BY must be used with a LIMIT clause

在子查询 order by v.uv desc 中增加 limit 。

另外, MaxCompute 1.0对于view的检查不够严格。比如在一个不需要检查 limit 的Project (odps.sql.validate.orderby.limit=false)中,创建了一个View。

create view table_view as select id from table_view order by id;

若访问此View:

select * from table_view;

MaxCompute 1.0不会报错,而MaxCompute 2.0会报如下错误信息:

FAILED: ODPS-0130071:[1,15] Semantic analysis exception - while resolving view xdj.xdj_view_limit - ORDER BY must be used with a LIMIT clause

generated.column.name.multi.window

说明:使用自动生成的 alias 的问题。

旧版MaxCompute会为select语句中的每个表达式自动生成一个alias,这个alias会最后显示在Console上。但是,它并不承诺这个alias的生成规则,也 不承诺这个alias的生成规则会保持不变,所以不建议用户使用自动生成的alias。

MaxCompute 2.0会对使用自动生成alias的情况给予警告,由于牵涉面较广,暂时无法直接给予禁止。

对于某些情况,MaxCompute的不同版本间生成的alias规则存在已知的变动,但因为已有一些线上作业依赖于此类alias,这些查询在 MaxCompute版 本升级或者回滚时可能会失败,存在此问题的用户,请修改您的查询,对于感兴趣的列,显式地指定列的别名。

示例

不推荐写法

select _c0 from (select count(*) from table_name) t;

• 建议改法:

select c from (select count(*) c from table_name) t;

non.boolean.filter

使用了非BOOLEAN过滤条件的问题。

MaxCompute不允许布尔类型与其他类型之间的隐式转换,但旧版MaxCompute会允许用户在某些情况下使用BIGINT作为过滤条件。MaxCompute 2.0 将不再允许,如果您的脚本中存在这样的过滤条件,请及时修改。示例如下:

错误写法:

select id, count(*) from table_name group by id having id;

报错信息:

FAILED: ODPS-0130071:[1,50] Semantic analysis exception - expect a BOOLEAN expression

正确改法:

select id, count(*) from table_name group by id having id <> 0;

post.select.ambiguous

在order by、cluster by、distribute by、sort by等语句中,引用了名字冲突的列的问题。

旧版MaxCompute中,系统会默认选取Select列表中的后一列作为操作对象,MaxCompute 2.0将会进行报错,请及时修改。示例如下:

错误写法:

select a, b as a from t order by a limit 10;

报错信息:

FAILED: ODPS-0130071:[1,34] Semantic analysis exception - a is ambiguous, can be both t.a or null.a

正确改法:

select a as c, b as a from t order by a limit 10;

本次推送修改会包括名字冲突但语义一样的情况,虽然不会出现歧义,但是考虑到这种情况容易导致错误,作为一个警告,希望用户进行修改。

duplicated.partition.column

在query中指定了同名的partition的问题。

旧版MaxCompute在用户指定同名partition key时并未报错,而是后一个的值直接覆盖了前一个,容易产生混乱。MaxCompute2.0将会对此情况进行 报错,示例如下:

错误写法一:

insert overwrite table partition (ds = '1', ds = '2') select \dots ;

实际上,在运行时 ds = `1' 被忽略。

正确改法:

insert overwrite table partition (ds = '2') select ... ;

错误写法二:

create table t (a bigint, ds string) partitioned by (ds string);

正确改法:

create table t (a bigint) partitioned by (ds string);

order.by.col.ambiguous

select list中alias重复,之后的order by子句引用到重复的alias的问题。

错误写法:

select id, id
from table_test
order by id;

正确改法:

select id, id id2
from table_name
order by id;

需要去掉重复的alias, order by子句再进行引用。

in.subquery.without.result

colx in subquery没有返回任何结果,则colx在源表中不存在的问题。

错误写法:

```
select * from table_name
where not_exist_col in (select id from table_name limit 0);
```

报错信息:

FAILED: ODPS-0130071:[2,7] Semantic analysis exception - column not_exist_col cannot be resolved

ctas.if.not.exists

目标表语法错误问题。

如果目标表已经存在,旧版MaxCompute不会做任何语法检查,MaxCompute 2.0则会做正常的语法检查,这种情况会出现很多错误信息,示例如下: 错误写法:

```
create table if not exists table_name
as
select * from not exist table;
```

报错信息:

FAILED: ODPS-0130131:[1,50] Table not found - table meta_dev.not_exist_table cannot be resolved

worker.restart.instance.timeout

旧版MaxCompute UDF每输出一条记录,便会触发一次对分布式文件系统的写操作,同时会向Fuxi发送心跳,如果UDF 10分钟没有输出任何结果,会得 到如下错误提示:

FAILED: ODPS-0123144: Fuxi job failed - WorkerRestart errCode:252,errMsg:kInstanceMonitorTimeout, usually caused by bad udf perf ormance.

MaxCompute 2.0的Runtime框架支持向量化,一次会处理某一列的多行来提升执行效率。但向量化可能导致原来不会报错的语句(2条记录的输出时间间隔不超过10分钟),因为一次处理多行,没有及时向Fuxi发送心跳而导致超时。

遇到这个错误,建议首先检查UDF是否有性能问题,每条记录需要数秒的处理时间。如果无法优化UDF性能,可以尝试手动设置batch row大小来绕开 (默认为1024):

set odps.sql.executionengine.batch.rowcount=16;

divide.nan.or.overflow

旧版MaxCompute不会做除法常量折叠的问题。

比如如下语句, 旧版MaxCompute对应的物理执行计划如下:

explain select if(false, 0/0, 1.0) from table_name; in task M1_Stg1: Data source: meta_dev.table_name TS: alias: table_name SEL: If(False, Divide(UDFToDouble(0), UDFToDouble(0)), 1.0) FS: output: None

由此可以看出, IF和Divide函数仍然被保留,运行时因为IF第一个参数为false,第二个参数Divide的表达式不需要求值,所以不会出现除零异常。

而MaxCompute 2.0支持除法常量折叠,所以会报错。如下所示:

错误写法:

select IF(FALSE, 0/0, 1.0)
from table_name;

报错信息:

FAILED: ODPS-0130071:[1,19] Semantic analysis exception - encounter runtime exception while evaluating function /, detailed mess age: DIVIDE func result NaN, two params are 0.000000 and 0.000000

除了上述的错误,还可能遇到overflow错误,比如:

错误写法:

```
select if(false, 1/0, 1.0)
from table_name;
```

报错信息:

FAILED: ODPS-0130071:[1,19] Semantic analysis exception - encounter runtime exception while evaluating function /, detailed mess age: DIVIDE func result overflow, two params are 1.000000 and 0.000000

正确改法:

建议去掉/0的用法,换成合法常量。

CASE WHEN常量折叠也有类似问题,比如:CASE WHEN TRUE THEN 0 ELSE 0/0,MaxCompute 2.0常量折叠时所有子表达式都会求值,导致除0错误。

CASE WHEN可能涉及更复杂的优化场景,比如:

select case when key = 0 then 0 else 1/key end from (select 0 as key from src union all select key from src) r;

优化器会将除法下推到子查询中,转换类似于:

M (select case when 0 = 0 then 0 else 1/0 end c1 from src UNION ALL select case when key = 0 then 0 else 1/key end c1 from src) r;

报错信息:

FAILED: ODPS-0130071:[0,0] Semantic analysis exception - physical plan generation failed: java.lang.ArithmeticException: DIVIDE func result overflow, two params are 1.000000 and 0.000000

其中UNION ALL第一个子句常量折叠报错,建议将SQL中的 CASE WHEN挪到子查询中,并去掉无用的CASE WHEN和去掉/0用法:

```
select c1 end
from (
select 0 c1 end from src
union all
select case when key = 0 then 0 else 1/key end) r;
```

small.table.exceeds.mem.limit

旧版MaxCompute支持Multi-way Join优化,多个Join如果有相同Join Key,会合并到一个FuxiTask中执行,比如下面例子中的/4_1_2_3_Stg1:

```
explain
select t1.*
from t1 join t2 on t1.c1 = t2.c1
join t3 on t1.c1 = t3.c1;
```

旧版MaxCompute物理执行计划:

In Job job0: root Tasks: M1_Stg1, M2_Stg1, M3_Stg1 J4_1_2_3_Stg1 depends on: M1_Stg1, M2_Stg1, M3_Stg1 In Task M1_Stg1: Data source: meta_dev.t1 In Task M2_Stg1: Data source: meta_dev.t2 In Task M3_Stg1: Data source: meta_dev.t3 In Task J4_1_2_3_Stg1: JOIN: t1 INNER JOIN unknown INNER JOIN unknown SEL: t1._col0, t1._col1, t1._col2 FS: output: None

如果增加MapJoin hint,旧版MaxCompute物理执行计划不会改变。也就是说对于旧版MaxCompute优先应用Multi-way Join优化,并且可以忽略用户指 定MapJoin hint。

```
explain
select /* +mapjoin(t1) */ t1.*
from t1 join t2 on t1.c1 = t2.c1
join t3 on t1.c1 = t3.c1;
```

旧版MaxCompute物理执行计划同上。

MaxCompute 2.0 Optimizer会优先使用用户指定的MapJoin hint,对于上述例子,如果t1比较大的话,会遇到类似错误:

FAILED: ODPS-0010000:System internal error - SQL Runtime Internal Error: Hash Join Cursor HashJoin_REL... small table exceeds, mem ory limit(MB) 640, fixed memory used ..., variable memory used ...

对于这种情况,如果MapJoin不是期望行为,建议去掉MapJoin hint。

sigkill.oom

同small.table.exceeds.mem.limit,如果用户指定了MapJoin hint,并且用户本身所指定的小表比较大。在旧版MaxCompute下有可能被优化成Multiway Join从而成功。但在MaxCompute 2.0下,用户可能通过设定 odps.sql.mapjoin.memory.max 来避免小表超限的错误,但每个MaxCompute worker有固定的内存限制,如果小表本身过大,则MaxCompute worker会由于内存超限而被杀掉,错误类似于:

Fuxi job failed - WorkerRestart errCode:9,errMsg:SigKill(OOM), usually caused by OOM(outof memory).

这里建议您去掉MapJoin hint,使用Multi-way Join。

wm_concat.first.argument.const

聚合函数 中关于WM_CONCAT的说明,一直要求WM_CONCAT第一个参数为常量,旧版MaxCompute检查不严格,比如源表没有数据,就算WM_CONCAT第一个参数为ColumnReference,也不会报错。

函数声明: string wm_concat(string separator, string str) 参数说明: separator: String类型常量,分隔符。其他类型或非常量将引发异常。

MaxCompute 2.0会在plan阶段便检查参数的合法性,假如WM_CONCAT的第一个参数不是常量,会立即报错。示例如下:

错误写法:

select wm_concat(value, ',') FROM src group by value;

报错信息:

FAILED: ODPS-0130071:[0,0] Semantic analysis exception - physical plan generation failed: com.aliyun.odps.lot.cbo.validator.Aggr egateCallValidator\$AggregateCallValidationException: Invalid argument type - The first argument of WM_CONCAT must be constant st ring.

pt.implicit.convertion.failed

srcpt是一个分区表,并有两个分区:

create table srcpt(key STRING, value STRING) partitioned by (pt STRING); alter table srcpt add partition (pt='pt1'); alter table srcpt add partition (pt='pt2');

对于以上SQL, String类型pt列,INT类型常量,都会转为DOUBLE进行比较。即使Project设置了 odps.sql.udf.strict.mode=true ,旧版 MaxCompute不会报错,所有pt都会过滤掉,而MaxCompute 2.0会直接报错。示例如下:

错误写法:

select key from srcpt where pt in (1, 2);

报错信息:

FAILED: ODPS-0130071:[0,0] Semantic analysis exception - physical plan generation failed: java.lang.NumberFormatException: ODPS-0123091:Illegal type cast - In function cast, value 'ptl' cannot be casted from String to Double.

建议避免STRING分区列和INT类型常量比较,将INT类型常量改成STRING类型。

having.use.select.alias

SQL规范定义Group by + Having子句是Select子句之前阶段,所以Having中不应该使用Select子句生成的Column alias。

示例

错误写法:

select id id2 from table_name group by id having id2 > 0;

报错信息:

FAILED: ODPS-0130071:[1,44] Semantic analysis exception - column id2 cannot be resolvedODPS-0130071:[1,44] Semantic analysis e xception - column reference id2 should appear in GROUP BY key

其中id2为Select子句中新生成的Column alias,不应该在Having子句中使用。

dynamic.pt.to.static

说明: MaxCompute2.0动态分区某些情况会被优化器转换成静态分区处理。

示例

insert overwrite table srcpt partition(pt) select id, 'pt1' from table_name;

会被转化成

insert overwrite table srcpt partition(pt='pt1') select id from table_name;

如果用户指定的分区值不合法,比如错误地使用了'\${bizdate}',MaxCompute 2.0语法检查阶段便会报错。详情请参见分区。

错误写法:

insert overwrite table srcpt partition(pt) select id, '\${bizdate}' from table_name limit 0;

报错信息:

FAILED: ODPS-0130071:[1,24] Semantic analysis exception - wrong columns count 2 in data source, requires 3 columns (includes dyn amic partitions if any)

旧版MaxCompute因为LIMIT 0, SQL最终没有输出任何数据,动态分区不会创建,所以最终不报错。

lot.not.in.subquery

说明: In subquery中NULL值的处理问题。

在标准SQL的IN 运算中,如果后面的值列表中出现NULL,则返回值不会出现false,只可能是NULL或者true。如1 in (null, 1, 2, 3)为true,而1 in (null, 2, 3)为NULL, null in (null, 1, 2, 3)为NULL。同理not in操作在列表中有NULL的情况下,只会返回false或者NULL,不会出现true。

MaxCompute 2.0会用标准的行为进行处理,收到此提醒的用户请注意检查您的查询,IN操作中的子查询中是否会出现空值,出现空值时行为是否与您预期相符,如果不符合预期请做相应的修改。

示例

select * from t where c not in (select accepted from c_list);

若accepted中不会出现NULL值,则此问题可忽略。若出现空值,则_c not in (select accepted from c_list) 原先返回true,则新版本返回 NULL。

● 正确写法

select * from t where c not in (select accepted from c_list where accepted is not null)

1.3. 导出SQL的运行结果

本文将通过示例,为您介绍导出MaxCompute SQL计算结果的方法。

⑦ 说明 本文中所有SDK部分仅以Java举例。

概述

您可以通过以下方法导出SQL的运行结果:

- 如果数据比较少,请使用SQLTask得到全部的查询结果。
- 如果需要导出某个表或者分区,请使用Tunnel直接导出查询结果。
- 如果SQL比较复杂,请使用Tunnel和SQL相互配合导出查询结果。
- DataWorks 可以方便地帮您运行SQL,同步数据,并支持定时调度,配置任务依赖的功能。
- 开源工具DataX可以帮助您方便地把MaxCompute中的数据导出到目标数据源。

SQLTask方式导出

SQLT ask使用SDK方法,直接调用MaxCompute SQL的接口SQLT ask.getResult(i),可以很方便地运行SQL并获得其返回结果。使用方法请参见SQLT ask。 使用SQLT ask时,请注意:

- SQLT ask.get Result (i)用于导出SELECT查询结果,不适用于导出show tables;等其他MaxCompute命令操作结果。
- SELECT语句返回给客户端的数据条数可以通过READ_TABLE_MAX_ROW进行设置,详情请参见项目空间操作。
- SELECT语句最多返回1万条数据至客户端。即如果在客户端(包括SQLTask)直接执行SELECT语句,相当于在SELECT语句最后加了Limit N。 如果使用CREATE TABLE XX AS SELECT或者INSERT INTO/OVERWRITE TABLE把结果固化到具体的表里则不受此限制。

Tunnel方式导出

如果您需要导出的查询结果是某张表的全部内容(或者是具体的某个分区的全部内容),可以通过Tunnel来实现,详情请参见命令行工具和基于SDK编 写的Tunnel SDK。

此处提供一个Tunnel命令行导出数据的简单示例,Tunnel SDK的编写适用于Tunnel命令行无法支持的场景,详情请参见批量数据通道概述。

tunnel d wc_out c:\wc_out.dat; 2016-12-16 19:32:08 - new session: 201612161932082d3c9b0a012f68e7 total lines: 3 2016-12-16 19:32:08 - file [0]: [0, 3), c:\wc_out.dat downloading 3 records into 1 file 2016-12-16 19:32:08 - file [0] start 2016-12-16 19:32:08 - file [0] OK. total: 21 bytes download OK

SQLTask配合Tunnel方式导出

SQLT ask不能处理超过1万条数据,而Tunnel方式可以,两者可以互补,因此可以基于两者实现超过1万条数据的导出。 代码实现的示例如下。

最佳实践·SQL

private static final String accessId = "userAccessId"; private static final String accessKey = "userAccessKey"; private static final String endPoint = "http://service.cn-shanghai.maxcompute.aliyun.com/api"; private static final String project = "userProject"; private static final String sql = "userSQL"; private static final String table = "Tmp " + UUID.randomUUID().toString().replace("-", " ");//用随机字符串作为临时表的名称。 private static final Odps odps = getOdps(); public static void main(String[] args) { System.out.println(table); runSql(); tunnel(); } /* * 下载SQLTask的结果。 * */ private static void tunnel() { TableTunnel tunnel = new TableTunnel(odps); try { DownloadSession downloadSession = tunnel.createDownloadSession(project, table); System.out.println("Session Status is : " + downloadSession.getStatus().toString()); long count = downloadSession.getRecordCount(); System.out.println("RecordCount is: " + count); RecordReader recordReader = downloadSession.openRecordReader(0, count); Record record; while ((record = recordReader.read()) != null) { consumeRecord(record, downloadSession.getSchema()); } recordReader.close(); } catch (TunnelException e) { e.printStackTrace(); } catch (IOException el) { e1.printStackTrace(); } } /* * 保存数据。 * 数据量少时直接打印后拷贝也是可行的。实际场景可以用Java.io写到本地文件,或者写到远端存储上保存起来。 * */ private static void consumeRecord(Record record, TableSchema schema) { System.out.println(record.getString("username")+","+record.getBigint("cnt")); } /* * 运行SQL,把查询结果保存成临时表。 * 此处保存数据的生命周期为1天,即使删除步骤出了问题,也不会太浪费存储空间。 * */ private static void runSql() { Instance i; StringBuilder sb = new StringBuilder("Create Table ").append(table) .append(" lifecycle 1 as ").append(sql); try { System.out.println(sb.toString()); i = SQLTask.run(getOdps(), sb.toString()); i.waitForSuccess(); } catch (OdpsException e) { e.printStackTrace(); } /* * 初始化MaxCompute的连接信息。 * */ private static Odps getOdps() { Account account = new AliyunAccount(accessId, accessKey); Odps odps = new Odps(account); odps.setEndpoint(endPoint); odps.setDefaultProject(project); return odps; }

DataWorks数据同步方式导出

DataWorks支持运行SQL并配置数据同步任务,以完成数据生成和导出需求。

1. 登录DataWorks控制台。 2. 在左侧导航栏, 单击工作空间列表。 3. 单击相应工作空间后的进入数据开发。 4. 新建业务流程。 i. 右键单击**业务流程**,选择**新建业务流程** ii. 输入业务名称。 iii. 单击新建。 5. 创建SQL节点。 i. 右键单击业务流程,选择新建 > MaxCompute > ODPS SQL。 ii. 填写节点名称为runsql, 单击提交。 iii. 配置ODPS SQL节点,配置完成后单击保存。 6. 创建数据同步节点。 i. 右键单击业务流程,选择新建>数据集成>离线同步。 ii. 填写节点名称为sync2mysql, 单击提交。 iii. 选择数据来源以及去向。 iv. 配置字段映射。 v. 配置通道控制。

- vi. 单击保存。
- 7. 将数据同步节点和ODPS SQL节点连线配置成依赖关系,ODPS SQL节点作为数据的产出节点,数据同步节点作为数据的导出节点。
- 8. 工作流调度配置完成后(可以直接使用默认配置),单击运行。数据同步的运行日志,如下所示。

```
2016-12-17 23:43:46.394 [job-15598025] INFO JobContainer -
任务启动时刻 : 2016-12-17 23:43:34
任务结束时刻 : 2016-12-17 23:43:46
任务总计耗时 : 11s
任务平均流量 : 31.36KB/s
记录写入速度 : 1668rec/s
读出记录总数 : 16689
读写失败总数 : 0
```

9. 执行如下SQL语句查看数据同步的结果。

select count(*) from result_in_db;

1.4. 分区剪裁合理性评估

本文为您介绍如何评估分区剪裁合理性。

背景信息

MaxCompute分区表是指在创建表时指定分区空间,即指定表内的几个字段作为分区列。使用数据时,如果指定了需要访问的分区名称,则只会读取 相应的分区,避免全表扫描,提高处理效率,降低费用。

分区剪裁是指对分区列指定过滤条件,使得SQL执行时只用读取表分区的部分数据,避免全表扫描引起的数据错误及资源浪费。但是分区失效的情况会 经常发生。

本文将从以下两方面介绍分区剪裁:

- 判断分区剪裁是否生效。
- 分区剪裁失效的场景分析。

判断分区剪裁是否生效

通过EXPLAIN命令查看SQL执行计划,用于判断SQL中的分区剪裁是否生效。

• 分区剪裁未生效。

```
explain
select seller_id
from xxxxx_trd_slr_ord_1d
where ds=rand();
```

从执行计划中可见, SQL读取了表的1344个分区, 即该表的所有分区。

• 分区剪裁生效

<pre>explain select seller_id from xxxxx_trd_slr_ord_1d where ds='20150801';</pre>
In Task M1 Stol: Data source:nd alr and id/datrd slr ord id/ds=20150801 TS: allas: nd alr and id/datrd slr_ord_id FIL: EQUAL nd alr and id/datrd slr_ord_id.ds, '20150801') SEL: pd_alr_ord_id/damd_slr_ord_id.seller_id

从执行计划中可见, SQL只读取了表的20150801分区。

分区剪裁失效的场景分析

自定义函数导致分区剪裁失效

当分区剪裁的条件中使用了用户自定义函数(或者部分内建函数)时,分区剪裁失效。所以,对于分区值的限定,如果使用了非常规函数,建议您 使用Explain命令查看执行计划,确定分区剪裁是否生效。

```
explain
select ...
from xxxxx_base2_brd_ind_cw
where ds = concat(SPLIT_PART(bi_week_dim(' ${bdp.system.bizdate}'), ',', 1), SPLIT_PART(bi_week_dim(' ${bdp.system.bizdate}'),
',', 2))
```

⑦ 说明 UDF已支持分区裁剪,详情请参见WHERE子句 (where_condition) 文中的说明。

● 使用JOIN时分区剪裁失效

在SQL语句中使用JOIN进行关联时:

- 如果分区剪裁条件放在WHERE子句中,则分区剪裁会生效。
- 如果分区剪裁条件放在ON子句中,从表的分区剪裁会生效,主表则不会生效。
- 下面针对三种JOIN具体说明:

• LEFT OUT ER JOIN

■ 分区剪裁条件均放在ON子句中



由上图可见, 左表进行了全表扫描, 只有右表的分区裁剪有效果。

■ 分区剪裁条件均放在WHERE子句中

```
set odps.sql.allow.fullscan=true;
explain
select a.seller_id
    ,a.pay_ord_pbt_1d_001
from xxxxx_trd_slr_ord_1d a
left outer join
    xxxxx_seller b
on a.seller_id=b.user_id
where a.ds='20150801';
and b.ds='20150801';
```



由上图可见,两张表的分区裁剪都有效果。

• RIGHT OUTER JOIN

与LEFT OUTER JOIN类似,如果分区剪裁条件放在ON子句中则只有RIGHT OUTER JOIN的左表生效。如果分区剪裁条件放在WHERE中,则两张表都会 生效。

• FULL OUT ER JOIN

分区剪裁条件只有都放在WHERE子句中才会生效,放在ON子句中都不会生效。

注意事项

- 分区剪裁如果失效影响较大,且不容易发现。因此,建议在代码提交时关注分区剪裁失效问题。
- 自定义函数中使用分区剪裁时,需要修改类或者在SQL语句前设置 set odps.sql.udf.ppr.deterministic = true;
 。详情请参见WHERE子句 (where_condition)。

1.5. 分组取出每组数据的前N条

本文将为您介绍如何对数据进行分组,取出每组数据的前N条数据。

示例数据

empno	ename	job	sal
7369	SMITH	CLERK	800.0
7876	SMITH	CLERK	1100.0
7900	JAMES	CLERK	950.0
7934	MILLER	CLERK	1300.0
7499	ALLEN	SALESMAN	1600.0
7654	MARTIN	SALESMAN	1250.0

empno	ename	job	sal
7844	TURNER	SALESMAN	1500.0
7521	WARD	SALESMAN	1250.0

实现方法

您可以通过以下两种方法实现:

• 取出每条数据的行号,再用WHERE语句进行过滤。

```
SELECT * FROM (
SELECT empno
, ename
, sal
, job
, ROW_NUMBER() OVER (PARTITION BY job ORDER BY sal) AS rn
FROM emp
) tmp
WHERE rn < 10;</pre>
```

● 使用UDTF实现Split函数。

详情请参见MaxCompute 学习计划中最后的示例。此方法可以更迅速地判断当前的序号,如果当前序号已经超过指定的条数(例如10条),便不再 处理,从而提高计算效率。

1.6. 多行数据合并为一行数据

本文为您介绍,如何使用SQL实现多行数据合并为一行数据。

示例数据

class	gender	name
1	М	LiLei
1	F	HanMM
1	М	Jim
1	F	HanMM
2	F	Kate
2	М	Peter

使用示例

• 示例1: 将class相同的names合并为一行,并对names去重。去重操作可通过嵌套子查询实现。

SELECT class, wm_concat(distinct ',', name) FROM students GROUP BY class;

```
⑦ 说明 wm_concat 是字符拼接函数,详情请参见WM_CONCAT。
```

输出结果如下。

class	names
1	LiLei,HanMM,Jim
2	Kate,Peter

• 示例2: 统计不同class对应的男女人数。

SELECT
class
, SUM(CASE WHEN gender = 'M' THEN 1 ELSE 0 END) AS cnt_m
, SUM(CASE WHEN gender = 'F' THEN 1 ELSE 0 END) AS cnt_f
FROM students
GROUP BY class;

输出结果如下。

class	cnt_m	cnt_f
1	2	2
2	1	1

1.7. 行转列及列转行最佳实践

本文基于示例为您介绍如何使用SQL实现行转列、列转行需求。

背景信息

行转列与列转行的示意图如下。

姓名	科目	成绩					
张三	语文	74	4-++				
张三	数学	83	行转列	姓名	语文	数学	物理
张三	物理	93		→ 张三	74	83	93
李四	语文	74		李四	74	84	94
李四	数学	84 -	列转行				
李四	物理	94]				

● 行转列

将多行数据转换成一行显示,或将一列数据转换成多列显示。

• 列转行

将一行数据转换成多行显示,或将多列数据转换成一列显示。

示例数据

为便于理解后续代码示例,本文为您提供源数据,并基于源数据提供相关转换示例。

• 创建用于实现行转列的源表并插入数据,命令示例如下。

```
create table rowtocolumn (name string, subject string, result bigint);
insert into table rowtocolumn values
('张三', '函文', 74),
('张三', '物理', 83),
('张三', '物理', 93),
('李四', '语文', 74),
('李四', '数学', 84),
('李四', '物理', 94);
```

• 创建用于实现列转行的源表并插入数据,命令示例如下。

```
create table columntorow (name string, chinese bigint, mathematics bigint, physics bigint);
insert into table columntorow values
('张三' , 74, 83, 93),
('李四' , 74, 84, 94);
```

行转列示例

您可以通过如下两种方法实现行转列:

• 方法一:使用 case when 表达式,灵活提取各科目(subject)的值作为单独的列,命令示例如下。

```
select name as 姓名,
max(case subject when '语文' then result end) as 语文,
max(case subject when '数学' then result end) as 数学,
max(case subject when '物理' then result end) as 物理
from rowtocolumn
group by name;
```

返回结果如下。

+	+	·+	+	+
姓名	语文	数学	物理	
· □ 张三 □ 李四	, 74 74	83 84	93 94	

 方法二:借助MaxCompute提供的内建函数实现,先基于CONCAT和WM_CONCAT函数合并科目和成绩为一列,然后通过KEYVALUE函数解析科目 (subject)的值作为单独的列。命令示例如下。

select name as 姓名,

```
keyvalue(subject, '语文') as 语文,
keyvalue(subject, '数学') as 数学,
keyvalue(subject, '物理') as 物理
```

```
from(
```

select name, wm_concat(';',concat(subject,':',result))as subject
from rowtocolumn
group by name);

返回结果如下。

+	+	+	+	+
姓名	语文	数学	物理	
│ 张三	74	83	93	I
│ 李四	74	84	94	I
+	+		+	+

列转行示例

您可以通过如下两种方法实现列转行:

• 方法一: 使用 union all , 将各科目 (chinese、mathematics、physics) 整合为一列, 命令示例如下。

```
--解除order by必须带limit的限制,方便列转行SQL命令对结果按照姓名排序。
set odps.sql.validate.orderby.limit=false;
--列转行SQL。
select name as 姓名, subject as 科目, result as 成绩
from(
    select name, '语文' as subject, chinese as result from columntorow
    union all
    select name, '数学' as subject, mathematics as result from columntorow
    union all
    select name, '物理' as subject, physics as result from columntorow)
order by name;
```

返回结果如下。

+ 姓名	+ 科目	+ 成绩	+
+ 张三 张三	+ 语文 数学	74 83	+
张三	物理	93	
李四		84	I I
学四 +	│ 物埋	94	 +

● 方法二:借助MaxCompute提供的内建函数实现,先基于CONCAT函数拼接各科目和成绩,然后基于TRANS_ARRAY和SPLIT_PART函数逐层拆解科目 和成绩作为单独的列。命令示例如下。

```
select name as 姓名,
    split_part(subject,':',1) as 科目,
    split_part(subject,':',2) as 成绩
from(
    select trans_array(1,';',name,subject) as (name,subject)
    from(
        select name,
        concat('语文',':',chinese,';','数学',':',mathematics,';','物理',':',physics) as subject
        from columntorow)tt)tx;
```

返回结果如下。

+	│ 科目	│ 成绩	
+	+	+	+
张三	语文	74	1
张三	数学	83	1
张三	物理	93	1
李四	语文	74	1
李四	数学	84	1
李四	物理	94	1
+	+		+

1.8. MaxCompute SQL中的关联操作

本文为您介绍MaxCompute SQL中的关联(JOIN)操作。

概述

JOIN类型如下所示。

类型	说明
INNER JOIN	输出符合关联条件的数据。
LEFT JOIN	输出左表的所有记录,以及右表中符合关联条件的数据。右表中不符合关联条件的行,输出NULL。
RIGHT JOIN	输出右表的所有记录,以及左表中符合关联条件的数据。左表中不符合关联条件的行,输出NULL。
FULL JOIN	输出左表和右表的所有记录,对于不符合关联条件的数据,未关联的另一侧输出 NULL。
LEFT SEMI JOIN	对于左表中的一条数据,如果右表存在符合关联条件的行,则输出左表。
LEFT ANTI JOIN	对于左表中的一条数据,如果右表中不存在符合关联条件的数据,则输出左表。

SQL语句中,同时存在JOIN和WHERE子句时,如下所示。

(SELECT * FROM A WHERE {	subquery_where_condition}	A)	A	
JOIN				
(SELECT * FROM B WHERE {	subquery_where_condition}	B)	В	
ON {on_condition}				
WHERE {where_condition}				

计算顺序如下:

- 1. 子查询中的WHERE子句(即 {subquery_where_condition})。
- 2. JOIN子句中的关联条件(即 {on_condition})。
- 3. JOIN结果集中的WHERE子句(即 {where_condition})。

因此,对于不同的JOIN类型,过滤条件在 {subquery_where_condition} 、 {on_condition} 和 {where_condition} 中时,查询结果可能一致, 也可能不一致。详情请参见场景说明。

示例数据

● 表A

建表语句如下。

CREATE TABLE A AS SELECT * FROM VALUES (1, 20180101), (2, 20180101), (2, 20180102) t (key, ds);

示例数据如下。

key	ds
1	20180101
2	20180101
2	20180102

● 表B

建表语句如下。

CREATE TABLE B AS SELECT * FROM VALUES (1, 20180101),(3, 20180101),(2, 20180102) t (key, ds);

示例数据如下。

key	ds
1	20180101
3	20180101
2	20180102

• 表A和表B的笛卡尔乘积如下。

a.key	a.ds	b.key	b.ds
1	20180101	1	20180101
1	20180101	3	20180101
1	20180101	2	20180102
2	20180101	1	20180101
2	20180101	3	20180101
2	20180101	2	20180102
2	20180102	1	20180101
2	20180102	3	20180101
2	20180102	2	20180102

场景说明

INNER JOIN

INNER JOIN对左右表执行笛卡尔乘积,然后输出满足ON表达式的行。

结论: 过滤条件在 {subquery_where_condition} 、 {on_condition} 和 {where_condition} 中时, 查询结果是一致的。

o 情况1: 过滤条件在子查询 {subquery_where_condition} 中。

```
SELECT A.*, B.*
FROM
(SELECT * FROM A WHERE ds='20180101') A
JOIN
(SELECT * FROM B WHERE ds='20180101') B
ON a.key = b.key;
```

结果如下。

a.key	a.ds	b.key	b.ds
1	20180101	1	20180101

○ 情况2: 过滤条件在JOIN的关联条件 {on_condition} 中。

```
SELECT A.*, B.*
FROM A JOIN B
ON a.key = b.key and A.ds='20180101' and B.ds='20180101';
```

笛卡尔积结果为9条,满足关联条件的结果只有1条,如下。

a.key	a.ds	b.key	b.ds
1	20180101	1	20180101

```
SELECT A.*, B.*
FROM A JOIN B
ON a.key = b.key
WHERE A.ds='20180101' and B.ds='20180101';
```

笛卡尔积的结果为9条,满足关联条件的结果有3条,如下。

a.key	a.ds	b.key	b.ds
1	20180101	1	20180101
2	20180102	2	20180102
2	20180101	2	20180102

对上述结果执行JOIN结果集中的过滤条件 A.ds='20180101' and B.ds='20180101' , 结果只有1条, 如下。

a.key	a.ds	b.key	b.ds
1	20180101	1	20180101

• LEFT JOIN

LEFT JOIN对左右表执行笛卡尔乘积,输出满足ON表达式的行。对于左表中不满足ON表达式的行,输出左表,右表输出NULL。

结论: 过滤条件在 {subguery_where_condition} 、 {on_condition} 和 {where_condition} 中时,查询结果不一致。

• 左表的过滤条件在 {subguery_where_condition} 和 {where_condition} 中时, 查询结果是一致的。

• 右表的过滤条件在 {subquery_where_condition} 和 {on_condition} 中时,查询结果是一致的。

• 情况1: 过滤条件在子查询 {subquery_where_condition} 中。

```
SELECT A.*, B.*
FROM
(SELECT * FROM A WHERE ds='20180101') A
LEFT JOIN
(SELECT * FROM B WHERE ds='20180101') B
ON a.key = b.key;
```

结果如下。

a.key	a.ds	b.key	b.ds
1	20180101	1	20180101
2	20180101	NULL	NULL

○ 情况2: 过滤条件在JOIN的关联条件 {on_condition} 中。

```
SELECT A.*, B.*
FROM A LEFT JOIN B
ON a.key = b.key and A.ds='20180101' and B.ds='20180101';
```

笛卡尔积的结果有9条,满足关联条件的结果只有1条。左表输出剩余不满足关联条件的两条记录,右表输出NULL。

a.key	a.ds	b.key	b.ds
1	20180101	1	20180101
2	20180101	NULL	NULL
2	20180102	NULL	NULL

```
SELECT A.*, B.*
FROM A LEFT JOIN B
ON a.key = b.key
WHERE A.ds='20180101' and B.ds='20180101';
```

笛卡尔积的结果为9条,满足ON条件的结果有3条。

a.key	a.ds	b.key	b.ds
1	20180101	1	20180101
2	20180101	2	20180102
2	20180102	2	20180102

对上述结果执行JOIN结果集中的过滤条件 A.ds='20180101' and B.ds='20180101' , 结果只有1条。

a.key	a.ds	b.key	b.ds
1	20180101	1	20180101

• RIGHT JOIN

RIGHT JOIN和LEFT JOIN是类似的,只是左右表的区别。

o 过滤条件在 {subquery_where_condition} 、 {on_condition} 和 {where_condition} 时, 查询结果不一致。

• 右表的过滤条件,在 {subquery_where_condition} 和 {where_condition} 中时,查询结果一致。

• 左表的过滤条件,放在 {subquery_where_condition} 和 {on_condition} 中时,查询结果一致。

• FULL JOIN

FULL JOIN对左右表执行笛卡尔乘积,然后输出满足关联条件的行。对于左右表中不满足关联条件的行,输出有数据表的行,无数据的表输出NULL。 结论:过滤条件在 {subquery_where_condition} 、 {on_condition} 和 {where_condition} 时,查询结果不一致。

o 情况1: 过滤条件在子查询 {subquery_where_condition} 中。

```
SELECT A.*, B.*
FROM
(SELECT * FROM A WHERE ds='20180101') A
FULL JOIN
(SELECT * FROM B WHERE ds='20180101') B
ON a.key = b.key;
```

结果如下。

a.key	a.ds	b.key	b.ds
1	20180101	1	20180101
2	20180101	NULL	NULL
NULL	NULL	3	20180101

○ 情况2: 过滤条件在JOIN的关联条件 {on_condition} 中。

```
SELECT A.*, B.*
FROM A FULL JOIN B
```

```
ON a.key = b.key and A.ds='20180101' and B.ds='20180101';
```

笛卡尔积的结果有9条,满足关联条件的结果只有1条。对于左表不满足关联条件的两条记录输出左表数据,右表输出NULL。对于右表不满足关联 条件的两条记录输出右表数据,左表输出NULL。

a.key	a.ds	b.key	b.ds
1	20180101	1	20180101
2	20180101	NULL	NULL
2	20180102	NULL	NULL
NULL	NULL	3	20180101
NULL	NULL	2	20180102

```
SELECT A.*, B.*
FROM A FULL JOIN B
ON a.key = b.key
WHERE A.ds='20180101' and B.ds='20180101';
```

笛卡尔积的结果有9条,满足关联条件的结果有3条。

a.key	a.ds	b.key	b.ds
1	20180101	1	20180101
2	20180101	2	20180102
2	20180102	2	20180102

对于不满足关联条件的表输出数据,另一表输出NULL。

a.key	a.ds	b.key	b.ds
1	20180101	1	20180101
2	20180101	2	20180102
2	20180102	2	20180102
NULL	NULL	3	20180101

对上述结果执行JOIN结果集中的过滤条件 A.ds='20180101' and B.ds='20180101' , 结果只有1条。

a.key	a.ds	b.key	b.ds
1	20180101	1	20180101

LEFT SEMI JOIN

LEFT SEMI JOIN将左表的每一条记录,和右表进行匹配。如果匹配成功,则输出左表。如果匹配不成功,则跳过。由于只输出左表,所以JOIN后的 WHERE条件中不涉及右表。

结论: 过滤条件在 {subquery_where_condition} 、 {on_condition} 和 {where_condition} 中时,查询结果是一致的。

○ 情况1: 过滤条件在子查询{subquery_where_condition}中。

SELECT A.* FROM (SELECT * FROM A WHERE ds='20180101') A LEFT SEMI JOIN (SELECT * FROM B WHERE ds='20180101') B ON a.key = b.key;

结果如下。

a.key	a.ds
1	20180101

◦ 情况2: 过滤条件在JOIN的关联条件 {on_condition} 中。

```
SELECT A.*
FROM A LEFT SEMI JOIN B
ON a.key = b.key and A.ds='20180101' and B.ds='20180101';
```

结果如下。

a.key	a.ds
1	20180101

SELECT A.*			
FROM A LEFT SEMI JOIN			
(SELECT * FROM B WHERE ds='20180101') E	3		
ON a.key = b.key			
WHERE A.ds='20180101';			

符合关联条件的结果如下。

a.key	a.ds
1	20180101

对上述结果执行JOIN结果集中的过滤条件 A.ds='20180101',结果如下。

a.key	a.ds
1	20180101

LEFT ANTIJOIN

LEFT ANTI JOIN将左表的每一条记录,和右表进行匹配。如果右表中的记录不匹配,则输出左表。由于只输出左表,所以JOIN后的WHERE条件中不能 涉及右表。LEFT ANTI JOIN常常用来实现NOT EXIST S语义。

结论: 过滤条件在 {subquery_where_condition} 、 {on_condition} 和 {where_condition} 中时, 查询结果不一致。

• 左表的过滤条件在 {subquery_where_condition} 和 {where_condition} 中时,查询结果是一致的。

• 右表的过滤条件在 {subquery_where_condition} 和 {on_condition} 中时,查询结果是一致的。

• 情况1: 过滤条件在子查询 {subquery_where_condition} 中。

SELECT A.*	
FROM	
(SELECT * FROM A WHERE ds='20180101') A	1
LEFT ANTI JOIN	
(SELECT * FROM B WHERE ds='20180101') E	3
ON a.key = b.key;	

结果如下。

a.key	a.ds
2	20180101

○ 情况2: 过滤条件在JOIN的关联条件 {on_condition} 中。

```
SELECT A.*
FROM A LEFT ANTI JOIN B
ON a.key = b.key and A.ds='20180101' and B.ds='20180101';
```

结果如下。

a.key	a.ds
2	20180101
2	20180102

SELECT A.*	
FROM A LEFT ANTI JOIN	
(SELECT * FROM B WHERE ds='20180101')	В
ON a.key = b.key	
WHERE A.ds='20180101';	

左表中符合关联条件的数据如下。

a.key	a.ds
2	20180101
2	20180102

对上述结果执行JOIN结果集中的过滤条件 A.ds='20180101',结果如下。

a.key	a.ds
2	20180101

注意事项

- INNER JOIN/LEFT SEMI JOIN左右表的过滤条件不受限制。
- LEFT JOIN/LEFT ANTIJOIN左表的过滤条件需放在 {subquery_where_condition} 或 {where_condition} 中, 右表的过滤条件需放在 {subquery_where_condition} 或 {on_condition} 中。
- RIGHT JOIN和LEFT JOIN相反,右表的过滤条件需放在 {subquery_where_condition} 或 {where_condition} 中,左表的过滤条件需放在 {subque ry_where_condition} 或 {on_condition} 。
- FULL OUT ER JOIN的过滤条件只能放在 {subquery_where_condition} 中。

2.数据迁移

本文为您介绍数据迁移的最佳实践,包含将其他业务平台的业务数据或日志数据迁移至MaxCompute,或将MaxCompute的数据迁移至其它业务平 台。

背景信息

传统关系型数据库不适合处理海量数据,如果您的数据存放在传统的关系型数据库且数据量庞大时,可以将数据迁移至MaxCompute。

MaxCompute为您提供了完善的数据迁移方案以及多种经典的分布式计算模型,能够快速地解决海量数据存储和计算问题,有效降低企业成本。

Dat aWorks为MaxCompute提供一站式的数据集成、数据开发、数据管理和数据运维等功能。其中:数据集成为您提供稳定高效和弹性伸缩的数据同步 平台。

最佳实践合集

• 迁移其它业务平台的业务数据至MaxCompute:

- 跨项目迁移MaxCompute数据,详情请参见跨项目迁移MaxCompute数据。
- · 迁移Hadoop数据至MaxCompute,详情请参见迁移Hadoop数据至MaxCompute。数据迁移和脚本迁移遇到的问题及解决方案请参见迁移自建 Hadoop数据至MaxCompute实践。
- 迁移Oracle数据至MaxCompute,详情请参见迁移Oracle数据至MaxCompute。
- 。 迁移Kafka集群数据至MaxCompute, 详情请参见迁移Kafka数据至MaxCompute。
- ◎ 迁移Elasticsearch集群数据至MaxCompute,详情请参见迁移Elasticsearch数据至MaxCompute。
- 迁移RDS数据至MaxCompute,详情请参见迁移RDS数据至MaxCompute。
- ◎ 迁移OSS的JSON数据至MaxCompute,详情请参见迁移OSS的JSON数据至MaxCompute。
- 迁移MongoDB的JSON数据至MaxCompute,详情请参见迁移MongoDB的JSON数据至MaxCompute。
- ◎ 迁移ECS上自建的MySQL数据库中的数据至MaxCompute,详情请参见迁移ECS自建MySQL数据库至MaxCompute。
- 迁移其它业务平台的日志数据至MaxCompute:
 - ◎ 通过Tunnel迁移日志数据至MaxCompute,详情请参见通过Tunnel迁移日志数据至MaxCompute。
 - 通过DataHub迁移日志数据至MaxCompute,详情请参见通过DataHub迁移日志数据至MaxCompute。
 - ◎ 通过DataWorks迁移日志数据至MaxCompute,详情请参见通过DataWorks迁移日志数据至MaxCompute。
- 迁移MaxCompute数据至其它业务平台:
 - ◎ 迁移MaxCompute数据至OSS,详情请参见迁移MaxCompute数据至OSS。
 - ◎ 迁移MaxCompute数据至OTS,详情请参见迁移MaxCompute数据至OTS。

MaxCompute处理业务数据和日志数据后,可以通过Quick B快速地以可视化方式展现数据处理结果,详情请参见基于MaxCompute的大数据B份析。

2.1. 数据迁移

本文为您介绍数据迁移的最佳实践,包含将其他业务平台的业务数据或日志数据迁移至MaxCompute,或将MaxCompute的数据迁移至其它业务平台。

背景信息

传统关系型数据库不适合处理海量数据,如果您的数据存放在传统的关系型数据库且数据量庞大时,可以将数据迁移至MaxCompute。

MaxCompute为您提供了完善的数据迁移方案以及多种经典的分布式计算模型,能够快速地解决海量数据存储和计算问题,有效降低企业成本。

Dat aWorks为MaxCompute提供一站式的数据集成、数据开发、数据管理和数据运维等功能。其中:数据集成为您提供稳定高效和弹性伸缩的数据同步 平台。

最佳实践合集

- 迁移其它业务平台的业务数据至MaxCompute:
 - 跨项目迁移MaxCompute数据,详情请参见跨项目迁移MaxCompute数据。
 - ◎ 迁移Hadoop数据至MaxCompute,详情请参见迁移Hadoop数据至MaxCompute。数据迁移和脚本迁移遇到的问题及解决方案请参见迁移自建 Hadoop数据至MaxCompute实践。
 - ◎ 迁移Oracle数据至MaxCompute,详情请参见迁移Oracle数据至MaxCompute。
 - 迁移Kafka集群数据至MaxCompute,详情请参见迁移Kafka数据至MaxCompute。
 - ◎ 迁移Elast icsearch集群数据至MaxCompute,详情请参见迁移Elast icsearch数据至MaxCompute。
 - 迁移RDS数据至MaxCompute,详情请参见迁移RDS数据至MaxCompute。
 - 迁移OSS的JSON数据至MaxCompute,详情请参见迁移OSS的JSON数据至MaxCompute。
 - ◎ 迁移MongoDB的JSON数据至MaxCompute,详情请参见迁移MongoDB的JSON数据至MaxCompute。
 - 。 迁移ECS上自建的MySQL数据库中的数据至MaxCompute,详情请参见迁移ECS自建MySQL数据库至MaxCompute。
- 迁移其它业务平台的日志数据至MaxCompute:
 - ◎ 通过Tunnel迁移日志数据至MaxCompute,详情请参见通过Tunnel迁移日志数据至MaxCompute。

- ◎ 通过DataHub迁移日志数据至MaxCompute,详情请参见通过DataHub迁移日志数据至MaxCompute。
- ◎ 通过DataWorks迁移日志数据至MaxCompute,详情请参见通过DataWorks迁移日志数据至MaxCompute。
- 迁移MaxCompute数据至其它业务平台:
 - 迁移MaxCompute数据至OSS,详情请参见迁移MaxCompute数据至OSS。
 - ◎ 迁移MaxCompute数据至OTS,详情请参见迁移MaxCompute数据至OTS。

MaxCompute处理业务数据和日志数据后,可以通过Quick B快速地以可视化方式展现数据处理结果,详情请参见基于MaxCompute的大数据BI分析。

2.2. MaxCompute跨项目迁移

本文为您介绍如何配置相同区域下不同的MaxCompute项目,以及如何实现数据迁移。

前提条件

请您首先完成教程《搭建互联网在线运营分析平台》的全部步骤,详情请参见业务场景与开发流程。

背景信息

本文使用的被迁移的原始项目为教程《搭建互联网在线运营分析平台》中的bigdata_DOC项目,您需要再创建一个迁移目标项目,用于存放原始项目的表、资源、配置和数据。

操作步骤

- 1. 创建迁移目标项目
 - 本文的MaxCompute项目即DataWorks的工作空间。
 - i. 登录DataWorks控制台,单击左侧导航栏中的工作空间列表。
 - ii. 选择区域为**华东1(杭州)**,单击创建工作空间。

● 基本配置		2 选择引擎		3 引擎详情	-	
基本信息						
* 工作空间名称	需要字母开头,只能包	含字母下划线和数字				
显示名	如果不填,默认为工作	空间名称				
 (株本) (地本) <			~	\checkmark		
100-00	Here Bene Construction A					
描述						
					-	
下一步 取	汉 消				-	
下一步	汉 消					
下─步 耶	_{怒消} 参数		描述			
下一步 取 分类	図消 参数 参数 工作空间名称		描述 工作空间名 母、下划约	š称的长度需要在3~23个5 訖 (_) 和数字。	字符,以字母开头,且只能包含字	
下一步 1	byii 参数 工作空间名称 显示名		描述 工作空间名 母、下划约 显示名不能 母、下划约	「称的长度需要在3~23个5 え(_) 和数字。 ジ超过23个字符,只能字母 え(_) 和数字。	字符,以字母开头,且只能包含字 母、中文开头,仅包含中文、字	
下─步 ■	Wii 参数 工作空间名称 显示名		描述 描述 工作空间名母、下划约 显示名不能 母、下划约 工作空间移 模式:	5称的长度需要在3~23个5 & () 和数字。 8超过23个字符,只能字母 & () 和数字。 模式是DataWorks新版推出	字符,以字母开头,且只能包含字 母、中文开头,仅包含中文、字 出的新功能,分为 简单模式 和 标准	
下─步 予 类	四消 参数 工作空间名称 显示名		描述 工作空间名 母、下划约 显示名不划约 曼、下划约 工作空间移 模式: 筒单模:	 「称的长度需要在3~23个号 ○ ○<	字符,以字母开头,且只能包含字 母、中文开头,仅包含中文、字 出的新功能,分为 简单模式和标准 作空间对应一个MaxCompute项	
下─步 ⑦ 炎 基本信息	(2) (2) (3) (3) (3) (3) (3) (3) (3) (3) (3) (3		描述 工作空刻名 显示下划名 显示、下划名 显示、下划名 工作空间者 模式: ● 简单模注	5杯的长度需要在3~23个5 (_)和数字。 5超过23个字符,只能字母 (_)和数字。 12式是DataWorks新版推出 式:指一个DataWorks工 法设置开发和生产环境,另	字符,以字母开头,且只能包含字 母、中文开头,仅包含中文、字 出的新功能,分为 简单模式和标准 作空间对应一个MaxCompute项 气能进行简单的数据开发,无法对	
下─步 予类 基本信息	Wii 参数 参数 工作空间名称 显示名		描述 描述 工作空间名 显示名不划约 显示名不划约 显示名不划约 工作空间移 模式 単、下 通行 近年道 東方 世本道	 (赤的长度需要在3~23个5 (_)和数字。 (超过23个字符,只能字母 (_)和数字。 (_)和数字。 (_)和数字。 (_)和数字。 (_)和数字。 (」)和数字。 (」)和数字。 	字符,以字母开头,且只能包含字 母、中文开头,仅包含中文、字 出的新功能,分为 简单模式和标准 作空间对应一个MaxCompute项 只能进行简单的数据开发,无法对 空制。 体内词对应两个MaxComputatio	
<u>下─</u> 步 予类 基本信息	Wiii · · · · · · · · · · · · · · · · · ·		描述 描述 工作空间名 显示で切り 显示下切り 显示下切り 工作空间名 工作空前 工作空前 ● 衛阜模記 ● 御泉の ● 新田模記 ● 新田規	 新称的长度需要在3~23个号 (_)和数字。 超过23个字符,只能字号 (_)和数字。 或是DataWorks新版推出 式是DataWorks新版推出 式"指一个DataWorks工 法设置开发和生产环境,反 流程以及表权限进行强制 式"指一个DataWorks工 以置开发和生产两种环境 	字符,以字母开头,且只能包含字 母、中文开头,仅包含中文、字 出的新功能,分为 简单模式 和 标准 作空间对应一个MaxCompute项 只能进行简单的数据开发,无法对 空制。 作空间对应两个MaxCompute项 竟,提升代码开发规范,并能够对	
下−步 予美 基本信息	(2)消 参数 工作空间名称 显示名 模式		描述 工作空间线 工作空间线 显示下切线 显示下划 基示下划 工作式: ● 简单无对 数据 ● 新規	5杯的长度需要在3~23个号 (_)和数字。 8超过23个字符,只能字号 (_)和数字。 12式是DataWorks新版推出 1式是DataWorks新版推出 1式:指一个DataWorks工 5、法理人及表权限进行通指 式:指一个DataWorks工 以设置开发和生产两种环境 1 1	字符,以字母开头,且只能包含字 母、中文开头,仅包含中文、字 出的新功能,分为 简单模式和标准 作空间对应一个MaxCompute项 只能进行简单的数据开发,无法对 空制。 作空间对应两个MaxCompute项 意,提升代码开发规范,并能够对 操作生产环境的表,保证生产表的	
下─步 予类 基本信息	Wiii · · · · · · · · · · · · · · · · · ·		描述 描述 工作空间划 显示で切り 显示下の均 曼、下名不切り 「作式: ● 筒り、据花 ● 筒り、振花 ● 「前・模式」 ● 「「「」 ● 「「」 ● 「「」 ● 「「」 ● 「「」 ● 「「」 ● 「「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「	 新称的长度需要在3~23个号 (_)和数字。 超过23个字符,只能字母 (_)和数字。 建式是DataWorks新版推出 式:指一个DataWorks工 法设置开发和生产环境,反 流程以及表权限进行强制 式:指一个DataWorks工 以置开发和生产两种环境 出一个DataWorks工 出一个DataWorks工 出一个DataWorks工 	字符,以字母开头,且只能包含字 子符,以字母开头,且只能包含字 母、中文开头,仅包含中文、字 出的新功能,分为 简单模式 和 标准 作空间对应一个MaxCompute项 只能进行简单的数据开发,无法对 空制。 作空间对应两个MaxCompute项 意,提升代码开发规范,并能够对 操作生产环境的表,保证生产表的 C别 。	
<u>下─</u> 步 予类 基本信息	Image: Second secon		描述 工作空间线 工作空可划 显示で切り 显示下切り 显示下切り 工作式: ● 简单 週期 ● 「簡単 ● 「簡単 ● 「「「」 ● 「「」 ● 「「」 ● 「「」 ● 「「」 ● 「「」 ● 「「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「」 ● 「 ● 「 ● 「 ● ● ● ● ● <td>5称的长度需要在3~23个号 2(_)和数字。 2超过23个字符,只能字母 2(_)和数字。 2 2 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3</td> <th>字符,以字母开头,且只能包含字 母、中文开头,仅包含中文、字 出的新功能,分为简单模式和标准 作空间对应一个MaxCompute项 只能进行简单的数据开发,无法对 控制。 作空间对应两个MaxCompute项 着,提升代码开发规范,并能够对 操作生产环境的表,保证生产表的 S別。</th>	5称的长度需要在3~23个号 2(_)和数字。 2超过23个字符,只能字母 2(_)和数字。 2 2 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 2 3 3 3 3 3 3 3 3 3 3 3 3 3	字符,以字母开头,且只能包含字 母、中文开头,仅包含中文、字 出的新功能,分为 简单模式 和 标准 作空间对应一个MaxCompute项 只能进行简单的数据开发,无法对 控制。 作空间对应两个MaxCompute项 着,提升代码开发规范,并能够对 操作生产环境的表,保证生产表的 S別 。	
<u>下−</u> 步 分类 基本信息	IXIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII		描述 描述 工件空切り 显示 くりり 显示、「なりり 夏、下名切り 夏、下名切り 夏、下名切り 夏、「、「なり」 東京 「「「」」」 「「」」 「「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 「」 <td>5称的长度需要在3~23个5 (_)和数字。 5超过23个字符,只能字母。 5超过23个字符,只能字母。 6【二)和数字。 12式是DataWorks新版推出 14式:指一个DataWorks工 15流程以及表权限进行强抗 15指一个DataWorks工 15流程以及表权限进行强抗 15指一个DataWorks工 15位严格控制,禁止随意批告。 16章单模式和标准模式的区 16章单模式和标准模式的区 16章面的数据结果是零</td> <th>字符,以字母开头,且只能包含字 母、中文开头,仅包含中文、字 出的新功能,分为简单模式和标准 作空间对应一个MaxCompute项 只能进行简单的数据开发,无法对 控制。 作空间对应两个MaxCompute项 意,提升代码开发规范,并能够对 景作生产环境的表,保证生产表的 C别。</th>	5称的长度需要在3~23个5 (_)和数字。 5超过23个字符,只能字母。 5超过23个字符,只能字母。 6【二)和数字。 12式是DataWorks新版推出 14式:指一个DataWorks工 15流程以及表权限进行强抗 15指一个DataWorks工 15流程以及表权限进行强抗 15指一个DataWorks工 15位严格控制,禁止随意批告。 16章单模式和标准模式的区 16章单模式和标准模式的区 16章面的数据结果是零	字符,以字母开头,且只能包含字 母、中文开头,仅包含中文、字 出的新功能,分为 简单模式和标准 作空间对应一个MaxCompute项 只能进行简单的数据开发,无法对 控制。 作空间对应两个MaxCompute项 意,提升代码开发规范,并能够对 景作生产环境的表,保证生产表的 C别。	

由于原始项目bigdata_DOC为简单模式,为方便起见,本文中DataWorks工作空间模式也为**简单模式(单环境)**。

工作空间名称全局唯一,建议您使用易于区分的名称,本例中使用的名称为clone_test_doc。

iv. 选择**计算引擎服务**为MaxCompute、按量付费,单击下一步。

v. 配置引擎详情, 单击创建工作空间。

分类	参数	描述
	实例显示名称	实例显示名称不能超过27个字符,仅支持字母、中文开头,仅包含中 文、字母、下划线和数字。
	MaxCompute项目名称	默认与DataWorks工作空间的名称一致。
MaxCompute	MaxCompute访问身份	开发环境的MaxCompute访问身份默认为 任务负责人 ,不可以修改。 生产环境的MaxCompute访问身份包括 阿里云主账号和阿里云子账 号。
	Quota组切换	Quota用来实现计算资源和磁盘配额。

2. 跨项目克隆

您可以通过**跨项目克隆**功能将原始项目bigdata_DOC的节点配置和资源复制到当前项目,详情请参见<mark>跨项目克隆实践</mark>。

? 说明

- 跨项目克隆无法复制表结构与数据。
- 跨项目克隆无法复制组合节点,需要您手动创建。

i. 单击原始项目bigdata_DOC右上角的**跨项目克隆**,跳转至相应的克隆页面。

Ш	数据开发	≗₿С⊕ы	🗰 dw_user_trace_log	ds_user_trace_log	tots_user_trace_log	Tpt_user_trace_log	듣 数据集成	Sq query	× (sa ods_user_trace_log	Sq rpt_user_t	rac
6	💸 DataStudio	biqdata_DOC bigdata_DOC								❷ 跨项目克隆	@ 运维中心 く	2

ii. 选择**克隆目标工作空间**为clone_test_doc,**业务流程**为您需要克隆的业务流程Workshop,勾选所有节点,单击**添加到待克隆**后单击右侧的**待克隆列表**。

=	合 创建	「売隆包 「売隆目标」	作空间:	O BRIA	× Ø			5 荷克駿州去
日二 克隆包列表	解决方案	: 请选择 ~	业务流程: Workshop	→ 提交人				•
	节点类型 提交时间:	: 请选择 ~ 大于等于: YYYY-MM-DD HH:	v 変更美型: 请选择 mm:ss ====		入节点ID YY-MM-DD HH:mm:ss		炊	
	3 🔽		名称	提交人	节点类型	变更类型	提交时间	操作
		1000408562	rpt_user_trace_log	defination (ODPS SQL	更新	2019-06-19 15:32:19	查看 克隆 添加到待克隆
		1000408514	ods_user_trace_log	diam'n a subscription a subscriptin a subscription a subscription a subscription a subscription	ODPS SQL	更新	2019-06-19 15:32:15	
		1000408559	start	eponen .	虚拟节点	更新	2019-06-19 15:32:11	
		1000408561	dw_user_trace_log	diamana ana amin'ny fanisa amin'ny fanisa amin'ny fanisa amin'ny fanisa amin'ny fanisa amin'ny fanisa amin'ny f	ODPS SQL	新増	2019-06-17 10:58:39	
		1000408545	getregion	40.010	函数	新增	2019-06-17 10:53:41	
		1000408531	GetAddr.jar	epination (JAR	新增	2019-06-17 10:50:10	
		1000408527	ip.dat	epinetica.	File	新增	2019-06-17 10:47:28	
4	添加到	持克隆	克羅逊中项				〈上一页 🚺	下页 > 每页显示: 10 ~

iii. 单击全部克隆,将选中的节点克隆至工作空间clone_test_DOC。

clone_test_doc v 创 默认 v 创		持克隆7项	全部克隆	
务流程: Workshop V 提交人: dtplus_docs V			•	
21 安慰: 请选择 ジ 节点: 请输入节点D				
确认克隆 	×			
① 克隆到目标工作空间:clone_test_doc rpt_user_trace_log_2019-06-24_dtplus_docs				
查看完隆包详情				
2 5 8	Ħ			

iv. 切换至您新建的项目,检查节点是否已完成克隆。

3. 新建数据表

跨项目克隆功能无法克隆您的表结构,因此您需要手动新建表。

• 对于非分区表,建议使用如下语句迁移表结构。

create table table_name as select * from **源库**MaxCompute**项目.表名 ;**

对于分区表,建议使用如下语句迁移表结构。

create table table_name partitioned by (分区列 string);

新建表后请将表提交到生产环境。更多建表信息,请参见新建数据表。

4. 数据同步

跨项目克隆功能无法复制原始项目的数据到新项目,因此您需要手动同步数据,本文中仅同步表rpt_user_trace_log的数据。

- i. 新建数据源。
 - a. 在数据集成页面,单击左侧导航栏上的数据源。
 - b. 在数据源管理页面,单击右上角新增数据源,并选择MaxCompute(ODPS)。
 - c. 填写您的数据源名称、ODPS项目名称、AccessKey ID、AccessKey Secret等信息,单击完成,详情请参见配置MaxCompute数据源。
- ii. 创建数据同步任务。
 - a. 在数据开发页面右键单击您克隆的业务流程Workshop下的数据集成,选择新建 > 离线同步。
 - b. 编辑您新建的数据同步任务节点,填写参数如下图所示。其中数据源bigdata_DOC是您的原始项目,数据源odps_first代表您当前的新 建项目,表名是您需要同步数据的表rpt_user_trace_log。完成后单击**调度配置**。

🖸 old2new 🗙 🧮 d	dw_user_trace_log 🛛 🧮 od	ls_user_trace_log 🛛 🗮 ot:	s_user_trace_log	rpt_user_trace_log				
								运维↗
01 选择数据源	55	如据来源			数据去向			调度
	在这	理配置数据的来源端和写入端	;可以是默认的数据	源,也可以是您创建的自有	3 数据源 查看支持的数据来源美			E E
* 数据源	ODPS Y	bigdata_DOC	~ ?	* 数据源	ODPS V	odps_first V	?	本
*表	rpt_user_trace_log				rpt_user_trace_log			
* 分区信息	dt = \${bizdate}	\bigcirc						
空字符串作为null	● 是 ● 否			* 分区信息	dt = \${bizdate}	0		
	数据			清理规则	写入前清理已有数据 (Insert 0	verwrite) 🗸 🗸		
				空字符串作为nuli	● 是 ● 否			

c. 单击使用工作空间根节点后, 提交数据同步任务。

									运维↗	
2	X 调度配置									
在这里配置数据	具体时间: 00.18 () 近: 默认调度时间,从0点到0点30分随机生成									
cron表达式: 00 18 00 ** ?										
ODPS V	依赖上—周期:									
rpt_user_trace_log										
dt = \${bizdate}	调度依赖 🕐 —									
) 是 🧿 否	依赖的上游节点:	请输入父	节点输出名称或输出表名	s 🗸 🛃	使用工作空间根节点					
	自动推荐									
	父节点输出名称		父节点输出表名	节点名	父节点ID	责任人	来源	操作		
	clone_test_doc_ro	ot		clone_test_doc_root		-	手动添加			
	本节点的输出:	请输入节	点输出名称							

- iii. 补数据
 - a. 单击左上角的图标,选择**全部产品 > 运维中心**。
 - b. 单击左侧导航栏中的周期任务运维 > 周期任务。
 - c. 右键单击您的数据同步任务,选择**补数据 > 当前节点**。
 - d. 本例中,需要补数据的日期分区为2019年6月11日到17日,您可以直接选择业务日期,进行多个分区的数据同步。完成设置后,单击确定。

下变入近古				,
* 补数据名称:	P_old2new			
*选择业务日期:	2019-06-11	2019-06-17	111 111	
* 当前任务:	old2new			
* 是否并行:	不并行	~		
				确定取消

e. 在**周期任务运维 > 补数据实例**页面,您可以查看补数据实例任务运行状态,显示运行成功则说明完成数据同步。 Ⅳ. 验证结果

您可以在**业务流程 > 数据开发**中新建ODPS SQL类型节点,执行如下语句查看数据是否完成同步。

select * from rpt_user_trace_log where dt BETWEEN '20190611' and '20190617';

2.3. Hadoop数据迁移MaxCompute最佳实践

本文为您介绍如何通过DataWorks数据同步功能,迁移HDFS数据至MaxCompute,或从MaxCompute迁移数据至HDFS。无论您使用Hadoop还是 Spark,均可以与MaxCompute进行双向同步。

前提条件

● 开通MaxCompute并创建项目。

本文以在华东1(杭州)地域创建项目bigdata_DOC为例。详情请参见开通MaxCompute和DataWorks。

搭建Hadoop集群。

进行数据迁移前,您需要保证Hadoop集群环境正常。本文使用阿里云EMR服务自动化搭建Hadoop集群,详情请参见创建集群。

本文使用的EMR Hadoop版本信息如下:

- EMR版本: EMR-3.11.0
- 集群类型: HADOOP
- ∘ 软件信息:

HDFS2.7.2/YARN2.7.2/Hive2.3.3/Ganglia3.7.2/Spark2.2.1/HUE4.1.0/Zeppelin0.7.3/Tez0.9.1/Sqoop1.4.6/Pig0.14.0/ApacheDS2.0.0/Knox0.13.0

Hadoop集群使用经典网络,地域为华东1(杭州),主实例组ECS计算资源配置公网及内网IP,高可用选择为否(非HA模式)。

步骤一:数据准备

- 1. Hadoop集群创建测试数据。
 - i. 通过阿里云账号登录阿里云E-MapReduce控制台。
 - ii. 在EMR控制台界面,选择目标项目并新建作业doc。本例中Hive建表语句如下。 关于EMR上新建作业更多信息请参见<mark>作业编辑</mark>。

CREATE TABLE IF NOT EXISTS hive_doc_good_sale(create_time timestamp, category STRING, brand STRING, buyer_id STRING, trans_num BIGINT, trans_amount DOUBLE, click_cnt BIGINT) PARTITIONED BY (pt string) ROW FORMAT DELIMITED FIELDS TERMINATED BY ',' lines terminated by '\n';

iii. 单击运行,出现 Query executed successfully 提示,则说明成功在EMR Hadoop集群上创建了表hive_doc_good_sale。



Ⅳ. 插入测试数据。您可以选择从OSS或其他数据源导入测试数据,也可以手动插入少量的测试数据。本文中手动插入数据如下。

insert into

hive_doc_good_sale PARTITION(pt =1) values('2018-08-21','外套','品牌A','lilei',3,500.6,7),('2018-08-22','生鲜','品牌B','l
ilei',1,303,8),('2018-08-22','外套','品牌C','hanmeimei',2,510,2),(2018-08-22,'卫浴','品牌A','hanmeimei',1,442.5,1),('201808-22','生鲜','品牌D','hanmeimei',2,234,3),('2018-08-23','外套','品牌B','jimmy',9,2000,7),('2018-08-23','生鲜','品牌A','jim
my',5,45.1,5),('2018-08-23','外套','品牌C','jimmy',5,100.2,4),('2018-08-24','生鲜','品牌C','peiqi',10,5560,7),('2018-08-24',' ','卫浴','品牌F','peiqi',1,445.6,2),('2018-08-24','外套','品牌A','ray',3,777,3),('2018-08-24','卫浴','品牌G','ray',3,122,3)
,('2018-08-24','外套','品牌C','ray',1,62,7);

V. 完成插入数据后,您可以执行 select * from hive_doc_good_sale where pt =1; 语句,检查Hadoop集群表中是否已存在数据可以用于迁移。

						■ 保存税指	- 2040/5.91 × 2079			
> select * from hive_doe_good_sale where pt =1;										
▶ 运行										
云行结果:										
hive_doc_good_s ale.create_time	hive_doc_good_s ale.category	hive_doc_good_s ale.brand	hive_doc_good_s ale.buyer_Id	hive_doc_good_s ale.trans_num	hive_doc_good_s ale.trans_amount	hive_doc_good_s ale.click_cnt	hive_doc_good_s ale.pt			
2018-08-21 00:00:0 0.0	95書	ARIA	lilei	3	500.6	7	1			
2018-08-22 00:00:0 0.0	95.00	品牌8	lilei	1	303.0	8	1			
2018-08-22 00:00:0).0	外套	8844C	hanmeimei	2	510.0	2	1			
null	238	品牌A	hanmeimei	1	442.5	1	1			
2018-08-22 00:00:0 0.0	239	品牌D	hanmeimei	2	234.0	3	1			
2018-08-23 00:00:0	95 2	品牌8	jimmy	9	2000.0	7	1			

- 2. 利用DataWorks新建目标表。
 - i. 登录DataWorks控制台。
 - ii. 在左侧导航栏, 单击工作空间列表。
 - iii. 在**工作空间列表**页面,单击相应工作空间后的数据开发。
 - iv. 在数据开发页面,右键单击目标工作流程,选择新建 > MaxCompute > 表。
 - v. 在弹出的新建表对话框中,填写表名,并单击提交。
 - ? 说明
 - 表名必须以字母开头,不能包含中文或特殊字符。
 - 如果绑定多个实例,则需要选择MaxCompute引擎实例。
 - vi. 在表的编辑页面,选择DDL模式。
 - vii. 在DDL模式对话框中输入建表语句,单击生成表结构,并确认操作。本示例的建表语句如下所示。

```
CREATE TABLE IF NOT EXISTS hive_doc_good_sale(
  create_time string,
  category STRING,
  brand STRING,
  buyer id STRING,
  trans_num BIGINT,
   trans_amount DOUBLE,
  click_cnt BIGINT
   )
   PARTITIONED BY (pt string) ;
```

在建表过程中,需要考虑Hive数据类型与MaxCompute数据类型的映射,当前数据映射关系请参见数据类型映射表。

上述步骤同样可通过odpscmd命令行工具完成,命令行工具安装和配置请参见安装并配置MaxCompute客户端。



⑦ 说明 考虑到部分Hive与MaxCompute数据类型的兼容问题,建议在odpscmd客户端上执行以下命令。

```
set odps.sql.type.system.odps2=true;
set odps.sql.hive.compatible=true;
```

viii. 单击提交到生产环境,完成表的创建。

ix. 完成建表后,单击左侧导航栏中的表管理,即可查看当前创建的MaxCompute表。



步骤二:数据同步

1. 新建自定义资源组。

由于MaxCompute项目所处的网络环境与Hadoop集群中的数据节点(data node)网络通常不可达,您可以通过自定义资源组的方式,将

Dat aWorks的同步任务运行在Hadoop集群的Master节点上(Hadoop集群内Master节点和数据节点通常可达)。

i. 查看Hadoop集群数据节点。

- a. 登录EMR控制台, 单击集群管理。
- b. 选择集群名称,并在左侧导航栏上单击**主机列表**。

您也可以通过单击上图中Master节点的ECS ID,进入ECS实例详情页。然后单击远程连接进入ECS,执行 hadoop dfsadmin -report 命 令查看data node。

DFS Used:: 0.05% Under replicated blocks: 0 Blocks with corrupt replicas: 0 Missing blocks: 0 Live datanodes (2): Name: 10.31.122.109:50010 (emr-worker-1.cluster-74503) Hostname: emr-worker-1.cluster-74503 Bostname: emr-worker-1.cluster-74503 Decommission Status: Normal Configured Capacity: 33373341696 (310.48 GB) DFS Used: 155725824 (140.51 MB) Non DFS Used: 325541808 (310.46 MB) DFS Used: 155725824 (140.51 MB) Non DFS Used: 325541808 (310.46 MB) DFS Used: 0.65% DFS Beenkining: 9.06% Configured Cache Capacity: 0 (0 B) Cache Used: 10 (0 B) DFS Used: 325451776 (310.48 GB) DFS Used: 325451776 (310.48 GB) DFS Used: 325451776 (310.48 GB) DFS Used: 325451776 (310.83 MB) DFS Used: 9.06% Configured Cache Capacity: 0 (0 B) Cache Used: 0 (0 B) C

⑦ 说明 本示例的data node只具有内网地址,很难与DataWorks默认资源组互通,所以需要设置自定义资源组,将master node 设置为执行DataWorks数据同步任务的节点。

ii. 新建任务资源组。

a. 在DataWorks控制台上,进入数据集成 > 自定义资源组管理页面,单击右上角的新增自定义资源组。

⑦ 说明 目前仅专业版及以上版本方可使用此入口。

b. 添加服务器时,需要输入ECS UUID和机器IP等信息(对于经典网络类型,需要输入服务器名称。对于专有网络类型,需要输入服务器UUID)。目前仅DataWorks V2.0华东2(上海)支持添加经典网络类型的调度资源,对于其他地域,无论您使用的是经典网络还是专有网络类型,在添加调度资源组时都请选择专有网络类型。

机器IP需要填写master node公网IP(内网IP有可能不可达)。ECS的UUID需要进入master node管理终端,通过命令dmidecode | grep UUID获取(如果您的hadoop集群并非搭建在EMR环境上,也可以通过该命令获取)。

[root@emr-header-1 logs]# dmidecode | grep UUID UUID: F631D86C-

- c. 添加服务器后,需要保证master node与DataWorks网络可达。如果您使用的是ECS服务器,需要设置服务器安全组。
 - 如果您使用的内网IP互通,请参见场景示例: ECS自建数据库的安全组配置。
 - 如果您使用的是公网IP,可以直接设置安全组公网出入方向规则。本文中设置公网入方向放通所有端口(实际应用场景中,为了您的数据安全,强烈建议设置详细的放通规则)。

<	bigdata12						8952Z	0 80 800000	NUMBER OF STREET	skliving gelefet
925333	1983-219 PM	9609 08	OREN OREN						1 0 × 6 H	4000000
*820709397	0 escats	110282	NORM	11(2,2)2	255728	152	6.76%	0.001		1877
	= x#	28	-1/-1	122/07/10	0.0.0.00		1	2018005174日 14:35	9 次	1 7532 I BPP
	0 300									

d. 完成上述步骤后,按照提示安装自定义资源组agent。当前状态显示为**可用**时,则新增自定义资源组成功。

當埋資源組 - hdfs				
			刷新	添加服务器
服务器名称/ECS UUID	服务圈IP	当前状态	已使用DMU	操作
F631D86C-	1.00.000.000	可用	0	修改 删除

如果状态为不可用,您可以登录master node,执行 tail -f/home/admin/alisatasknode/logs/heartbeat.log 命令查看DataWorks与 master node之间心跳报文是否超时。

trooteemr-neader-1 logs1# 1	RIS dIS -IS /user/nive/warenouse/nive_auc_good_sale/	
Found 1 items		
druxr-xx - hive hadoop	8 2818-89-83 17:46 /user/hive/warehouse/hive_doc_good_sale/pt=1	
[root@emr-header-1 logs]# 1	ail -f /home/admin/alisatasknode/logs/heartbeat.log	
2018-09-06 21:47:34,440 IN	O [pool-6-thread-1] [HeartbeatReporter.java:184] [] - heartbeat start, current status:2	
2018-09-06 21:47:34,465 IN	70 [pool-6-thread-1] [HeartbeatReporter.java:133] [] - heartbeat end# cost time:0.025s	
2018-09-06 21:47:39,465 IN	70 [pool-6-thread-1] [HeartbeatReporter.java:184] [] - heartbeat start, current status:2	
2018-09-06 21:47:39,491 IN	O [pool-6-thread-1] [HeartbeatReporter.java:133] [] - heartbeat end cost time:0.026s	
2018-09-06 21:47:44,491 IN	70 [pool-6-thread-1] [HeartbeatReporter.java:184] [] - heartbeat start, current status:2	
2018-09-06 21:47:44,515 IN	70 [pool-6-thread-1] [HeartbeatReporter.java:133] [] - heartbeat ends cost time:0.024s	
2018-09-06 21:47:49,516 IN	O [pool-6-thread-1] [HeartbeatReporter.java:184] [] - heartbeat start, current status:2	
2018-09-06 21:47:49,538 IN	70 [pool-6-thread-1] [HeartbeatReporter.java:133] [] - heartbeat end# cost time:0.022s	
2018-09-06 21:47:54,539 IN	O [pool-6-thread-1] [HeartbeatReporter.java:184] [] - heartbeat start, current status:2	
2018-09-06 21:47:54,555 IN	70 [pool-6-thread-1] [HeartbeatReporter.java:133] [] - heartbeat end cost time:0.016s	
A		

- 2. 新建数据源。
 - Dat aWorks新建工作空间后,默认数据源odps_first。因此只需要添加Hadoop集群数据源。更多详情请参见配置HDFS数据源。
 - i. 进入数据集成页面,单击左侧导航栏中的数据源。
 - ii. 在数据源管理页面,单击右上角的新增数据源。
 - iii. 在新增数据源页面中,选择数据源类型为HDFS。

iv. ț	真写HDFS数据源	的各配置项。						
	新增HDFS数据源		×					
	* 数据源名称:	自定义名称						
	数据源描述:							
	* 适用环境:	✔ 开发生产						
	* DefaultFS :	格式:hdfs://ServerIP:Port	0					
	测试连通性:	测试连通性						
			上一步 完成					
	配置		说明					
	数据源名称		数据源名称必须以字母、数字、下划线组合,且不能以数字和下划线开头。					
	数据源描述		对数据源进行简单描述,不得超过80个字符。					
			可以选择 开发 或生产环境。					
	适用环境		⑦ 说明 仅标准模式工作空间会显示此配置。					
	DefaultFS		对于EMR Hadoop集群而言,如果Hadoop集群为HA集群,则此处地址为 hdfs://emr-header-的IP:8020 。如果Hadoop集群为非HA集群,则此处地址为 hdfs://emr-header-1的IP:900 。 。 本实验中的emr-header-1与DataWorks通过公网连接,因此此处填写公网IP并放通安全组。	1				

v. 完成配置后, 单击**测试连通性**。

vi. 测试连通性通过后,单击**完成**。

⑦ 说明 如果EMR Hadoop集群设置网络类型为专有网络,则不支持连通性测试。

- 3. 配置数据同步任务 。
 - i. 在数据开发页面的左侧菜单栏顶部,单击 图标,选择数据集成 > 离线同步。
 - ii. 在新建节点对话框中,输入节点名称和目标文件夹,单击提交。
 - iii. 成功创建数据同步节点后,单击工具栏中的转换脚本按钮。



- iv. 单击提示对话框中的确认,即可进入脚本模式进行开发。
- v. 单击工具栏中的**导入模板**按钮。



vi. 在导入模板对话框中,选择来源类型、数据源、目标类型及数据源,单击确认。



- vii. 新建同步任务完成后,通过导入模板已生成了基本的读取端配置。
 - 此时您可以继续手动配置数据同步任务的读取端数据源,以及需要同步的表信息等。本示例的代码如下所示,更多详情请参见HDFS Reader。



"trans_num",
"trans_amount",
"click_cnt"
1,
"table": "hive_doc_good_sale"
}
},
"setting": {
"errorLimit": {
"record": "1000"
},
"speed": {
"throttle": false,
"concurrent": 1,
"mbps": "1",
}
}
},
"type": "job",
"version": "1.0"

其中,path参数为数据在Hadoop集群中存放的位置。您可以在登录Master Node后,执行 hdfs_dfs_ls_ls_/user/hive/warehouse/hive_doc _good_sale 命令确认。对于分区表,您可以不指定分区,DataWorks数据同步会自动递归到分区路径。

troot8emr-header-1 logs]# hdfs dfs -ls /user/hive/warehouse/hive_doc_good_sale/ Found 1 items Fawary-x- - hive hadoop 8 2818-89-83 17:46 /user/hive/warehouse/hive_doc_good_sale/pt

viii. 完成配置后,单击运行。如果提示任务运行成功,则说明同步任务已完成。如果运行失败,可以通过日志进行排查。

步骤三: 查看结果

- 1. 在DataStudio页面的左侧导航栏,单击临时查询。
- 2. 选择新建 > ODPS SQL。

		临时查询	2 B C C G	create	_table_ddl ×	a workshop		国 运行日志		In ftp_取到问题
	数据开发		2 文件夹	m	PT (F)	ن ش الج				
*	组件管理	→ 临时查问	8/62	> ODPS	SQL			TS ods_user_	info	_d (
8	1 (1)			SHEL	L STRI Beimer S	NG COMMENT	用户	'ID', 性别',		
Ē.					age_rang zodiac S	e STRING COM TRING COMMEN	MEN IT	▼ *年龄段*。 星座*		
Ľ	手动业务流程 New									
					dt STRIN);					

- 3. 选择新建 > ODPS SQL。
- 4. 编写并执行SQL语句,查看导入hive_doc_good_sale的数据。

SQL语句如下所示:

--查看是否成功写入MaxCompute。 select * from hive_doc_good_sale where pt=1;

⑦ 说明 您也可以在odpscmd命令行工具中输入 select * FROM hive_doc_good_sale where pt =1; , 查询表结果。

如果您想实现MaxCompute数据迁移至Hadoop,步骤与上述步骤类似,不同的是同步脚本内的reader和writer对象需要对调,具体实现脚本如下。

{
"configuration": {
"reader": {
"plugin": "odps",
"parameter": {
"partition": "pt=1",
"isCompress": false,
"datasource": "odps_first",
"column": [
"create_time",
"category",
"brand",
"buyer_id",
"trans_num",
"trans_amount",
"click_cnt"
1,
"table": "hive_doc_good_sale"
}
},
Hannah Hanna Hanna (

"writer": { "plugin": "hdfs", "parameter": { "path": "/user/hive/warehouse/hive_doc_good_sale", "fileName": "pt=1", "datasource": "HDFS_data_source", "column": [{ "name": "create_time", "type": "string" }, { "name": "category", "type": "string" }, { "name": "brand", "type": "string" }, { "name": "buyer_id", "type": "string" }, { "name": "trans_num", "type": "BIGINT" }, { "name": "trans_amount", "type": "DOUBLE" }, { "name": "click cnt", "type": "BIGINT" }], "defaultFS": "hdfs://47.100.XX.XX:9000", "writeMode": "append", "fieldDelimiter": ",", "encoding": "UTF-8", "fileType": "text" } }, "setting": { "errorLimit": { "record": "1000" }, "speed": { "throttle": false, "concurrent": 1, "mbps": "1", } } }, "type": "job", "version": "1.0" }

⑦ 说明 您需要参见HDFS Writer,在运行上述同步任务前,对Hadoop集群进行设置。在运行同步任务后,手动复制同步过去的文件。

2.4. Oracle数据迁移MaxCompute最佳实践

本文将为您介绍如何通过DataWorks数据集成,迁移Oracle上的数据至MaxCompute。

前提条件

- 准备DataWorks环境
 - i. 开通MaxCompute和DataWorks。
 - ii. 开通DataWorks。
 - iii. 在DataWorks上完成创建业务流程,本例使用DataWorks简单模式。详情请参见创建业务流程。
- 准备Oracle环境

本文中的Oracle安装在云服务器ECS上,ECS具体配置如下。为了让网络互通,您需要给ECS配置公网IP,并且配置ECS的安全组规则放通Oracle数据 库的常用端口1521。关于ECS安全组配置详情请参见修改安全组规则。

实例ID/名称	标签		监控	可用区 👻	IP地址	状态 ▼	网络类型 🔻	配置	付费方式 ▼	操作
i-bp182xds5u68 oracle测试-gc	۲	•	Ы	华东 1 可用区 H	47.111 0 (公) 172.16 (私有)	●运行中	专有网络	4 vCPU 8 GiB (I/O优化) ecs.c5.xlarge 5Mbps (峰值)	按量 2019年7月28日 21:15 创建	管理 远程连接 ▼ 更改实例规格 更多 ▼

如上图所示,本文中的ECS规格为ecs.c5.xlarge,使用专有网络,区域为华东1(杭州)。

背景信息

本文需要使用DataWorks Oracle Reader读取Oracle中的测试数据,详情请参见Oracle Reader。

准备Oracle测试数据

1. 进入Oracle图形化操作界面,新建表DTSTEST.GOOD_SALE,主要包括create_time、category、brand、buyer_id、trans_num、trans_amount、click_cnt这7列。

🔂 Oracle SQL Developer 表 DTSTEST.GOOD_SALE@本地								
文件(E) 编辑(E) 查看(V) 导航(N) 运行(R) 小组(M)	ŢĮ	LD.	窗口(W) 帮助(Ð				
	80							
连接 × I	- [_ "7)	如使用"页 ×	& 本地 × 🖽 GOOD_	SALE			
💠 - 🔞 🝸 🖓 🖶	3	刘 数	τ据│Model│约束シ	条件 授权 统计信息 顧	帔发器 闪回	相关性 详细资	料 分区 索3	SQL
🗐 Oracle 连接	â	* 📈	🔂 ▼ 操作					
	U		COLUMN_NAME	DATA_TYPE	NULLABLE	DATA_DEFAULT	COLUMN_ID	COMMENTS
		1	CREATE_TIME	TIMESTAMP(6)	Yes	(null)	1	(null)
		2	CATEGORY	VARCHAR2(20 BYTE)	Yes	(null)	2	(null)
		3	BRAND	VARCHAR2(20 BYTE)	Yes	(null)	3	(null)
INC_DATETIME		4	BUYER_ID	VARCHAR2(20 BYTE)	Yes	(null)	4	(null)
RANDOM_ID		5	TRANS_NUM	NUMBER(38,0)	Yes	(null)	5	(null)
THE DAMONE CEDITIC		6	TRANS_AMOUNT	FLOAT	Yes	(null)	6	(null)
査找数据库対象 × 〔	_	7	CLICK_CNT	NUMBER(38,0)	Yes	(null)	7	(null)
🗟 本地	•							2

2. 插入测试数据,本文中手动插入数据如下。

```
insert into good_sale values('28-12月-19','厨具','品牌A','hanmeimei','6','80.6','4');
insert into good_sale values('21-12月-19','生鲜','品牌B','lilei','7','440.6','5');
insert into good_sale values('29-12月-19','衣服','品牌C','lily','12','351.9','9');
commit;
```

3. 完成数据插入之后,执行如下语句查看表数据。

select * from good_sale;

通过DataWorks将数据从Oracle迁移至MaxCompute

- 1. 登录DataWorks控制台。
- 2. 在DataWorks上创建目标表。用以接收从Oracle迁移的数据。
 - i. 右键单击已创建的业务流程,选择新建 > MaxCompute > 表。
 - ii. 在新建表页面,选择引擎类型并输入表名。
 - iii. 在表的编辑页面,单击DDL模式。
 - iv. 在DDL模式对话框,输入建表语句,单击生成表结构。

```
CREATE TABLE good_sale
(
create_time string,
category string,
brand string,
buyer_id string,
trans_num bigint,
trans_amount double,
click_cnt bigint
```

在建表过程中,需要考虑Oracle数据类型与MaxCompute数据类型的映射,Oracle Reader支持的数据类型请参见类型转换列表。

- 3. 新建Oracle数据源,详情请参见配置Oracle数据源。
- 4. 创建离线同步节点。

^{);}

v. 单击提交到生产环境。

i.

ii. iii.	成功创建数据同步 [;] 置。	节点后,选择 数 :	据源 为您刚阝	刚添加的Ora	cle数据源,	表 为您刚刚创建	的测i	式表格,选	择同]名映射。	其他参数	保持默认酉	5
		oracle2maxcompute x	good_sale	🛛 🗸 oracle2m	axcompute	🗰 bank_data							
			一 在 这 里 配 置 数 据 的 ³	来源端和写入端;『	可以是默认的数据	源,也可以是您创建的自有	数据源						
	01 选择数据源		数据来源					1	数据 】	疴			
	* 数据源	Oracle	V Oracle2Max	«Compute 🗸 🗸	0	* 数据源	ODPS			odps_first		?	
	* 表	DTSTEST.GOOD_SALE				生产项目名	dla_proj	ect					
	数据过滤	请参考相应SQL语法填写 键字)。该过滤语句诵》	号where过滤语句(2 常用作增量同步	不要填写where关	0		good_s	ale					
	切分键	根据配置的字段进行数	屠分片,实现并发诱	卖取	?	分区信息	无分区偏	恴					
			教展新览		0	清理规则	写入前	青理已有数据 (Ins	sert Ov	verwrite)			
			Man 1930			空字符串作为null	● 是	() 否					
	02 字段映射		数据来源				数据去	句					
		源头表字段	类型 🖉	3				目标表字段		类型		同名映射	
		CREATE_TIME	TIMESTAMP	•			••	create_time		STRING			
		CATEGORY	VARCHAR2	•			••	category		STRING			
		BRAND	VARCHAR2	•			-0	brand		STRING			

iv. 单击 ▶ 图标运行代码。

v. 您可以在运行日志查看运行结果。

验证结果

- 1. 右键单击业务流程,选择新建 > MaxCompute > ODPS SQL。
- 2. 在新建节点对话框中输入节点名称,并单击提交。
- 3. 在ODPS SQL节点编辑页面输入如下语句。

--查看是否成功写入MaxCompute。 select * from good_sale;

- 4. 单击 🕟 图标运行代码。
- 5. 您可以在**运行日志**查看运行结果。

2.5. Kafka数据迁移MaxCompute最佳实践

本文为您介绍如何使用DataWorks数据集成,将Kafka集群上的数据迁移至MaxCompute。

前提条件

- 开通MaxCompute和DataWorks。
- 在DataWorks上完成创建业务流程,本例使用DataWorks简单模式。详情请参见创建业务流程。
- 搭建Kafka集群

进行数据迁移前,您需要保证自己的Kafka集群环境正常。本文使用阿里云EMR服务自动化搭建Kafka集群,详细过程请参见Kafka快速入门。

本文使用的EMR Kafka版本信息如下:

- EMR版本: EMR-3.12.1
- 集群类型: Kafka
- 。 软件信息:Ganglia 3.7.2, ZooKeeper 3.4.12, Kafka 2.11-1.0.1, Kafka-Manager 1.3.3.16

Kafka集群使用专有网络,区域为华东1(杭州),主实例组ECS计算资源配置公网及内网IP。

背景信息

Kafka是一款分布式发布与订阅的消息中间件,具有高性能、高吞量的特点被广泛使用,每秒能处理上百万的消息。Kafka适用于流式数据处理,主要 应用于用户行为跟踪、日志收集等场景。

一个典型的Kafka集群包含若干个生产者(Producer)、Broker、消费者(Consumer)以及一个Zookeeper集群。Kafka集群通过Zookeeper管理自身 集群的配置并进行服务协同。 Topic是Kafka集群上最常用的消息的集合,是一个消息存储逻辑概念。物理磁盘不存储Topic,而是将Topic中具体的消息按分区(Partition)存储在 集群中各个节点的磁盘上。每个Topic可以有多个生产者向它发送消息,也可以有多个消费者向它拉取(消费)消息。

每个消息被添加到分区时,会分配一个Offset(偏移量,从0开始编号),是消息在一个分区中的唯一编号。

步骤一:准备Kafka数据

您需要在Kafka集群创建测试数据。为保证您可以顺利登录EMR集群Header主机,以及保证MaxCompute和DataWorks可以顺利和EMR集群Header主机 通信,请您首先配置EMR集群Header主机安全组,放行TCP 22及TCP 9092端口。

1. 登录EMR集群Header主机地址。

- i. 进入EMR Hadoop控制台。
- ii. 在顶部导航栏,单击集群管理。
- iii. 在显示的页面,找到您需要创建测试数据的集群,进入集群详情页。
- iv. 在集群详情页面,单击主机列表,确认EMR集群Header主机地址,并通过SSH连接远程登录。
- 2. 创建测试Topic。

执行如下命令创建测试所使用的Topic testkafka。

kafka-topics.sh --zookeeper emr-header-1:2181/kafka-1.0.1 --partitions 10 --replication-factor 3 --topic testkafka --create

3. 写入测试数据。

执行如下命令,可以模拟生产者向Topic testkafka中写入数据。由于Kafka用于处理流式数据,您可以持续不断的向其中写入数据。为保证测试结果,建议写入10条以上的数据。

kafka-console-producer.sh --broker-list emr-header-1:9092 --topic testkafka

您可以同时再打开一个SSH窗口,执行如下命令,模拟消费者验证数据是否已成功写入Kafka。当数据写入成功时,您可以看到已写入的数据。

kafka-console-consumer.sh --bootstrap-server emr-header-1:9092 --topic testkafka --from-beginning

步骤二:在DataWorks上创建目标表

在DataWorks上创建目标表用以接收Kafka数据。

- 1. 进入**数据开发**页面。
 - i. 登录DataWorks控制台。
 - ii. 在左侧导航栏, 单击工作空间列表。
 - iii. 单击相应工作空间后的数据开发。
- 2. 右键单击业务流程,选择新建 > MaxCompute > 表。
- 3. 在弹出的新建表对话框中,填写表名,并单击提交。

? 说明

- 表名必须以字母开头,不能包含中文或特殊字符。
- 如果绑定多个实例,则需要选择MaxCompute引擎实例。
- 4. 在表的编辑页面,选择DDL模式。
- 5. 在DDL模式对话框中, 输入如下建表语句, 单击生成表结构。

```
CREATE TABLE testkafka
(
 kev
               string,
value
               string,
 partition1
               string,
 timestamp1
               string,
 offset
               string,
                string,
 ±123
 event_id
               string,
               string
 tag
);
```

其中的每一列,对应于DataWorks数据集成Kafka Reader的默认列:

- __key__表示消息的key。
- __value__表示消息的完整内容。
- __partition__表示当前消息所在分区。
- __headers__表示当前消息headers信息。
- __offset__表示当前消息的偏移量。
- __timestamp__表示当前消息的时间戳。

您还可以自主命名,详情参见Kafka Reader。

6. 单击提交到生产环境并确认。

步骤三:同步数据

1. 新建独享数据集成资源组。

由于当前Dat aWorks的默认资源组无法完美支持Kaf ka插件,您需要使用独享数据集成资源组完成数据同步。详情请参见<mark>新增和使用独享数据集成资源</mark> <mark>组</mark>。

- 2. 新建数据集成节点。
 - i.
 - ii.
- 3.
- 4.
- 5. 配置脚本,示例代码如下。

最佳实践·数据迁移

```
{
    "type": "job",
    "steps": [
      {
           "stepType": "kafka",
           "parameter": {
               "server": "47.xxx.xxx.xxx:9092",
               "group.id": "console-consumer-83505"
},
               "valueType": "ByteArray",
                "column": [
                  "__key__",
"__value__",
"__partition__",
"__timestamp__",
"__offset__",
                  "'123'",
                   "event_id",
                  "tag.desc"
               ],
               "topic": "testkafka",
               "keyType": "ByteArray",
                "waitTime": "10",
               "beginOffset": "0",
               "endOffset": "3"
           },
           "name": "Reader",
           "category": "reader"
       },
       {
           "stepType": "odps",
           "parameter": {
               "partition": "",
               "truncate": true,
               "compress": false,
               "datasource": "odps_first",
               "column": [
                  "key",
                   "value",
                  "partition1",
                   "timestamp1",
                   "offset",
                   "t123",
                   "event_id",
                   "tag"
               ],
               "emptyAsNull": false,
              "table": "testkafka"
           },
           "name": "Writer",
           "category": "writer"
       }
   ],
    "version": "2.0",
    "order": {
       "hops": [
        {
              "from": "Reader",
               "to": "Writer"
          }
      ]
   },
    "setting": {
      "errorLimit": {
         "record": ""
      },
      "speed": {
          "throttle": false,
           "concurrent": 1,
      }
   }
}
```

您可以通过在Header主机上使用**kaf ka-consumer-groups.sh --boot st rap-server emr-header-1:9092 --list** 命令查看group.id参数,及 消费者的Group名称。

。 命令示例

kafka-consumer-groups.sh --bootstrap-server emr-header-1:9092 --list

○ 返回结果

_emr-client-metrics-handler-group console-consumer-69493 console-consumer-83505 console-consumer-21030 console-consumer-45322 console-consumer-14773

以console-consumer-83505为例,您可以根据该参数在Header主机上使用kafka-consumer-groups.sh --bootstrap-server emr-header-1:9092 --describe --group console-consumer-83505命令确认beginOffset及endOffset参数。

命令示例。

kafka-consumer-groups.sh --bootstrap-server emr-header-1:9092 --describe --group console-consumer-83505

0	返		结果
---	---	--	----

TOPIC HOST	PARTITION CLIENT-ID	CURRENT-OFFSET	LOG-END-OFFSET	LAG	CONSUMER-ID
testkafka	6	0	0	0	-
-	-				
test	6	3	3	0	-
-	-				
testkafka	0	0	0	0	-
-	-				
testkafka	1	1	1	0	-
-	-				
testkafka	5	0	0	0	-
-	-				

6. 配置调度资源组。

i. 在节点编辑页面的右侧导航栏,单击调度配置。

ii. 在资源属性区域,选择调度资源组为您创建的独享数据集成资源组。

⑦ 说明 如果您需要将Kafka的数据周期性(例如每小时)写入MaxCompute,您可以使用beginDateTime及endDateTime参数,设置 数据读取的时间区间为1小时,然后每小时调度一次数据集成任务。详情请参见Kafka Reader。

7. 单击 💽 图标运行代码。

8. 您可以在运行日志查看运行结果。

后续步骤

您可以新建一个ODPS SQL节点运行SQL语句,查看从Kafka同步数据至MaxCompute是否成功。详情请参见<mark>使用临时查询运行SQL语句(可选</mark>)。

2.6. Elasticsearch数据迁移至MaxCompute

本文为您介绍如何通过DataWorks数据同步功能,迁移阿里云Elasticsearch集群上的数据至MaxCompute。

前提条件

- 已开通MaxCompute服务。
 开通指导,详情请参见开通MaxCompute。
- 已开通DataWork服务。
 开通指导,详情请参见开通DataWorks。
- 在DataWorks上已完成创建业务流程。
 本例使用DataWorks简单模式,详情请参见创建业务流程。
- 已搭建阿里云Elasticsearch集群。

进行数据迁移前,您需要保证自己的阿里云Elasticsearch集群环境正常。搭建阿里云Elasticsearch集群的详细过程,请参见快速入门。

本示例中阿里云Elasticsearch的具体配置如下:

- 地域: 华东2(上海)
- 可用区: 上海可用区B

○版本: 5.5.3 with Commercial Feature

背景信息

Elasticsearch是一个基于Lucene的搜索服务器,它提供了一个多用户分布式的全文搜索引擎。Elasticsearch是遵从Apache开源条款的一款开源产品, 是当前主流的企业级搜索引擎。

阿里云Elast icsearch提供Elast icsearch 5.5.3 with Commercial Feature、6.3.2 with Commercial Feature、6.7.0 with Commercial Feature及商业插件X-pack服务,致力于数据分析、数据搜索等场景服务。在开源Elast icsearch基础上提供企业级权限管控、安全监控告警、自动报表生成等功能。

操作步骤

- 1. 在Elasticsearch上创建源表。详情请参见通过DataWorks将MaxCompute数据同步至Elasticsearch。
- 2. 在MaxCompute上创建目标表。
 - i. 登录DataWorks控制台。
 - ii. 在左侧导航栏, 单击**工作空间列表**。
 - iii. 在工作空间列表页面,单击相应工作空间后的数据开发。
 - iv. 在**数据开发**页面,右键单击目标工作流程,选择**新建 > MaxCompute > 表**。
 - v. 在弹出的新建表对话框中,填写表名,并单击提交。
 - ? 说明
 - 表名必须以字母开头,不能包含中文或特殊字符。
 - 如果绑定多个实例,则需要选择MaxCompute引擎实例。
 - vi. 在表的编辑页面,单击DDL模式。
 - vii. 在DDL模式对话框,输入如下建表语句,单击生成表结构。

create table elastic2mc_bankdata (age string, job string, marital string, education string, default string, housing string, loan string, contact string,

month string, day of week string);

viii. 单击提交到生产环境。

```
3. 同步数据。
```

- i.
- ii.

iii.

iv.

```
v.
```

vi. 配置脚本。

示例代码如下。代码释义请参见Elast icsearch Reader。

```
{
   "type": "job",
   "steps": [
       {
           "stepType": "elasticsearch",
           "parameter": {
               "retryCount": 3,
               "column": [
                   "age",
                   "job",
                   "marital",
                    "education",
                   "default",
                    "housing",
                    "loan",
                    "contact",
                    "month",
                    "day of week".
```

```
"duration",
                   "campaign",
                   "pdays",
                   "previous",
                   "poutcome",
                   "emp_var_rate",
                   "cons_price_idx",
                   "cons_conf_idx",
                   "euribor3m",
                   "nr_employed",
                   "y"
               ],
               "scroll": "1m",
               "index": "es index",
               "pageSize": 1,
               "sort": {
                  "age": "asc"
},
               "type": "elasticsearch",
               "connTimeOut": 1000,
               "retrySleepTime": 1000,
               "endpoint": "http://es-cn-xxxx.xxxx.xxxx.com:9200",
               "password": "xxxx",
               "search": {
                  "match_all": {}
               },
               "readTimeOut": 5000,
               "username": "xxxx"
           },
           "name": "Reader",
           "category": "reader"
       },
       {
           "stepType": "odps",
           "parameter": {
              "partition": "",
               "truncate": true,
               "compress": false,
               "datasource": "odps_first",
               "column": [
                   "age",
                   "job",
                   "marital",
                   "education",
                   "default",
                   "housing",
                   "loan",
                   "contact",
                   "month",
                   "day_of_week",
                   "duration",
                   "campaign",
                   "pdays",
                   "previous",
                   "poutcome",
                   "emp_var_rate",
                   "cons_price_idx",
                   "cons_conf_idx",
                   "euribor3m",
                   "nr_employed",
                   "y"
              ],
                "emptyAsNull": false,
               "table": "elastic2mc_bankdata"
           },
           "name": "Writer",
           "category": "writer"
       }
    ],
    "version": "2.0",
    "order": {
       "hops": [
         {
              "from": "Reader",
           "to": "Writer"
```

```
}
]
}
,
"setting": {
    "errorLimit": {
        "record": "0"
    },
    "speed": {
        "throttle": false,
        "concurrent": 1,
        "dmu": 1
    }
}
```

⑦ 说明 您可以在创建的阿里云Elasticsearch集群的基本信息中,查看公网地址和公网端口信息。

vii. 单击o图标运行代码。

viii. 您可以在运行日志查看运行结果。

4. 查看结果。

- i. 右键单击业务流程,选择新建 > MaxCompute > ODPS SQL。
- ii. 在新建节点对话框中输入节点名称,并单击提交。
- iii. 在ODPS SQL节点编辑页面输入如下语句。

SELECT * FROM elastic2mc_bankdata;

- ⅳ. 单击 图标运行代码。
- v. 您可以在运行日志查看运行结果。

2.7. JSON数据从MongoDB迁移至MaxCompute

本文为您介绍如何通过DataWorks的数据集成功能,将从MongoDB提取的JSON字段迁移至MaxCompute。

前提条件

- 开通MaxCompute和DataWorks。
- 开通DataWorks。
- 在DataWorks上完成创建业务流程,本例使用DataWorks简单模式。详情请参见创建业务流程。

在MongoDB上准备测试数据

1. 账号准备。

在数据库内新建用户,用于DataWorks添加数据源。本示例执行如下命令。

db.createUser({user:"bookuser",pwd:"123456",roles:["root"]})

新建用户名为bookuser,密码为123456,权限为root。

2. 数据准备。

```
将数据上传至MongoDB数据库。本示例使用阿里云的<mark>云数据库MongoDB版</mark>,网络类型为VPC(需申请公网地址,否则无法与DataWorks默认资源组
互通),测试数据如下。
```

大数据计算服务

```
{
    "store": {
       "book": [
          {
               "category": "reference",
               "author": "Nigel Rees",
               "title": "Sayings of the Century",
               "price": 8.95
               },
           {
               "category": "fiction",
               "author": "Evelyn Waugh",
               "title": "Sword of Honour",
               "price": 12.99
               },
           {
               "category": "fiction",
               "author": "J. R. R. Tolkien",
               "title": "The Lord of the Rings",
               "isbn": "0-395-19395-8",
               "price": 22.99
               }
                  ],
       "bicycle": {
           "color": "red",
           "price": 19.95
              }
                  },
       "expensive": 10
           }
```

3. 在MongoDB的DMS控制台,本示例使用的数据库为admin,集合为userlog。执行如下命令,查看已上传的数据。

db.userlog.find().limit(10)

通过DataWorks将JSON数据从MongoDB迁移至MaxCompute

- 1. 登录DataWorks控制台。
- 2. 在DataWorks上创建目标表。用以接收从MongoDB迁移的数据。
 - i. 右键单击已创建的**业务流程**,选择新建 > MaxCompute > 表。
 - ii. 在新建表页面,选择引擎类型并输入表名。
 - iii. 在表的编辑页面,单击DDL模式。
 - iv. 在DDL模式对话框,输入建表语句,单击生成表结构。

create table mqdata (mqdata string);

- v. 单击提交到生产环境。
- 3. 新增MongoDB数据源,详情请参见配置MongoDB数据源。
- 4. 创建离线同步节点。
 - i.

```
ii.
```

iii.

iv.

v.

vi. 输入如下脚本。

```
{
   "type": "job",
   "steps": [
   {
       "stepType": "mongodb",
       "parameter": {
          "datasource": "mongodb_userlog",//数据源名称。
           "column": [
              {
              "name": "store.bicycle.color", //JSON字段路径,本例中提取color值。
              "type": "document.String" //非一层子属性以最终获取的类型为准。假如您选取的JSON字段为一级字段,例如本例中的expensiv
e,则直接填写string即可。
             }
           ],
          "collectionName": "userlog" //集合名称。
          },
       "name": "Reader",
       "category": "reader"
       },
       {
           "stepType": "odps",
           "parameter": {
          "partition": "",
          "isCompress": false,
           "truncate": true,
           "datasource": "odps_first",
           "column": [
           "mqdata" //MaxCompute表列名。
          ],
           "emptyAsNull": false,
           "table": "mqdata"
           },
           "name": "Writer",
           "category": "writer"
           }
           ],
           "version": "2.0",
           "order": {
           "hops": [
           {
           "from": "Reader",
           "to": "Writer"
           1
          },
           "setting": {
           "errorLimit": {
           "record": ""
           },
           "speed": {
           "concurrent": 2,
           "throttle": false,
           }
           }
       }
```

vii. 单击 🕟 图标运行代码。

viii. 您可以在运行日志查看运行结果。

验证结果

- 1. 右键单击业务流程,选择新建 > MaxCompute > ODPS SQL。
- 2. 在新建节点对话框中输入节点名称,并单击提交。
- 3. 在ODPS SQL节点编辑页面输入如下语句。

SELECT * from mqdata;

- 4. 单击 🕟 图标运行代码。
- 5. 您可以在运行日志查看运行结果。

2.8. RDS迁移至MaxCompute实现动态分区

本文为您介绍如何使用DataWorks数据集成同步功能自动创建分区,动态地将RDS中的数据迁移至MaxCompute大数据计算服务。

前提条件

- 准备DataWorks环境
 - i. 开通MaxCompute和DataWorks。
 - ii. 开通DataWorks。
 - iii. 在DataWorks上完成创建业务流程,本例使用DataWorks简单模式。详情请参见创建业务流程。
- 新增数据源
 - 新增MySQL数据源作为数据来源,详情请参见配置MySQL数据源。
 - ◎ 新增MaxCompute数据源作为目标数据源接收RDS数据,详情请参见配置MaxCompute数据源。

自动创建分区

i.

准备工作完成后,需要将RDS中的数据定时每天同步到MaxCompute中,自动创建按天日期的分区。详细的数据同步任务的操作和配置请参见DataWorks数据开发和运维。

- 1. 登录DataWorks控制台。
- 2. 在MaxCompute上创建目标表。
 - ii. 单击相应工作空间后的数据开发。
 - iii. 右键单击已创建的业务流程,选择新建 > MaxCompute > 表。
 - iv. 在**新建表**页面,选择引擎类型并输入**表名**。
 - v. 在表的编辑页面,单击DDL模式。
 - vi. 在DDL模式对话框, 输入如下建表语句, 单击生成表结构。

```
CREATE TABLE IF NOT EXISTS ods_user_info_d (
uid STRING COMMENT '用户ID',
gender STRING COMMENT '性别',
age_range STRING COMMENT '年龄段',
zodiac STRING COMMENT '星座'
)
PARTITIONED BY (
dt STRING
);
```

vii. 单击提交到生产环境。

3. 新建离线同步节点。

- i.
- ii.

iii. 选择数据来源和数据去向。

DI rds_sync ×				< >
	🐿 🕫 🔍 [J]			运维
01 选择数据源	数据来源		数据去向	收起
	在这里配置数据的来源满和写入端;可	以是默认的数据源,也可以是您创建	的自有数据源查看支持的数据来源类型	refer
* 数据源:	MySQL v etta, paraladaap, jaag v	? * 数据源:	ODPS · adjoint ·	?
*表:	`ods_user_info_d' ×	*表:	ods_user_info_d ~	
	添加数据源	+	一键生成目标表	
数据过渡:	请参考相应SQL语法填写where过滤语句(不 要填写where关键字)。该过滤语句通常用作 增量同步	? * 分区信息:	dt = \${bizdste}	
		清理规则:	写入前清理已有数据 (Insert Overwrite) ~	
切分键:	uid	⑦ 压缩:	• 不压缩 🔵 压缩	
	数据预览	空字符串作为null:	○是●否	

4. 配置分区参数。

- i. 在右侧导航栏上, 单击**调度配置**。
- ii. 在基础属性区域,设置参数。参数值默认为系统自带的时间参数 \${bizdate} ,格式为yyyymmdd。

⑦ 说明 默认参数值与数据去向中的分区信息值对应。调度执行迁移任务时,目标表的分区值会被自动替换为任务执行日期的前一 天,默认情况下,您会在当前执行前一天的业务数据,这个日期也叫做业务日期。如果您需要使用当天任务运行的日期作为分区值,则需 自定义参数值。

自定义参数设置:用户可以自主选择某一天和格式配置,如下所示:

- 后N年: \$[add_months(yyyymmdd,12*N)]
- 前N年: \$[add_months(yyyymmdd,-12*N)]
- 前N月: \$[add_months(yyyymmdd,-N)]
- 后N周: \$[yyyymmdd+7*N]
- 后N月: \$[add_months(yyyymmdd,N)]
- 前N周: \$[yyyymmdd-7*N]
- 后N天: \$[yyyymmdd+N]
- 前N天: \$[yyyymmdd-N]
- 后N小时: \$[hh24miss+N/24]
- 前N小时: \$[hh24miss-N/24]
- 后N分钟: \$[hh24miss+N/24/60]
- 前N分钟: \$[hh24miss-N/24/60]

? 说明

- 使用中括号([])编辑自定义变量参数的取值计算公式,例如 key1=\$[yyyy-mm-dd]。
- 默认情况下,自定义变量参数的计算单位为天。例如 \$[hh24miss-N/24/60] 表示 (yyyymmddhh24miss-(N/24/60 * 1天))
 的计算结果,然后按 hh24miss 的格式取时分秒。
- 使用add_months的计算单位为月。例如 \$[add_months(yyyymmdd,12 N)-M/24/60] 表示 (yyyymmddhh24miss-(12 * N * 1月))-(M/24/60 * 1天) 的结果,然后按 yyyymmdd 的格式取年月日。

5. 单击 🕟 图标运行代码。

6. 您可以在**运行日志**查看运行结果。

补数据实验

如果您的数据中存在大量运行日期之前的历史数据,需要实现自动同步和自动分区。您可以通过DataWorks的运**维中心**,选择当前的同步数据节点, 使用**补数据**功能实现。

1. 在RDS端按照日期筛选出历史数据。

您可以在同步节点数据来源区域设置数据过滤条件。

● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●					
 教羅際: MySQL 、 ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●		T. J. 🔍 🗇			
* 表: ods_user_jinfo_d' × * 表: ods_user_jinfo_d ~ 添加数据源 + 健生成目标表 数据过滤: S(bizdate) ? * 分区信息: dt = S(bizdate) ? 初分键: uid ? : : · · · · 数据预宽 愛子符串作为null: ① 星 ③ 否 · · · · ·	* 数据源:	MySQL × rds.watatop.log ×	? * 数据源:	ODPS × dipu_fare: ×	?
添加数据源・ 一键生成目标表 数据过滤: ⑤[bizdate] ⑦ * 分区信息: dt = ⑤[bizdate] ⑦ 近分键: uid ⑦ 指理规则: 写入前清理已有数据 (Insert Overwrite) ~ 灯分键: uid ⑦ 圧缩: ● 不压缩 ① 数据预览 空字符串作为null: ○ 星 ● 否	*表:	`ods_user_info_d` × V	*表:	ods_user_info_d ~	
数据过滤: \${bizdate} ⑦ * 分区信息: dt = \${bizdate} ⑦ 清理规则: 写入前清理已有数据 (Insert Overwrite) ~ 切分键: uid ⑦ 正缩: ③ 不压缩 〇 压缩 数据预览 空字符串作为null: 〇 是 ④ 否		添加数据源 +		一键生成目标表	
清理规则: 写入前清理已有数据 (Insert Overwrite) 、 切分键: uid ② 正缩: ③ 不压缩 〇 压缩 数据预览 空字符串作为null: 〇 是 ④ 否	数据过滤:	\${bizdate}	⑦ * 分区信息:	dt = \${bizdate}	
初分键: uid 数据預覧 空字符串作为null: ○ 是 ● 否			注理切问。	E》 新注理日本数据 (Insort Ourspurite)	
切分键: uid ⑦ 正缩: ● 不压缩 压缩 数据预览 空字符串作为null: ○ 星 ● 否			有理规则:	与大則有理已有数据 (insert Overwrite)	
数据预览 空字符串作为null: ○ 是 • 否	切分键:	uid	⑦ 压缩:	● 不压缩 ○ 压缩	
		数据预览	空字符串作为null:	○是 • 否	

2. 执行补数据操作。详情请参见执行补数据并查看补数据实例

3. 在运行的日志中查看对RDS数据的抽取结果。

从运行结果中可以看到MaxCompute已自动创建分区。

运行日志	
Alibaba DI Console Copyright 2018 Ali Start Job[16961870 89484710656] The Job[16961870] 2018-12-02 03:31:2 Reader: mysql	:, Build 201805310000 . Lbaba Group, All rights reserved .)], traceId [283789484710656#79023#None#None#228255635341196741#None#None#rds_sync], running in Pipeline[baseco will run in PhysicsPipeline [basecommon_group_283789484710656_oxs] with requestId [4f44180d-300c-47c3-8ea3-805 25 :
	<pre>column=[["uid","gender","age_range","zodiac"]] connection=[[{"datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource":""datasource:""datasource":""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datasource:""datas</pre>
Writer: odps	<pre>isCompress=[false] partition=[dt=20180913] truncate=[true] datasource=[odps_first] column=[["uid","gender","age_range","zodiac"]] emptyAsNull=[false] table=[ods_user_info_d]</pre>
Setting: 2018-12-02 03:31:2 2018-12-02 03:31:3	errorLimit=[{"record":""}] speed=[{"concurrent":1,"dmu":1,"mbps":"10","throttle":true}] 26 : State: 1(SUBMIT) Total: 0R 0B Speed: 0R/s 0B/s Error: 0R 0B Stage: 0.0% 26 : State: 3(RUN) Total: 0R 0B Speed: 0R/s 0B/s Error: 0R 0B Stage: 0.0%

4. 运行结果验证。在MaxCompute客户端执行如下命令,查看数据写入情况。

SELECT count(*) from ods_user_info_d where dt = 20180913;

Hash实现非日期字段分区

如果您的数据量较大,或没有按照日期字段对第一次全量的数据进行分区,而是按照省份等非日期字段分区,则此时数据集成操作将不能实现自动分区。这种情况下,您可以按照RDS中某个字段进行Hash,将相同的字段值自动存放到这个字段对应值的MaxCompute分区中。

1. 将数据全量同步到MaxCompute的一个临时表,创建一个SQL脚本节点。执行如下命令。

<pre>drop table if exists ods_user_t;</pre>
CREATE TABLE ods_user_t (
dt STRING,
uid STRING,
gender STRING,
age_range STRING,
zodiac STRING);
将MaxCompute表中的数据存入临时表。
insert overwrite table ods user t select dt,uid,gender,age range,zodiac from ods user info d;

2. 创建同步任务的节点mysql_to_odps,即简单的同步任务。将RDS数据全量同步到MaxCompute,无需设置分区。

	1 b 🗉		<u>{}}</u>				
01 选择数据源		数	u 据来 源		数据去向		
		在这里配置	置数据的来源端和写入端;可	似是默认的数据源,也可以是您创建的目	自有数据源查看支持的数据来	源关型	
*数据源:	ODPS		oden.live ~	? * 数据源:	ODPS v	odes.line ~	?
*表:	ods_user_t			*表:	ods_user_d		
分区信息:	无分区信息					一键生成目标表	
压缩:	• 不压缩 🔵 压	缩		* 分区信息:	dt = \${bizdate}	0	
空字符串作为null:	〇 是 📀 否			清理规则:	写入前清理已有数据 (Inser	t Overwrite) 🗸 🗸	
		数据	预览	压缩:	• 不压缩 🔵 压缩		
				空字符串作为null:	○ 是 • 百		

3. 使用SQL语句进行动态分区到目标表,命令如下。

drop table if exists ods_user_d;
//创建一个ODPS分区表(最终目的表)。
CREATE TABLE ods_user_d (
uid STRING,
gender STRING,
age_range_STRING,
zodiac STRING
)
PARTITIONED BY (
dt STRING
);
//执行动态分区SQL,按照临时表的字段dt自动分区,dt字段中相同的数据值,会按照这个数据值自动创建一个分区值。
// 例如 dt 中有些数据是 20181025 ,会自动在 ODPS 分区表中创建一个分区, dt=20181025。
// 动态分区 SQL 如下 。
//可以注意到SQL中select的字段多写了一个dt,就是指定按照这个字段自动创建分区。
insert overwrite table ods_user_d partition(dt)select dt,uid,gender,age_range,zodiac from ods_user_t;
//导入完成后,可以把临时表删除,节约存储成本。
drop table if exists ods_user_t;

在MaxCompute中您可以通过SQL语句完成数据同步。详细的SQL语句介绍请参见阿里云大数据利器MaxCompute学习之--分区表的使用。

4. 将三个节点配置成一个工作流, 按顺序执行。

◇ 数据集成			
▣ 数据同步		Sq 临时表	0
◇ 数据开发			
Sa ODPS SQL	•	Di mysal to odps	•
Mr ODPS MR			
Ⅵ 虚拟节点			
Py PyODPS		▼ Sq 目标表	
Sh Shell			
☐ SQL组件节点			

5. 查看执行过程。您可以重点观察最后一个节点的动态分区过程。

20181203065434115g3a2eqsa
view:
://logview.odps.aliyun.com/logview/?h=http://service.odps.aliyun.com/api&p=DataWorks_D0C&i=20181203065434115g3a2eqsa&token=V0NaNDFxUmpni
Queueing
imary:
purce cost: cpu 0.00 Core * Min, memory 0.00 GB * Min
rts:
aworks_doc.ods_user_t: 20028 (119496 bytes)
Duts:
aworks_doc.ods_use 20028 (119176 bytes)
run time: 0.000
run mode: service job
run engine: execution engine
ance count: 1
time: 0.000
ance time:
: 0.000, max: 0.000, avg: 0.000
it records:

6. 运行结果验证。在MaxCompute客户端执行如下命令,查看数据写入情况。

SELECT count(*) from ods_user_d where dt = 20180913;

2.9. JSON数据从OSS迁移至MaxCompute

本文为您介绍如何通过Dat aWorks数据集成,将JSON数据从OSS迁移至MaxCompute,并使用MaxCompute内置字符串函数GET_JSON_OBJECT提取JSON 信息。

前提条件

- 开通MaxCompute和DataWorks。
- 开通DataWorks。
- 在DataWorks上完成创建业务流程,本例使用DataWorks简单模式。详情请参见创建业务流程。
- 将JSON文件重命名为后缀为 .txt 的文件,并上传至OSS。本文中OSS Bucket地域为华东2(上海)。示例文件如下。

大数据计算服务

```
{
   "store": {
       "book": [
          {
              "category": "reference",
              "author": "Nigel Rees",
              "title": "Sayings of the Century",
              "price": 8.95
           },
           {
              "category": "fiction",
              "author": "Evelyn Waugh",
              "title": "Sword of Honour",
              "price": 12.99
           },
           {
               "category": "fiction",
               "author": "J. R. R. Tolkien",
               "title": "The Lord of the Rings",
               "isbn": "0-395-19395-8",
               "price": 22.99
           }
         ],
         "bicycle": {
            "color": "red",
             "price": 19.95
        }
   },
   "expensive": 10
}
```

将JSON数据从OSS迁移至MaxCompute

- 1. 新增OSS数据源。详情请参见配置OSS数据源。
- 2. 在Dat aWorks上新建数据表,用于存储迁移的JSON数据。 i.
 - ii. 在**新建表**页面,选择引擎类型并输入**表名**。
 - iii. 在表的编辑页面,单击DDL模式。
 - iv. 在DDL模式对话框,输入如下建表语句,单击生成表结构。

```
create table mqdata (mq_data string);
```

```
v. 单击提交到生产环境。
```

```
3. 新建离线同步节点。
```

```
i.
ii.
iii.
iv.
```

```
v.
```

vi. 修改JSON代码后,单击⊙按钮。

示例代码如下。

```
{
    "type": "job",
    "steps": [
      {
           "stepType": "oss",
           "parameter": {
              "fieldDelimiterOrigin": "^",
              "nullFormat": "",
              "compress": "",
              "datasource": "OSS_userlog",
              "column": [
                 {
                     "name": 0,
                     "type": "string",
                      "index": 0
                 }
               ],
               "skipHeader": "false",
              "encoding": "UTF-8",
              "fieldDelimiter": "^",
              "fileFormat": "binary",
              "object": [
                  "applog.txt"
              ]
           },
           "name": "Reader",
           "category": "reader"
       },
       {
           "stepType": "odps",
           "parameter": {
              "partition": "",
              "isCompress": false,
              "truncate": true,
              "datasource": "odps_first",
              "column": [
                 "mqdata"
              ],
              "emptyAsNull": false,
              "table": "mqdata"
           },
           "name": "Writer",
           "category": "writer"
       }
    ],
    "version": "2.0",
    "order": {
      "hops": [
         {
              "from": "Reader",
              "to": "Writer"
          }
      ]
    },
    "setting": {
      "errorLimit": {
         "record": ""
      },
       "speed": {
          "concurrent": 2,
          "throttle": false,
       }
   }
```

结果验证

- 1. 新建ODPS SQL节点。
 - i. 右键单击业务流程,选择**新建 > MaxCompute > ODPS SQL**。

```
ii. 在新建函数对话框中, 输入函数名称, 单击提交。
```

iii. 在ODPS SQL节点编辑页面输入如下语句。

```
--查询表mq_data数据。
```

```
SELECT * from mqdata;
--获取JSON文件中的EXPENSIVE值。
SELECT GET_JSON_OBJECT(mqdata.MQdata,'$.expensive') FROM mqdata;
```

iv. 单击 **●**图标运行代码。

```
v. 您可以在运行日志查看运行结果。
```

2.10. MaxCompute数据迁移至OTS

本文为您介绍如何将MaxCompute数据迁移至表格存储OTS(Table Store)。

前提条件

- 开通MaxCompute和DataWorks。
- 开通DataWorks。
- 在DataWorks上完成创建业务流程,本例使用DataWorks简单模式。详情请参见创建业务流程。

操作步骤

- 1. 在DataWorks上创建表。
 - i. 登录DataWorks控制台。
 - ii. 在左侧导航栏,单击**工作空间列表**。
 - iii. 单击相应工作空间后的数据开发。
 - iv. 右键单击已创建的业务流程,选择新建 > MaxCompute > 表。
 - v. 在新建表页面,选择引擎类型并输入表名。
 - vi. 在表的编辑页面,单击DDL模式。
 - vii. 在DDL模式对话框,输入如下建表语句,单击生成表结构。

```
create table Transs
(name string,
id bigint,
gender string);
```

viii. 单击提交到生产环境。

- 2. 为表Transs导入数据。
 - i. 在**数据开发**页面,单击<mark>西</mark>图标。
 - ii. 在数据导入向导对话框,至少输入3个字母来搜索需要导入数据的表,单击下一步。
 - iii. 选择数据导入方式为上传本地数据,单击选择文件后的浏览...。选择本地数据文件,配置导入信息。
 - 示例数据如下。

```
qwe,145,F
asd,256,F
xzc,345,M
rgth,234,F
ert,456,F
dfg,12,M
ttyj,4,M
bfg,245,M
nrtjeryj,15,F
rwh,2344,M
trh,387,F
srjeyj,67,M
saerh,567,M
```

- iv. 单击下一步。
- v. 选择目标表字段与源字段的匹配方式。
- vi. 单击导入数据。

```
3. 在表格存储控制台上创建表。
```

- i. 登录表格存储控制台, 创建实例。详情请参见创建实例。
- ii. 创建数据表Trans。详情请参见创建数据表。

```
4. 在DataWorks中新增数据源。
```

```
i.
```

ii. 在左侧导航栏,单击**工作空间列表**。

iii.

- iv.
- v. 单击右上角**新增数据源**,并选择数据类型为**ODPS**。
- vi. 在**新增ODPS数据源**对话框中配置参数,并单击**完成**。详情请参见<mark>配置MaxCompute数据源</mark>。
- vii. 新增OTS数据源,详情请参见配置OTS数据源。
- 5. 配置MaxCompute (ODPS) Reader和表格存储 (OTS) Writer。
 - i.
 - ii.
 - iii.
 - .
 - iv.
 - v.

vi. 修改JSON代码后,单击⊙图标。

代码如下。

```
{
    "type": "job",
    "steps": [
      {
           "stepType": "odps",
           "parameter": {
              "partition": [],
              "datasource": "odps_first",
              "column": [
                 "name",
                 "id",
                 "gender"
              ],
              "table": "Transs"
           },
           "name": "Reader",
           "category": "reader"
       },
       {
           "stepType": "ots",
           "parameter": {
              "datasource": "Transs",
               "column": [
                {
                     "name": "Gender",
                     "type": "STRING"
                 }
               ],
               "writeMode": "UpdateRow",
               "table": "Trans",
               "primaryKey": [
                 {
                     "name": "Name",
                     "type": "STRING"
                 },
                  {
                     "name": "ID",
                    "type": "INT"
                  }
              ]
           },
           "name": "Writer",
           "category": "writer"
       }
    ],
    "version": "2.0",
    "order": {
      "hops": [
         {
              "from": "Reader",
              "to": "Writer"
          }
      ]
    },
    "setting": {
       "errorLimit": {
         "record": "0"
       },
       "speed": {
          "throttle": false,
           "concurrent": 1,
           "dmu": 1
       }
   }
}
```

6. 在表格存储控制台中查看新增的表数据。

i. 登录表格存储控制台。

```
ii. 在左侧导航栏上, 单击全部实例。
```

iii. 单击实例名称进入**实例管理**页面。在数据表列表区域,单击要查看的数据表名称。

iv. 单击顶部数据管理页签, 查看新增的表数据。

2.11. MaxCompute数据迁移至OSS

本文为您介绍如何使用DataWorks的数据同步功能将MaxCompute数据迁移至对象存储OSS(Object Storage Service)。

前提条件

- 开通MaxCompute和DataWorks。
- 开通DataWorks。
- 在DataWorks上完成创建业务流程,本例使用DataWorks简单模式。详情请参见创建业务流程。

操作步骤

- 1. 在DataWorks上创建表。
 - i. 登录DataWorks控制台。
 - ii. 在左侧导航栏,单击**工作空间列表**。
 - iii. 单击相应工作空间后的数据开发。
 - iv. 右键单击已创建的业务流程,选择新建 > MaxCompute > 表。
 - v. 在新建表页面,选择引擎类型并输入表名。
 - vi. 在表的编辑页面,单击DDL模式。
 - vii. 在DDL模式对话框,输入如下建表语句,单击生成表结构。

```
create table Transs
(name string,
id string,
gender string);
```

viii. 单击提交到生产环境。

- 2. 为表Transs导入数据。
 - i. 在数据开发页面,单击<mark>西</mark>图标。
 - ii. 在数据导入向导对话框,至少输入3个字母来搜索需要导入数据的表,单击下一步。
 - iii. 选择数据导入方式为上传本地数据,单击选择文件后的浏览...。选择本地数据文件,配置导入信息。

示例数据如下。

```
qwe,145,F
asd,256,F
xzc,345,M
rgth,234,F
ert,456,F
dfg,12,M
tyj,4,M
bfg,245,M
nrtjeryj,15,F
rwh,2344,M
trh,387,F
srjeyj,67,M
saerh,567,M
```

- iv. 单击下一步。
- v. 选择目标表字段与源字段的匹配方式。

```
vi. 单击导入数据。
```

- 3. 在OSS控制台上创建表。
 - i. 登录OSS控制台,创建Bucket。详情请参见创建存储空间。
 - ii. 上传文件*qwee.csv*至OSS。详情请参见上传文件。

⑦ 说明 请确保qwee.csv文件中的字段与表Transs的字段完全一致。

```
4. 在DataWorks上新增数据源。
```

```
i.
ⅲ.
ⅲ.
ⅳ. 在左侧导航栏上,单击数据源,进入数据源管理页面。
Ⅳ. 单击右上角新增数据源,并选择数据类型为ODPS。
```

```
vi. 在新增ODPS数据源对话框中配置参数,并单击完成。详情请参见配置MaxCompute数据源。
   vii. 新增OSS数据源,详情请参见配置OSS数据源。
5. 配置MaxCompute (ODPS) Reader和对象存储 (OSS) Writer。
    i.
    ii.
   iii.
   iv.
   v.
   vi. 修改JSON代码后,单击 图标。
      示例代码如下。
       {
           "order":{
               "hops":[
                {
                      "from":"Reader",
                      "to":"Writer"
                 }
              ]
           },
           "setting":{
              "errorLimit":{
                 "record":"0"
              },
               "speed":{
                  "concurrent":1,
                  "dmu":1,
                  "throttle":false
              }
           },
           "steps":[
              {
                  "category":"reader",
                  "name":"Reader",
                  "parameter":{
                      "column":[
                         "name",
                         "id",
                         "gender"
                      ],
                      "datasource":"odps_first",
                      "partition":[],
                      "table":"Transs"
                  },
                  "stepType":"odps"
               },
               {
                  "category":"writer",
                  "name":"Writer",
                  "parameter":{
                     "datasource":"Trans",
                      "dateFormat":"yyyy-MM-dd HH:mm:ss",
                      "encoding":"UTF-8",
                      "fieldDelimiter":",",
                     "fileFormat":"csv",
                      "nullFormat":"null",
                      "object":"qweee.csv",
                      "writeMode":"truncate"
                  },
                  "stepType":"oss"
               }
           1.
           "type":"job",
           "version":"2.0"
       }
```

6. 在OSS控制台中查看新增的表数据。详情请参见下载文件。

2.12. 迁移ECS自建MySQL数据库至MaxCompute

本文为您介绍如何使用独享数据集成资源,将您在ECS上自建的MySQL数据库中的数据,迁移到MaxCompute。

前提条件

● 已拥有至少一个绑定专有网络VPC的ECS(请勿使用经典网络),并在ECS上安装好MySQL数据库,数据库中已创建好用户和测试数据。本文中ECS自 建MySQL的测试数据创建语句如下。

```
CREATE TABLE IF NOT EXISTS good_sale(
  create_time timestamp,
  category varchar(20),
  brand varchar(20),
   buyer_id varchar(20),
  trans_num varchar(20),
  trans_amount DOUBLE,
  click_cnt varchar(20)
  );
insert into good_sale values('2018-08-21','coat','brandA','lilei',3,500.6,7),
('2018-08-22','food','brandB','lilei',1,303,8),
('2018-08-22','coat','brandC','hanmeimei',2,510,2),
('2018-08-22','bath','brandA','hanmeimei',1,442.5,1),
('2018-08-22','food','brandD','hanmeimei',2,234,3),
('2018-08-23','coat','brandB','jimmy',9,2000,7),
('2018-08-23','food','brandA','jimmy',5,45.1,5),
('2018-08-23','coat','brandE','jimmy',5,100.2,4),
('2018-08-24','food','brandG','peiqi',10,5560,7),
('2018-08-24', 'bath', 'brandF', 'peiqi', 1, 445.6, 2),
('2018-08-24','coat','brandA','ray',3,777,3),
('2018-08-24', 'bath', 'brandG', 'ray', 3, 122, 3),
('2018-08-24','coat','brandC','ray',1,62,7) ;
```

• 请记录好您的ECS的私有IP、专有网络和虚拟交换机信息。

三 (一)阿里云	
<	实例类型: I/O优化
实例 注情	操作系统: CentOS 7.2 64位
大学和学生	弹性网卡: eni-u 🚽 🕞
本文 向 世昭	公网IP: 196 L
本实际进行	弹性公网IP: - 📕
本实例操作记录	私有IP: 192.168.3.121 📕
本实例安全组	辅助私网IP: 管理辅助私网IP
本实例安全防护	带宽计费方式:按使用流量
	当前使用带宽: 5Mbps (峰值)
	专有网络: vpc-uf6h
	虚拟交换机: vsw-uf ulggvn4s L

• ECS上的安全组已放通MySQL数据库所使用的端口(默认为3306),详情请参见添加安全组规则,请记录好您的安全组名称。

☰ (-) 阿里云				
<	sg-uf652kdu	1u8wro	1m a Dw_	workshop / vpc-uf6ha
安全组规则	sg-uf652	3wrqmv		
安全组内实例列表	入方向 出方向			
安全组内弹性网卡	□ 授权策略 协	协议类型	端口范围	授权类型(全部) ▼
	□ 允许 自	自定义 TCP	1433/1433	IPv4地址段访问
	□ 允许 自	自定义 TCP	3306/3306	IPv4地址段访问

- 已成功创建DataWorks工作空间。本文使用DataWorks简单模式工作空间,计算引擎为MaxCompute。请保证您的ECS与DataWorks工作空间处于同一个地域,创建方法请参见创建工作空间。
- 已完成独享数据集成资源的购买,并且绑定了ECS所在的专有网络VPC。请注意独享资源组必须与ECS同一可用区,详情请参见新增和使用独享数据 集成资源组。完成绑定后,您可以在资源组列表查看到您的独享资源组。

	(上海)	-			Q 搜索文档、控制	制台、API、解决:	方案和资源	费用 工单	备案 企	业支持	官网	۶	۵. .	≓' ⑦
DataWorks	D	DataWorks / 资源组列表												产品动态
概览		独享资源组 公共资源组	自定义资源组											
工作空间列表		新增独享资源组 请输入搜索	e键词 Q											
資源組列表 1 计算引擎列表 へ		资源组名称	备注	类型	状态	到期时间	资源组数	资源组使用	操作					
MaxCompute				调度资源组	✓ 运行 中	2020-03- 18	1	0	查看信息 共 修改归属工(广容 缩容 ; 作空间	奏费 专有)	网络绑定	运维助手	beta
Graph Compute 交互式分析		(Min. Statulit. or	Table at a field of a	调度资源组	● 已释 放				扩容 缩容	重新购买	有网络绑	定 修改	日雇工作的	2(8)
	<	1000,000	100.00	數据集成资源 组	① 已释 放				扩容 缩容	重新购买	行网络绑	定 修改!		20
		xiangcui_vpc	向翠	数据集成资源 组	✓ 运行 中	2020-03- 18	1	0	查看信息 技	广容 缩容 结	奏要 专有)	网络绑定	修改归属	工作空间

● 在专有网络绑定处查看专有网络、交换机和安全组信息是否和ECS一致。

Data	Works / 资源组列表						
	返回						
ŝ	资源组: xiangcui_	vpc	\sim				
	资源组名称	类型	可用区	专有网络	交换机	安全组	状态
	xiangcui_vpc	数据集成资源组	cn-shanghai-f	an official characteries		a - Nebalitica	已绑定
	xiangcui_vpc	数据集成资源组	cn-shanghai-f	vpc-u hyq8b	vsw-u /n4s	sg-u rqmv	已绑定

背景信息

独享资源可以保障您的数据快速、稳定地传输。您购买的独享数据集成资源和需要访问的数据源(即本文中的ECS自建MySQL数据库)必须在同地域同可用区,且和Dat aWorks工作空间同地域。

操作步骤

- 1. 在DataWorks上创建MySQL数据源。
 - i. 使用主账号登录DataWorks控制台。

ii. 在工作空间列表单击进入数据集成。

	(上海) ▼			Q 搜索文档、控制台、API、解决方案环	回過源 奏用 工单	备案 企业 支持	寺 宮岡 돈	Ŭ. Å _、 ©	简体
DataWorks	DataWorks / 工作空间列表	1						产品动态	4 帮助文档
概览		版本到期日为 2022年12月5日				医太舟纲 医太阳	「明 帝要版本注册	的开始喜欢海纳	的可能拥有
工作空间列表	- HUISCHEDSZE SEMENK , /	W44398079 20224-129950 .				NR44-713R NR44-X	196 III.48/3X/441+10	299-2010-00043E	ARCHINES
资源组列表	创建工作空间 请	输入工作空间/显示名	Q						С
计算引率列表 ^	工作空间名称/显示名	模式	创建时间	管理员	状态	开通服务	操作		
MaxCompute	shuai_test shuai_test ⊡	简单模式 单环境	2020-02-28 (16:37:4	2) dataworks	✔正常	~	工作空间配置 进入数据集成	开通服务配置 进入 进入数据服务 更多	数据开发

- iii. 在左侧导航栏,单击**数据源**。
- ⅳ. 单击数据源管理页面右上角的新增数据源。
- v. 在新增数据源页面,单击MySQL。
- vi. 在**新增MySQL数据源**对话框中,配置各项参数,详情请参见配置MySQL数据源。
 - 本文以**连接串模式**为例,在**JDBC URL**处输入您刚刚记录的ECS私有地址和MySQL的默认端口号3306。

新增MySQL数据源						\times			
* 数据源类型:	🔵 阿里云实例模式 💽	连接串模式							
* 数据源名称:	自定义名称								
数据源描述:									
* JDBC URL :	jdbc:mysql://ServerIP:Por	dbc:mysql://ServerIP:Port/Database							
* 用户名 :									
* 密码 :									
资源组连通性:	资源组名称	类型	连通状态	测试时间	操作	?			
	公共资源	公共资源组 未测试 测试连通性							
注意:如果测试不通,可能的原因为: 数据库没有启动,请确认已经正常启动。 DataWorks无法访问数据库所在网络,请确保网络已和阿里云打通。 DataWorks被数据库所在网络防火墙禁止,请添加<mark>白名单。</mark> 数据库域名无法被正确解析,请确认域名可以被正常解析访问。 									
					上一步	▲完成			

⑦ 说明 当前VPC环境下的自建MySQL数据源暂不支持测试连通性,因此连通性测试失败是正常现象。

vii. 单击相应资源组后的测试连通性。

数据同步时,一个任务只能使用一种资源组。您需要在每种资源组上单独测试连通性,以保证同步任务使用的数据集成资源组能够与数据源 连通,否则将无法正常执行数据同步任务。详情请参见<mark>配置资源组与网络连通</mark>。

viii. 测试连通性通过后,单击**完成**。

2. 创建MaxCompute表。

您需要通过DataWorks创建一个表,用于接收来自MySQL的测试数据。

- i. 单击左上角的■图标,选择全部产品 > DataStudio (数据开发)。
- ii. 新建一个**业务流程**,详情请参见创建业务流程。

iii. 右键单击新建的业务流程,选择新建 > MaxCompute > 表。



iv. 输入您的MaxCompute表名称,本例中使用和MySQL数据库表一样的名称good_sale。单击DDL模式后,输入您的建表语句并生成表结构。





CREATE TABLE IF NOT EXISTS good_sale(create_time string, category STRING, buyer_id STRING, trans_num BIGINT, trans_amount DOUBLE, click_cnt BIGINT); v. 输入表的中文名后,单击提交到生产环境,完成MaxCompute表good_sale的创建。

DDL模式 从生产环境加载	提交到生产环境	ā
	表名	good_sale
Max	shuai_test	
写,	mysqltest	
基本属性		
中文名	good_sale	
一级主题	请选择	

- 3. 配置数据集成任务。
 - i. 右键单击业务流程,选择新建 > 数据集成 > 离线同步,创建一个数据集成任务。

 ▲ mysqltest 参 数 新建 ≫ 数 新建 > 数据集成 > 函 新建 > 数据集成 > 函 新建 > 数据服务 > 函 添加到解決方案 通 印 → 函 自 打开解決方案 	✔ 业务流程				
 > → 数 新建 > 数据集成 > 高线同步 > № M: 看板 MaxCompute> > ⊗ 数 修改属性 数据服务 > > ⊙ 通 添加到解決方案 通用 > > 圖 自 打开解決方案 自定义 > 	🗸 🚠 mysqltes	it			
▶ № Mi 看板 MaxCompute> ▶ 圖 核心屋性 数据服务 > ▶ 圖 添加到解決方案 通用 > ▶ 圖 自 打开解決方案 自定义 >	> 😑 数	新建 >	数据集成	>	离线同步
 ✓ Solution (Section 2) ✓ Solution (Sec	> 🚺 Ma	看板	MaxComput	e≻	
 ✓ O 通 添加到解決方案 → 通用 →	✔ 🮯 数	修改属性	数据服务	>	
→	✓ [○] 通	添加到解决方案	通用	>	
	> 🔡 自		自定义	>	
		删除			

ii. 选择您的数据来源为您刚添加的MySQl数据源,数据去向为默认MaxCompute数据源odps_first,单击**转换脚本**切换数据集成任务为脚本模 式。

此时,如果产生报错或您无法选择数据来源的表,都属于正常现象,直接转换为脚本模式即可。

DI MySQL2MaxCompute	• Sines		10.000					
	ि 🛛 🕄	Ø						
		在这里配置数据的来源	就和写入端;可	可以是默认的数据源,也可以	以是您创建的自有	有数据源查看支持的数据来源		
01 选择数据源		数据来源				数据	祛向	
* 数据源	MySQL	∽ shuai		0	* 数据源	ODPS V	odps_first ~	0
*表	请选择				生产项目名:	shuai_test		
						请选择		
数据过滤	请参考相应SQL语法填 键字) 该过滤语句诵	写where过滤语句(不要 1100日に増量同步	要填写where关	?	清理规则	写入前清理已有数据 (Insert	Overwrite) V	
	NEJ / & KAZIMUNA SAS			空	字符串作为null	● 是 🧿 否		
切分键	根据配置的字段进行数	如据分片,实现并发读取		?				
		数据预览						

iii. 单击页面右侧的数据集成资源组配置,选中已购买的独享资源组。
 如果未切换任务资源组为数据集成独享资源,后续您的任务将无法成功运行。
iv. 填写数据集成任务脚本内容如下。

```
{
   "type": "job",
   "steps": [
      {
          "stepType": "mysql",
          "parameter": {
             "column": [//源列名
                 "create_time",
                 "category",
                 "brand",
                 "buyer_id",
                  "trans_num",
                  "trans_amount",
                  "click_cnt"
              ],
              "connection": [
                 {
"datasource": "shuai",//源数据源
                     "table": [
                       "good_sale"//源数据库表名,此处必须为方括号数组格式。
                     1
                 }
             ],
              "where": "",
              "splitPk": "",
              "encoding": "UTF-8"
           },
           "name": "Reader",
           "category": "reader"
       },
       {
           "stepType": "odps",
           "parameter": {
             "partition": "",
             "truncate": true,
             "datasource": "odps_first",//目标数据源
              "column": [//目标列名
                 "create_time",
                 "category",
                 "brand",
                 "buyer_id",
                 "trans_num",
                 "trans_amount",
                  "click_cnt"
             1,
              "emptyAsNull": false,
              "table": "good_sale"//目标表名
          },
          "name": "Writer",
          "category": "writer"
      }
   ],
   "version": "2.0",
   "order": {
      "hops": [
         {
             "from": "Reader",
             "to": "Writer"
          }
      ]
   },
   "setting": {
      "errorLimit": {
         "record": "0"
      },
      "speed": {
         "throttle": false,
          "concurrent": 2
      }
   }
}
```

Di MySQL2MaxCompute ×				10 per ette
🖱 🖸 🖸 🐻		8		
43	"trans_amount",			
44	"click_cnt"			
45				
46 em	ptyAsNull": false	,		
	DIE : good_sale			
40 j), 40 "pamo":	"Writer"			
	rv". "writer"			
51 }	iy i writter			
52 1.				
53 "version": "2.0	".			
54 "order": {				
55 "hops": [
56 {				
57 "fr	om": "Reader",			
运行日志				
2020-02-29 16:53:16.432 [job-22 2020-02-29 16:53:16.433 [job-22 0.000s All Task WaitReader 2020-02-29 16:53:16.434 [job-22 2020-02-29 16:53:16.434 [job-22	26478048] INFO Metri 26478048] INFO Local Fime 0.000s Percent 26478048] INFO LogRe 26478048] INFO JobCo	CReportUtil JobContaine age 100.00% portUtil - ontainer -	reportJobMetri rCommunicator - T ; report datax log	c is turn off otal 11 records, is turn off
任务启动时刻	: 2020-02-29 16:52:	53		
仕分结束町刻	: 2020-02-29 16:53:	16		
任务平均流量	: 158	2/5		
记录写入速度	: 0rec	:/s		
读出记录总数	:	11		
读写失败总数				
2020-02-29 16:53:16 INFO ======				

v. 单击运行, 您可以在下方的运行日志查看数据是否已传输到MaxCompute。

执行结果

您可以新建一个ODQP SQL类型的节点,用于查询当前MaxCompute表中的数据。



输入您的查询语句 select * from good_sale ; , 单击运行,即可看到当前已传入MaxCompute表中的数据。

Image: Second	Di MyS	QL2Ma	xComput	e So	quer	у	×				-		Ŧ		with				
1 odps sql author: author: 4 create time: 2020-02-29 14:12:21 5 creater time: 2020-02-29 14:12:21 6 select * from good_sale ; Select * from good_sale ; Image: Select * from good_sale ; Select * fro		₿	ᡗ	ه (۱		\$	\odot												
A B C D E F G 1 create_time category brand buyer_id trans_num trans_amount click_cnt 1 create_time category brand buyer_id trans_num trans_amount click_cnt 1 create_time category brandC hanmeimei 2 510.0 2 1 create_time category brandE jimmy 9 2000.0 7 1 create_time category brandE jimmy 5 510.0 2 1 f 2018-08-22 00:0:00 coat brandE jimmy 5 000.0 7 1 f 2018-08-23 00:0:00 coat brandE jimmy 5 100.2 4 1 2018-08-23 00:0:00 coat brandG peiqi 10 100.2 4 1 2018-08-24 00:0:00 food brandF peiqi 10 45.6 2 1 10 2018-08-24 00:0:00 brandA ray		od ** au cr **	ps sql ****** thor: eate t: ******* ct * fi	******* ime:202 ******* rom goc	***** 20-02 ***** od_sa	2-29 1 2*****	***** 4:12: ****	21 ****											
$ \begin{array}{c c c c c c c c c c c c c c c c c c c $	运行	志		结果[1]	;	×													
1 create_time category brand buyer_id trans_num trans_amount click_cnt 4 2018-08-22 00:00:0 coat brandC hanmeimei 2 510.0 2 5 2018-08-22 00:00:0 coat brandB jimmy 9 2000.0 7 6 2018-08-23 00:00:0 food brandB jimmy 5 45.1 5 7 2018-08-23 00:00:0 coat brandE jimmy 5 100.2 4 6 2018-08-23 00:00:0 coat brandE jimmy 5 100.2 4 6 2018-08-23 00:00:0 coat brandE jimmy 5 100.2 4 6 2018-08-24 00:00:0 food brandG peiqi 10 5560.0 7 7 2018-08-24 00:00:00 bath brandF peiqi 1 45.6 2 10 2018-08-24 00:00:00 coat brandA ray 3 777.0	m			Α			В			С		D		E		F			
4 2018-08-22 00:0:00 coat brandC hanmeimei 2 510.0 2 5 2018-08-22 00:0:00 coat brandB jimmy 9 2000.0 7 6 2018-08-23 00:0:00 food brandA jimmy 5 45.1 5 7 2018-08-23 00:0:00 coat brandE jimmy 5 100.2 4 8 2018-08-23 00:0:00 coat brandE jimmy 5 100.2 4 8 2018-08-24 00:0:00 food brandF peiqi 10 5560.0 7 9 2018-08-24 00:0:00 bath brandF peiqi 1 445.6 2 10 2018-08-24 00:0:00 coat brandA ray 3 777.0 3		1	create_tir	ne	✓ ca	ategory		~	brand		✓ buyer.	_id	~	trans_num	 trans 	_amount	~	click_cnt	
5 2018/08/23 00:000 coat brandB jimmy 9 2000.0 7 6 2018/08/23 00:000 food brandA jimmy 5 45.1 5 7 2018/08/23 00:000 coat brandA jimmy 5 100.2 4 8 2018/08/24 00:000 food brandG peiqi 10 5560.0 7 9 2018/08/24 00:000 food brandF peiqi 1 445.6 2 10 2018/08/24 00:000 coat brandF peiqi 1 777.0 3	<u>_111</u>	4	2018-08-:	22 00:00:0	00 co	bat			brandC		hanm	eimei		2	510.0	0		2	
6 2018/08/23 00:000 food brandA jimmy 5 45.1 5 7 2018/08/23 00:000 coat brandE jimmy 5 100.2 4 8 2018/08/24 00:000 food brandG peiqi 10 5560.0 7 9 2018/08/24 00:000 bath brandF peiqi 1 445.6 2 10 2018/08/24 00:000 coat brandA ray 3 777.0 3	la.	5	2018-08-:	23 00:00:0	00 co	bat			brandB		jimmy			9	2000	.0		7	
7 2018/08/23 00:000 coat brandE jimmy 5 100.2 4 8 2018/08/24 00:000 food brandG peiqi 10 5560.0 7 9 2018/08/24 00:000 bath brandF peiqi 1 445.6 2 10 2018/08/24 00:000 coat brandF peiqi 1 560.0 7		6	2018-08-3	23 00:00:0	00 fo	bod			brandA		jimmy			5	45.1			5	
8 2018/08/24 00:00:00 food brandG peiqi 10 5560.0 7 9 2018/08/24 00:00:00 bath brandF peiqi 1 445.6 2 10 2018/08/24 00:00:00 coat brandF peiqi 1 445.6 2 10 2018/08/24 00:00:00 coat brandA ray 3 777.0 3	~	7	2018-08-3	23 00:00:0	00 co	bat			brandE		jimmy			5	100.3	2		4	
9 2018-08-24 00:00:00 bath brandi- peiqi 1 445.6 2 10 2018-08-24 00:00:00 coat brandi- ray 3 777.0 3		8	2018-08-3	24 00:00:0	00 fo	od			brandG		peiqi			10	5560	1.0 -		7	
10 2018-08-24 00 00:00 Coat BrandA ray 3 ////.0 3	<u>///</u>	9	2018-08-2	24 00:00:0	00 ba	ath			brandF		peiqi			1	445.0	b		2	
11 2019 00 24 00:00 both brandC ray 2 122 0 2		10	2018-08-2	24 00:00:0 24 00:00-0	00 CO	oat			brandA brandC		ray			3	177.0	u n		ა ა	

2.13. Amazon Redshift数据迁移至MaxCompute

本文为您介绍如何通过公网环境将Amazon Redshift数据迁移至MaxCompute。

前提条件

• 准备Amazon Redshift集群环境及数据环境。

您可以登录AWS官网,获取创建Redshift集群的详细操作内容,详情请参见Amazon Redshift集群管理指南。

i. 创建Redshift集群。如果已有Redshift集群,您可以直接使用已有的Redshift集群。

e,	aws 服务 → 资源组 → ★			۵	▼ 首尔 ▼ 支持 ▼
≡	Amazon Redshift > 集群				
::	集群			C 空泡業群	操作 🔻 创建集群
控制面板	Q 搜察		所有状态		< 1 > @
。 集群	集群 ▲ 状态	▽ 已用存储容量 ▽ CPU 利用率	▽ 快照 ▽ 通知	标签	∇
入 査询			无集群		
_		创建 Am	azon Redshift 集群		
》 。 编辑路			创建集群		

ii. 在Redshift集群中准备好需要迁移的Amazon Redshift数据。

假设,已在public schema中准备好了TPC-H数据集。数据集使用MaxCompute 2.0数据类型和Decimal 2.0数据类型。

● 准备MaxCompute的项目环境

操作详情请参见<mark>准备工作</mark>。

以**新加坡(新加坡)**区域为例,创建作为迁移目标的MaxCompute项目。由于TPC-H数据集使用MaxCompute 2.0数据类型和Decimal 2.0数据类型,因此本文创建的项目为MaxCompute 2.0版本。

	〒西亚 (▼			Q 搜索文档、控制台、AI	PI、解决方案和资源	费用	工単	留案	企业	支持	官网	>_	۵.	Ä	0	简体	0
DataWorks	DataWorks / 工作空间	列表												7	"品动态	5 #R	文档
瓶克 工作空间列表	当前使用的是 标准服	。. 版本到期日为 2020年9月24日 。							版本升级	版本现	朝童	服本详情	書 购	灭独享资	源组	购买资源	2
资源组列表	创建工作空间	请输入工作空间/显示名	Q														C
报警资源	工作空间名称/显示名	模式	创建时间	管理员	状	态		开通服务	5		操作						
计算引擎列表 へ MaxCompute	Jakarta Jakarta	简单模式 单环境	2020年8月21日 16:42:26		~	"正常		Ŵ			进入首) 进入运行 工作空(页 进入数 律中心 进 间配置 修	対据集点 主入数排 ■改服∮	成 进入数 属地图 进 時配置 夏	対展开发 主入数据 ■多	服务	
Graph Compute														共1]	₫ <	1	>

● 开通阿里云OSS服务。

开通阿里云OSS服务详情请参见开通OSS服务。

背景信息

将Amazon Redshift数据迁移至MaxCompute的流程如下。



序号	描述
0	将Amazon Redshift数据导出至Amazon S3数据湖(简称S3)。
2	通过对象存储服务OSS的在线迁移上云服务,将数据从S3迁移至OSS。
3	将数据从OSS迁移至同区域的MaxCompute项目中,并校验数据完整性和正确性。

步骤一:将Amazon Redshift数据导出至S3

Amazon Redshift支持IAM角色和临时安全凭证(AccessKey)认证方式。您可以基于这两种认证方式通过Redshift UNLOAD命令将数据导出至S3。将 Amazon Redshift数据导出至S3的详细操作内容请参见<mark>卸载数据</mark>。

两种认证方式的UNLOAD命令格式如下:

● 基于IAM角色的UNLOAD命令

```
-- 通过UNLOAD命令将表customer的内容导出至S3。
```

```
UNLOAD ('SELECT * FROM customer')
TO 's3://bucket_name/unload_from_redshift/customer/customer_' --S3 Bucket。
IAM_ROLE 'arn:aws:iam::****:role/MyRedshiftRole'; --角色ARN。
```

• 基于AccessKey的UNLOAD命令

```
-- 通过UNLOAD命令将表customer的内容导出至S3。
UNLOAD ('SELECT * FROM customer')
TO 's3://bucket_name/unload_from_redshift/customer/customer_' --S3 Bucket。
Access_Key_id '<access-key-id>' --IAM用户的Access Key ID。
Secret_Access_Key '<secret-access-key>' --IAM用户的Access Key Secret。
Session_Token '<temporary-token>'; --IAM用户的临时访问令牌。
```

UNLOAD命令导出的数据格式如下:

● 默认格式

命令示例如下。

UNLOAD ('SELECT * FROM customer') TO 's3://bucket_name/unload_from_redshift/customer/customer_' IAM ROLE 'arn:aws:iam::****:role/redshift_s3_role';

执行成功后,导出以竖线())分隔的文本文件。您可以登录S3控制台,在对应Bucket中查看导出的文本文件。

Amazon S3 >bucket > unload_from_redshift					
-bucket					
概述					
Q 键入前缀并按输入来搜索。按 ESC 可清除。					
▲ 上传 + 创建文件夹 下载 操作 ✓				亚太区域(新加坡)	C
				正在查看1到4	
□ 名称▼	上次修改时间 🗸	大小▼	存储类别▼		
Customer_0000_part_00	-	-	-		
Customer_0001_part_00	-	-			
Customer_0002_part_00	-	-	-		
Customer_0003_part_00					
				正在查看1到4	

导出的文本文件格式如下。

- 4ICustomer#000000004I
- 4
 4(Customer#000000004))
 al accoul

 5
 5(Customer#000000005))
 ave to unwind, foxes cajole accord
- 6 6lCustomer#00000006l ons. even deposits boost according to the slyly bold packages. final accounts cajole requests
- mic, express theodolites. express, even pinto beans among the expl 7ICustomer#0000000071 8 8lCustomer#00000008l gt b slyly regular theodolites kindle blithely courts. carefully even
- theodolites haggle slyly a
- 9 (Customer#00000009):
 10 (Customer#000000009):
 10 (Customer#00000000010
 r deposits haggle. furl
- 11 11/Customer#000000011 Slyty, quickly even pinto beans promise above the slyty regular pinto
- beans I
- hely regular requests nag, ironic theodolites boost quickly alongi 12 12lCustomer#000000012 after the close frays. carefully bold notornis use ironic requests. blithelyl
- 13
 13ICustomer#00000013

 14
 14ICustomer#000000014
- 15 15lCustomer#000000015 DI platelets. regular deposits detect asymptotes. blithely unusual packages nag slyly at the flufl

• PARQUET格式

以PARQUET格式导出,便于其它引擎直接读取数据。命令示例如下。

UNLOAD ('SELECT * FROM customer')
TO 's3://bucket_name/unload_from_redshift/customer_parquet/customer_'
FORMAT AS PARQUET
<pre>IAM_ROLE 'arn:aws:iam::xxxx:role/redshift_s3_role';</pre>

执行成功后,您可以在对应Bucket中查看导出的文件。PARQUET文件比文本文件更小,数据压缩率更高。

Amazon S3 >bucket > unload_from_redshift > customer_parquet					
-bucket					
概述					
Q, 健入前缀并按输入来搜索。按 ESC 可满除。					
▲ 上线 ◆ 前建文件夹 下级 提作 >				亚太区域(新加坡)	c
				正在查看1到1	
□ 名称▼	上次修改时间 ▼	大小▼	存储类别▼		
customer_0002_part_00.parquet	6月 27, 2020 7:20:54 下午 GMT+0800	1.3 MB	标准		
				正在查看1到1	

本文以IAM角色认证及导出PARQUET格式为例介绍数据迁移操作。

1. 新建Redshift类型的IAM角色。

i. 登录IAM控制台,单击创建角色。

aws 服务 女凝组	* *	↓ ◆妹 • 支持 •
Identity and Access Management (IAM)	角色	
控制周載 ◆ 防時問題 但 用 角 一 角 一 角 一 角 一 角 一 角 一 角 一 角 一 一	什么显 IAM 角色? IAM 角色是感觉做住的实体技艺们影响安全方法。以下为实体示例: 单 植物产中容 IAM 用户 - 在需要方 AVG 普通小式指针的 EC2 实践上运行的应用图单代码 - 小電量常过的炉中中空的漂通行操作以组织域加加的 系统 - 公司首求中华就会是句句 SAML—足迹能的用户 IAM 角色派的思想, 使其法为置安全的接下访问问题的方法。 Mm 角色派的思想 - Mm 香色素的思想	×
访问分析器 存档规则 分析器	 ・ W11 HELX目 ・ W11 HELX目 ・ 常见角色方面 	
先证报告	ERFAC BRARC	2 0

ii. 在创建角色页面的选择一个使用案例区域,单击Redshift。在选择您的使用案例区域单击Redshift-Customizable后,单击下一步:权限。

选择一个使用案例				
常见使用案例 EC2 Allows EC2 instances to ca	all AWS services on your be	ehalf.		
Lambda Allows Lambda functions t	o call AWS services on you	r behalf.		
Allows Lambda functions t 或者选择一个服务以查看 API Gateway AWS Backup AWS Backup AWS Chatbot AWS Support Amplify AppStream 2.0 AppSync Application Discovery Service Batch Chime CloudFormation CloudFormation CloudFormation CloudFormation CloudHSM CloudFormation CloudHSM CloudWatch Events CodeBuild CodeDeploy 选择您的'使用案例	o call AWS services on you	r behalf. ElastiCache Elastic Beanstalk Elastic Container Service Elastic Container Service Elastic Transcoder ElasticLoadBalancing Forecast GameLift Global Accelerator Glue Greengrass GuardDuty Health Organizational View IAM Access Analyzer Inspector IoT IoT SiteWise IoT Things Graph KMS	Kinesis Lake Formation Lambda Lex License Manager Machine Learning Macie Managed Blockchain MediaConvert Migration Hub OpsWorks Personalize Purchase Orders OLDB RAM RDS Redshift Rekognition	RoboMaker S3 SMS SNS SWF SageMaker Security Hub Service Catalog Step Functions Storage Gateway Systems Manager Textract Transfer Trusted Advisor VPC WorkLink WorkMail
Redshift - Scheduler Allow Redshift Scheduler t	to call Redshift on your beh	alf.		
*必填				取消 下一步: 权限

2. 添加读写S3的权限策略。在创建角色页面的Attach权限策略区域,输入S3,选中AmazonS3FullAccess,单击下一步:标签。

创建角色	(1) (2) (3) (4)
▼ Attach 权限策略	
选择一个或多个要附加到新角色的策略。	
创建策略	C
筛选策略 → Q <mark>S3</mark>	显示 4 个结果
策略名称 ▼	用作
AmazonDMSRedshiftS3Role	\mathcal{F}
AmazonS3FullAccess	Permissions policy (1)
AmazonS3ReadOnlyAccess	Permissions policy (1)
QuickSightAccessForS3StorageMar	nagementAnalyticsReadOnly 无

3. 为IAM角色命名并完成IAM角色创建。

i. 单击下一步: 审核, 在创建角色页面的审核区域, 配置角色名称和角色描述, 单击创建角色, 完成IAM角色创建。

创建角色		1 2 3 4
审核		
在创建此角色之前在下面提供必	需的信息并审	核此角色。
	角色名称*	redshift_s3_role
		请使用字母数字和'+=,-@'字符。 最长 64 个字符。
	角色描述	Allows Redshift clusters to call AWS services on your behalf.
		最长 1000 个字符。请使用字母数字和'+=,.@'字符。
D	信任的实体	AWS 服务: redshift.amazonaws.com
	策略	🖡 AmazonS3FullAccess 🕜
	权限边界	未设置权限边界

ii. 返回IAM控制台,在搜索框输入redshift_s3_role,单击redshift_s3_role角色名称,获取并记录角色ARN。

执行UNLOAD命令迁移数据时会使用角色ARN访问S3。

› redshift_s3_role 要	
角色 ARN 角色描述 实例配置文件 ARN	am.aws.iam::::::::::::::::::::::::::::::::::::
路径 创建时间	/ 2020-06-27 18:45 UTC+0800
上一次活动 最大会话持续时间	2020-06-27 23:19 UTC+0800 (41 天前) 1 小时编辑
限 信任关系 标签 访问顾问 撤消会话	
策略名称 ▼	第踏类型 ▼
AmazonS3FullAccess	AWS 托管策略

- 4. 为Redshift集群添加创建好的IAM角色,获取访问S3的权限。
 - i. 登录Amazon Redshift控制台,在右上角选择区域为**亚太地区(新加坡)**。
 - ii. 在左侧导航栏,单击集群,选中已创建好的Redshift集群,在操作下拉列表选择管理IAM角色。
 - iii. 在管理IAM角色页面,单击搜索框右侧的▼图标,选择redshift_s3_role。单击添加IAM角色 > 完成,将具备S3访问权限 的redshift_s3_role添加至Redshift集群。
- 5. 将Amazon Redshift数据导出至S3。
 - i. 返回Amazon Redshift控制台。

ii. 在左侧导航栏,单击编辑器,执行UNLOAD命令将Amazon Redshift数据以PARQUET格式导出至S3的Bucket目录。 命令示例如下。

UNLOAD ('SELECT * FROM customer') TO 's3://bucket_name/unload_from_redshift/customer_parquet/customer_' FORMAT AS PARQUET IAM_ROLE 'arn:aws:iam::xxxx:role/redshift_s3_role'; ______UNLOAD ('SELECT * FROM orders') TO 's3://bucket_name/unload_from_redshift/orders_parquet/orders_' FORMAT AS PARQUET IAM_ROLE 'arn:aws:iam::xxxx:role/redshift_s3_role'; UNLOAD ('SELECT * FROM lineitem') TO 's3://bucket_name/unload_from_redshift/lineitem_parquet/lineitem_' FORMAT AS PARQUET IAM_ROLE 'arn:aws:iam::xxxx:role/redshift_s3_role'; UNLOAD ('SELECT * FROM nation') TO 's3://bucket_name/unload_from_redshift/nation_parquet/nation_' FORMAT AS PARQUET IAM_ROLE 'arn:aws:iam::xxxx:role/redshift_s3_role'; UNLOAD ('SELECT * FROM part') TO 's3://bucket_name/unload_from_redshift/part_parquet/part_' FORMAT AS PARQUET IAM_ROLE 'arn:aws:iam::xxxx:role/redshift_s3_role'; UNLOAD ('SELECT * FROM partsupp') TO 's3://bucket_name/unload_from_redshift/partsupp_parquet/partsupp_' FORMAT AS PARQUET IAM_ROLE 'arn:aws:iam::xxxx:role/redshift_s3_role'; UNLOAD ('SELECT * FROM region') TO 's3://bucket_name/unload_from_redshift/region_parquet/region_' FORMAT AS PAROUET IAM_ROLE 'arn:aws:iam::xxxx:role/redshift_s3_role'; UNLOAD ('SELECT * FROM supplier') TO 's3://bucket_name/unload_from_redshift/supplier_parquet/supplier_' FORMAT AS PAROUET IAM_ROLE 'arn:aws:iam::xxxx:role/redshift_s3_role';

⑦ 说明 编辑器支持一次提交多条UNLOAD命令。

iii. 登录S3控制台,在S3的Bucket目录下检查导出的数据。

格式为符合预期的PARQUET格式。

Amazon S3 > unload_from_redshift				
-bucket				
2012 2017				
Q 還入筋漿并按輸入来搜索。按 ESC 可清除。				
▲ 上传 + (回起文件表 下版] 超作 →				亚太区域(新加坡) 2
				正在查看 1 到 8
□ 名称▼	上次修改时间 ▼	大小▼	存储类別▼	
customer_parquet	-	-		
ineitem_parquet				
nation_parquet	-	-	-	
C b orders_parquet			-	
Depart_parquet	**			
Departsupp_parquet	**			
egion_parquet				
supplier_parquet	-			

步骤二:将导出至S3的数据迁移至对象存储服务OSS

MaxCompute支持通过OSS的**在线迁移上云服务**,将S3的数据迁移至OSS,详情请参见AWS S3迁移教程。在线迁移上云服务处于公测状态,您需要提 工单联系客服,并由在线服务团队开通后才可使用。

1. 登录OSS管理控制台,创建保存迁移数据的Bucket,详情请参见创建存储空间。

最佳实践·数据迁移

対象存储 / / 文件管理 () 通过 SDK 管理文件																
sy migrati										读写权限 私有	类型 标准存储	(本地冗余)	区域新加坡	创建时间 2020年8月1	10日 14:34	
概览		上传文件	新建目录	碎片管理	授权	北量操作 ∨	刷新							请输入文件名前缀匹配	Q	
文件管理	>		文件名						文件大小	存储类型		更新时间			操作	
权限管理	>		unload_from	m_redshift/											删除	
展研设署	>															

- 2. 创建RAM用户并授予相关权限。
 - i. 登录RAM访问控制台,创建RAM用户,详情请参见创建RAM用户。
 - ii. 选中新创建的用户登录名称,单击添加权限,为新创建的RAM用户授予AliyunOSSFullAccess(存储空间读写权限) 和AliyunMGWFullAccess(在线迁移权限),单击确定>完成。
 - iii. 在左侧导航栏,单击**概览**。在**概览**页面的**账号管理**区域,单击**用户登录地址**链接,使用新创建的RAM用户登录阿里云控制台。
- 3. 在AWS侧准备可编程及访问S3的IAM用户。
 - i. 登录53控制台。
 - ii. 在导出的数据目录上单击右键,选择**获取总大小**,获取迁移目录的数据大小和文件个数。

获取总大小									
Amazon S3 >	-bucket								
-bu	ıcket								
概述	1性 权限	管理	接入点						
Q 键入前缀并按	输入来搜索。按 ESC 可	「清除。							
土 上传 + 创	建文件夹 下载	操作 ~						亚太区域(新加坡)	c
								正在查看1到	2
□ 名称 ▼					上次修改时间 🔻	大小▼	存储类别 🔻		
🗌 👺 tpch_10	0m_data				-	-	-		
🔽 🗁 unload_	from_redshift				-	-	-		
	打开							正在宣看1到	2
	下载为								
	获取总大小								
	更改存储类								
	还原								

■ 获取迁移目录的数据大小和文件个数

Amazon S3 > bucket	获得尺寸 ×
概述 属性 权限 管理 接入点	选择: 0 个对象。1 个文件夹 总大小: 32.8 MB 对象总数: 18
Q 键入前缀并按输入来搜索。按 ESC 可清除。	■ unload_from_redshift/ 18 个对象 - 32.8 MB
▲ 上传 + 创建文件夹 下载 操作 ~	
□ 名称 ▼	
E tpch_100m_data	
unicad_from_redshift	

iii. 登录IAM用户控制台, 单击添加用户。

aws 📭 🤋	19201 → ►					金線・支持・
Identity and Access Management (IAM)						2 0
控制图板	Q 按用户名或访问密钥查找用户					显示 3 个结果
▼ 访问管理	□ 用户名 •	组	访问證明使用期限	密码使用期限	上一次活动	MFA
组		and the second second	10 M T		100	1000
用户		A CONTRACTOR OF A CONTRACTOR OFTA CONTRACTOR O	1 mm	100		1000
策略						100

iv. 在添加用户页面,配置用户名。在选择AWS访问类型区域,选中编程访问,单击下一步:权限。

添加用户 1 2	3 4 5
设置用户详细信息 您可以一次添加多个具有相同访问类型和权限的用户。了解更多	
用户名* Read_S3	
◎ 添加其他用户	
选择 AWS 访问类型	
选择这些用户将如何访问 AWS。在最后一步中提供访问密钥和自动生成的密码。 了解更多	
访问类型* 📝 編程访问 为 AWS API、CLI、SDK 和其他开发工具启用 访问密钥 ID 和 私有访问密钥	3
• ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲ ▲	

v. 在添加用户页面, 单击直接附加现有策略。在搜索框中输入S3, 选中AmazonS3ReadOnlyAccess策略, 单击下一步:标签。

添加用户		1 2 3 4 5
▼ 设置权限		
梁·将用户添加到组 从现有用户复制权限	直接附加现有策略	
创建策略		2
筛选策略 ↓ ♀ ♀ 53		显示 4 个结果
策略名称 ▼	类型	用作
AmazonDMSRedshiftS3Role	AWS 托管	无
AmazonS3FullAccess	AWS 托管	Permissions policy (1)
AmazonS3ReadOnlyAccess	AWS 托管	Permissions policy (1)
QuickSightAccessForS3StorageManagementAnalyticsRead	AWS 托管	无

vi. 单击下一步: 审核 > 创建用户,完成IAM用户创建并记录密钥信息。
 创建在线迁移任务时会使用该密钥信息。

気力の人	用户		1 2 3 4
۲	成功 您已成功创建了以下所示的用户。您可以通 管理控制台。这是最后一次这些凭证可供了	ē看和下载用户安全凭证。您还可以通过电子邮件 下载。不过,您可以随时创建新的凭证。	向用户发送说明来登录到 AWS
	具有 AWS 管理控制台访问权限的用户可在	E以下位置登录: https://signin.aws	.amazon.com/console
& 下载	ŧ.csv		
	用户	访问密钥 ID	私有访问密钥
	Read S3	and the second se	显示

4. 创建在线迁移数据地址。

i. 登录<mark>阿里云数据在线迁移控制台。</mark>在左侧导航栏,单击**数据地址**。

ii.	(可选)	如果未开通在线迁移服务,	请在弹出的对话框中单击 去申请 。	在 在线迁移公测申请 页面,	填写信息,并单击 提交 。
	(-1/22 /	和朱木// 适任场在9月777		庄庄 3 2 19 4 (3) 午前 (3) (3)	实马伯心, 八千田 足入 。

在线迁移公测申请	
* 联系人:	
* 联系电话:	
钉钉号:	
* 迁移数据源类型:	 ・ 线下数据(有专线或者VPN)) ECS数据 ・ 阿里NAS ○ 阿里OSS ● 亚马逊AWS ・
* 迁移目的端:	● 阿里云OSS ○ 阿里云 NAS ○ 线下 NAS (有VPN或者专线) 阿里云OSS
* 迁移类型:	○ 一次性迁移 ○ 定时同步 (仅支持NAS)
* 迁移数据量预估(GB):	
* 迁移数据文件数预估(个):	
* 是否跨region:	○是 ○否
备注:	
	submit

iii. 在**管理数据地址**页面,单击**创建数据地址**,配置数据源及目标地址相关参数,单击**确定**。参数详情请参见迁移实施。

■ 数据源

创建数据地址	(1)如需更多帮助请参考产品手册	×
数据地址可以作为迁移任务	的 [源地址] 或者 [目的地址],数据地址创建成功之后,您可以	
创建迁移任务		
数据类型	AWS S3 ~ ②如何获取S3数据地址的相关信息	
* 数据名称	s3-to-oss 9/63	
* Endpoint ⑦	http://s3.ap-southeast-1.amazonaws.com	
* Bucket	-bucket	
* Prefix 🕐	迁移全部数据 <mark>迁移部分数据</mark> unload_from_redshift/	
* Access Key Id 🕐	(***C#C#C#C#C#C#C#C#C#C#C#C#C#C#C#C#C#C#	
* Secret Access Key ⑦		
		B
	取消	角定

⑦ 说明 Access Key ID和Access Key Secret 为IAM用户的密钥信息。

■ 目标地址

<u> </u>	 如需更多帮助请参考产品手册 	×
数据地址可以作为迁移任务	各的 [源地址] 或者 [目的地址]。数据地址创建成功之后,您可以	
- 创建迁移仕务		
数据类型	OSS > ⑦加回茲取OSS教授他让於相关信息	
* 数据名称	oss_data_address 16/63	
* 数据所在区域	新加坡	
* OSS Endpoint	https://pss-ap-southeast-1 alivuncs.com	
	nigosroo gradulouar Luigunoscom	
* Access Key Id		
* Access Key Secret ?)	
* OSS Bucket	×	
OSS Prefix ?	unload_from_redshift/	
	请选择或输入迁移文件的prefix (不填代表迁移全部)	
	取消	角定

⑦ 说明 Access Key ID和Access Key Secret 为RAM用户的密钥信息。

5. 创建在线迁移任务。

i. 在左侧导航栏,单击**迁移任务**。

ii. 在迁移任务列表页面,单击创建迁移任务,配置相关信息后,单击创建。参数详情请参见迁移实施。

■ 任务配置

创建迁移任务			(1)如需更多帮助;	青参考产品手册	×
任务配置			性能调优		
迁移数据地址					
* 任务名称	s3-to-oss-job			13/63	
	如果无可用数据	(源/目的) 地址	L, 请您先创建数据i	也址	
* 源地址 ⑦	[s3] s3-to-oss			\sim	
	aws-sg-s3-buck	et:unload_from	_redshift/		
* 目的地址 ⑦	[oss] oss_data	address		\sim	
迁移策略	https://oss-ap-so migration1/unloa	utheast-1.aliyu d_from_redshif	ncs.com:sg- t/		
迁移方式 ②	全量迁移	增量迁移			
	全量数据迁移完 任务多次提交全	;成后任务将立即 量迁移,仅迁移	」停止,不再对增量数 多更新的数据	数据进行迁移。同	
多版本迁移	不使用	使用			
	多版本迁移会扫 目的地址。	描忽源站文件的	前有版本,并全部	(按顺序) 迁移到	
迁移文件起点时间 ⑦	迁移全部	指定时间			
文件覆盖方式	最后修改时间优	洗 条件覆盖	盖 全覆盖	不覆盖	
	对于同名文件, 1.如果源LastM 过。 2.如果源LastM 3.如果源LastM -若二者的Siz - 否则(Size,	优先判断二者的 Nodified > 目的I Nodified > 目的I Nodified == 目的 ze或Content-Type 、Content-Type	LastModified,即最 LastModified,则比 LastModified,则执 与LastModified,则执 pe有其一不相等,则 都相等),文件将被	新后修改时间。 一 文件将被执行跳 行覆盖。 线判断: 助行覆盖。 執行数过。	
				取消 下	-步

■ 性能调优

过移任务		(1)如需更多	多帮助请参考产品	手册
任务配置	<u></u>	性	能调优	
·据预估 为保障顺利完成迁移	任务,准确统计迁移进	挂度和成功率,请尽量准	确评估您的迁移存储	諸量
🍟 和迁移文件个数。 如	何评估迁移数据量			
待迁移存储量	33		MB	~
待迁移文件个数	18		个	\sim
<u>遥</u> 控制	0点 3点 6点	9点 12点 15	点 18点 21点	į 24
(每天)限流时间段	\mapsto 0	0-1		
最大流量(MB/s)	5		添加	
最大流量(MB/s) 开始	5 结束	隕流	添加	

⑦ 说明 待迁移存储量和待迁移文件个数是您通过S3控制台获取到的待迁移数据大小和文件个数。

iii. 创建的迁移任务会自动运行。请您确认**任务状态**为已完成,表示迁移任务成功结束。

数据迁移服务	迁移任务列表	迁移目标区域: 全部区域	~			②数据在线迁移用户	手册 自然证移任务	은 刷新
V 201120984								
 NetSLITIPOLY 	任务省称	迁穆方式	迁移目标区	城 源地址	目的地址	创建时间	任务状态	操作
闪电立方	s3-to-oss-iob	全最迁移	新加坡	s3-to-oss	oss_data_address	2020年8月10日 19:35	 已完成 	19月 1月時
~ 在线迁移服务								
迁移任务								< 1 >
数据地址								

iv. 在迁移任务的右侧单击管理,查看迁移任务报告,确认数据已经全部迁移成功。

数据迁移服务	く 迁移任务列表 / 任务报告 任务名	称: s3-to-oss-job 任务状态: • 已完成				重试 启动 停止	○周新
~ 离线迁移服务	迁移源站信息			迁移目标站信息			
内电立方 ~ 在线迁移服务 迁移任务 数编地址	散電振振 Endpoint Bucket Prefix AccessKayyid SacraAccessKay 特定任何指量 特定任何指量	s3 http://is3.ap-doutheast-1 amazonaws.com aws-s9-s3-bucket unload_from_redshift/ 		数据迁移至 Endpoint Bucket Prefix AccessKey/d SecretAccessKey	oss https://oss-ap-southeast-1.alyuncs.c sg-migration unload_from_redshift/	om	
	迁移策略			流量时间规划图			筆置
	是否迁称增盛数据 增量迁称间隔	KII -		909			
	增量迁移次数	-			0 0 0 0 0 0 0	0 0 0 0 0 0	0 0 0
	待迁移存储量			文件数量			
	总迁移存储量	32.84 MB	日光成汪修存编量 💿 得汪修存储量	总迁移文件个数	17	 日成功汪稼文件数 特汪稼文件数 	大败汪修个教
	已完成迁移存储量 迁移完成率	32.84 MB		已成功迁移文件数 失败迁移个数 待迁移文件数	17 0 0	100%	
				迁移完成座	100%		

v. 登录OSS管理控制台。

vi. 在左侧导航栏,单击Bucket列表。在Bucket列表区域单击创建的Bucket。在Bucket页面,单击**文件管理**,查看文件迁移结果。

对象存储 /	/ 文	件管理											⑦ 通过 SDI	K 管理文件
sp migra	ritor								读写权	限 私有	类型 标准存储 (本地冗余)	区域新加坡	创建时间 2020年8月1	10日 14:34
概览			上传文件	新建目录 碎片管	遭 授权	批量操作 🗸	刷新						请输入文件名前缀匹配	Q
文件管理	>			文件名				文件大小	7	略类型	更新时间			操作
权限管理	>		•	v unload_from_red	hift/									
基础设置	>	Γ	_	customer_parquet/										删除
冗余与容错	>		_	lineitem_parquet/										删除
传输管理	>		_	nation_parquet/										删除
日志管理	>			orders_parquet/										删除
数据处理	>		_	part_parquet/										删除
数据统计	>		_	partsupp_parquet/										删除
			_	region_parquet/										删除
				supplier_parquet/										# 🖂

步骤三: 将数据从OSS迁移至同区域的MaxCompute项目

您可以通过MaxCompute的LOAD命令将OSS数据迁移至同区域的MaxCompute项目中。

LOAD命令支持STS认证和AccessKey认证两种方式,AccessKey认证方式需要使用明文AccessKey ID和AccessKey Secret。STS认证方式不会暴露 AccessKey信息,具备高安全性。本文以STS认证方式为例介绍数据迁移操作。

1. 在DataWorks的临时查询界面或MaxCompute客户端(odpscmd),使用Redshift集群数据的DDL,创建与迁移数据相对应的表。

临时查询功能详情请参见使用临时查询运行SQL语句(可选)。命令示例如下。

CREATE TABLE customer(C_CustKey int , C Name varchar(64) . C Address varchar(64) , C_NationKey int , C_Phone varchar(64) , C_AcctBal decimal(13, 2) , C MktSegment varchar(64) , C_Comment varchar(120) , skip varchar(64)); CREATE TABLE lineitem(L_OrderKey int , L_PartKey int , L SuppKey int , L_LineNumber int , L_Quantity int , L_ExtendedPrice decimal(13, 2) , L_Discount decimal(13, 2) , L_Tax decimal(13, 2) , L_ReturnFlag varchar(64) , L_LineStatus varchar(64) , L_ShipDate timestamp , L CommitDate timestamp , L_ReceiptDate timestamp ,

L_ShipInstruct varchar(64) , L_ShipMode varchar(64) , L_Comment varchar(64) , skip varchar(64)): CREATE TABLE nation(N_NationKey int , N_Name varchar(64) , N RegionKey int , N Comment varchar(160) , skip varchar(64)); CREATE TABLE orders (O_OrderKey int , O_CustKey int , O_OrderStatus varchar(64) , O TotalPrice decimal(13, 2) , O_OrderDate timestamp , O_OrderPriority varchar(15) , O_Clerk varchar(64) , O ShipPriority int , O Comment varchar(80) , skip varchar(64)); CREATE TABLE part(P_PartKey int , P_Name varchar(64) , P_Mfgr varchar(64) , P Brand varchar(64) . P Type varchar(64) , P_Size int , P_Container varchar(64) , P RetailPrice decimal(13, 2) , P Comment varchar(64) , skip varchar(64)); CREATE TABLE partsupp(PS PartKey int , PS_SuppKey int , PS_AvailQty int , PS SupplyCost decimal(13, 2) , PS Comment varchar(200) , skip varchar(64)); CREATE TABLE region(R RegionKey int , R_Name varchar(64) R_Comment varchar(160) , skip varchar(64)); CREATE TABLE supplier(S_SuppKey int , S Name varchar(64) , S Address varchar(64) , S_NationKey int , S_Phone varchar(18) , S_AcctBal decimal(13, 2) , S Comment varchar(105) . skip varchar(64));

本文的TPC-H数据集使用MaxCompute 2.0数据类型和Decimal 2.0数据类型,所以创建的项目为MaxCompute 2.0版本。如果您的项目需要设置使用2.0数据类型,请在命令前添加如下语句一起提交执行。

setproject odps.sql.type.system.odps2=true; setproject odps.sql.decimal.odps2=true;

2. 创建具备访问OSS权限的RAM角色并授权。详情请参见STS模式授权。

3. 分次执行LOAD命令,将OSS的全部数据加载至创建的MaxCompute表中,并执行SQL命令查看和校验数据导入结果。LOAD命令详情请参见LOAD。

LOAD OVERWRITE TABLE orders FROM LOCATION 'oss://endpoint/oss_bucket_name/unload_from_redshift/orders_parquet/' --OSS存储空间位置。 ROW FORMAT SERDE 'org.apache.hadoop.hive.ql.io.parquet.serde.ParquetHiveSerDe' WITH SERDEPROPERTIES ('odps.properties.rolearn'='acs:ram::xxx:role/xxx_role') STORED AS PARQUET;

⑦ 说明 如果导入失败,请提工单联系MaxCompute团队处理。

执行如下命令查看和校验数据导入结果。

SELECT * FROM orders limit 100;

返回结果示例如下。

1	se	lect * from or	ders limit 100;								
运行	旧志	结果11	结果[2]	×						x	
		A	B	c	D	E	F	G	н		
▦	1	o_orderkey ~	o_custkey 🗸	o_orderstatus 🗸	o_totalprice 🗸	o_orderdate 🗸	o_orderpriority ~	o_clerk 🗸	o_shippriority	~ o_c	omment
Lad	2	255150214	2499461	F	261534.21	1993-04-21	5-LOW	Clerk#000036188	0	ula	r, express i
<u>um</u>	3	255150215	8424601	0	63673.46	1995-08-26	4-NOT SPECIFIED	Clerk#000096205	0	are	carefully a
	4	255150240	7267403	0	214230.06	1996-06-29	2-HIGH	Clerk#000013750	0	he	special pa
_	5	255150241	13235423	0	54816.94	1997-02-27	5-LOW	Clerk#000094211	0	e th	e blithely
~	6	255150242	12746899	0	46135.66	1995-06-20	1-URGENT	Clerk#000088919	0	the	ly. sometir
_	7	255150243	7107253	F	332747.02	1992-09-07	2-HIGH	Clerk#000058057	0	ely.	final dugc
111	8	255150244	6569728	F	145541.45	1993-10-30	2-HIGH	Clerk#000052064	0	are	fully final c
	9	255150245	2415874	F	311040.82	1992-07-01	5-LOW	Clerk#000023718	0	ntic	ingly regu
	10	255150246	13565806	0	260672.11	1995-07-10	3-MEDIUM	Clerk#000019110	0	nus	ual, ironic
	11	255150247	482515	F	20633.44	1994-04-21	5-LOW	Clerk#000067998	0	slo	w pinto be
	12	255150272	9628064	F	53396.22	1994-09-07	2-HIGH	Clerk#000028958	0	nal	requests.
	13	255150273	13867210	0	98256.89	1996-01-13	2-HIGH	Clerk#000085190	0	ack	tages cajo
	14	255150274	11170573	0	221079.64	1995-12-01	1-URGENT	Clerk#000074552	0	cka	iges sleep
	15	255150275	3816889	F	9341.42	1992-05-06	1-URGENT	Clerk#000091622	0	ush	y regular a
	16	255150276	10439924	F	149961.05	1993-03-06	5-LOW	Clerk#000084821	0	bea	ins mold c
	17	255150277	8788966	0	65552.01	1998-07-23	1-URGENT	Clerk#000076999	0	blit	thely. foxe:
	18	255150278	11128561	F	29456.5	1992-10-08	5-LOW	Clerk#000016410	0	the	slyly final
	19	255150279	2707846	F	45665.88	1994-11-28	2-HIGH	Clerk#000086676	0	lyly	express p
	20	255150304	741007	F	138694.94	1993-06-28	4-NOT SPECIFIED	Clerk#000051881	0	. sly	yly regular
	21	255150305	4087003	F	92537.42	1995-01-04	2-HIGH	Clerk#000063680	0	per	nding pack

4. 通过表的数量、记录的数量和典型作业的查询结果,校验迁移至MaxCompute的数据是否和Redshift集群的数据一致。

i. 登录Amazon Redshift控制台,在右上角选择区域为**亚太地区(新加坡)**。在左侧导航栏,单击<mark>编辑器</mark>,输入如下命令执行查询操作。

SELECT 1_returnflag, 1_linestatus, SUM(1_quantity) as sum_qty, SUM(1_extendedprice) AS sum_base_price, SUM(1_extendedprice*(1-1_discount)) AS sum_disc_price, SUM(1_extendedprice*(1-1_discount)*(1+1_tax)) AS sum_charge, AVG(1_quantity) AS avg_qty, AVG(1_extendedprice) AS avg_price, AVG(1_discount) AS avg_disc, COUNT(*) AS count_order FROM lineitem GROUP BY 1_returnflag, 1_linestatus ORDER BY 1_returnflag,1_linestatus;

返回结果示例如下。

l_returnflag ⊽	l_linestatus ⊽	sum_qty ⊽	sum_base_price ⊽	sum_disc_price ⊽	sum_charge ⊽	avg_qty ⊽
А	F	3774200	5320753880.69	5054096266.6828	5256751331.449234	25
Ν	F	95257	133737795.84	127132372.6512	132286291.229445	25
Ν	0	7679822	10823487077.24	10282025059.1390	10693158047.350890	25
R	F	3785523	5337950526.47	5071818532.9420	5274405503.049367	25

ii. 通过DataWorks的临时查询界面或MaxCompute客户端(odpscmd),执行上述命令,验证返回结果是否与Redshift集群的返回结果一致。 返回结果示例如下。

Sq Mig	Migration_to_MaxCompute x											
1 2 3 4 5 6 7	<pre>select L_returnflag, L_linestatus, sum(l_quantity) as sum_qty, sum(l_extendedprice) as sum_base_price, sum(L_extendedprice*(1-l_discount)) as sum_disc_price, sum(l_extendedprice*(1-l_discount)*(1+Ltax)) as sum_charge, avg(l_quantity) as avg_qty, avg(l_extendedprice) as avg_price, avg(L_discount) as avg_disc, count(*) as count_order from lineitem group by l_returnflag, l_linestatus rorder by l_returnflag,l_linestatus limit 100;</pre>											
运行	日志	结果[1]	×									
⊞	-	Α	В	с	D	E	F	G	н	1	J	
	1	Lreturnflag 🗸	Llinestatus ~	sum_qty ~	sum_base_price	sum_disc_price	✓ sum_charge ✓	avg_qty v	avg_price ~	avg_disc v	count_order	
<u> 111</u>	2	A	F	3774200	5320753880.69	5054096266.6828	5256/51331.449234	25.53/58/11685499/	36002.123829	0.050145	147790	
	3	N	0	7679822	10823487077 24	10282025059 139	10693158047 35089	25 5384548876681	35992 388424	0.049394	300716	
	5	R	F	3785523	5337950526.47	5071818532.942	5274405503.049367	25.5259438574251	35994.029214	0.049989	148301	

2.14. BigQuery数据迁移至MaxCompute

本文为您介绍如何通过公网环境将谷歌云GCP(Google Cloud Platform)的BigQuery数据集迁移至阿里云MaxCompute。

前提条件

类别	平台	要求	参考文档
环境及数据	谷歌云GCP	 已开通谷歌BigQuery服务,并准备好环境及待迁移的数据集。 已开通谷歌Cloud Storage服务,并创建存储分区(Bucket)。 	如果您没有相关环境及数据集,可参考如下内 容准备: • BigQuery: BigQuery快速入门和创建数据集 • Cloud Storage: Cloud Storage快速入 门和创建存储分区
	阿里云	 已开通MaxCompute、DataWorks服务并创建项目空间。 以印度尼西亚(雅加达)区域为例,创建作为迁移目标的 MaxCompute项目。 已开通对象存储服务OSS并创建存储分区(Bucket)。 已开通OSS的在线迁移上云服务。 	如果您没有相关环境,可参考如下内容准备: • MaxCompute、DataWorks:准备工作和创 建MaxCompute项目 • OSS:开通OSS服务和创建存储空间 • 在线迁移上云服务:提工单或在线申请
能 旦	谷歌云GCP	已创建具备访问谷歌Cloud Storage权限的IAM用户。	通过JSON使用IAM权限
処ち	阿里云	已创建具备存储空间读写权限和在线迁移权限的RAM用户及RAM角色。	创建RAM用户和STS模式授权
又基	谷歌云GCP	无。	无
区场	阿里云	开通OSS服务的区域与MaxCompute项目在同一区域。	无

背景信息

将BigQuery数据集迁移至阿里云MaxCompute的流程如下。



最佳实践·数据迁移

序号	描述
2	通过对象存储服务OSS的 在线迁移上云服务 ,将数据从谷歌Cloud Storage迁移至OSS。
3	将数据从OSS迁移至同区域的MaxCompute项目中,并校验数据完整性和正确性。

步骤一:将BigQuery数据集导出至谷歌Cloud Storage

您可以使用bq命令行工具执行 bg extract 命令,将BigQuery数据集导出至谷歌Cloud Storage。

1. 登录Google Cloud控制台,创建存储迁移数据的分区(Bucket)。操作详情请参见创建存储分区。

=	Google Cloud Platform		Q 搜索产品和资源			~	2.	9 1	e e
- 1	Storage	← 存储分区详情							
•	浏览器	timiger.							
<i></i>	监控	对象 配置 权限 保留	生命周期						
₽	转移	in in the second second							
6	本地传输	存储分区 > 🗖							
-	Transfer Appliance	上传文件 UPLOAD FOLDER 创建文件	夹 管理保全 删除						
\$	设置	Filter by object or folder name prefix							
		□ 名称		大小	Created time 🔞				
				-	-				:

2. 使用bq命令行工具,查询TPC-DS数据集中表的DDL脚本并下载至本地设备。操作详情请参见使用INFORMATION_SCHEMA获取表元数据。

BigQuery不提供诸如 show create table 之类的命令来查询表的DDL脚本。BigQuery允许您使用内置的用户自定义函数UDF来查询特定数据集中 表的DDL脚本。DDL脚本示例如下。

Row	10_	
1	CREATE OR REPLACE TABLE 'uuuuu uus0.tpcds_100gb.household_demographics' (hd_demo_sk INT64, hd_icroome_band_sk INT64, hd_icroup.otential STRING, hd_dep_count INT64, hd_vehicle_count INT64	
2	CREATE OR REPLACE TABLE ' , , tpcds_100gb.web_sales' (ws_sold_date_sk.INT64, ws_ship_date_sk.INT64, ws_ship_date_sk.INT64, ws_bill_custem_sk.INT64, ws_bill_custem_sk.INT64,	

3. 通过bq命令行工具执行 bg extract 命令,将BigQuery数据集中的表依次导出至谷歌Cloud Storage的目标存储分区(Bucket)。相关操作及导出的数据格式和压缩类型详情请参见导出表数据。

导出命令示例如下。 bq extract --destination_format AVRO --compression SNAPPY tpcds_100gb.web_site gs://bucket_name/web_site/web_site-*.avro.snappy;

4. 查看存储分区,检查数据导出结果。

步骤二:将导出至谷歌Cloud Storage的数据迁移至对象存储服务OSS

对象存储服务OSS支持通过**在线迁移上云服务**,将谷歌Cloud Storage的数据迁移至OSS,详情请参见谷歌云GCP迁移教程。在线迁移上云服务处于公测 状态,您需要<mark>提工单</mark>联系客服,并由在线服务团队开通后才可使用。

- 1. 预估需要迁移的数据,包括迁移存储量和迁移文件个数。您可以使用gsutil工具或通过存储日志查看待迁移存储分区(Bucket)的存储量。详情请参见获取存储分区信息。
- 2. (可选)如果您未创建存储分区,请登录OSS管理控制台,创建保存迁移数据的分区(Bucket),详情请参见创建存储空间。

沈兼存稿 / / 文件管理 ⑦ 通过 SDK 管理文											OK 管理文件				
sp-migrat	ion.	2								读写权限 私	美国 美型	标准存储 (本地冗余)	区域 印度尼西亚 (雅加达)	创建时间 2020年8月	21日 10:48
概览		上传文件	新建目录	碎片管理	授权	批量操作 🗸	刷新						历史版本	请输入文件名前缀匹配	Q
文件管理	>		文件名						文件大小			存储类型			操作
权限管理	>	- 1	tpc_ds_100)gb/											删除
基础设置	>														

3. (可选)如果您未创建RAM用户,请创建RAM用户并授予相关权限。

i. 登录RAM访问控制台,创建RAM用户,详情请参见创建RAM用户。

ii.	选中新创建的用户登录名称,	单击 添加权限 ,	为新创建的RAM用户授予 AliyunOSSFullAcce s	s(存储空间读写权限)
	和AliyunMGWFullAccess(在线迁移权限)	单击 确定 > 完成 。	

iii. 在左侧导航栏,单击**概览**。在**概览**页面的**账号管理**区域,单击**用户登录地址**链接,使用新创建的RAM用户登录<mark>阿里云控制台</mark>。

- 4. 在GCP侧准备一个以编程方式访问谷歌Cloud Storage的用户。操作详情请参见通过JSON使用IAM权限。
 - i. 登录IAM用户控制台,选择一个有权限访问BigQuery的用户。在操作列,单击_E > 创建密钥。

ii. 在弹出的对话框,选择JSON,单击创建。将JSON文件保存至本地设备,并单击完成。

iii. 在创建服务账号页面,单击选择角色,选择Cloud Storage > Storage Admin,授予用户访问谷歌Cloud Storage的权限。
 5. 创建在线迁移数据地址。

5. 则建住线江杨数据地址。

- i. 登录<mark>阿里云数据在线迁移控制台。</mark>在左侧导航栏,单击**数据地址**。
- ii. (可选)如果您未开通在线迁移服务,请在弹出的对话框中单击**去申请**。在**在线迁移公测申请**页面,填写信息,并单击提交。

在线迁移公测申请	
* 联系人:	
* 联系电话:	
钉钉号:	
* 迁移数据源类型:	・ 线下数据(有专线或者VPN) ECS数据 ・ 阿里NAS ● 阿里OSS 亚马逊AWS ・ 微软 AZURE 勝讯 COS 百度 BOS ● 又拍云 七牛 金山云KS3 HTTP源
* 迁移目的端:	 ○ 阿里云OSS ○ 阿里云 NAS ○ 线下 NAS (有VPN或者专线)
* 迁移类型:	○ 一次性迁移 ○ 定时同步 (仅支持NAS)
* 迁移数据量预估(GB):	
* 迁移数据文件数预估(个):	
* 是否跨region∶	○是○否
音注:	
	submit

⑦ 说明 在线迁移公测申请页面的迁移数据源类型中如果没有谷歌云GCP,请您选择其中一种数据源类型,并在备注中指明实际数据 源类型。

iii. 在**管理数据地址**页面,单击**创建数据地址**,配置数据源及目标地址相关参数,单击**确定**。参数详情请参见迁移实施。

■ 数据源

创建数据地址	①如需更多帮助请参考产品手册	×
◎ 数据地址可以作为迁	移任务的 [源地址] 或者 [目的地址]。数据地址	
- 创建成切之后, 芯叮	以创建迁移任务	
数据类型	Google Storage ~	
* 数据名称	google_storage_ds 17/63	
* Bucket	as tocds 100ab	
Desfiel		
Preiix	迁移全部数据 迁移部分数据	
Key File	印 点击上传JSON文件	
	未选择任何文件	
		Ű
	取消	靛

⑦ 说明 Key File是步骤4下载的JSON文件。

创建数据地址	①如需更多帮助请参考产品手册	×
数据地址可以作为迁 创建成功之后,您可	移任务的 [源地址] 或者 [目的地址]。数据地址 以 创 <mark>建迁移任务</mark>	
数据类型	OSS ~	
* 数据名称	migration_destination_ds 24/63	
* 数据所在区域	印度尼西亚 (雅加达) 🗸 🗸	
* OSS Endpoint	http://oss-ap-southeast-5-internal.aliyun \lor	
* Access Key Id ⑦		
* Access Key Secret ⑦	•••••	
* OSS Bucket		
OSS Prefix ⑦	tpc_ds_100gb/ / 请选择或输入迁移文件的prefix (不填代表迁移 今年1)	
	取消	定

⑦ 说明 Access Key ID和Access Key Secret 为RAM用户的密钥信息。

- 6. 创建在线迁移任务。
 - i. 在左侧导航栏,单击**迁移任务**。
 - ii. 在迁移任务列表页面,单击创建迁移任务,配置相关信息后,单击创建。参数详情请参见迁移实施。

住部調洗 ・任祭名称 「雪志子可用欺」(湯/目的)地址、清怨先想建数講 「「雪志泽 ・		「「「「」」「「」」「」」「」」「」」「」」「」」「」」「」」「」」」「」」	ни т и 10
	任务配置	置 性能调	优
・任务名称 gs-to-oss-job 13/63 知果无可用数据(源目的)地址,请您先创建数据 地址 源选择 ・源地址() 请选择 ・目的地址() 请选择 工修方式() 全量正修 增量正修 数据同步 文件裏盖方式 原作校の切旧优先 条件要盖 全要盖 不要盖 全要差 文件裏盖方式 原作校の切旧优先 条件要盖 全要盖 「「「」」」 小田菜園 文件報告の式 2. 如果源LastModified = 目的LastModified, 即此 文件保格状分育込式。 2. 如果源LastModified = 目的LastModified, 即此 次件報地分词放式。 2. 如果源LastModified = 目的LastModified, 即此 元件報助分词题式。	迁移数据地址		
	*任务名称	gs-to-oss-job	13/63
		如果于可用数据 (酒/日始) 地址 法	你生创建新居
 ▲ 原形地山 ② 请应至年 ◆ 目的地址 ③ 请选择 ◆ 目的地址 ④ 请选择 正移方式 ③ 全量正移 『雪量正形 数据同步 文件覆盖方式 新二春次改时间优先 条件覆盖 全覆盖 文件覆盖方式 文件覆盖方式 文件覆盖方式 新二春次改时间优先 条件覆盖 全覆盖 文件覆盖(大売判断二者的LastModified, 即時 「所必改时间。 如果混LastModified < 目的LastModified, 即此 文件将被执行那近。 如果混LastModified < 目的LastModified, 则此 文件将被执行那道。 如果混LastModified < 目的LastModified, 则此 文件将被执行那道。 如果混LastModified > 目的LastModified, 则此 方で置先。 如果混LastModified > 目的LastModified, 则此 方で置差。 如果混LastModified > 目的LastModified, 则此 方で置差。 五、四月二 本 (大戸田市) (大中報) での加速度を早助请参考产品手助 在多配置 での加速度を早助请参考产品手動 在多配置 での加速度を用助请参考产品手動 での加速度を用 での加速度を用助请参考 (本) からに、 なに、 からに、 なに、 からに、 なに、 のに、	. 1544	如来无可用致痛(减;日四)地狂,頃 地址	
 ◆目的地址 ② 講选择 壬移方式 ③ 全星壬修 常星壬移 致振同步 式伴覆盖方式 局后修改时间优先 条件覆盖 全覆盖 不要意 文件覆盖方式 局后修改时间优先 条件覆盖 全覆盖 文件覆盖方式 局后修改时间优先 条件覆盖 全覆盖 文件覆盖方式 如用源LastModified >目的LastModified,即時 G/m にないのからい。 如用源LastModified >目的LastModified,即時 ○加累源LastModified >目的LastModified,则此 公共確認,astModified >目的LastModified,则此 公共確認,astModified >目的LastModified,则此 公式電影 (Size, Content-Type有具一不相等,则 が行電意。 - る二者的Size或Content-Type有具一不相等,则 が行電意。 - る二者的Size或Content-Type有具一不相等,则 が行電話。 - る三者のSize或Content-Type有則時等),文件将被 か行電話。 - るごの別(Size, Content-Type引用時),文件将被 か行意込。 での加需更多報助请参考广品手册 を留所 を留所 なの目的には、 ののに なのに	* //\$7484E (?)	· 肩边洋	~
	*目的地址⑦	请选择	\sim
迁移方式 ② 全量壬修 営量壬修 営業間後 文件覆蓋方式 馬后修及时间优先、条件覆盖、全覆盖 不電差 別日尾名大件、优先判断二者的让astModified、即時におけれのはいけ、別はたけにおけいろいし。 1. 如果源LastModified <= 目的LastModified、则はたいでであ。 3. 如果源LastModified <= 目的LastModified、则はたいでであ。 3. 可思源LastModified <= 目的LastModified、则はたいであ。 3. 可思源LastModified == 目的LastModified、则はたいであ。 3. 可思源LastModified == 目的LastModified、则はたいであ。 3. 可思察LastModified == 目的LastModified、则はたいであ。 3. 可思察LastModified == 目的LastModified、则はたいであ。 3. 正常記 4. 正常記	迁移策略		
文件覆盖方式 福島修改时间优先 条件覆盖 全覆盖 不要盖 对于同名文件,优先判断二者的LastModified,即断 后修改时间。 1. 如果源LastModified < 目的LastModified,则此 文件将被计行部达。 2. 如果源LastModified > 目的LastModified,则此 文件将被计行部达。 3. 如果源LastModified > 目的LastModified,则比 文件将被计行部达。 3. 如果源LastModified == 目的LastModified,则比 文件将被计行题达。 3. 如果源LastModified == 目的LastModified,则比 文件将被计行题达。 "在二者的Size或Content-Type有其一不指等,则 计行题选。 "否则(Size、Content-Type有其一不相等,则 计行题选。 "否则(Size、Content-Type有其一不相等,则 计行题选。 "否则(Size、Content-Type有其一不相等),文件将被 计分别达。 建计移任务 ①如需更多帮助请参考广品手册 建计移任务 ①如需更多帮助请参考广品手册 全方配置数 ①如需更多帮助请参考广品手册 金属野館(NELSE) ①如需更多帮助请参考广品手册 金属野館 ①如需更多帮助请参考广品手册 金属野館 ①如需更多帮助请参考广品手册 金属子的 ①如需更多帮助请参考广品手册 金属子的 ①如需更多帮助请参考广品手册 金属型数 ①如需更多能。 金属。 ① 金属。 ① 金属的 ① 金属型数 ① 金属型数 ① 金属型 ① 金属型 ①	迁移方式⑦	全量迁移 增量迁移 数据同步	
文件覆盖方式 不要盖 对于同名文件,优先判断二者的LastModified,即振 后修改时间。 1. 如果源LastModified < 目的LastModified,则此 文件将被功行跳过。 2. 如果源LastModified > 目的LastModified,则此 文件将被功行跳过。 2. 如果源LastModified > 目的LastModified,则此 文件将被助行跳过。 3. 如果源LastModified == 目的LastModified,则此 文件将被助行题注。 3. 如果源LastModified == 目的LastModified,则 此劳用信: 否则 (Size, Content-Type有用每),文件将被 执行激过。 · 不同 (Size, Content-Type有用每),文件将被 执行激过。 W消 ①如需更多帮助请参考产品手册 建注移任务 ①如需更多帮助请参考产品手册 使活動力: 在診過优 建注移任务 ①如需更多帮助请参考产品手册 使活動力: 在診過优 建注移任务 ①如需更多帮助请参考产品手册 修正修改件个数 如同平位迁移双属量 (每天)限流时间段 个 成点 9点 12点 15点 18点 21点 2 (每天)限流时间段 ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● <td< td=""><td></td><td>最后修改时间优先 条件要盖 全部</td><td>要盖</td></td<>		最后修改时间优先 条件要盖 全部	要盖
対于同名文件,优先判断二者的LastModified,則結 后修改时间。 1、如果源LastModified < 目的LastModified,則此 文件将被执行部达。 2、如果源LastModified > 目的LastModified,则批 行覆盖。 3.如果源LastModified >==目的LastModified,则批 行覆盖。 3.如果源LastModified ===目的LastModified,则批 行覆盖。 -石二者的Size或Content-Type有其一不相等,贝 执行部道。 -石二者的Size或Content-Type有其一不相等。贝 执行部过。 -石二者的Size或Content-Type有相等),文件将描 执行部过。 建注移任务 ①如需更多帮助请参考产品手册 任告配置 生態明优 数据预估 ② 方保障顺利完成迁移任务,准确统计迁移进度和成功案,请尽量推 确评估您的迁移存储量和迁移文件个数、如何评估迁移政境量 每注移存储量 GB 停迁移存储量 GB 6点 9点<12点	文件覆盖方式	不要盖	
1. 如果源LastModified < 目的LastModified,则此 文件将被执行题达。 3. 如果源LastModified > 目的LastModified,则 任寒刑部: - 若二者的Size或Content-Type有其一不相等,则 执行意意。 - 否则(Size、Content-Type都相等),文件将被 执行部边。 * 可则(Size、Content-Type都相等),文件将被 执行部边。 * 可则(Size、Content-Type都相等),文件将被 力行部边。 * 可则(Size、Content-Type都有量。 * 可则(Size、Content-		对于同名文件,优先判断二者的Lastl 后修改时间。	Vlodified, 即看
2.如果源LastModified > 目的LastModified、则执行覆盖。 3.如果源LastModified == 目的LastModified、则 建球川町: - 石舎的Size或Content-Type有其一不相等。贝 水行覆盖。 - 否別(Size、Content-Type有其一不相等。贝 水行覆盖。 - 否別(Size、Content-Type有其一不相等。贝 地行源込。 *諸调优 建迁移任务 ①如需更多帮助请参考产品手册 任务配置 生能時优 数据数估 ①如需更多帮助请参考产品手册 ● 女子保障限利完成迁移任务、准确统计迁移进度和成功率、请尽量准 ● 少方保障限利完成迁移存储量和迁移文件个教。如何评估迁移数据量 ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●		1. 如果源LastModified < 目的LastM 文件将被执行跳过。	odified,则此
3. 如果要LastModified == 目的LastModified,则 出版時間: - 名二書的StzeniContent-Type有具一不相等,则 地行電益: - 否则 (Size, Content-Type有相等),文件将被 水行電益: - 否则 (Size, Content-Type有相等),文件将被 水行電益: - 否则 (Size, Content-Type有相等), 文件 建注移任务 ①如需更多帮助请参考产品手册 任容配置 住能唱优 数据预估 文 为保障週間利完成迁移行储量和迁移文件へ数、如何评估迁移数据量 停迁移存储量 68 停迁移存储量 68 修迁移存储量 68 停迁移存储量 68 (每天)限流时间段 个 最大流量(MB/s) 5 万治 添加 开始 4束 展流 現流 現作 不设置限流		 如果源LastModified > 目的LastM 行要盖。 	odified, 则执
· 古二書的Sze或Content-Type有具一不相等, 與 此行調證。 · 古二書別 (Size, Content-Type有具一不相等, 與 此行調註。 · 古川 (Size, Content-Type有則用等), 文件特徴 执行調註。 ** 建迁移任务 ①如需更多帮助请参考产品手册 任务配置 性前明优 数据预估 ② 方保障順利完成迁移午储量和迁移文件个数、如何评估迁移数据量 侍迁移存储量 68 侍迁移存储量 0点 3点 6点 9点 12点 15点 18点 21点 2 (每天)限流时间段 · 子设置限流		3. 如果源LastModified == 目的Lastl 继续判断·	Modified,则
→ 古製 (Size, Content-Type都相等),文件特徴 取消 下 第週 (Size, Content-Type都相等),文件特徴 第週 (Size, Content-Type都相等),文件特徴 第週 (Size, Content-Type都相等),文件特徴 第週 (Size, Content-Type都相等),文件特徴 建訂移任务 ①如需更多帮助请参考产品手册 任务配置 住邸明优 数据预估 ② 为保障顺利完成迁移行储量和迁移文件个数、如何评估迁移数据量 每 68 停迁移存储量 68 停迁移存储量 68 (每天)限流时间段 東大流量(MB/s) 5 万分置限流 万次置限流		- 若二者的Size或Content-Type有其	其一不相等,则
取行動正、 取消 下 診调化 建迁移任务 ①如需更多帮助请参考产品手册 任告配置 生筋明坑 数据预估 ② 为保障顺利完成迁移任务,准确统计迁移进度和成功率、请尽量准 微子保障顺利完成迁移存储量 ② 为保障顺利完成迁移行储量和迁移文体个数、如何评估迁移数据量 停迁移存储量 6 9点 9点 12点 15点 18点 21点 2 (每天)限流时间段 重大流量(MB/s) 5 万始 須束 限流 一 不设置限流		- 否则 (Size、Content-Type都相)	〕 ,文件将被
取消 下 諸期代 金山田県の名参考产品手册 建迁移任务 ①如需更多報助请参考产品手册 任务配置 金近期成 数据预估 金山田県の 愛 カ保障順利売成迁移任务, 生商統計迁移进度和成功率, 请尽量准 金 から開から成正移任务, 生商統計迁移理成和成功率, 请尽量准 金 から開から成正移任务, 生商統計迁移理備量和过移文体个数, 知の戸市に注移数増量 金 からごをする 金 からごをする 金 から、 金 ・ 金 ・ ・ ・		执行跳过。	
取消 下 謝湖 优 ④如需更多帮助请参考产品手册 建迁移任务 ④如需更多帮助请参考产品手册 任告配厚优 性能厚优 数据预估 ● ② 为保障顺利完成迁移任务、准确统计迁移进度和成功率、请尽量准 發 为保障顺利完成迁移存储量和迁移文件个数。如何平估迁移数属量 ● 停迁移存储量 ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ●			
任務問題 住前明优 数据预估 ② 为保障顺利完成迁移任务,准确统计迁移进度和成功率,请尽量推 确评估您的迁移存储量 ● 務評格完的迁移存储量 ● 68 ● 68 ● 68 ● 68 ● 68 ● 68 ● 68 ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● ● <t< th=""><th>(本)エジタ/エタ</th><th></th><th></th></t<>	(本)エジタ/エタ		
数据预估	NEL工作/1175	①如需更多希助唷	参考产品手册
→ 別保珈((阿利売成ご移任务、准确统计迁移进度和成功率、请尽量准确;正估您的迁移存储量和迁移文件个数、如何评估迁移数据量 待迁移存储量 侍迁移存储量 侍迁移存储量 侍迁移文件个数 侍迁物文件个数 の点 3点 6点 9点 12点 15点 18点 21点 2 の点 3点 6点 9点 12点 15点 18点 21点 2 和式量(MB/s) 日 日 和式量(MB/s) 日 日 日 日 日 日 日 日 日 日 日 日 日 日 日 日 日 日 日 日 日 日 日 日 日 日 日 日	建过物任务配		参考产品手册 优
待迁移存储量 GB 待迁移文件个数 个 流量控制 0点 3点 6点 9点 12点 15点 18点 21点 2 (每天)限流时间段 + 最大流量(MB/s) 5 万始 结束 限流 操作 不设置限流	2年11月27日 任务配 数据预估		参考产品手册 优
待迁移文件个数 个 流量控制 0点 3点 6点 9点 12点 15点 18点 21点 2 (每天)限流时间段	(任务配) (任务配) 数据预估 ② 为保障顺利完成 确评估您的迁移	①如壽更多幣如南 置 性範疇 江移任务,准确统计迁移进度和成功函 存储量和迁移文件个数。如何评估迁行	参考产品手册 优 ^医 ,请尽量准 多数据量
待迁移文件个数 个 流量控制 0点 3点 6点 9点 12点 15点 18点 21点 2 (每天)限流时间段 5 最大流量(MB/s) 5 万始 结束 限流 操作 不设置限流	(任务配) (任务配) 数据预估 ② 为保障顺利完成 确评估忽的迁移 待迁移存储量	①如論更多帮助请 【 住能调 【 住能调 【 任能调 》 【 任能调 》 》 》 </td <td>参考产品手册 优 ^{医,}请尽量准 多数据量 GB →</td>	参考产品手册 优 ^{医,} 请尽量准 多数据量 GB →
流量控制 0点 3点 6点 9点 12点 15点 18点 21点 2 (每天)限流时间段	▲正本7日子子 任务部: 数据预估 》为保障顺利完成 确评估您的迁移 待迁移存储量	①如論史多常如请 查 住爺湯 查 住爺湯 适移任务,准确统计迁移进度和成功3 存储量和迁移文件个数。如何评估迁行	参考产品手册 优 ∝,请尽量准 多数据量 GB ~
(每天)限流时间段 5 添加 最大流量(MB/s) 5 添加 开始 结束 限流 操作 不设置限流 5	任务配 数据预估 》为保障顺利完成 确评估您的迁移 待迁移存储量 待迁移文件个数	①如論史多報助请 置 性能调 迁移任务、准确统计迁移进度和成功 定移任务、加何评估迁移	参考产品手册 优 区,请尽量准 多数据量 GB ~ 个 ~
最大流量(MB/s) 5 添加 开始 结束 限流 操作 不设置限流	★上にや1155 任务部: 数据预估 为保障顺利完成 确评估您的迁移 待迁移存储量 待迁移存储量 待迁移存储量 洗量控制	田如嘉史多報初頃 世態調 任態調 廷稼任务,准确统计迁移进度和成功强 存储量和迁移文件个数。如何评估迁行 日 日 日 日 日 日 日	参考产品手册 优 匹,请尽量准 多数据量 GB ~ 185 215 2
	任务配 数据预估 》为保障顺利完成 确评估您的迁移 待迁移存储量 待迁移文件个数 流量控制 ((每天)限流时间段	① 如論更多常切情 【 世能得 江移任务、准确统计迁移进度和成功理 存储量和迁移文件个数。如何评估迁行 ① ① ① ③ ③ ③ ⑤ ⑤ ③ ③ ③ ③ ⑤ ⑤ ⑤ ③ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑦ ⑤ ⑤ ⑦ ⑤ ⑦ ⑤ ⑦ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑤ ⑦ ⑤ ⑦ ⑤ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦ ⑦	参考产品手册 优 区,请尽量准 多数据量 GB ~ 18点 21点 2 ; ;
开始 结束 限流 操作 不设置限流		① 如嘉史多報初頃 世態调 世態调 过移任务,准确统计迁移进度和成功 存储量和迁移文件个数。如何评估迁和 ① 点 3点 6点 9点 12点 15点 5	参考产品手册 优 医,请尽量准 多数振量 GB ~ 18点 21点 2 18点 21点 2 115 115 115 115 115 115 115 11
不设置限流	★上工や1155 任务部: 数据预估 为保障顺利完成 希评估您的迁移 待迁移存储量 待迁移文件个数 流量控制 (每天)限流时间段 最大流量(MB/s)	□ 如論更多報初请	参考产品手册 优 S、 请尽量准 多数据量 GB ~ 18点 21点 2 13点 21点 2 13点 21点 2
	★生工や1135 任务部 数据预估 》 为保障顺利完成 希评估您的迁移 待迁移存储量 待迁移存储量 待迁移文件个数 流量控制 (每天)限流时间段 最大流量(MB/s) 开始	① 如嘉史多報初頃 【 世前間 【 世前間 ご移任务、准确统计迁移进度和成功 巧存储量和迁移文件个数。如何评估迁移 ① □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □	参考产品手册 优 S. 请尽量准 多数張量 GB ~ 个 ~ 18点 21点 2 流加 操作
	任务部 数据预估	① 如扁更多常初頃 【生能调 【 【 【 【 【 【 】 【 】 【 】 【 】 【 】 【 】 【 】 【 】 】 【 】 】 【 】 】 【 】 】 】 】 【 】 】 【 】 】 】 】 】 【 】 】 】 【 】 】 】 【 】 】 】 【 】 】 】 】 【 】 】 】 】 】 【 】 】 】 【 】 】 【 】 】 】 】 】 】 】 【 】 】 】 【 】 】 】 】 】 【 】 】 】 【 】 】 】 【 】 】 】 【 】 】 【 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】	参考产品手册 优 医,请尽量准 多数振量 GB 〜 18点 21点 2 18点 21点 2 13点 21点 2 13点 21点 2
		① 如論更多報知頃 【 【 化論 印 《 日 《 日 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化 化	参考产品手册 优 低,请尽量准 多数据量 GB ~ 18点 21点 2 18点 21点 2 減加 操作
			参考产品手册 优 低 構 の量准 多数据量 GB 〜 18点 21点 2 18点 21点 2 注 添加 操作
		① 知識更多報知頃 【 【 】 【 】 【 】 【 】 【 】 【 】 】 】 【 】 】 】 【 】 】 】 】 】 【 】 】 】 】 】 】 】 】 】 】 【 】 】 】 】 】 】 】 】 】 】 】 】 】 【 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 【 】 】 】 】 】 】 】 】 】 】 】 】 】 【 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】 】	参考产品手册 优 低 、 请尽量准 多数振量 GB へ 18点 21点 2 添加 操作
	★生工や1155 任务部 数据预估 》 为保障順利式統 希洋塔/280计译 待迁移存储量 待迁移文件个数 流量控制 (每天)限流时间段 最大流量(MB/s) 开始	① 知識更多報初頃 【 世齢調 ご び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び び	参考产品手册 优 低 低 低 低 低 低 低 低 低
	★ LET #71253 任务部 数据预估 》 为保障顺利式成 希洋都存储量 待迁都存储量 待迁都存储量 待迁都存储量 (每天)限流时间段 最大流量(MB/s) 开始		参考产品手册 优 S、 请尽量准 多数据量 GB ~ 18点 21点 2 13点 21点 2 13点 21点 2
		C 外山論史多年初頃 C 外山論史多年初頃 C 外山論史多年初頃 C 小 前 12 小 15 小	参考产品手册 优 ぶ、请尽量准 多数据量
		C 火山震史多年初頃 C 火山震史多年初頃 C 人工震 C 人工 C 人工	参考产品手册 优 低 低 低 低 低 低 低 低 低

⑦ 说明 待迁移存储量和待迁移文件个数是您通过Google Cloud控制台获取到的待迁移数据大小和文件个数。

iii. 创建的迁移任务会自动运行。请您确认任务状态为已完成,表示迁移任务成功结束。

iv. 在迁移任务的右侧单击管理, 查看迁移任务报告, 确认数据已经全部迁移成功。

- v. 登录OSS管理控制台。
- vi. 在左侧导航栏,单击Bucket列表。在Bucket列表区域单击创建的Bucket。在Bucket页面,单击**文件管理**,查看数据迁移结果。

步骤三:将数据从OSS迁移至同区域的MaxCompute项目

您可以通过MaxCompute的LOAD命令将OSS数据迁移至同区域的MaxCompute项目中。

LOAD命令支持STS认证和AccessKey认证两种方式,AccessKey认证方式需要使用明文AccessKey ID和AccessKey Secret。STS认证方式不会暴露 AccessKey信息,具备高安全性。本文以STS认证方式为例介绍数据迁移操作。

1. 在DataWorks的临时查询界面或MaxCompute客户端(odpscmd),修改已获取到的BigQuery数据集中表的DDL,适配MaxCompute数据类型,创 建与迁移数据相对应的表。

临时查询功能详情请参见使用临时查询运行SQL语句(可选)。命令示例如下。

CREATE OR REPLACE TABLE `****.tpcds_100gb.web_site` (web_site_sk INT64, web_site_id STRING, web_rec_start_date STRING, web_rec_end_date STRING, web_name STRING, web_open_date_sk INT64, web_close_date_sk INT64, web class STRING, web_manager STRING, web_mkt_id INT64, web mkt class STRING, web_mkt_desc STRING, web_market_manager STRING, web_company_id INT64, web_company_name STRING, web_street_number STRING, web_street_name STRING, web_street_type STRING, web_suite_number STRING, web_city STRING, web_county STRING, web_state STRING, web zip STRING, web country STRING, web_gmt_offset FLOAT64, web_tax_percentage FLOAT64) Modify the INT64 and FLOAT64 fields to obtain the following DDL script: CREATE TABLE IF NOT EXISTS <your_maxcompute_project>.web_site_load (web_site_sk BIGINT, web_site_id STRING, web_rec_start_date STRING, web_rec_end_date STRING, web name STRING, web_open_date_sk BIGINT, web_close_date_sk BIGINT, web class STRING, web_manager STRING, web_mkt_id BIGINT, web_mkt_class STRING, web_mkt_desc STRING, web_market_manager STRING, web_company_id BIGINT, web_company_name STRING, web_street_number STRING, web_street_name STRING, web_street_type STRING, web_suite_number STRING, web city STRING, web county STRING, web_state STRING, web_zip STRING, web_country STRING, web_gmt_offset DOUBLE, web_tax_percentage DOUBLE

);

BigQuery和MaxCompute数据类型的对应关系如下。

BigQuery数据类型	MaxCompute数据类型
INT64	BIGINT
FLOAT 64	DOUBLE
NUMERIC	DECIMAL、DOUBLE
BOOL	BOOLEAN

BigQuery数据类型	MaxCompute数据类型
STRING	STRING
BYTES	VARCHAR
DATE	DATE
DAT ET IME	DATETIME
TIME	DATETIME
TIMESTAMP	TIMESTAMP
STRUCT	STRUCT
GEOGRAPHY	STRING

2. (可选)如果您未创建RAM角色,请创建具备访问OSS权限的RAM角色并授权。详情请参见STS模式授权。

3. 执行LOAD命令,将OSS的全部数据加载至创建的MaxCompute表中,并执行SQL命令查看和校验数据导入结果。LOAD命令一次只能加载一张表,有多个表时,需要执行多次。LOAD命令详情请参见LOAD。

LOAD OVERWRITE TABLE web_site

FROM LOCATION 'oss://oss-<your_region_id>-internal.aliyuncs.com/bucket_name/tpc_ds_100gb/web_site/' --OSS存储空间位置。 ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.avro.AvroSerDe' WITH SERDEPROPERTIES ('odps.properties.rolearn'='<Your_RoleARN>', 'mcfed.parquet.compression'='SNAPPY')

WITH SERVERWORKTIES ('odps.properties.rolearn'='<rour_kolearn>', 'mcred.parquet.compression'='SNAPPY') STORED AS AVRO;

⑦ 说明 如果导入失败,请提工单联系MaxCompute团队处理。

执行如下命令查看和校验数据导入结果。

SELECT * FROM web_site;

返回结果示例如下。

							web_open_date_sk			web_manager			web_market_manager	web_company_na
1		16	AAAAA	1997-08-16	1999-08-16	site_2	2450679	2447029	Unkno	Herbert H	Hard new	Basic othe	Albert Leung	cally
2			AAAAA	1999-08-17	2001-08-15	site_2	2450679	2447029		Charles C	New, differ	Basic othe	Keith Frazier	cally
3			AAAAA	2001-08-16	N	site_2	2450679	2447029	Unkno	Charles C	Broad, ne	Basic othe	Keith Frazier	cally
4			AAAAA	1997-08-16		site_0	2450807		Unkno	Ronald Sh	Grey lines	Well simila	Joe George	cally
5			AAAAA	1997-08-16	2000-08-15	site_0	2450798	2443498	Unkno	Tommy J	Completely	Lucky pas	David Myers	ese
6			AAAAA	2000-08-16		site_0	2450798	2443498	Unkno	Tommy J	Completely	Particular,	David Myers	ese
7			AAAAA	1997-08-16		site_3	2450654		Unkno	Jimmy Pope	Leaders m	Home new	Eldon Snow	anti
8			AAAAA	1997-08-16	1999-08-16	site_0	2450781	2447131	Unkno	Harold Wi	As strong	Deeply sm	James Harris	anti
9			AAAAA	1999-08-17	2001-08-15	site_0	2450781	2447131	Unkno	Harold Wi	Wide, final	Deeply sm	Edward George	ought
10		6	AAAAA	2001-08-16		site_0	2450781	2447131	Unkno	Harold Wi	Wide, final	Thin roles	John Sheppard	ought
1			AAAAA	1997-08-16	2000-08-15	site_3	2450645	2446995	Unkno	Peter Cas	Just popul	Formerly I	Kelvin Lynch	pri
12	2		AAAAA	2000-08-16		site_3	2450645	2446995	Unkno	Adam Sto	As private	Formerly I	Jarvis Allen	pri
13			AAAAA	1997-08-16	1999-08-16	site_3	2450628	2443328	Unkno	Lewis Wolf	Severe, us	Individual	James Bernard	able
14			AAAAA	1999-08-17	2001-08-15	site_3	2450628	2443328	Unkno	Jason Silva	Severe, us	Individual	Jeffrey Martin	able
15		24	AAAAA	2001-08-16		site_3	2450628	2443328	Unkno	John Ward	Severe, us	Individual	Philip Sampson	able
10	6		AAAAA	1997-08-16		site_1	2450756		Unkno	Moses Hi	New home	Enough s	William Reyes	ese
1		8	AAAAA	1997-08-16	2000-08-15	site_1	2450747	2447097	Unkno	Charles P	Only, temp	Physical wi	Gerald Craft	anti
18	8		AAAAA	2000-08-16		site_1	2450747	2447097	Unkno	Marshall	Labour car	Physical wi	Scott Bryant	anti
19			AAAAA	1997-08-16	1999-08-16	site_1	2450730	2443430	Unkno	Dwight A	New, impo	Companie	Frank Cooper	able
20			AAAAA	1999-08-17	2001-08-15	site_1	2450730	2443430	Unkno	Dwight A	Right effor	Companie	Casey Banks	able
2			AAAAA	2001-08-16		site_1	2450730	2443430	Unkno	Richard F	Right effor	Acres see	Casey Banks	able
2			AAAAA	1997-08-16		site_2	2450705		Unkno	John Tho	About rura	Political af	James Brewer	anti
2		14	AAAAA	1997-08-16	2000-08-15	site_2	2450696	2443396	Unkno	Robert Ar	Necessary,	Associatio	Gilbert Chapman	anti
24			AAAAA	2000-08-16		site_2	2450696	2443396	Unkno	James Au	Necessary,	Associatio	Zachery Oneil	anti

4. 通过表的数量、记录的数量和典型作业的查询结果,校验迁移至MaxCompute的数据是否和BigQuery的数据一致。

2.15. 日志数据迁移至MaxCompute

2.15.1. 概述

日常工作中,企业通常会对实时日志数据进行开发。其中:日志数据来源可以为ECS、容器、移动端、开源软件、网站服务或JavaScript。本文为您介绍如何通过Tunnel、DataHub、LogHub以及DataWorks数据集成将日志数据迁移至MaxCompute。

方案	说明	适用场景
Tunnel	通过MaxCompute的Tunnel功能,将日志数据上传 至MaxCompute。 详情请参见通过Tunnel迁移日志数据至 MaxCompute。	Tunnel主要用于批量上传数据至离线表,适用于离 线计算的场景。

方案	说明	适用场景
DataHub	DataHub数据迁移功能通过Connector实现。 DataHub Connector可以将DataHub中的流式数据 同步至MaxCompute。您只需要向DataHub中写入 数据,并在DataHub中配置同步功能,便可以在 MaxCompute中使用这些数据。 详情请参见通过DataHub迁移日志数据至 MaxCompute。	此方法多用于公测和自研。DataHub用于实时上传数 据,主要适用于流式计算场景。 数据上传后会保存到实时表,后续会在几分钟内通过 定时任务的形式同步到MaxCompute离线表,供离 线计算使用。
DataWorks数据集成	通过DataWorks配置离线同步节点和同步任务将日志 数据同步至MaxCompute。 详情请参见 <u>通过DataWorks数据集成迁移日志数据至</u> MaxCompute。	此方法为定时任务,配置一次可以多次执行同步任 务。

2.15.2. 通过Tunnel迁移日志数据至MaxCompute

本文为您介绍如何通过Tunnel上传日志数据至MaxCompute。

前提条件

- 安装MaxCompute客户端,详情请参见安装并配置MaxCompute客户端。
- 将日志数据保存至本地。本文使用的示例数据为loghub.csv。

背景信息

Tunnel是MaxCompute的批量上传数据工具,适用于离线计算场景。Tunnel详细信息请参见Tunnel使用说明。

操作步骤

1. 在MaxCompute客户端(odpscmd)执行如下命令创建表 loghub,用于存储上传的日志数据。

```
---打开新类型数据开关,此命令需要和SQL语句一起提交。
```

```
set odps.sql.type.system.odps2=true;
--创建表loghub。
CREATE TABLE loghub
(
client_ip STRING,
receive_time STRING,
topic STRING,
id STRING,
name VARCHAR(32),
salenum STRING
);
```

2. 执行如下命令将日志数据上传至MaxCompute。

Tunnel u D:\loghub.csv loghub;

上述命令中需要指定如下两个参数:

- D:\loghub.csv:本地日志数据文件存储路径。
- loghub: MaxCompute中存储日志数据的表名。

⑦ 说明 使用Tunnel数据不支持通配符或正则表达式。

3. 执行如下命令查询数据是否成功导入至表中。

SELECT * FROM loghub;

返回结果如下,表示导入成功。



2.15.3. 通过DataHub迁移日志数据至MaxCompute

本文为您介绍如何通过Dat aHub迁移日志数据至MaxCompute。

前提条件

授权访问MaxCompute的账号已开通以下权限:

• MaxCompute中项目的CreateInstance权限。

• MaxCompute中表的查看、修改和更新权限。

授权操作详情请参见<mark>权限列表</mark>。

背景信息

DataHub是阿里云流式数据(Streaming Data)的处理平台。数据上传至DataHub后保存在实时表里,后续会在5分钟内通过定时任务的形式同步到 MaxCompute离线表里,供离线计算使用。

您只需要创建DataHub Connector,指定相关配置,即可创建将DataHub中流式数据定期归档到MaxCompute的同步任务。

操作步骤

1. 在MaxCompute客户端(odpscmd)执行如下语句创建表,用于存储DataHub同步的日志数据。示例语句如下。

CREATE TABLE test(f1 string, f2 string, f3 double) partitioned by (ds string);

- 2. 在DataHub上创建项目。
 - i. 登录DataHub控制台,在左上角选择地域。
 - ii. 在左侧导航栏, 单击**项目管理**。
 - iii. 在**项目列表**区域,单击右上角**新建项目**。
 - iv. 在新建项目页面,填写名称及描述,单击创建。
- 3. 创建Topic。
 - i. 在**项目列表**区域,单击项目右侧的**查看**。
 - ii. 在项目详情页面,单击右上角**新建Topic**,进入**新建Topic**页面,填写配置参数。

新建Topic		\times
创建方式	○ 直接创建 ● 导入MaxCompute表结构	
Project名称	MaxCompute Project名称	
Table名称	MaxCompute Table名称	
AccessId	訪问MaxCompute的AccessId	
AccessKey	访问MaxCompute的AccessKey	
	F	步

iii. 单击下一步,完善Topic信息。

? 说明

- Schema与MaxCompute表为对应关系,DataHub Topic Schema的字段名称、类型和顺序必须与MaxCompute表字段完全一致, 如果三个条件中的任意一个不满足,则Connector创建失败。
- 支持将TUPLE和BLOB类型的DataHub Topic同步到MaxCompute表中。
- 系统默认最多可以创建20个Topic。如果需要创建更多,请提交工单申请。
- DataHub Topic的Owner或Creator账号有操作Connector的权限,包括创建和删除等。
- 4. 向新建的Topic中写入数据。
 - i. 单击新建Topic右侧的查看。
 - ii. 在Topic详情页面,单击右上角**同步**。
 - iii. 在新建Connector页面,单击MaxCompute,配置相关参数后,单击创建。
- 5. 查看Connector详细信息。
 - i. 在左侧导航栏, 单击**项目管理**。
 - ii. 在**项目列表**区域,单击项目右侧的查看。
 - iii. 在Topic列表区域,单击Topic右侧的查看。
 - iv. 在Topic详情页面, 单击Connector页签, 查看创建好的Connector。
 - v. 单击Connector右侧的查看,即可查看Connector详细信息。

DataHub默认每五分钟或当数据达到60 MB时强行向MaxCompute离线表中投递一次数据。同步点位代表已经同步数据的条数。

详情:		0.00000000	G					×
同步点位	2020年7月14日 15:0	7:18	同步延迟(秒)	861			
TimestampUnit	MICROSECOND		Endpoint					
Project			Table		in the second			
分区模式			分区字段					
ID 🕸	状态 ♪	同步时间 小		同步点	位♪	脏数据量	操作	
0	EXECUTING	2020年7月14日 15:07:18		-1		0	重启	
						重启	停止	

6. 执行如下语句测试日志数据是否投递成功。

SELECT * FROM test; 返回结果如下,表示投递成功。 运行日志 结果[1] В Α 1 id 위 name ✓ salenum 三星1 1000 13 30 14 31 三星2 3000 三星2 606 16 32 二星3 2000 锤子 中兴 40 19 60 三星11 909 20 61 三星12 606 三星13 21 62 2000

2.15.4. 通过DataWorks数据集成迁移日志数据至MaxCompute

本文为您介绍如何通过数据集成功能同步LogHub数据至MaxCompute。

背景信息

日志服务支持以下数据同步场景:

• 跨地域的LogHub与MaxCompute等数据源的数据同步。

- 不同阿里云账号下的LogHub与MaxCompute等数据源间的数据同步。
- 同一阿里云账号下的LogHub与MaxCompute等数据源间的数据同步。
- 公共云与金融云账号下的LogHub与MaxCompute等数据源间的数据同步。

以B账号进入数据集成配置同步任务,将A账号的LogHub数据同步至B账号的MaxCompute为例,跨阿里云账号的特别说明如下:

1. 使用A账号的AccessKey ID和AccessKey Secret创建LogHub数据源。

此时B账号可以同步A账号下所有日志服务项目的数据。

- 2. 使用A账号下的RAM用户A1的AccessKey ID和AccessKey Secret创建LogHub数据源。
 - o A账号为RAM用户A1赋予日志服务的通用权限,即 AliyunLogFullAccess 和 AliyunLogReadOnlyAccess ,详情请参见创建RAM用户及授 权。
 - A账号给RAM用户A1赋予日志服务的自定义权限。

```
主账号A进入RAM控制台 > 权限管理 > 权限策略管理页面,单击新建授权策略。
```

相关的授权请参见访问控制RAM和RAM子用户访问。

根据下述策略进行授权后,B账号通过RAM用户A1只能同步日志服务project_name1以及project_name2的数据。

```
"Version": "1",
"Statement": [
"Action": [
"log:Get*",
"log:List*",
"log:CreateConsumerGroup",
"log:UpdateConsumerGroup",
"log:DeleteConsumerGroup",
"log:ListConsumerGroup",
"log:ConsumerGroupUpdateCheckPoint",
"log:ConsumerGroupHeartBeat",
"log:GetConsumerGroupCheckPoint"
],
"Resource": [
"acs:log:*:*:project/project_name1",
"acs:log:*:*:project/project name1/*",
"acs:log:*:*:project/project_name2",
"acs:log:*:*:project/project_name2/*"
],
"Effect": "Allow"
}
```

新建LogHub数据源

- 1. 登录DataWorks控制台,单击相应工作空间后的进入数据集成。
- 2. 单击左侧导航栏中的数据源,即可跳转至工作空间管理 > 数据源管理页面。
- 3. 在**数据源管理**页面,单击右上角的**新增数据源**。
- 4. 在新增数据源对话框中,选择数据源类型为LogHub。
- 5. 填写新增LogHub数据源对话框中的配置。

参数	描述					
数据源名称	数据源名称必须以字母、数字、下划线组合,且不能以数字和下划线开头。					
数据源描述	对数据源进行简单描述,不得超过80个字符。					
LogHub Endpoint	LogHub的Endpoint,格式为 http://example.com 。详情请参见服务入口。					
Project	输入项目名称。					
AccessKey ID	访问密钥中的AccessKey ID,您可以进入控制台的 用户信息管理 页面进行复制。					
AccessKey Secret	访问密钥中的AccessKey Secret , 相当于登录密码。					

- 6. 单击测试连通性。
- 7. 连通性测试通过后,单击完成。

新建离线同步节点

- 1. 在数据源页面,单击左上角的图标,选择全部产品 > DataStudio(数据开发)。
- 2. 在**数据开发**页面,鼠标悬停至<mark>、</mark>图标,单击**业务流程**。
- 3. 在新建业务流程对话框中,输入业务流程名称和描述,单击新建。
- 4. 展开业务流程,右键单击数据集成,选择新建 > 离线同步。
- 5. 在新建节点对话框中, 输入节点名称, 并选择目标文件夹。
- 6. 单击**提交**,进入离线节点编辑页面。

通过向导模式配置同步任务

1. 在离线节点编辑页面,选择数据来源。



参数	描述
数据源	输入LogHub数据源的名称。
Logstore	导出增量数据的表的名称。该表需要开启Stream,可以在建表时开启,或者使用UpdateTable接口开 启。
日志开始时间	数据消费的开始时间位点,为时间范围(左闭右开)的左边界,为yyyyMMddHHmmss格式的时间字符 串(例如20180111013000)。该参数可以和DataWorks的调度时间参数配合使用。
日志结束时间	数据消费的结束时间位点,为时间范围(左闭右开)的右边界,为yyyyMMddHHmmss格式的时间字符 串(例如20180111013010)。该参数可以和DataWorks的调度时间参数配合使用。
批量条数	一次读取的数据条数,默认为256。

⑦ 说明 您可以进行数据预览,此处仅选择LogHub中的几条数据展现在预览框。由于您在进行同步任务时,会指定开始时间和结束时间, 会导致预览结果和实际的同步结果不一致。

- 2. 选择MaxCompute数据源及目标表。
- 3. 选择字段的映射关系。
- 4. 在通道控制中配置作业速率上限和脏数据检查规则。
- 5. 确认当前节点的配置无误后,单击左上角的保存。
- 6. 运行离线同步节点。

您可以通过以下两种方式运行离线同步节点:

○ 直接运行(一次性运行)

单击节点编辑页面工具栏中的运行图标,直接在页面运行。

⑦ 说明 运行之前需要配置自定义参数的具体取值。

。 调度运行

单击节点编辑页面工具栏中的**提交**图标,提交离线同步节点至调度系统,调度系统会根据配置的属性,从第2天开始自动定时运行。

×				调度
基础属性 ② 一				
				血缘
				ぞ 关 系
	描述:			版本
	参数:	startTime=\$[yyyymmddhh24miss-10/24/60] endTime=\$[yyyymmddhh24miss-5/24/60]	?	<i>(</i> ±
				約

如上图所示,设置开始时间为系统前10分钟,结束时间为系统前5分钟:startTime=\$[yyyymmddhh24miss-10/24/60] endTime=\$[yyyymmddhh24miss-5/24/60]。

X 调度配置		调
时间属性 🕐 -		配置
生成实例方式:	🧿 T+1次日生成 🔷 发布后即时生成	
时间属性:	● 正常调度 ● 空跑调度	版本
出错重试:		
生效日期:	1970-01-01 💼	
暂停调度:		
调度周期:	分钟 🗸	
定时调度:		
开始时间:	00:00	
时间间隔:	05 〇 分钟	
结束时间:	23.59 🕥	
cron表达式:	00*/5 00-23**?	
依赖上—周期:		

如上图所示,设置离线同步节点的调度周期为分钟,从00:00~23:59每5分钟调度一次。

通过脚本模式配置离线同步节点

1. 成功创建离线同步节点后,单击工具栏中的转换脚本。

\$	💥 DataStudio		■ ~						
	数据开发 2 閲 C O	L)	□ 离线同步	×					
*	Q 文件名称/创建人	V.		Þ	ſ	ځ		Ø	
Q	> 解决方案	00 00							在这里
G	◆ 业务流程		01 选择数	居源				数	民来源

- 2. 单击提示对话框中的确认,即可进入脚本模式进行开发。
- 3. 单击工具栏中的导入模板。

Di write	e_result	•			
Ľ	\odot	Þ	ſ	ل	🗟 💽 🗱
1 2	{	"type	": "j	ob",	导入模板

- 4. 在导入模板对话框中,选择从来源端的LogHub数据源同步至目标端的ODPS数据源的导入模板,单击确认。
- 5. 导入模板后,根据自身需求编辑代码,示例脚本如下。

```
{
"type": "job",
"version": "1.0",
"configuration": {
"reader": {
"plugin": "loghub",
"parameter": {
"datasource": "loghub_lzz",//数据源名,需要和您添加的数据源名一致。
"logstore": "logstore-ut2",//目标日志库的名字,LogStore是日志服务中日志数据的采集、存储和查询单元。
"beginDateTime": "${startTime}",//数据消费的开始时间位点,为时间范围(左闭右开)的左边界。
"endDateTime": "${endTime}",//数据消费的结束时间位点,为时间范围(左闭右开)的右边界。
"batchSize": 256,//一次读取的数据条数,默认为256。
"splitPk": "",
"column": [
"key1",
"key2",
"key3"
]
}
},
"writer": {
"plugin": "odps",
"parameter": {
"datasource": "odps_first",//数据源名,需要和您添加的数据源名一致。
"table": "ok",//目标表名。
"truncate": true,
"partition": "",//分区信息。
"column": [//目标列名。
"key1",
"key2",
"key3"
]
}
},
"setting": {
"speed": {
"mbps": 8,//作业速率上限。
"concurrent": 7//并发数。
}
}
}
}
```

3.数据开发 3.1.日期数据格式转换:STRING、TIMESTAMP、 DATETIME互相转换

本文为您介绍STRING、TIMESTAMP与DATETIME类型数据间的转换方法,帮助您在实际业务处理过程中,快速对标找到合适的日期转换方法,提升业 务处理效率。

日期数据格式转换常见场景如下:

- STRING转换为TIMESTAMP
- STRING转换为DATETIME
- TIMESTAMP转换为STRING
- TIMESTAMP转换为DATETIME
- DATETIME转换为TIMESTAMP
- DATETIME转换为STRING

STRING转换为TIMESTAMP

• 应用场景

将STRING类型数据转换为TIMESTAMP类型(格式为 yyyy-mm-dd hh:mi:ss.ff3)的日期值。

• 实现方法

使用CAST函数进行转换。

● 使用限制

输入的STRING类型数据的格式至少要满足 yyyy-mm-dd hh:mi:ss 要求。

- 使用示例
 - 示例1: 使用CAST函数,将STRING类型数据 2009-07-01 16:09:00 转换为TIMESTAMP类型。命令示例如下。

```
--返回2009-07-01 16:09:00.000。
select cast('2009-07-01 16:09:00' as timestamp);
```

○ 示例2: 错误使用CAST函数的命令示例如下。

--返回NULL。输入数据格式不满足要求。至少要包含yyyy-mm-dd hh:mi:ss格式。 select cast('2009-07-01' as timestamp);

STRING转换为DATETIME

● 应用场景

将STRING类型数据转换为DATETIME类型(格式为 yyyy-mm-dd hh:mi:ss)的日期值。

- 实现方法
 - 方法一:使用CAST函数进行转换。
 - 方法二: 使用TO DATE函数进行转换。
- 使用限制
 - 使用CAST函数时, 输入的STRING类型数据的格式必须要满足 yyyy-mm-dd hh:mi:ss 要求。
 - 使用TO_DATE函数时,需要指定format参数的取值为 yyyy-mm-dd hh:mi:ss 。
- 使用示例
 - 示例1: 使用CAST函数,将STRING类型数据 2009-07-01 16:09:00 转换为DATETIME类型。命令示例如下。

```
--返回2009-07-01 16:09:00。
select cast('2009-07-01 16:09:00' as datetime);
```

○ 示例2: 使用TO_DATE函数,指定format参数,将STRING类型数据 2009-07-01 16:09:00 转换为DATETIME类型。命令示例如下。

--返回2009-07-01 16:09:00。 select to_date('2009-07-01 16:09:00','yyyy-mm-dd hh:mi:ss');

○ 示例3: 错误使用CAST函数的命令示例如下。

```
--返回NULL。输入数据格式不满足要求。必须为yyyy-mm-dd hh:mi:ss格式。
select cast('2009-07-01' as datetime);
```

```
○ 示例4: 错误使用TO_DATE函数的命令示例如下。
```

```
--返回NULL。输入数据格式不满足要求。必须为yyyy-mm-dd hh:mi:ss格式。
select to_date('2009-07-01','yyyy-mm-dd hh:mi:ss');
```

TIMESTAMP转换为STRING

● 应用场景

将TIMESTAMP类型(格式为 yyyy-mm-dd hh:mi:ss.ff3)的日期值转换为STRING类型。

- 实现方法
 - 方法一:使用CAST函数进行转换。
 - 方法二:使用TO_CHAR函数按照format参数指定的格式进行转换。
- 使用示例
 - 示例1:使用CAST函数,将TIMESTAMP类型数据
 2009-07-01 16:09:00
 转换为STRING类型。为构造TIMESTAMP类型数据,总共需要使用2次
 CAST函数。命令示例如下。

```
--返回2009-07-01 16:09:00。
select cast(cast('2009-07-01 16:09:00' as timestamp) as string);
```

示例2:使用TO_CHAR函数,将TIMESTAMP类型数据 2009-07-01 16:09:00 转换为STRING类型。为构造TIMESTAMP类型数据,需要使用到1次 CAST函数。命令示例如下。

```
--返回2009-07-01 16:09:00。
select to_char(cast('2009-07-01 16:09:00' as timestamp),'yyyy-mm-dd hh:mi:ss');
```

TIMESTAMP转换为DATETIME

● 应用场景

将TIMESTAMP类型(格式为 yyyy-mm-dd hh:mi:ss.ff3)的日期值转换为DATETIME类型(格式为 yyyy-mm-dd hh:mi:ss)的日期值。

- 实现方法
 - 方法一:使用CAST函数进行转换。
 - 方法二: 使用TO_DATE函数进行转换。
- 使用限制

使用TO_DATE函数时,需要指定format参数的取值为 yyyy-mm-dd hh:mi:ss 。

- 使用示例
 - 示例1:使用CAST函数,将TIMESTAMP类型数据
 2009-07-01 16:09:00
 转换为DATETIME类型。为构造TIMESTAMP类型数据,总共需要使用2次
 CAST函数。命令示例如下。

```
--返回2009-07-01 16:09:00。
```

select cast(cast('2009-07-01 16:09:00' as timestamp) as datetime);

示例2:使用TO_DATE函数,指定format参数,将TIMESTAMP类型数据
 2009-07-01 16:09:00
 转换为DATETIME类型。为构造TIMESTAMP类型数据,需要使用到1次CAST函数。命令示例如下。

```
--返回2009-07-01 16:09:00。
```

select to_date(cast('2009-07-01 16:09:00' as timestamp),'yyyy-mm-dd hh:mi:ss');

DATETIME转换为TIMESTAMP

• 应用场景

将DATETIME类型(格式为 yyyy-mm-dd hh:mi:ss)的日期值转换为TIMESTAMP类型(格式为 yyyy-mm-dd hh:mi:ss.ff3)的日期值。

- 实现方法
 - 使用CAST函数进行转换。
- 使用示例

使用CAST函数,将DATETIME类型的日期值转换为TIMESTAMP类型。为构造DATETIME类型数据,需要使用到1次GETDATE函数。命令示例如下。

```
--返回2021-10-14 10:21:47.939。
select cast(getdate() as timestamp);
```

```
DATETIME转换为STRING
```

```
● 应用场景
```

将DATETIME类型(格式为 yyyy-mm-dd hh:mi:ss)的日期值转换为STRING类型。

- 实现方法
 - 方法一:使用CAST函数进行转换。
- 方法二:使用TO_CHAR函数按照format参数指定的格式进行转换。
- 使用示例
 - 示例1:使用CAST函数,将DATETIME类型的日期值转换为STRING类型。为构造DATETIME类型数据,需要使用到1次GETDATE函数。命令示例如 下。

```
--返回2021-10-14 10:21:47。
select cast(getdate() as string);
```

○ 示例2:使用TO_CHAR函数,将DATETIME类型的日期值转换为指定格式的STRING类型。为构造DATETIME类型数据,需要使用到1次GETDATE函数。命令示例如下。

```
--返回2021-10-14 10:21:47。
select to_char (getdate(),'yyyy-mm-dd hh:mi:ss');
--返回2021-10-14。
select to_char (getdate(),'yyyy-mm-dd');
--返回2021。
select to_char (getdate(),'yyyy');
```

3.2. 基于MaxCompute UDF将IPv4或IPv6地址转换为归属 地

随着大数据平台发展,现已可以处理多类型的非结构化、半结构化数据,其中将IP地址转换为归属地是常见的一种场景。本文为您介绍如何通过 MaxCompute UDF实现将IPv4或IPv6地址转换为归属地。

前提条件

请确认已满足如下条件:

● 已安装MaxCompute客户端。

更多安装并配置MaxCompute客户端信息,请参见安装并配置MaxCompute客户端。

• 已安装MaxCompute Studio、已连接MaxCompute项目、已创建MaxCompute Java Module。

更多操作信息,请参见安装MaxCompute Studio、管理项目连接和创建MaxCompute Java Module。

背景信息

要实现将IPv4或IPv6地址转换为归属地,必须要有IP地址库,您需要下载IP地址库文件并以资源形式上传至MaxCompute项目。开发MaxCompute UDF,并基于IP地址库文件注册函数,从而在SQL语句中调用函数将IP地址转换为归属地。

注意事项

本文提供的IP地址库文件,仅供验证该最佳实践使用,请您结合实际业务情况,自行维护IP地址库文件。

操作流程

基于MaxCompute UDF将IPv4或IPv6地址转换为归属地的操作流程如下:

1. 步骤一:上传IP地址库文件

将IP地址库文件作为资源上传至MaxCompute项目,后续创建的MaxCompute UDF会依赖此资源。

- 2. 步骤二:编写MaxCompute UDF
- 在IntelliJ IDEA上编写MaxCompute UDF代码。
- 3. 步骤三: 注册MaxCompute UDF

将MaxCompute UDF注册为函数。

4. 步骤四:调用MaxCompute UDF转换IP地址为归属地

在SQL语句中调用注册好的函数将IP地址转换为归属地。

步骤一:上传IP地址库文件

- 下载IP地址库文件至本地,解压得到ipv4.txt和ipv6.txt,并放置于MaxCompute客户端的安装目录 ...\odpscmd_public\bin 下。
 本文提供的IP地址库文件,仅供验证该最佳实践使用,请您结合实际业务情况,自行维护IP地址库文件。
- 2. 登录MaxCompute客户端,进入目标MaxCompute项目。
- 3. 执行 add file 命令,将ipv4.txt和ipv6.txt以File类型资源上传至MaxCompute项目。

命令示例如下。

```
add file ipv4.txt -f;
add file ipv6.txt -f;
```

更多添加资源信息,请参见<mark>添加资源</mark>。

步骤二:编写MaxCompute UDF

1. 创建Java Class对象。

后续步骤中编写的MaxCompute UDF代码会用到此处创建的Java Class。

i. 进入Intellij IDEA界面,在Project区域,右键单击Module的源码目录(即src > main > java),选择new > Java Class。



ii. 在New Java Class对话框,输入Class名称,按下Enter键并在代码编辑区域输入代码。

您需要依次创建3个Java Class对象,Class名称及对应代码如下,代码可直接复制使用,无需修改。

IpUtils

```
package com.aliyun.odps.udf.utils;
import java.math.BigInteger;
import java.net.Inet4Address;
import java.net.Inet6Address;
import java.net.InetAddress;
import java.net.UnknownHostException;
import java.util.Arrays;
public class IpUtils {
   /**
    * 将字符串形式的ip地址转换为long
    * @param ipInString
    * 字符串形式的ip地址
    * @return 返回long形式的ip地址
    */
   public static long StringToLong(String ipInString) {
      ipInString = ipInString.replace(" ", "");
       byte[] bytes;
       if (ipInString.contains(":"))
          bytes = ipv6ToBytes(ipInString);
       else
           bytes = ipv4ToBytes(ipInString);
      BigInteger bigInt = new BigInteger(bytes);
        System.out.println(bigInt.toString());
       return bigInt.longValue();
   }
   /**
    * 将字符串形式的ip地址转换为long
    * @param ipInString
   * 字符串形式的ip地址
```

```
* @return bigint的string形式的ip地址
 */
public static String StringToBigIntString(String ipInString) {
   ipInString = ipInString.replace(" ", "");
   byte[] bytes;
   if (ipInString.contains(":"))
       bytes = ipv6ToBytes(ipInString);
   else
       bytes = ipv4ToBytes(ipInString);
   BigInteger bigInt = new BigInteger(bytes);
   return bigInt.toString();
}
/**
 * 将整数形式的ip地址转换为字符串形式
 * @param ipInBigInt
 * 整数形式的ip地址
 * @return 字符串形式的ip地址
 */
public static String BigIntToString(BigInteger ipInBigInt) {
   byte[] bytes = ipInBigInt.toByteArray();
   byte[] unsignedBytes = Arrays.copyOfRange(bytes, 1, bytes.length);
   // 去除符号位
   try {
       String ip = InetAddress.getByAddress(unsignedBytes).toString();
        return ip.substring(ip.indexOf('/') + 1).trim();
   } catch (UnknownHostException e) {
       throw new RuntimeException(e);
    3
}
/**
 * ipv6地址转有符号byte[17]
 */
private static byte[] ipv6ToBytes(String ipv6) {
   byte[] ret = new byte[17];
   ret[0] = 0;
   int ih = 16:
   boolean comFlag = false;// ipv4混合模式标记
   if (ipv6.startsWith(":"))// 去掉开头的冒号
       ipv6 = ipv6.substring(1);
   String groups[] = ipv6.split(":");
    for (int ig = groups.length - 1; ig > -1; ig--) {// 反向扫描
       if (groups[ig].contains(".")) {
           // 出现ipv4混合模式
           byte[] temp = ipv4ToBytes(groups[ig]);
           ret[ib--] = temp[4];
           ret[ib--] = temp[3];
           ret[ib--] = temp[2];
           ret[ib--] = temp[1];
           comFlag = true;
        } else if ("".equals(groups[ig])) {
           // 出现零长度压缩,计算缺少的组数
           int zlg = 9 - (groups.length + (comFlag ? 1 : 0));
           while (zlg-- > 0) {// 将这些组置0
               ret[ib--] = 0;
               ret[ib--] = 0;
           }
       } else {
           int temp = Integer.parseInt(groups[ig], 16);
           ret[ib--] = (byte) temp;
ret[ib--] = (byte) (temp >> 8);
       }
    }
   return ret;
}
/**
 * IPv4地址转有符号byte[5]
 */
private static byte[] ipv4ToBytes(String ipv4) {
   byte[] ret = new byte[5];
    ret[0] = 0;
   // 先找到ip地址字符串中.的位置
   int position1 = ipv4.indexOf(".");
int position2 = ipv4.indexOf(".", position1 + 1);
    int position3 = ipv4.indexOf(".", position2 + 1);
```

```
// 将每个.之间的字符串转换成整型
    ret[1] = (byte) Integer.parseInt(ipv4.substring(0, position1));
    ret[2] = (byte) Integer.parseInt(ipv4.substring(position1 + 1,
           position2));
    ret[3] = (byte) Integer.parseInt(ipv4.substring(position2 + 1,
           position3));
    ret[4] = (byte) Integer.parseInt(ipv4.substring(position3 + 1));
    return ret;
}
/**
 * @param ipAdress ipv4或ipv6字符串
 * @return 4:ipv4, 6:ipv6, 0:地址不对
 * @throws Exception
 */
public static int isIpV4OrV6(String ipAdress) throws Exception {
   InetAddress address = InetAddress.getByName(ipAdress);
   if (address instanceof Inet4Address)
        return 4;
    else if (address instanceof Inet6Address)
       return 6;
    return 0;
}
/*
 * 验证ip是否属于某个IP段
 * ipSection ip段 (以'-'分隔)
 * ip 所验证的ip号码
 * /
public static boolean ipExistsInRange(String ip, String ipSection) {
   ipSection = ipSection.trim();
    ip = ip.trim();
   int idx = ipSection.indexOf('-');
   String beginIP = ipSection.substring(0, idx);
   String endIP = ipSection.substring(idx + 1);
    return getIp2long(beginIP) <= getIp2long(ip)</pre>
           && getIp2long(ip) <= getIp2long(endIP);
}
public static long getIp2long(String ip) {
   ip = ip.trim();
   String[] ips = ip.split("\\.");
    long ip2long = 0L;
    for (int i = 0; i < 4; ++i) {
       ip2long = ip2long << 8 | Integer.parseInt(ips[i]);</pre>
    }
    return ip2long;
}
public static long getIp2long2(String ip) {
   ip = ip.trim();
    String[] ips = ip.split("\\.");
   long ip1 = Integer.parseInt(ips[0]);
    long ip2 = Integer.parseInt(ips[1]);
   long ip3 = Integer.parseInt(ips[2]);
    long ip4 = Integer.parseInt(ips[3]);
    long ip2long = 1L * ip1 * 256 * 256 * 256 + ip2 * 256 * 256 + ip3 * 256
          + ip4;
    return ip2long;
}
public static void main(String[] args) {
    System.out.println(StringToLong("2002:7af3:f3be:ffff:ffff:ffff:ffff:ffff"));
    System.out.println(StringToLong("54.38.72.63"));
}
private class Invalid{
   private Invalid()
    {
    }
}
```

}

■ lpV40bj

```
package com.aliyun.odps.udf.objects;
public class IpV4Obj {
   public long startIp ;
   public long endIp ;
   public String city;
   public String province;
   public IpV4Obj(long startIp, long endIp, String city, String province) {
       this.startIp = startIp;
       this.endIp = endIp;
       this.city = city;
       this.province = province;
    }
    @Override
   public String toString() {
       return "IpV4Obj{" +
               "startIp=" + startIp +
               ", endIp=" + endIp +
", city='" + city + '\'' +
               ", province='" + province + '\'' +
               '}';
    }
   public void setStartIp(long startIp) {
       this.startIp = startIp;
    }
   public void setEndIp(long endIp) {
       this.endIp = endIp;
    }
   public void setCity(String city) {
       this.city = city;
    }
   public void setProvince(String province) {
      this.province = province;
    }
    public long getStartIp() {
       return startIp;
    }
   public long getEndIp() {
       return endIp;
    }
   public String getCity() {
       return city;
    }
   public String getProvince() {
      return province;
    }
}
```

∎ lpV60bj

```
package com.aliyun.odps.udf.objects;
public class IpV60bj {
  public String startIp ;
   public String endIp ;
   public String city;
   public String province;
   public String getStartIp() {
       return startIp;
    }
   @Override
   public String toString() {
       return "IpV6Obj{" +
               "startIp='" + startIp + '\'' +
", endIp='" + endIp + '\'' +
", city='" + city + '\'' +
                ", province='" + province + '\'' +
                '}';
   }
   public IpV60bj(String startIp, String endIp, String city, String province) {
       this.startIp = startIp;
       this.endIp = endIp;
       this.city = city;
       this.province = province;
    }
   public void setStartIp(String startIp) {
       this.startIp = startIp;
    }
   public String getEndIp() {
       return endIp;
    }
   public void setEndIp(String endIp) {
       this.endIp = endIp;
    }
    public String getCity() {
       return city;
    }
   public void setCity(String city) {
       this.city = city;
    }
   public String getProvince() {
       return province;
    }
   public void setProvince(String province) {
       this.province = province;
    }
}
```

2. 编写MaxCompute UDF代码。

	<u>File E</u> dit <u>V</u> iew <u>N</u> avigate <u>C</u> ode	Analyze <u>R</u> efactor <u>B</u> uild R <u>u</u> n <u>T</u> ools VC <u>S</u> <u>W</u> indov	w MaxCompute Help MCUDF - IpV4C
N	MCUDF 〉 udf 〉 src 〉 main 〉 java 〉 ④	lpV4Obj	
1. Z: Structure	Project * MCUDF idea out scripts target udf examples oc.adjun.odps.exa oc.adjun.odps.exa	New 2 X Cut Ctrl+X Copy ▶ D Paste Ctrl+V Find Usages Alt+F7 Find in Path Ctrl+Shift+F Replace in Path Ctrl+Shift+R	 Java Class Kotlin File/Class File Scratch File Ctrl+Alt+Shift+Insert Package Python Package FXML File acchage info inva
orer 😚 Project Explorer	src main publictest.osql test test	Analyze ▶ Refactor ▶ Clean Python Compiled Files ▶ Add to Favorites ▶ Reformat Code Ctrl+Alt+L Optimize Imports Ctrl+Alt+O Delete Delete	Pockage Intojata Python File Python File MaxCompute Python MaxCompute Java MaxCompute SQL 脚本 HTML File Kotlin Script
5 Job Expl	org > target mpom.xml ill udf.iml v warehouse v example_project > resources_	Build Module 'udf' Rebuild ' <default>' Ctrl+Shift+F9 ▶ Run 'All Tests' Ctrl+Shift+F10 ऄ Debug 'All Tests' \$\$ Run 'All Tests' with Coverage</default>	lige Kotlin Worksheet and JavaFXApplication Edit File Templates
vorites	▶ ■ ads_log ▼ ■ kmeans_in	 Create 'All Tests' Show in Explorer Directory <u>Path</u> Open in Terminal 上传到 DataWorks Set MaxCompute project 	
¥ 2: Fan	Process finished w	Local <u>H</u> istory Reload from Disk	
	i≡ <u>6</u> : TODO ▶ <u>4</u> : Run 🗵 Termin	→ [€] Compare With Ctrl+D	

i. 在Project区域,右键单击Module的源码目录(即src > main > java),选择new > MaxCompute Java。

ii. 在Create new MaxCompute java class对话框,单击UDF并填写Name后,按Enter键并在代码编写区域输入代码。



例如Java Class名称为IpLocation。代码内容如下,代码可直接复制使用,无需修改。

```
package com.aliyun.odps.udf.udfFunction;
import com.aliyun.odps.udf.ExecutionContext;
import com.aliyun.odps.udf.UDF;
import com.aliyun.odps.udf.UDFException;
import com.aliyun.odps.udf.utils.IpUtils;
import com.aliyun.odps.udf.objects.IpV4Obj;
import com.aliyun.odps.udf.objects.IpV6Obj;
import java.io.*;
import java.util.ArrayList;
import java.util.Comparator;
import java.util.List;
import java.util.stream.Collectors;
public class IpLocation extends UDF {
   public static IpV4Obj[] ipV4ObjsArray;
   public static IpV6Obj[] ipV6ObjsArray;
   public IpLocation() {
       super();
    }
    @Override
    public void setup(ExecutionContext ctx) throws UDFException, IOException {
       //IPV4
        if(ipV4ObjsArray==null)
```

```
BufferedInputStream bufferedInputStream = ctx.readResourceFileAsStream("ipv4.txt");
           BufferedReader br = new BufferedReader(new InputStreamReader(bufferedInputStream));
           ArrayList<IpV4Obj> ipV4ObjArrayList=new ArrayList<>();
           String line = null;
           while ((line = br.readLine()) != null) {
               String[] f = line.split("\\|", -1);
               if(f.length>=5)
                    long startIp = IpUtils.StringToLong(f[0]);
                   long endIp = IpUtils.StringToLong(f[1]);
                    String city=f[3];
                    String province=f[4];
                    IpV40bj ipV40bj = new IpV40bj(startIp, endIp, city, province);
                    ipV4ObjArrayList.add(ipV4Obj);
               }
            }
           br.close();
           List<IpV40bj> collect = ipV40bjArrayList.stream().sorted(Comparator.comparing(IpV40bj::getStartIp)).collect(
Collectors.toList());
           ArrayList<IpV4Obj> basicIpV4DataList=(ArrayList)collect;
           IpV4Obj[] ipV4Objs = new IpV4Obj[basicIpV4DataList.size()];
           ipV4ObjsArray = basicIpV4DataList.toArray(ipV4Objs);
       }
        //IPV6
       if(ipV6ObjsArray==null)
       {
           BufferedInputStream bufferedInputStream = ctx.readResourceFileAsStream("ipv6.txt");
           BufferedReader br = new BufferedReader(new InputStreamReader(bufferedInputStream));
           ArrayList<IpV6Obj> ipV6ObjArrayList=new ArrayList<>();
           String line = null;
           while ((line = br.readLine()) != null) {
               String[] f = line.split("\\|", -1);
               if(f.length>=5)
               {
                   String startIp = IpUtils.StringToBigIntString(f[0]);
                   String endIp = IpUtils.StringToBigIntString(f[1]);
                    String city=f[3];
                   String province=f[4];
                   IpV6Obj ipV6Obj = new IpV6Obj(startIp, endIp, city, province);
                    ipV6ObjArrayList.add(ipV6Obj);
               }
           }
           br.close();
           List<IpV60bj> collect = ipV60bjArrayList.stream().sorted(Comparator.comparing(IpV60bj::getStartIp)).collect(
Collectors.toList());
           ArrayList<IpV6Obj> basicIpV6DataList=(ArrayList)collect;
           IpV6Obj[] ipV6Objs = new IpV6Obj[basicIpV6DataList.size()];
           ipV6ObjsArray = basicIpV6DataList.toArray(ipV6Objs);
       }
   }
   public String evaluate(String ip) {
       if(ip==null||ip.trim().isEmpty()||!(ip.contains(".")||ip.contains(":")))
        {
           return null;
       }
       int ipV40rV6=0;
       try {
           ipV4OrV6= IpUtils.isIpV4OrV6(ip);
        } catch (Exception e) {
           return null;
       //如果是IPv4
       if(ipV40rV6==4)
        {
           int i = binarySearch(ipV4ObjsArray, IpUtils.StringToLong(ip));
           if(i>=0)
           {
               IpV4Obj ipV4Obj = ipV4ObjsArray[i];
               return ipV4Obj.city+","+ipV4Obj.province;
            }else{
               return null;
           3
        }else if(ipV4OrV6==6)//如果是IPv6
        {
            int i = binarySearchTPV6(inV6ObisArray. InUtils StringToBigIntString(in));
```

```
if(i>=0)
           {
              IpV6Obj ipV6Obj = ipV6ObjsArray[i];
              return ipV6Obj.city+","+ipV6Obj.province;
           }else{
             return null;
          }
       }else{//如果不符合IPv4或IPv6格式
          return null;
       }
   }
   @Override
   public void close() throws UDFException, IOException {
      super.close();
   }
   private static int binarySearch(IpV4Obj[] array,long ip){
      int low=0;
       int hight=array.length-1;
      while (low<=hight)
      {
           int middle=(low+hight)/2;
          if((ip>=array[middle].startIp)&&(ip<=array[middle].endIp))</pre>
          {
              return middle;
          if (ip < array[middle].startIp)</pre>
             hight = middle - 1;
           else {
              low = middle + 1;
          }
       }
       return -1;
   }
   private static int binarySearchIPV6(IpV6Obj[] array,String ip){
      int low=0;
      int hight=array.length-1;
       while (low<=hight)
       {
           int middle=(low+hight)/2;
          if((ip.compareTo(array[middle].startIp)>=0)&&(ip.compareTo(array[middle].endIp)<=0))</pre>
          {
              return middle;
          }
          if (ip.compareTo(array[middle].startIp) < 0)</pre>
              hight = middle - 1;
           else {
             low = middle + 1;
          }
       }
       return -1;
   }
   private class Invalid{
      private Invalid()
       {
       }
   }
}
```

调试MaxCompute UDF,确保代码可以运行成功。
 更多调试操作,请参见通过本地运行调试UDF。
 i. 右键单击编写完成的MaxCompute UDF脚本,选择Run。

ii. 在Run/Debug Configuratio	ns对话框,配置下图幻	[框所示运行参数,单击 OK 。				
Run/Debug Configurations				×		
+- 🖻 🖯 🖊 🔺 🔻 »	Name: IpLocation	Allow parall	el r <u>u</u> n 📃 <u>S</u> tore as	s project file 🤹		
 Application MaxCompute Java 	Main <u>c</u> lass:	com.aliyun.odps.udf.udfFunction.IpLocat	ion			
	<u>V</u> M options:			+ **		
♦ IpLocation	Program a <u>rg</u> uments:			+ **		
 MaxCompute Python MaxCompute SQL 	Working directory:	(IdeaProjects\Proje	ct2\MCUDF	+ 🖿		
F Templates	Environment variables:					
	Use classpath of m <u>o</u> dule:	📭 udf		•		
		□ Include dependencies with "Provided" scope				
	JRE:	Default (1.8 - SDK of 'udf' module)		-		
	Shorten command <u>l</u> ine:	user-local default: none - java [options]	className [args]	•		
	<u>E</u> nable capturing form	n snapshots				
	*MaxCompute project: lo	cal 🔹	exampoject 🔻	+		
	*MaxCompute table: w	c_in2		•		
	*Table partition: p	2=1,p1=2		ie:p1=1,p2=2		
	*Table columns: co	blc,colb		ie:c1,c2		
	Download Record limit:	100 Data Column Separator: ,				
		_				
?			OK Cancel	Apply		
返回无报错,说明代码运行成」	为,即可继续执行后续;	步骤。如有报错,请按照IntelliJ IDEA	报错信息处理。			
⑦ 说明 运行参数可参照	图示数据填写。					

步骤三:注册MaxCompute UDF

1. 右键单击已经编译成功的MaxCompute UDF脚本,选择**Deploy to server…**。

IJ	<u>F</u> ile	<u>E</u> dit	<u>V</u> iew	<u>N</u> avigate	<u>C</u> ode	Analyz	e	<u>R</u> efactor	<u>B</u> uild	R <u>u</u> n	<u>T</u> ools	VC <u>S</u>	<u>W</u> indov	w M	axCompute
м	ICUDF) ud	f ∕src	angle main $ angle$ ja	ava 👌 🧿	IpLocat	tio	n							
oject	P	roject	Ŧ							e	÷	\$	- 0	Low	er.class $ imes$
nd 🗶 📕	× 🔨	MCU .ic	DF C:\ dea ut	Users\zhao	huiten\l	deaProj	ec	ts\Project2	MCUD	1F			147	0	
= 7: Structure	•	sc ta u	ripts rget d f exam	ples									150 151 152 153		
lorer) 	Co Co da src	m.aliyun.od ıta	lps.exan	nples		New					154		•
Project Exp		~	∎ ma	ain java META-I	NF		× Ö	Cu <u>t</u> Copy <u>P</u> aste						Ctrl+	⊦X ▶ ⊦V
Explorer 🔸				C IpLocat C IpUtils C IpV4Ob C IpV6Ob	ion 9j 9j			Find <u>U</u> sag Analy <u>z</u> e <u>R</u> efactor	es	nniled	Files			Alt+	F7
dol 冬		►	E ter	Cower resources publictest.c	osql			Add to F <u>a</u> Browse Ty <u>R</u> eformat	vorites /pe Hier Code	rarchy			Ctr	Ctrl+	► •H +L
		► ■ m	targe pom.	t xml				Optimi <u>z</u> e <u>D</u> elete	Imports	; P			Ctrl	+Alt+ Dele	•O
	varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse varehouse			►ăĢ	Recompile Run 'IpLoo Debug 'Ip Run 'IpLoo	e 'IpLoc cation.m Locatio	' ation.ja nain()' n.main nain()' y	ava' 0' with Co	verag	Ctrl+S Ctrl+Sh e	hift+ ift+F	F9 10			
orites				ads_log kmeans_in _schen data kmeans_ou schen	na ut na		<u>∿</u> ≥	Edit 'IpLoo Show in E File <u>P</u> ath Open in T Deploy to	cation.m xplorer erminal server.	nain()'			Ctrl+/	Alt+F	12
¥ 2: Fav	Run:	*	IpLoca	<mark>∦</mark> data tion ×	_		G	Local <u>H</u> ist Reload fro	ory om Disk	c					•

2. 在Package a jar, submit resource and register function对话框中, 配置参数信息。

更多参数解释,请参见打包、上传及注册。

Package a jar, submit resource and register function	×
*MaxCompute project: doc_test_dev (service.cn-hangzhou.maxcompute.aliyun.com) +	
*Resource file: \IdeaProjects\Project2\MCUDF\udf\target\udf-1.0-SNAPSHOT.jar	
*Resource name: udf-1.0-SNAPSHOT.jar	
Resource comment:	
lash ose	
Receiver and the second s	
Extra resources: ipv4.bt	
and the second	
*Main class: com.aliyun.odps.udf.udfFunction.lpLocation	-
*Function name: ipv4_ipv6_aton	
✓ Force update if already exists	
? OK Cano	el

Extra resources必须选中步骤一中上传的IP地址库文件ipv4.txt和ipv6.txt。假设注册好的函数名称为ipv4_ipv6_aton。

步骤四:调用MaxCompute UDF转换IP地址为归属地

1. 登录MaxCompute客户端。

- 2. 执行SQL SELECT语句调用MaxCompute UDF将IPv4或IPv6地址转换为归属地。
 - 命令示例如下。
 - 转换IPv4地址为归属地

select ipv4_ipv6_aton('116.11.34.15');

返回结果如下。

北海市,广西壮族自治区

◦ 转换IPv6地址为归属地

select ipv4_ipv6_aton('2001:0250:080b:0:0:0:0:0');

返回结果如下。

保定市,河北省

3.3. IntelliJ IDEA Java UDF开发最佳实践

Intellij IDEA是Java语言的集成开发环境,可以帮助您快速的开发Java程序。本文为您详细介绍如何使用Intellij IDEA的插件MaxCompute Studio进行Java UDF开发,实现大写字母转换为小写字母。

前提条件

请确认已在Intellij IDEA上完成如下准备工作:

- 1. 安装MaxCompute Studio
- 2. 创建MaxCompute项目连接
- 3. 创建MaxCompute Java Module

操作步骤

- 1. 编写Java UDF。
 - i. 在Project区域,右键单击Module的源码目录(即src > main > java),选择new > MaxCompute Java。



ii. 在Create new MaxCompute java class对话框,单击UDF并填写Name后,按Enter键。例如Java Class名称为Lower。



Name为创建的MaxCompute Java Class名称。如果还没有创建Package,在此处填写packagename.classname,会自动生成Package。

iii. 在代码编写区域写入如下代码。



UDF代码示例如下。

```
package <packagename>;
import com.aliyun.odps.udf.UDF;
public final class Lower extends UDF {
    public String evaluate(String s) {
        if (s == null) {
            return null;
        }
            return s.toLowerCase();
    }
}
```

2. 调试UDF,确保可以运行成功。

i. 在Java目录下,右键单击编写完成的Java脚本,选择Run。

ii. 在Run/Debug Configurations对话框,配置运行参数。

- 🖻 🖬 🎤 🔺 🔻 🛤 🖓	Name: MyFirstUDF		Share through VCS ⑦	Allow parallel ru
MaxCompute Java	Main <u>c</u> lass:	MyFirstUDF		
MaxCompute Python	⊻M options:			+ 2
MaxCompute SQL	Program arguments:			+ 2
🖉 Templates	Working directory:	C:\Users\livang\MaxCompute Studio		
	Environment variables:			
	Environment variables.			
	Use classpath of m <u>o</u> dule:	test_UDF		`
		Include dependencies with "Provided" scope	9	
	JRE:	Default (1.8 (3) - SDK of 'test_UDF' module)		\
	Shorten command <u>line</u> :	user-local default: none - java [options] classr	ame [args]	`
	<u>Enable capturing form</u>	snapshots		
	*MaxCompute project: loc	cal	v example_project	t ~ +
	*MaxCompute table: wc	_in1		
	*Table columns: co	<u>in</u>		ie:c1,e
	Download Record limit:	100		
	Before launch: Build, Acti	ivate tool window		
	+ - /			
	🔨 Build			
	🗌 Show this page 🗹 Act	ivate tool window		
			OK C	ancel Apply

- MaxCompute table: UDF运行时需要使用的MaxCompute表的名称。
- Table columns: UDF运行时需要使用的MaxCompute表的列信息。
- iii. 单击OK,运行结果如下图。

🗊 Project ▼ 😳 崇 🕸 🕅	C Lower.java ×
> 🖿 test	1 package wd_udf;
> 🖿 target	2 import com. aliyun. odps. udf. UDF;
warehouse	3 public class Lower extends UDF {
MyFirstModule.iml	4
m pom.xml	5 public String evaluate (String s) {
> scripts	6 if ("null".equals(s)) {
target	7 return null:
v warehouse	8
> example_project	<pre>9 return s. toLowerCase();</pre>
MySecondProject2	
✓ ■ tables	
> table_1	
✓ ■ upperabc	
≝ _schema_	
🚔 data	
> wc_in1	
MySecondProject.iml	Lower
Run: 🔥 Lower ×	
"D:\Program Files\Java\jdk1.8.0_181\bi	n/java.exe″
aliyun	
Process finished with exit code 0	

- 3. 注册MaxCompute UDF。
 - i. 在UDF Java文件上单击右键,选择**Deploy to server...**。

ii. 在Package a jar, submit resource and register function对话框,配置如下参数。

Package a jar, submit resource and register function	×
*MaxCompute project: c	• +
*Resource file: (
*Resource name:	
Resource comment:	
Extra resources:	
*Main class: AggrAvg	
*Function name:	
✓ Force update if already exists	
(?)	Cancel

- MaxCompute project: UDF所在的MaxCompute项目名称。由于UDF本身是在连接的MaxCompute项目下编写的,此处保持默认值即可。
- Resource file: UDF依赖的资源文件路径。此处保持默认值即可。
- Resource name: UDF依赖的资源。此处保持默认值即可。
- Function name: 注册的函数名称,即后续SQL中调用的UDF名称。例如Lower_test。
- iii. 单击OK,完成UDF注册。

4. 调用UDF。

在左侧导航栏单击**Project Explore**,在目标MaxCompute项目上单击右键,选择**Open in Console**并在Console区域输入调用UDF的SQL语句,按Enter键运行即可。



select Lower_test('ALIYUN');

返回结果如下。表明使用Intellij IDEA上开发的Java UDF函数Lower_test已经可用了。

_									
÷		+							
	_c0								
÷									
	aliyun								
÷									
1	records	(at most	10000	supported)	fetched	bу	instance	tunnel.	

3.4. 使用MaxCompute分析IP来源最佳实践

×

本文为您介绍如何使用MaxCompute分析IP来源,包括下载、上传IP地址库数据、编写UDF函数和编写SQL四个步骤。

前提条件

- 开通MaxCompute和DataWorks。
- 开通DataWorks。
- 在DataWorks上完成创建业务流程,本例使用DataWorks简单模式。详情请参见创建业务流程。

背景信息

淘宝IP地址库的查询接口为IP地址字串,使用示例如下。

{"code":0,"data":{"ip":"114.114.114","country":"^{China #},"area":"","region":"^{Jiangsu}","city":"^{Nanjing}" ","county":"XX","isp":"XX","country_id":"CN","area_id":"","region_id":"320000","city_id":"320100","county_id":"xx","isp_id":"xx"}}

在MaxCompute中禁止使用HTTP请求,因此目前可以通过如下三种方式实现在MaxCompute中查询IP:

• 用SQL将数据下载至本地,再发起HTTP请求查询。

⑦ 说明 效率低下,且淘宝IP库查询频率需要小于10 QPS,否则拒绝请求。

• 下载IP地址库至本地,再进行查询。

⑦ 说明 效率低下,且不利于数据仓库等产品分析使用。

• 将IP地址库定期维护上传至MaxCompute,进行连接查询。

⑦ 说明 比较高效,但是IP地址库需要自己定期维护。

下载IP地址库数据

- 1. 获取地址库数据。本文提供示例IP地址库数据UTF-8格式的不完整的地址库demo。
- 2. 下载示例地址库数据至本地,示例如下。

```
0,16777215,"0.0.0.0","0.255.255.255","","","","内网IP","内网IP","内网IP"
16777216,16777471,"1.0.0.0","1.0.0.255","澳大利亚","","","","",""
16777472,16778239,"1.0.1.0","1.0.3.255","中国","福建省","福州市","","电信"
```

示例数据说明如下:

- 数据格式为UTF-8。
- 前四个数据是IP地址的起始地址与结束地址。前两个是十进制整数形式,后两个是点分形式。IP地址段为整数形式,以便计算IP是否属于这个网段。

```
⑦ 说明 如果您需要使用自己的IP地址,请自行下载IP地址库
```

上传IP地址库数据

1. 在MaxCompute客户端执行如下语句, 创建表ipresource存放IP地址库数据。

```
DROP TABLE IF EXISTS ipresource ;
CREATE TABLE IF NOT EXISTS ipresource
(
    start_ip BIGINT
   ,end_ip BIGINT
   ,start_ip_arg string
   ,country STRING
   ,area STRING
   ,county STRING
   ,county STRING
   ,isp STRING
```

);

2. 执行如下Tunnel命令,上传本地示例IP地址库数据至表ipresource。

odps@ workshop_demo>tunnel upload D:/ipdata.txt.utf8 ipresource;

上述命令中, D:/ipdata.txt.utf8为IP地址库数据本地存放路径。更多命令说明请参见Tunnel命令。

您可以执行如下语句验证数据是否上传成功。

--查询表中数据条数。

select count(*) from ipresource;

3. 执行如下SQL语句,查看表ipresource前10条的样本数据。

select * from ipresource limit 10;

返回结果如下。

J	ob Queueing			
+	start_ip	end_ip	start_ip_arg end_ip_arg country area city county isp	
	3395369026	3395369026	~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~	
	3395369027	3395369028	″202.97.56.67″ ″202.97.56.68″ ″中国″ ″黑龙江省″ ″″ ″″ ″电信″	
I	3395369029	3395369029	″202.97.56.69″ ″202.97.56.69″ ″中国″ ″安徽省″ ″合肥市″ ″″ ″电信″	
I	3395369030	3395369030	″202.97.56.70″ ″202.97.56.70″ ″中国″ ″湖南省″ ″长沙市″ ″″ ″电信″	
I	3395369031	3395369033	″202.97.56.71″ ″202.97.56.73″ ″中国″ ″黑龙江省″ ″″ ″″ ″″ ″	
I	3395369034	3395369034	″202.97.56.74″ ″202.97.56.74″ ″中国″ ″湖南省″ ″长沙市″ ″″ ″电信″	
I	3395369035	3395369036	″202.97.56.75″ ″202.97.56.76″ ″中国″ ″黑龙江省″ ″″ ″″ ″″ ″	
	3395369037	3395369037	″202.97.56.77″ ″202.97.56.77″ ″中国″ ″江苏省″ ″南京市″ ″″ ″电信″	
	3395369038	3395369038	″202.97.56.78″ ″202.97.56.78″ ″中国″ ″湖南省″ ″长沙市″ ″″ ″电信″	
I	3395369039	3395369040	″202.97.56.79″ ″202.97.56.80″ ″中国″ ″黑龙江省″ ″″ ″″ ″″	

编写UDF函数

- 1. 进入**数据开发**页面。
 - i. 登录DataWorks控制台。
 - ii. 在左侧导航栏,单击**工作空间列表**。
 - iii. 单击相应工作空间后的数据开发。
- 2. 新建Python资源。
 - i. 右键单击业务流程,选择新建 > MaxCompute > 资源 > Python。
 - ii. 在新建资源对话框中,填写资源名称,并勾选上传为ODPS资源,单击新建。
 - iii. 在Python资源中输入如下代码后,单击回图标。

```
from odps.udf import annotate
@annotate("string->bigint")
class ipint(object):
    def evaluate(self, ip):
        try:
            return reduce(lambda x, y: (x << 8) + y, map(int, ip.split('.')))
        except:
            return 0
```

iv. 单击<mark>⊡</mark>图标。

- 3. 新建函数。
 - i. 右键单击已创建的业务流程,选择新建 > MaxCompute > 函数。
 - ii. 在新建函数对话框中, 输入函数名称, 单击新建。

⑦ 说明 如果绑定了多个MaxCompute引擎,则需要选择MaxCompute引擎实例。

iii. 在函数的编辑页面, 配置各项参数。

注册函数		
	函数类型:	其他函数
MaxC	ompute引擎实例:	A DESCRIPTION OF A DESC
	函数名:	
	责任人:	~ · · · · · · · · · · · · · · · · · · ·
	* 类名:	
	*资源列表:	
	描述:	
	命令格式:	
	参数说明:	
	返回值:	
	示例:	

参数	描述
函数类型	选择函数类型,包括数学运算函数、聚合函数、字符串处理函数、日期函数、窗口函数和其他函数。
MaxCompute引擎实例	默认不可以修改。
函数名	UDF函数名,即SQL中引用该函数所使用的名称。需要全局唯一,且注册函数后不支持修改。
责任人	默认显示。
	实现UDF的主类名,必填。
类名	⑦ 说明 当资源类型为Python时,类名格式为Python资源名称.类名(资源名称中的.py无需填写)。
	今教从六进27场,主任按照正式未轻于工作穴洞击司运物从次流,改造
资源列表	元至的又什石标,又行使糊些即直找本工作至时中已添加的资源,必填。 多个文件之间,使用英文逗号(,)分隔。
描述	针对当前UDF作用的简单描述。
命令格式	该UDF的具体使用方法示例,例如 test 。
参数说明	支持输入的参数类型以及返回参数类型的具体说明。
返回值	返回值,例如1,非必填顶。
示例	函数中的示例,非必填项。

4. 单击工具栏中的3图标。

5. 提交函数。

- i. 单击工具栏中的丽图标。
- ii. 在**提交新版本**对话框中,输入**备注**。
- ⅲ. 单击确认。

在SQL中使用UDF函数分析IP来源

- 1. 右键单击业务流程,选择**新建 > MaxCompute > ODP5 SQL**。
- 2. 在新建节点对话框中输入节点名称,并单击提交。
- 3. 在ODPS SQL节点编辑页面,输入如下语句。

select * from ipresource
WHERE ipint('1.2.24.2') >= start_ip
AND ipint('1.2.24.2') <= end_ip</pre>

- 4. 单击 🕟 图标运行代码。
- 5. 您可以在运行日志查看运行结果。

3.5. 解决DataWorks 10 MB文件限制问题最佳实践

本文为您介绍如何解决在DataWorks上执行MapReduce作业时,大于10 MB的JAR和资源文件不能上传至DataWorks的问题,方便您使用调度功能定期 执行MapReduce作业。

前提条件

请下载并安装MaxCompute客户端,详情请参见安装并配置MaxCompute客户端。

操作步骤

1. 在MaxCompute客户端上执行如下命令上传大于10 MB的资源。

```
--添加资源。
add jar C:\test_mr\test_mr.jar -f;
```

2. 通过MaxCompute客户端上传的资源,在DataWorks左侧资源列表中不显示。因此需要执行如下命令查看资源列表,确认上传是否成功。

--**查看资源。** list resources;

3. 减小JAR文件。DataWorks执行MapReduce作业的时候,需要在本地执行,所以保留Main函数即可。

```
jar
-resources test_mr.jar,test_ab.jar --资源在客户端注册后直接引用。
-classpath test_mr.jar --减小JAR文件策略:在gateway上提交包含Main函数的Mapper和Reducer,不需要提交额外的三方依赖,其他都可以放在resour
ces com.aliyun.odps.examples.mr.test_mr wc_in wc_out中。
```

3.6. 实现指定用户访问特定UDF最佳实践

本文为您介绍如何实现将资源(表、UDF等)设置为仅能被指定的用户访问。此方法涉及数据的加密解密算法,属于数据安全管控范畴。

前提条件

您需要提前安装MaxCompute客户端,以实现指定UDF被指定用户访问的操作。详情请参见安装并配置MaxCompute客户端。

背景信息

设置用户访问权限的常见方法有如下几种:

• Package方案,通过打包授权进行权限精细化管控。

Package用于解决跨项目空间的数据共享及资源授权问题。通过Package授予用户开发者角色后,用户拥有所有权限,风险不可控。详情请参见基于 Package跨项目访问资源。

○ 下图为DataWorks开发者角色的权限。



由上图可见,开发者角色对工作空间中的Package、Functions、Resources和Table默认有全部权限,不符合权限配置的要求。

○ 下图为通过DataWorks添加子账号并赋予开发者角色的权限。



由此可见,通过打包授权和DataWorks默认的角色都不能满足特定用户访问指定UDF的需求。例如,授予子账号 RAM\$xxxxx.pt@aliyun-test.com:ramtest 开发者角色,则默认该子账号拥有当前工作空间中全部对象的所有操作权限,详情请参见用户授权。

• 在DataWorks中新建角色进行权限管控。

在DataWorks工作配置页面的**MaxCompute高级配置**页面,可以对自定义用户角色进行权限管控。在该页面只能针对某个表或项目进行授权,不能 对资源和UDF进行授权。

⑦ 说明 更多有关DataWorks工作空间的MaxCompute属性介绍,请参见MaxCompute高级配置。

• Role Policy结合Project Policy实现指定用户访问指定UDF。

通过Policy可以精细化地管理具体用户对特定资源的具体权限粒度。

⑦ 说明 为了安全起见,建议初学者使用测试项目来验证Policy。

因此您可以通过Policy方案实现特定UDF被指定用户访问:

- 如果您不想让其他用户访问工作空间内具体的资源,在DataWorks中添加数据开发者权限后,再根据Role Policy的操作,在MaxCompute客户端将其 配置为拒绝访问权限。
- 如果您需要指定用户访问指定资源,在DataWorks中添加数据开发者权限后,再根据Project Policy的操作,在MaxCompute客户端将其配置为允许 访问权限。

操作步骤

- 1. 创建默认拒绝访问UDF的角色。
 - i. 在客户端输入如下命令创建角色denyudfrole。

create role denyudfrole;

ii. 创建Policy授权文件,如下所示。

```
{
    "Version": "1", "Statement"
    [{
    "Effect":"Deny",
    "Action":["odps:Read","odps:List"],
    "Resource":"acs:odps:*:projects/sz_mc/resources/getaddr.jar"
    },
    {
    "Effect":"Deny",
    "Action":["odps:Read","odps:List"],
    "Resource":"acs:odps:*:projects/sz_mc/registration/functions/getregion"
    }
  ] }
```

iii. 设置Role Policy。

在客户端执行如下命令,设置Role Policy文件的存放路径。

put policy /Users/yangyi/Desktop/role_policy.json on role denyudfrole;

iv. 在客户端执行如下命令查看Role Policy。

get policy on role denyudfrole;

返回结果如下。
odps <mark>0 get policy on role denyudtrole;</mark> {
"Statement": [{
"Action": ["odps:Read",
"odps:List"],
"Effect": "Deny",
"Resource": ["acs:odps:*:projects/sz_mc/resources/getaddr.jar"]},
ζ.
"Action": ["odps:Read",
"odps:List"],
"Effect": "Deny",
"Resource": ["acs:odps:*:projects/sz_mc/registration/functions/getre
gion"]}],
"Version": "1"}

v. 在客户端执行如下命令添加子账号至role denyudfrole。

grant denyudfrole to RAM\$xxxx.pt@aliyun-test.com:ramtest;

- 2. 验证拒绝访问UDF的角色是否创建成功。
 - i. 登录客户端输入 whoami; 确认角色。



ii. 通过 show grants; 查看当前登录用户权限。



通过查询发现该RAM子账号有两个角色,一个是role_project_dev(即DataWorks默认的开发者角色),另一个是刚自定义创建的 denyudfrole。

iii. 验证自建UDF以及依赖的包的权限。



通过上述验证发现,该子账号在拥有DataWorks开发者角色的前提下,没有自建UDF(get region)的读权限。您还需要结合Project Policy来 实现该UDF只能被指定的用户访问。

3. 配置Project Policy。

i. 编写Policy。

```
{
    "Version": "1", "Statement":
    [{
    "Effect":"Allow",
    "Principal":"RAM$yangyi.pt@aliyun-test.com:yangyitest",
    "Action":["odps:Read","odps:List","odps:Select"],
    "Resource":"acs:odps:*:projects/sz_mc/resources/getaddr.jar"
    },
    {
    "Effect":"Allow",
    "Principal":"RAM$xxxx.pt@aliyun-test.com:yangyitest",
    "Action":["odps:Read","odps:List","odps:Select"],
    "Resource":"acs:odps:*:projects/sz_mc/registration/functions/getregion"
    })
}
```

ii. 设置Role Policy。

在客户端执行如下命令,设置Role Policy文件的存放路径。

put policy /Users/yangyi/Desktop/project_policy.json;

iii. 在客户端执行如下命令查看Role Policy。

get policy;

返回结果如下。



iv. 通过 whoami; 和 show grants; 进行验证。



- v. 运行SQL任务,查看是否仅指定的RAM子账号能够查看指定的UDF和依赖的包。
 - 指定的RAM子账号查看指定的UDF。

odps@ <mark>sele</mark> ID = 2019011409 Log view: http://logview.c J00DA2MTU2Myx7I]	ect getregion odps.aliyun.co	('172. 2 om/logview/?h	 =http://	service.od	lps.aliyur CJdLCJFZr	1.com/api8 nZlY30i0i	kp=sz_n JBbGxvc	
biI6IjEifQ== Job Queueing.								
 41_job_0	STAGES	STATUS TERMINATED	 TOTAL 1	COMPLETED	RUNNING Ø	PENDING Ø	ΒΑϹΚΙ	
				00% ELAPS				
Inputs: Jubputs: Job run time: 18 Job run mode: fu Job run engine: 41: instance input re output re	2.000 ixi job execution en : count: 1 :: 18.000 : time: min: 16.000, :coords: coords: AdhocSink1: :							
▶+ [美国,美国,, ++								
查看依赖包。								
adps@desc reso Name Owner Type Comment CreatedTime LastModifiedTime LastModifiedTime LastUpdator Size	urce getaddr.jar;	getaddr.jar ALTYUN\$ JAR IDE RESOURCE UP 2018-05-24 19:5 2018-05-24 19:5 1353716 2320103-0552	tëaliyun-te DATE TO ODP 1:16 1:16	st.com 5 /home/admin. -76-00	/oxs-base-bi:	z-phoenix/te	mp/4d3efccz0sl9eiiri	n53o5n4/ç

3.7. PyODPS节点实现结巴中文分词

本文为您介绍如何使用DataWorks的PyODPS类型节点,借助开源结巴中文分词包实现对中文字段的分词并写入新的表,以及如何通过闭包函数使用自 定义词典进行分词。

前提条件

- 请首先确保您已经完成DataWorks工作空间的创建,本示例使用绑定多个MaxCompute计算引擎的**简单模式**工作空间,详情请参见<mark>创建工作空</mark>间。
- 请在Git Hub下载开源结巴分词中文包。



背景信息

PyODPS集成了MaxCompute的Python SDK。您可以在DataWorks的PyODPS节点上直接编辑Python代码,并使用MaxCompute的Python SDK。关于 PyODPS节点的详情,请参见PyODPS节点。

本文主要为您介绍如何使用PyODPS节点实现结巴中文分词。

- 借助开源包实现结巴中文分词
- 自定义的词典实现结巴分词

↓ 注意 本文的操作仅作为代码示例,不建议用于实际的生产环境。

借助开源包实现结巴中文分词

- 1. 创建业务流程。
 - i. 登录DataWorks控制台。
 - ii. 在左侧导航栏,单击**工作空间列表**。
 - iii. 选择工作空间所在地域后,单击相应工作空间后的进入数据开发。
 - iv. 鼠标悬停至**■**图标,单击**业务流程**。
 - v. 在新建业务流程对话框中, 输入业务名称和描述, 单击新建。

□ 注意 业务名称必须是大小写字母、中文、数字、下划线(_)以及小数点(.),且不能超过128个字符。

- 2. 上传jieba-master.zip包。
 - i. 展开新建的业务流程下的MaxCompute,右键单击资源,选择新建 > Archive。
 - ii. 在新建资源对话框中,配置各项参数后,单击新建。

新建资源		×
* 引擎类型:		
*引擎实例:	IIIFirst 华东1 (杭州)	
* 目标文件夹:	业务流程 axCompute	
* 文件类型:	Archive	
	✓ 上传为ODPS资源本次上传,资源会同步上传至ODPS中	
* 上传文件 :	jieba-master.zip (23.31M)	
*资源名称:	jieba-master.zip	
		新建取消

参数	描述						
	从下拉列表中选择该资源所在的计算引擎。						
引擎类型	⑦ 说明 如果您所在的工作空间仅绑定一个实例,则不会显示该参数。						
引擎实例	任务所绑定的MaxCompute引擎名称。						
目标文件夹	默认当前所在文件夹的路径,您可以进行修改。						
	此处选择Archive类型。						
文件类型	⑦ 说明 如果该资源包已经在MaxCompute(ODPS)客户端上传过,请取消勾选上传为ODPS资源,否则 上传会报错。						
上传文件	单击 点击上传 ,在本地选择已下载的文件jieba-master.zip后,单击 打开 。						
资源名称	资源的名称,无需和上传的文件名保持一致,但需要符合以下规范: ■ 资源名称仅包含中文、字母、数字、英文句号(.)、下划线(_)和短划线(-)。 ■ 资源类型为Archive时,资源名称和文件名的后缀必须一致,且后缀名包含.zip、.tgz、.tar.gz或.tar。						

- iii. 在工具栏中, 单击回图标。
- iv. 在提交新版本对话框中, 输入变更描述, 单击确定。
- 3. 创建测试数据表。
 - i. 展开新建的业务流程下的MaxCompute,右键选择表 > 新建表。
 - ii. 在新建表对话框中, 输入表名, 单击新建。

⑦ 说明 本示例表名示例为jieba_test。

iii. 单击**DDL模式**,输入以下建表DDL语句。

⑦ 说明 本教程准备了两列测试数据,您在后续开发过程中可以选择一列进行分词。

- ₩. 在弹出的对话框中,单击确定。
- v. 在基本属性区域, 输入表的中文名, 单击提交到生产环境。
- vi. 在提交生产确认对话框中,选中我已悉知风险,确认提交,单击确认。
- 4. 以同样的方式创建存放测试结果的数据表jieba_result, DDL语句如下所示。

```
CREATE TABLE jieba_result (
    `chinese` string
);
```

⑦ 说明 本例仅对测试数据的chinese列进行分词处理,因此结果表仅有一列。

- 5. 单击测试数据下载分词测试数据。
- 6. 上传测试数据:
 - i. 在数据开发页面,单击 🗗 图标。
 - ii. 在数据导入向导对话框中,输入需要导入数据的测试表jieba_test并选中,单击下一步。
 - iii. 单击**浏览**,上传您下载至本地的*jieba_test.csv*文件,单击下一步。
 - iv.选中按名称匹配,单击导入数据。
- 7. 创建PyODPS 2节点。

i. 展开业务流程中的MaxCompute,右键单击数据开发,选择新建 > PyODPS 2。

- ii. 在新建节点对话框中, 输入节点名称, 并选择目标文件夹, 单击提交。
 - ? 说明
 - 节点名称必须是大小写字母、中文、数字、下划线(_)和小数点(.),且不能超过128个字符。
 - 本示例的节点名称为word_split。
- iii. 在word_split节点, 输入下述PyODPS代码。

```
def test(input_var):
   import jieba
   import sys
  reload(sys)
  sys.setdefaultencoding('utf-8')
   result=jieba.cut(input_var, cut_all=False)
   return "/ ".join(result)
hints = \{
   'odps.isolation.session.enable': True
1
libraries =['jieba-master.zip'] #引用您的jieba-master.zip压缩包。
iris = o.get table('jieba test').to df() #引用您的jieba test表中的数据。
example = iris.chinese.map(test).execute(hints=hints, libraries=libraries)
print(example) #查看分词结果,分词结构为MAP类型数据。
abci=list(example) #将分词结果转为list类型的数据。
i = 0
for i in range(i,len(abci)):
  pq=str(abci[i])
   o.write_table('jieba_result',[pq]) #通过循环,逐条写入数据至结果表jieba_result中。
   i+=1
else:
   print("done")
```

ⅳ. 单击工具栏中的凹图标。

```
v. 单击工具栏中的 图标, 在参数对话框中, 从调度资源组下拉列表选择需要使用的资源组, 单击确定。
```

⑦ 说明 调度资源组详细说明,请参见DataWorks资源组概述。

- vi. 在页面下方的运行日志区域,查看结巴分词程序的运行结果。
- 8. 创建和运行ODPS SQL节点。
 - i. 展开业务流程中的MaxCompute,右键单击数据开发,选择新建 > ODPS SQL。
 - ii. 在新建节点对话框中,输入节点名称,并选择目标文件夹,单击提交。

⑦ 说明 节点名称必须是大小写字母、中文、数字、下划线(_)和小数点(.),且不能超过128个字符。

iii. 在节点的编辑页面, 输入以下SQL语句。

select * from jieba_result;

- ⅳ. 单击工具栏中的 28标。
- v. 单击工具栏中的 图标, 在参数对话框中, 从调度资源组下拉列表选择需要使用的资源组, 单击确定。

⑦ 说明 调度资源组详细说明,请参见DataWorks资源组概述。

- vi. 在MaxCompute计算成本估计对话框中,确认预估费用后,单击运行。
- vii. 在页面下方的运行日志区域,查看运行结果。

自定义的词典实现结巴分词

如果开源结巴分词的词库无法满足您的需求,您可以选择使用自定义的词典。

PyODPS自定义函数可以读取上传至MaxCompute的资源(表资源或文件资源)。此时,自定义函数需要写为闭包函数或Callable类。如果您需要引用 复杂的自定义函数,则可以使用DataWorks的注册MaxCompute函数功能,详情请参见<mark>注册MaxCompute函数</mark>。

本文以使用闭包函数的方式,引用上传至MaxCompute的资源文件(即自定义词典)key_words.txt。

- ⑦ 说明 本示例的资源文件名为key_words.txt。
- 1. 展开业务流程中的MaxCompute,右键单击资源,选择新建 > File。
- 2. 在新建资源对话框中,配置各项参数,单击确定。

新建资源		×
* 引擎类型:		
* 引擎实例:	■_First 华东1 (杭州)	
*目标文件夹:	业务流 axCompute	
* 文件类型:	File	
	大文件 (内容超过500KB)	
	✓ 上传为ODPS资源本次上传,资源会同步上传至ODPS中	
*资源名称:		
	新建	取消

参数	描述							
	从下拉列表中选择该资源所在的计算引擎。							
引擎类型	⑦ 说明 如果您所在的工作空间仅绑定一个实例,则不会显示该参数。							
引擎 买例	任务所绑定的MaxCompute引擎名称。							
目标文件夹	默认当前所在文件夹的路径,您可以进行修改。							
	此处选择File类型。							
文件类型	⑦ 说明 如果您从本地上传词典文件至DataWorks,则文件必须为UTF-8编码格式。							
上传文件	单击 点击上传 ,在本地选择文件key_words.txt后,单击 打开 。							
资源名称	资源名称仅包含中文、字母、数字、英文句号(.)、下划线(_)和短划线(-)。							

3. 在key_words.txt资源编辑页面,输入自定义词典的内容。格式如下。

```
○ 一个词占一行。
  。每一行包括词、词频(可省略)和词性(可省略),使用空格分隔,且不可以颠倒顺序。
4. 单击工具栏中的 图标提交资源。
5. 创建一个PyODPS 2节点,并输入如下代码。
   def test(resources):
      import jieba
      import sys
      reload(sys)
      sys.setdefaultencoding('utf-8')
       fileobj = resources[0]
      def h(input_var):#在嵌套函数h()中,执行词典加载和分词。
          import jieba
          jieba.load_userdict(fileobj)
         result=jieba.cut(input_var, cut_all=False)
          return "/ ".join(result)
      return h
   hints = {
       'odps.isolation.session.enable': True
   libraries =['jieba-master.zip'] #引用您的jieba-master.zip压缩包。
   iris = o.get_table('jieba_test').to_df() #引用您的jieba_test表中的数据。
   file_object = o.get_resource('key_words.txt') #get_resource()引用odps资源。
   example = iris.chinese.map(test, resources=[file_object]).execute(hints=hints, libraries=libraries) #map调用函数, 并传递resour
   ces参数。
   print(example) #查看分词结果,分词结构为MAP类型数据。
   abci=list(example) #将分词结果转为list类型数据。
   for i in range(i,len(abci)):
      pg=str(abci[i])
      o.write_table('jieba_result',[pq]) #通过循环,逐条写入数据至结果表jieba_result中。
      i+=1
   else:
```

print("done")

6. 运行代码,对比引用自定义词典前后的结果。

3.8. PyODPS节点实现避免将数据下载到本地

本文为您介绍PyODPS如何避免将数据下载到本地。

背景信息

PyODPS提供了多种方便下载数据到本地的方法。因此,在设备允许的情况下,可以把数据下载到本地处理,然后再上传至MaxCompute。但是这种操作非常低效,数据下载到本地进行处理,无法使用MaxCompute的大规模并行能力。当数据量大于10 MB时,不建议进行本地数据处理。常见的将数据 下载到本地的操作如下:

- Head、Tail和To_pandas方法的调用。通常,可以调用 head、tail 方法返回少量数据进行数据探查,当数据量较大时,建议调用Persisit方法,将 数据直接保存在MaxCompute表中。详情请参见执行。
- 在表或SQL实例上直接执行Open_reader方法获取表数据。当数据量大时,建议使用PyODPS DataFrame(从MaxCompute表创建)和MaxCompute SQL来处理数据,以替代本地数据处理这种比较低效的方式。

示例代码

- 将一份JSON串数据按Key-Value对展开成一行,示例代码如下。
- 本地测试,通过 head() 方法返回少量数据进行测试。

最佳实践·数据开发

In [12]: df.head(2) json 0 {"a": 1, "b": 2} 1 {"c": 4, "b": 3} In [14]: from odps.df import output In [16]: @output(['k', 'v'], ['string', 'int']) ...: def h(row): ...: import json ...: for k, v in json.loads(row.json).items(): ...: yield k, v ...: In [21]: df.apply(h, axis=1).head(4) k v 0 a 1 1 b 2 2 c 4 3 b 3 ● 线上生产,通过 persist() 方法将结果存回MaxCompute表。 In [14]: from odps.df import output In [16]: @output(['k', 'v'], ['string', 'int']) ...: def h(row):

- ...: import json ...: for k, v in json.loads(row.json).items(): ...: yield k, v
- ...:
- In [21]: df.apply(h, axis=1).persist('my_table')

4.计算优化 4.1. SQL调优

本文为您介绍常见的SQL问题以及优化示例。

数据倾斜优化

数据倾斜产生的根本原因是少数Worker处理的数据量远远超过其他Worker处理的数据量,因此少数Worker的运行时长远远超过其他Worker的平均运行 时长,导致整个任务运行时间超长,造成任务延迟。

● Join操作导致的数据倾斜

Join操作导致数据倾斜的原因是Join on的Key分布不均匀。假设大表A和小表B执行Join操作,运行如下语句。

SELECT * FROM A JOIN B ON A.value = B.value;

复制该语句的Logview链接并打开,双击执行Join操作的作业,可以看到此时在[Long-tails]区域存在长尾现象,表示数据已经倾斜了。



您可通过如下方法进行优化:

○ 由于表B是个小表并且没有超过512 MB,您可将上述语句优化为MapJoin语句后执行,语句如下。

SELECT /* + MAPJOIN(B) */ * FROM A JOIN B ON A.value = B.value;

○ 将倾斜的Key用单独的逻辑来处理。假设两边的Key中有大量NULL数据导致了倾斜,则需要在Join前先过滤掉NULL数据或者补上随机数,然后再进行Join,示例如下。

SELECT * FROM A JOIN B ON CASE WHEN A.value IS NULL THEN CONCAT('value', RAND()) ELSE A.value END = B.value;

在实际场景中,如果您发现已经数据倾斜,但无法获取导致数据倾斜的Key信息,可以使用如下方法查看数据倾斜。

--执行如下语句产生数据倾斜。

SELECT * FROM a JOIN b ON a.key=b.key; --您可以执行如下SQL, 查看Key的分布,判断执行Join操作时是否会有数据倾斜。 SELECT left.key, left.cnt * right.cnt FROM (select key, count(*) AS cnt FROM a GROUP BY key) LEFT JOIN (SELECT key, COUNT(*) AS cnt FROM b GROUP BY key) RIGHT ON left.key=right.key;

• Group By倾斜

造成Group By倾斜的原因是Group By的Key分布不均匀。

假设表A内有两个字段(Key, Value),表内的数据量较大且Key值分布不均匀,运行语句如下所示。

SELECT key, COUNT(value) FROM A GROUP BY key;

当表中的数据足够大时,您会在Logview中发现长尾现象。解决此问题,您需要在执行SQL前设置防倾斜的参数,设置语句为 set odps.sql.groupb y.skewindata=true 。

● 错误使用动态分区造成的数据倾斜

动态分区SQL时,在MaxCompute中会默认增加一个Reduce,用来将相同分区的数据合并在一起。此操作可以:

○ 减少MaxCompute系统产生的小文件,使后续处理更快速。

○ 避免一个Worker输出文件很多时占用内存过大。

如果引入的Reduce导致分区数据倾斜,则会发生长尾。因为相同的数据最多只会有10个Worker处理,所以数据量大,则会发生长尾,示例如下。

INSERT OVERWRITE TABLE A2 PARTITION(dt) SELECT SPLIT_PART(value,'\t',1) AS field1, SPLIT_PART(value,'\t',2) AS field2, dt FROM A WHERE dt='20151010';

这种情况下,不建议使用动态分区,优化语句如下。

INSERT OVERWRITE TABLE A2 PARTITION (dt='20151010')
SELECT
SPLIT_PART(value,'\t',1) as field1,
SPLIT_PART(value,'\t',2) as field2
FROM A
WHERE dt='20151010';

数据倾斜优化的详情请参见其它计算长尾调优。

窗口函数优化

如果SQL语句中使用了窗口函数,通常每个窗口函数会形成一个Reduce作业。如果窗口函数较多,会消耗过多的资源。您可以对符合下述条件的窗口 函数进行优化:

- 窗口函数在OVER关键字后面要完全相同,要有相同的分组和排序条件。
- 多个窗口函数在同一层SOL中执行。

符合上述2个条件的窗口函数会合并为一个Reduce执行。SQL示例如下所示。

```
SELECT
RANK()OVER(PARTITION BY A ORDER BY B desc) AS RANK,
ROW_NUMBER()OVER(PARTITION BY A ORDER BY B desc) AS row_num
FROM MyTable;
```

子查询优化

子查询如下所示。

```
SELECT * FROM table_a a WHERE a.coll IN (SELECT coll FROM table_b b WHERE xxx);
```

当此语句中的table_b子查询返回的col1的个数超过1000个时,系统会报错为 records returned from subquery exceeded limit of 1000 。此时您 可以使用Join语句来代替,如下所示。

SELECT a.* FROM table_a a JOIN (SELECT DISTINCT coll FROM table_b b WHERE xxx) c ON (a.coll = c.coll)

? 说明

- 如果没有使用DISTINCT关键字,而子查询表c返回的结果中有相同的col1的值,可能会导致a表的结果数变多。
- DISTINCT关键字会导致查询在同一个Worker中执行。如果子查询数据量较大,会导致查询比较慢。
- 如果业务上已经确保子查询中col1列值无重复,您可以删除DIST INCT关键字,以提高性能。

Join语句优化

当两个表进行Join操作时,建议在如下位置使用WHERE子句:

- 主表的分区限制条件可以写在WHERE子句中(最好先用子查询过滤)。
- 主表的WHERE子句建议写在SQL语句最后。
- 从表分区限制条件不要写在WHERE子句中,建议写在ON条件或者子查询中。

示例如下。

```
SELECT * FROM A JOIN (SELECT * FROM B WHERE dt=20150301) B ON B.id=A.id WHERE A.dt=20150301;
```

SELECT * FROM A JOIN B ON B.id=A.id WHERE B.dt=20150301; --不建议使用。此语句会先执行Join操作后进行分区裁剪,导致数据量变大,性能下降。

SELECT * FROM (SELECT * FROM A WHERE dt=20150301)A JOIN (SELECT * FROM B WHERE dt=20150301)B ON B.id=A.id;

4.2. JOIN长尾优化

本文为您介绍执行SQL时JOIN阶段常见的数据倾斜场景以及对应的解决办法。

背景信息

MaxCompute SQL在JOIN阶段会将JOIN Key相同的数据分发到同一个Instance上进行处理。如果某个Key上的数据量比较多,会导致该Instance执行时间 比其他Instance执行时间长。执行日志中该JOIN Task的大部分Instance都已执行完成,但少数几个Instance一直处于执行中,这种现象称之为长尾。 数据量倾斜导致长尾的现象比较普遍,严重影响任务的执行时间,尤其是在双十一等大型活动期间,长尾程度比平时更为严重。例如,某些大型店铺 的浏览量远远超过一般店铺的浏览量,当用浏览日志数据和卖家维表关联时,会按照卖家ID进行分发,导致某个Inst ance处理的数据量远远超过其他 Inst ance, 而整个任务会因为这个长尾的Inst ance无法结束。

您可以从以下四个方面进行长尾处理考虑:

- 如果两张表里有一张大表和一张小表,可以考虑使用MAP JOIN,对小表进行缓存,具体的语法和说明请参见SELECT语法。
- 如果两张表都比较大,就需要先尽量去重。
- 从业务上考虑,寻找两个大数据量的Key执行笛卡尔积的原因,从业务上进行优化。
- 小表LEFT JOIN大表,直接LEFT JOIN较慢。先将小表和大表进行MAP JOIN,得到小表和大表的交集中间表,且这个中间表一定是不大于大表的(Key倾斜程度与表的膨胀大小成正比)。然后小表再和这个中间表进行LEFT JOIN,这样操作的效果等于小表LEFT JOIN大表。

查看数据倾斜

执行如下步骤查看JOIN是否发生数据倾斜:

1. 打开SQL执行时产生的Logview日志,查看每个FuxiTask的详细执行信息。Long-Tails(115)表示有115个长尾。

	TaskName	Fatal/Finished/TotalInstCo	I/O Records	FinishedPercentage	Status	StartTime	EndTime	Latency(s)	TimeLine		查看
1	M39_Stg1	0/1674/1674	4030702	100%	Terminate	2016-04-21 03:2	2016-04-21 03:5	26:32			
2	M2_Stg3	<mark>0/3/</mark> 3	4952006	100%	Terminate	2016-04-21 03:2	2016-04-21 03:2	24			
3	M1_Stg3	0/42/42	7957958	100%	Terminate	2016-04-21 03:2	2016-04-21 03:3	2:9			
4	M26_Stg11	0/9/9	5500630	100%	Terminate	2016-04-21 03:2	2016-04-21 03:3	1:23			
5	M30_Stg12	0/96/96	7328747	100%	Terminate	2016-04-21 03:2	2016-04-21 03:3	1:56			
e	M7_Stg5	0/9/9	6870146	100%	Terminate	2016-04-21 03:2	2016-04-21 03:3	57			
7	J3_1_2_Stg3	0/1999/1999	7958937	100%	Terminate	2016-04-21 03:3	2016-04-21 03:3	33			
8	J10_3_7_St	0/1999/1999	8645334	100%	Terminate	2016-04-21 03:3	2016-04-21 03:4	9:7			
9	J28_10_26	0/1999/1999	8508021	100%	Terminate	2016-04-21 03:4	2016-04-21 03:4	47			
J10	3 7 Sta5 🕷								_		
Smar	tFilter Failed(0)) Terminated(1999) All	1999) Long	g-Tails(115)	ncy chart					Latency: {"min":"5","avg":"16",	"max":"8:5
	uxiInstance	IP & Path StdOut Std	Err Débug	Status Finished	Percentage	Start Time 🔶	EndTime	Latency(s)	TimeLine		1001
0	110_3_7_St	10.182.82.1 🧊 🎵	*	Terminate	00%	2016-04-21 03:3	2016-04-21 03:3	1:16			扁鹊
1	010_3_7_St	10.182.91.9 🧊 🛛	*	Terminate 1	00%	2016-04-21 03:3	2016-04-21 03:3	1:32			扁鹊
2	010_3_7_St	10.182.82.9 📋 📑	*	Terminate 1	00%	2016-04-21 03:3	2016-04-21 03:3	34			扁鹊

2. 单击Fuxilnstance后的 圆图标,查看StdOut 中Instance读入的数据量。

例如, Read from 0 num:52743413 size:1389941257 表示JOIN输入读取的数据量是1389941257行。如果Long-Tails中Instance读取的数据量远 超过其它Instance读取的数据量,则表示是因为数据量导致长尾。

常见场景及解决方案

• MAP JOIN方案: JOIN倾斜时,如果某路输入比较小,可以采用MAP JOIN避免分发引起的长尾。

MAP JOIN的原理是将JOIN操作提前到Map端执行,这样可以避免因为分发Key不均匀导致数据倾斜。MAP JOIN使用限制如下:

◎ MAP JOIN使用时,JOIN中的从表比较小才可用。所谓从表,即LEFT OUT ER JOIN中的右表,或者RIGHT OUT ER JOIN中的左表。

○ MAP JOIN使用时,对小表的大小有限制,默认小表读入内存后的大小不能超过512 MB。用户可以通过如下语句加大内存,最大为8192 MB。

set odps.sql.mapjoin.memory.max=8192

⑦ 说明 odps.sql.mapjoin.memory.max设置过大可能导致OOM (Out Of Memory)内存溢出,请您谨慎操作。

MAP JOIN的使用方法非常简单,在SQL语句中 SELECT 后加上 /*+ mapjoin(b) */即可,其中b代表小表(或者是子查询)的别名。举例如下。

select	/*+ mapj	oin(b)	*/	
	a.c2			
	,b.c3			
from				
	(select	c1		
		,c2		
	from	t1) a
left out	er join			
	(select	c1		
		,c3		
	from	t2) b
on	a.c1 = b	.c1;		

• JOIN因为热点值导致长尾

如果是因为热点值导致长尾,并且JOIN的输入比较大无法用MAP JOIN,可以先将热点Key取出,对于主表数据用热点Key切分成热点数据和非热点数 据两部分分别处理,最后合并。以淘宝的浏览量日志表关联商品维表取商品属性为例:

i. 取出热点Key

将浏览量大于50000的商品ID取出到临时表。

ii. 取出非热点数据。

将主表 (sdwd_tb_log_pv_di) 和热点key表 (topk_item) 外关联后通过条件 bl.item_id is null , 取出关联不到的数据即非热点商品的日 志数据,此时需要用MAP JOIN。再用非热点数据关联商品维表,因为已经排除了热点数据,不会存在长尾。

```
select ...
from
       (select *
       from dim_tb_itm
where ds = '${bizdate}'
        ) a
right outer join
       (select /*+ mapjoin(b1) */
                b2.*
        from
                (select item_id
                from topk_item
where ds = '${bizdate}'
                ) b1
        right outer join
               (select *
from dwd_tb_log_pv_di
                 where ds = '${bizdate}'
                         url_type = 'ipv'
                 and
                 ) b2
                 b1.item_id = coalesce(b2.item_id,concat("tbcdm",rand())
        on
        where b1.item_id is null
        ) 1
        a.item_id = coalesce(l.item_id,concat("tbcdm",rand());
on
```

iii. 取出热点数据。

将主表(sdwd_tb_log_pv_di)和热点Key表(topk_item)内关联,此时需要用MAP JOIN,取到热点商品的日志数据。同时,需要将商品维表 (dim_tb_itm)和热点Key表(topk_item)内关联,取到热点商品的维表数据,然后将第一部分数据外关联第二部分数据,因为第二部分只有 热点商品的维表,数据量比较小,可以用MAP JOIN避免长尾。

```
select /*+ mapjoin(a) */
from
       (select /*+ mapjoin(b1) */
                b2.*
       from
                (select item_id
                from
                         topk_item
                where ds = '${bizdate}'
                )b1
        ioin
                (select *
                        dwd_tb_log_pv_di
                from
                where ds = '${bizdate}'
                and url_type = 'ipv'
and item_id is not null
                ) b2
                (b1.item_id = b2.item_id)
        on
       ) ]
left outer join
       (select /*+ mapjoin(al) */
                a2.*
        from
                (select item_id
                from topk_item
                where ds = '${bizdate}'
                ) al
        join
               (select *
               from dim_tb_itm
                where ds = '${bizdate}'
                ) a2
                (al.item_id = a2.item_id)
        on
        ) a
        a.item id = 1.item id;
on
```

iv. 将步骤2和步骤3的数据通过 union all 合并后即得到完整的日志数据,并且关联了商品的信息。

• 通过设置odps.sql.skewjoin参数解决长尾问题。

此方法简单方便,但是如果倾斜的值发生变化需要修改代码重新执行命令,且变化无法提前预知。另外,如果倾斜值较多也不方便在参数中设置, 需要根据实际情况选择拆分代码或者参数设置。参数设置的操作步骤如下:

i. 开启功能。

set odps.sql.skewjoin=true

ii. 设置倾斜的Key及对应的值。

set odps.sql.skewinfo=skewed_src:(skewed_key) [("skewed_value")]

其中: skewed_key代表倾斜的列, skewed_value代表倾斜列上的倾斜值。

• 通过SkewJoin Hint避免热值倾斜。SkewJoin Hint详情请参见SKEWJOIN HINT。

○ 使用方法

--方法1: Hint表名(注意Hint的是表的alias) 。
select /*+ skewjoin(a) */ * from T0 a join T1 b on a.c0 = b.c0 and a.c1 = b.c1;
--方法2: Hint表名和认为可能产生倾斜的列,例如表a的c0和c1列存在数据倾斜。
select /*+ skewjoin(a(c0, c1)) */ * from T0 a join T1 b on a.c0 = b.c0 and a.c1 = b.c1 and a.c2 = b.c2;
--方法3: Hint表名和列,并提供发生倾斜的key值。如果是STRING类型,需要加上引号。例如(a.c0=1 and a.c1="2")和(a.c0=3 and a.c1="4")的值都存
在数据倾斜。
select /*+ skewjoin(a(c0, c1)((1, "2"), (3, "4"))) */ * from T0 a join T1 b on a.c0 = b.c0 and a.c1 = b.c1 and a.c2 = b.c2;

⑦ 说明 方法3直接指定值的处理效率比方法1和方法2(不指定值)高。

○ 定位倾斜Join

以如下Logview为例,耗时最长的FuxiTask为J5_3_4,定位倾斜Join的步骤如下。



a. 单击J5_3_4,在Fuxi Instance页面观察J5的Instances,可以看到J5_3_4#215_0的耗时最长,且I/O record和I/O Bytes远远大于其它Instance。

Fusi In	stance of Fuxi Task:	J5_3_4	 La	tency: (min.00-00-01, a	vg:00:00:06, max:00:48	:05}		Sma	rtFilter A	VI (1111) Failed	(0) Terminated (1111) Long-Tails (8) Da	ta-Skews (10)	Latency Chart C
	J5_3_4 ×													
			Sensor	I/O Record	1/O Bytes		StdOut	StdErr	Debug	Progress 💲	Start Time 💲		Latency 💲	
240	J5_3_4#215_0	11.159.130.168 🖵		304.4 M/304.4 M	16.91 GB/17.56 GB			R		100%	2020/11/05 02:14:03	2020/11/05 03:02:08	00:48:05	
	J5_3_4¥352_0	11.169.130.34 🖵		69.3 M/69.3 M	3.7 GB/3.39 GB					100%	2020/11/05 02:14:03	2020/11/05 02:18:36	00:05:33	
	J5_3_4#1053_0	11.169.128.163 🗢		3.5 M/3.5 M	189.12 MB/177.91 MB					100%	2020/11/05 02:14:03	2020/11/05 02:14:25	00:00:22	
	J5_3_4#295_0			3.5 M/3.4 M	173.95 MB/164.04					100%	2020/11/05 02:14:04	2020/11/05 02:14:25	00.00.21	
	J5_3_4¥454_0	11.169.140.143 🖵			6.35 MB/2.74 MB					100%	2020/11/05 02:14:06	2020/11/05 02:14:23	00:00:17	
	J5_3_4¥238_0	11.0.80.43 🖵			88.44 MB/92.05 MB					100%	2020/11/05 02:14:04	2020/11/05 02:14:20	00:00:16	
	J5_3_4¥442_0	11.168.93.219 🗢		87.4 K/85.3 K	7.1 MB/3.39 MB					100%	2020/11/05 02:14:06	2020/11/06 02:14:22	00:00:16	
502	IE 3 44534 0	11 100 100 14		2014/2014	127 20 MD/117 01 MD					1000	2020/11/05 02:14:02	2020/11/05 0244-19	00.0045	

b. 此时,可以认为J5节点发生了数据倾斜,继续需要确定倾斜的是哪个Join。单击倾斜Instance的StdOut和任一个不倾斜节点的StdOut,查看 内容。通常,网页无法显示所有StdOut的内容,您可以单击Download,查看全部内容。

Stdout		×
Auto	refresh	
	neWrite& writes records: 208304624	
	2020-11-05 03:01:16] Brader StreamlineBead7 reads records: 298497024	
	[2020-11-05 03:01:16] Writer StreamLineWrites writes records: 298497024	
	[2020-11-05 03:01:17] Reader StreamLineRead7 reads records: 298599424	
	[2020-11-05 03:01:17] Writer StreamLineWrite8 writes records: 298599424	
	[2020-11-05 03:01:17] Reader StreamLineRead7 reads records: 298701824	
	[2020-11-05 03:01:17] Writer StreamLineWrite8 writes records: 298701824	
	[2020-11-05 03:01:18] Reader StreamLineRead7 reads records: 298804224	
	[2020-11-05 03:01:18] Writer StreamLineWrite8 writes records: 298804224	
	[2020-11-05 03:01:19] Reader StreamLineRead7 reads records: 298906624	
	[2020-11-05 03:01:19] Writer StreamLineWrite8 writes records: 298906624	
	[2020-11-05 03:01:20] Reader StreamLineRead7 reads records: 299009024	
	[2020–11–05 03:01:20] Writer StreamLineWrite8 writes records: 299009024	
	[2020-11-05 03:01:20] Reader StreamLineRead7 reads records: 299111424	
	[2020-11-05 03:01:20] Writer StreamLineWrite8 writes records: 299111424	
	[2020-11-05 03:01:21] Reader StreamLineRead7 reads records: 299213824	
	[2020-11-05 03:01:21] Writer StreamLineWrite8 writes records: 299213824	
	[2020-11-05 03:01:22] Reader StreamLineRead7 reads records: 299316224	
	[2020-11-05 03:01:22] Writer StreamLineWrite8 writes records: 299316224	
	[2020-11-05 03:01:23] Reader StreamLineRead7 reads records: 299418624	
	[2020-11-05 03:01:23] Writer StreamLineWrite8 writes records: 299418624	
	[2020-11-05 03:01:23] Reader StreamLineRead7 reads records: 299521024	
	[2020-11-05 03:01:23] Writer StreamLineWrite8 writes records: 299521024	
	[2020-11-05 03:01:25] Reader StreamLineRead7 reads records: 299623424	
	[2020-11-05 03:01:25] Writer StreamLineWrites writes records: 209023424	
	[2020-11-05 0310125] Keader StreamLineKead/ reads records: 209/25824	
	[2020-11-05 03:01:25] Writer StreamLinewrite0 writes records, 2009/2024	
	[2020-11-05 03:01:20] Keduer StreamLineKedu/ redus recurss: 27020224	
	[2020-11-05 03:01:20] white streamLinewisted whites fetures, 25020224	
	[2020-11-05 03:01:26] Weider Stream Indexista writes records: 20030624	
	[2020-11-05 03:01:27] Reader Stream ineRead reads records: 30003024	
		a

c. Download的内容如下图所示,可以看到倾斜Instance的StreamLineRead7的record count远远大于非倾斜Instance,因此可以认为 StreamLineWrite7到SreamLineRead7的这个Shuffle发生了倾斜。



d. 在DAG热点图界面,单击鼠标右键选择**expland all**,即可找到StreamLineWrite7和StreamLineRead7。



e. 可以看到MergeJoin2发生了倾斜,倾斜的是MergeJoin2的StreamLineRead7这一路。此时,进一步追溯StreamLineRead7的输入节点,可以看 到是dim_hm_item和dim_tb_itm_brand Join之后的结果,再与StreamLineRead4(输入是dim_tb_brand表)进行Join,即MergeJoin2。



f. 此时根据这些表名,在SQL语句中定位,可以发现是Left Outer Join发生了倾斜,而倾斜的是t1表,在SQL中添加 /*+ skewjoin(t1) */ 即 可。



4.3. 其它计算长尾调优

除了Join之外还有其它计算长尾现象产生,本文将为您介绍典型的长尾问题的场景及其解决方案。

长尾问题是分布式计算中最常见的问题之一。造成长尾问题的主要原因是数据分布不均,导致各个节点的工作量不同,整个任务需要等最慢的节点完成才能结束。

为了避免一个Worker单独运行大量的工作,需要把工作分给多个Worker去执行。

Group By 长尾

问题原因

Group By Key出现长尾,是因为某个Key内的计算量特别大。

解决办法

您可以通过以下两种方法解决:

```
• 对SQL进行改写,添加随机数,把长Key进行拆分。举例如下。
```

SELECT Key, COUNT(*) AS Cnt FROM TableName GROUP BY Key;

不考虑Combiner, Mapper会Shuffle到Reducer上,然后Reducer再做Count操作。对应的执行计划是 Mapper > Reducer。但是如果对长尾的Key再进行一次工作再分配,就变成如下语句。

```
-- 假设长尾的Key已经找到是KEY001。
SELECT a.Key
, SUM(a.Cnt) AS Cnt
FROM (
   SELECT Key
, COUNT(*) AS Cnt
FROM TableName
GROUP BY Key,
   CASE
   WHEN Key = 'KEY001' THEN Hash(Random()) % 50
   ELSE 0
   END
) a
GROUP BY a.Key;
```

由上可见,这次的执行计划变成了Mapper>Reducer>Reducer。虽然执行的步骤变长了,但是长尾的Key经过2个步骤的处理,整体的时间消耗可能 反而有所减少。

⑦ 说明 如果数据的长尾并不严重,用此方法人为地增加一次Reducer的过程,最终的消耗时间可能反而更长。

• 通过设置系统参数优化长尾问题。
set odps.sql.groupby.skewindata=true.

此设置为通用性的优化策略,无法针对具体的业务进行分析,得出的结果不一定是最优的。您可以根据实际的数据情况,用更加高效的方法来改写 SQL。

Distinct 长尾

对于Distinct,将长Key进行拆分的策略已经不生效了。此场景下,您可以考虑通过其它方式解决。

解决办法

--原始SQL**,不考虑**uid**为空的情况。** SELECT COUNT(uid) AS Pv , COUNT(DISTINCT uid) AS Uv FROM UserLog;

可以改写成如下语句。

```
SELECT SUM(PV) AS Pv
, COUNT(*) AS UV
FROM (
SELECT COUNT(*) AS Pv
, uid
FROM UserLog
GROUP BY uid
) a;
```

该方法是把Distinct改成了普通的Count,这样计算压力不会落到同一个Reducer上。而且这样改写后,既能支持前面提到的Group By优化,系统又能执 行Combiner,性能会有较大的提升。

动态分区长尾

问题原因

• 动态分区功能为了整理小文件,会在最后启用一个Reduce,对数据进行整理,所以如果使用动态分区写入数据时有倾斜,就会发生长尾。

• 一般情况下,滥用动态分区功能也是产生这类长尾的一个常见原因。

解决办法

如果已经确定需要把数据写入某个具体分区,则可以在插入数据的时候指定需要写入的分区,而不是使用动态分区。

通过 Combiner解决长尾

对于MapRedcuce作业,使用Combiner是一种常见的长尾优化策略。通过Combiner,减少Mapper Shuffle到Reducer的数据,可以大大减少网络传输的开销。对于MaxCompute SQL,这种优化会由系统自动完成。

 ⑦ 说明
 Combiner只是Map端的优化,需要保证执行Combiner的结果是一样的。以WordCount为例,传2个
 (KEY,1)
 和传1个
 (KEY,2)

 的结果是一样的。但是在做平均值时,便不能直接在Combiner里把
 (KEY,1)
 和 (KEY,2)
 合并成 (KEY,1.5)
 。

通过系统优化解决长尾

针对长尾这种场景,除了前面提到的Local Combiner, MaxCompute系统本身还做了一些优化。例如,在运行任务的时候,日志里突然打出如下的内容 (+N backups 部分)。

M1_Stg1_job0:0/521/521[100%] M2_Stg1_job0:0/1/1[100%] J9_1_2_Stg5_job0:0/523/523[100%] J3_1_2_Stg1_job0:0/523/523[100%] R6_3_9_S tg2_job0:1/1046/1047[100%]

M1_stgl_job0:0/521/521[100%] M2_stgl_job0:0/1/1[100%] J9_1_2_Stg5_job0:0/523/523[100%] J3_1_2_Stg1_job0:0/523/523[100%] R6_3_9_S tg2_job0:1/1046/1047[100%]

M1_stg1_job0:0/521/521[100%] M2_stg1_job0:0/1/1[100%] J9_1_2_stg5_job0:0/523/523[100%] J3_1_2_stg1_job0:0/523/523[100%] R6_3_9_s tg2_job0:1/1046/1047(+1 backups)[100%]

M1_stg1_job0:0/521/521[100%] M2_stg1_job0:0/1/1[100%] J9_1_2_stg5_job0:0/523/523[100%] J3_1_2_stg1_job0:0/523/523[100%] R6_3_9_s tg2_job0:1/1046/1047(+1 backups)[100%]

可以看到1047个Reducer,有1046个已经完成了,但是最后一个一直没完成。系统识别出这种情况后,自动启动了一个新的Reducer,运行一样的数据,然后取运行结束较早的数据归并到最后的结果集里。

通过业务优化解决长尾

虽然前面的优化策略有很多,但仍然不能解决所有问题。有时碰到的长尾问题,还需要从业务角度上去考虑是否有更好的解决方法。

- 实际数据可能包含非常多的噪音。例如,需要根据访问者的ID进行计算,看每个用户的访问记录的行为。需要先去掉爬虫的数据(现在的爬虫已越来 越难识别),否则爬虫数据很容易在计算时长尾。类似的情况还有根据xxid进行关联的时候,需要考虑关联字段是否存在为空的情况。
- 一些特殊的业务情况。例如,ISV的操作记录,在数据量、行为方式上会和普通个人有很大的区别。那么可以考虑针对大客户,使用特殊的分析方式 进行单独处理。
- 数据分布不均匀的情况下,尽量不要使用常量字段做Distribute by字段来实现全排序。

4.4. 长周期指标的计算优化方案

本文为您介绍如何对长周期指标的计算进行优化。

实验背景

电子商务公司在电商数据仓库和商业分析场景中,经常需要计算最近N天的访客数、购买用户数、老客数等类似的指标。这些指标需要根据一段时间内 的累积数据进行计算。

通常,这些指标的计算方式为从日志明细表中查询数据进行计算。例如,运行如下SQL语句计算商品最近30天的访客数。

select item_id --商品id
,count(distinct visitor_id) as ipv_uv_ld_001
from 用户访问商品日志明细表
where ds <= \${bdp.system.bizdate}
and ds >=to_char(dateadd(to_date(\${bdp.system.bizdate},'yyyymmdd'),-29,'dd'),'yyyymmdd')
group by item_id;

⑦ 说明 代码中的变量都是DataWorks的调度变量,仅适用于DataWorks的调度任务。下文不再重复说明。

当每天的日志量很大时,SELECT操作需要大量的Map Instance,运行上面的代码需要的Map Instance个数太多,甚至会超过99999个Instance的限制个 数,导致Map Task无法顺利执行。

实验目的

在不影响性能的情况下计算长周期的指标。

影响性能的问题根源是多天汇总数据量过大,建议您使用构建临时表的方式对每天的数据进行轻度汇总,这样可以去掉很多重复数据,减少数据量。

实验方案

1. 构建中间表, 每天汇总一次。

对于上述示例,构建 item_id+visitior_id 粒度的日汇总表,记作A。

```
insert overwrite table mds_itm_vsr_xx(ds='${bdp.system.bizdate} ')
select item_id,visitor_id,count(1) as pv
from
  (
  select item_id,visitor_id
  from 用户访问商品日志明细表
  where ds =${bdp.system.bizdate}
  group by item_id,visitor_id
  ) a;
```

2. 计算多天的数据, 依赖中间表进行汇总。

对A进行30天的汇总。

```
select item_id
    ,count(distinct visitor_id) as uv
    ,sum(pv) as pv
from mds_itm_vsr_xx
where ds <= '${bdp.system.bizdate} '
and ds >= to_char(dateadd(to_date('${bdp.system.bizdate} ','yyyymmdd'),-29,'dd'),'yyyymmdd')
group by item_id;
```

影响及思考

上述方法对每天的访问日志明细数据进行单天去重,从而减少了数据量,提高了性能。缺点是每次计算多天数据的时候,都需要读取N个分区的数据。 您可以通过增量累计方式计算长周期指标的方式,不需要读取N个分区的数据,而是把N个分区的数据压缩合并成一个分区的数据,让一个分区的数据 包含历史数据的信息。

场景示例

计算最近1天店铺商品的老买家数。老买家是指过去一段时间有购买的买家(例如过去30天)。

一般情况下,老买家数计算方式如下所示。

```
select item_id --商品id
    ,buyer_id as old_buyer_id
from 用户购买商品明细表
where ds < ${bdp.system.bizdate}
and ds >=to_char(dateadd(to_date(${bdp.system.bizdate},'yyyymmdd'),-29,'dd'),'yyyymmdd')
group by item_id
    ,buyer_id;
```

改进思路:

- 维护一张店铺商品和买家购买关系的维表A,记录买家和店铺的购买关系、第一次购买时间、最近一次购买时间、累计购买件数、累计购买金额等信息。
- 每天使用最近1天的支付明细日志更新表A的相关数据。
- 计算老买家数时,判断最近一次购买时间是否是30天之内,从而做到最大程度上的数据关系对去重,减少计算输入数据量。

5.作业诊断 5.1. 通过Logview诊断慢作业

在实际业务开发过程中,企业通常要求作业能在期望的时间节点前产出结果,并根据结果做进一步决策,这就需要作业开发人员及时关注作业运行状态,识别并优化慢作业。您可以通过MaxCompute的Logview功能诊断慢作业。本文为您介绍导致出现慢作业的原因及如何查看慢作业并提供对应的解 决措施。

背景信息

MaxCompute提供的Logview功能会记录作业的全部运行阶段日志信息,为查看和调式作业提供指导依据。您可以通过作业运行结果中的Log View处获取Logview链接。MaxCompute提供了2个版本的Logview功能,推荐您使用Logview 2.0,页面加载速度、设计风格更优。更多Logview 2.0信息,请参见Logview 2.0。

出现慢作业的常见情形如下:

• 计算资源不足

当MaxCompute项目的计费模式为包年包月时,如果某一时间段内提交的作业数量较多或小文件过多,会导致购买的计算资源(CU)全部占满,作 业变为排队状态。

• 数据倾斜

当处理的数据量很大或某些作业专门用于处理特定数据时,会导致出现大部分作业已经运行结束但某些作业迟迟不结束的长尾(Long-Tails)情况。

• 代码逻辑问题

当SQL或UDF逻辑低效,或没有使用最优的参数设定时,会导致出现Fuxi Task的运行时间很长,但Fuxi Task下每个Fuxi Instance的运行时间都很均匀的情况。作业、Fuxi Task、Fuxi Instance间的关系请参见作业详情。

计算资源不足

问题现象

提交作业后,有如下两种表现:

- 现象一:作业一直显示 Job Queueing... 状态。
 - 可能因其他作业占用了资源组的资源,出现作业排队情况。您可以通过如下步骤查看作业等待时长:

i. 在作业运行结果信息中获取Logview链接并在浏览器中打开。

odps@ doc_test_dev≻select ╡	· · · · · · · · · · · · · · · · · · ·					
ID = 20210709						
Log view:						
http://logview.odps.aliyun.	.com/logview/?h=http	://service.cn-hangzh	ou. maxcompute. aliyu	n.com/api&p=doc_	_test_dev&i=20210	0709024821160g3o56ua2&token=
d1BoQUJWTDc5ZkJzY11Pdy9v						:01J1YWQiXSwiRWZm
ZWN0IjoiQWxsb3ciLCJSZXNv				المراجع القادر		_ lfV0sI1Z1cnNpb24i
0iIxIn0=						
Job Queueing						
Job Queueing						
Job Queueing						

ii. 在SubStatusHistory页签的Description列查看Waiting for scheduling对应的Latency值,即为等待时长。

Job Detail:	s Result SourceXML SQL Script		History SubStatusHistory C	OperationHistory
Code	Description	StartTime	Latency	TimeLine
1010	Waiting for scheduling	2021/07/09 10:48:21.332	00:00:00.004	
1020	Waiting for execution	2021/07/09 10:48:21.336	00:00:00.001	
1030	Preparing for execution	2021/07/09 10:48:21.337	00:00:00.009	
1032	Task is executing	2021/07/09 10:48:21.346	00:00:00	

• 现象二: 作业有进展但执行速度慢。

提交作业后,由于所需计算资源较多,当前的资源组不能同时启动所有的Fuxi Instance,作业虽然有进度但执行不快。您可以通过如下步骤查看作业 执行状态:

- i. 在作业运行结果信息中获取Logview链接并在浏览器中打开。
- ii. 在Job Details页签的Fuxi Instance区域,单击Latency Chart,查看作业运行状态图。



Fuxi Insta	ince of Fuxi 1	ask: M11						Sn	artFilter All (3) Failed 🚺	Terminated (3)	Long-Tails (0)	Data-Skews (0)	Latency Chart C
M1 ×					M11 ×									
													E.	

产生原因

针对上述现象,您可以按照如下操作确认产生原因:

- 1. 进入MaxCompute管家。
- 2. 在左侧导航栏,单击**配额**。

Compute 🧮	包年包月配额组					配额组 > 请输入	✓	新建配線組 设置分时
既吃	配额组监控							
项目	配数组 💲	预留CU最小配额 👙	预留CU最大配额 👙	非预留CU最大配额 👙	配额组标签 👙	♡ 包含项目个数 💲	状态 👙	▽ 損作
配额	默认预付薅Quota	50	50	0		1	正常	修改 删除
F <u>√k</u>							±	1 祭 〈 1 〉 10 祭/页
۱) Mi								
主体	按量付费配額组							
睦								
ZIR								
			您好,按量付费的配	额不支持展示,您可以查看按量付费	息项目列表或按量付费作业快	照列表,谢谢!		

- 3. 在**包年包月配额组**区域,单击MaxCompute项目对应的配额组。
- 4. 在资源消耗页签的预留CU资源使用趋势图中,单击计算资源使用量最高的点,记录时间点。

2021-07-06 05:53:19 ~	2021-07-06 11:53:19 📋	
	预留CU资源使用趋势	
80		2021-07-06,11:45:00
60		预留CU最小配额: 64 预留CU贯用量: 63.87
40		
20		\sum

- 5. 在左侧导航栏,单击**作业**后,在右侧单击**作业管理**。
- 6. 在作业管理页面,根据记录的时间点填写日期范围,在作业状态下拉列表选择Running,单击确定。
- 7. 在作业列表区域,单击CPU使用占比(%)后的 图标,将作业按照 CPU使用量占比降序排列。
 - 如果某个作业占用CPU特别大,单击作业操作列的Logview,在Fuxi Instance区域查看I/O Bytes。当I/O Bytes只有1 MB或几十KB,且作业的 并行度很高(多个Fuxi Instance)时,表明读取表的小文件过多,需要合并小文件或调整并行度。

○ 如果CPU占比均匀说明同时提交了多个大作业, 占满了计算资源。您需要增购计算资源或将作业使用按量计费资源运行。

解决措施

- 合并小文件:
- 调整并行度:

MaxCompute的并行度会根据输入的数据量和作业复杂度自动推测执行,一般不需要调节,理想情况并行度越大速度处理越快。但是对于包年包月 资源组,资源组可能会占满,导致作业都在等待资源,作业运行变慢。您可以通 过odps.stage.mapper.split.size、odps.stage.reducer.num、odps.stage.joiner.num或odps.stage.num参数调整并行度。更多参数信息,请参 见SET操作。

• 购买计算资源:

增购计算资源,请参见升级。

使用按量计费资源:

购买按量计费规格,并通过MaxCompute管家设置包年包月项目使用按量计费资源。

数据倾斜

问题现象

Fuxi Task中大多数Fuxi Instance都已经结束了,但是部分Fuxi Instance迟迟不结束,出现长尾(Long-Tails)情况。

您可以在Logview的Job Details页签的Fuxi Instance区域,单击Long-Tails,查看出现长尾情况的Fuxi Instance。

Fuel Instance of Fuel Task: J3_1_2 Latency: (min.000.00:01, avg.02:27:05, max:38:54:14) SmartFilter I All (376) Faminated (368) Rumeing (18) Latency											Latency (Chart 🔿				
	× J3_1,2×															
No.			Path	Sensor	I/O Record	I/O Bytes	Status	StdOut	StdErr	Debug	Progress	Start Time	End T	ime ‡	Latency	\$
											30%		4		38:54:14	
45	J3_1_2#13_0	11.246.50.143 🗢									30%	2020/03/04 05:58:5			38:54:14	
73	J3_1_2#165_0	11.246.48.236 🖵									53N	2020/03/04 05:58:5			38:54:14	
177	J3_1_2#258_0	11.246.48.211 🖵				0 B/0 B					53N	2020/03/04 05:56:5			38:54:14	
203	J3_1_2#280_0	11.246.49.42 🗢									63%	2020/03/04 05:56:5			38:54:14	
225	J3_1_2#29_0	11.246.52.20 🖵									53%	2020/03/04 05:56:5			38:54:14	
297	J3_1_2#363_0	11.246.50.138 🖵				0 B/0 B					63N	2020/03/04 05:56:5			38:54:14	
301	J3_1_2#366_0	11.246.50.156 🖵									30%	2020/03/04 05:56:5			38:54:14	
313	J3_1_2#44_0	11.246.49.71 🖵									63%	2020/03/04 05:56:5			38:54:14	
321	J3_1_2#50_0	11.246.49.45 🖵				0 B/0 B					30%	2020/03/04 05:56:5			38:54:14	
322	J3_1_2#51_0	11.194.251.145 🖵				0 B/0 B					53N	2020/03/04 05:56:5			38:54:14	

产生原因

Fuxi Instance处理的数据多或Fuxi Instance负责处理特殊数据。

解决措施

数据倾斜的详细解决措施,请参见数据倾斜优化。

代码逻辑问题

问题现象

提交作业后,有如下两种表现:

• 现象一:数据膨胀。FuxiTask的输出数据量比输入数据量大很多。

输入、输出数据量可通过FuxiTask区域的**I/O Record和I/O Bytes**参数获取。如下图所示,1GB的数据经过处理变为了1TB,一个FuxiInstance处 理1TB的数据,会降低运行效率。

SQL_0_0_0)_job_0						
Fuxi Task	Failed/T	erminated/ALL	I/O Record	I/O Bytes		Status	Sensor
M1	0/10,054	1/10,054	793.6 M/1.3 T	1.95 TB/14.88 TB		Terminated	
R2_1	0/4,000/4,000(+2 backups)		1.3 T/8.8 G	14.88 TB/189.14 GB		Terminated	
R3_2	0/4,000/4,000		8.8 G/5.6 M	189.14 GB/113.22 GB	;	Terminated	

• 现象二: UDF执行效率低。

某个FuxiTask执行效率低,且该FuxiTask中有UDF。当UDF执行超时报错 Fuxi job failed - WorkerRestart errCode:252,errMsg:kInstanceMoni torTimeout, usually caused by bad udf performance 。您可以通过如下步骤查看UDF的位置及执行速度:

i. 在作业运行结果信息中获取Logview链接并在浏览器中打开。

ii. 在作业执行图区域,双击运行速度慢或运行失败的FuxiTask,在Operator算子图中查看UDF的位置。如下图所示。

最佳实践·作业诊断



iii. 在Fuxi Instance区域,单击StdOut,查看UDF的执行速度。

通常, Speed(records/s) 在百万或者十万级别。

[2021-07-12 15:35:16] Table reader	TableScan1 has read 3	records							
[2021-07-12 15:35:16] Table writer	AdhocSink1 has produce	d 7 records							
[2021-07-12 15:35:16] Table writer AdhocSinkl has produced 7 records									
[2021-07-12 15:35:16] Table writer	cursor AdhocSink1 proc	ess data elapsed	time(ms): 50.636;						
CursorId	OutputCount	InnerTime(ms)	Speed(records/s)	Others					
AdhocSink1			50 {	{"ByteEncodingCount":3, "ByteEncodingInBytes":21, "CompressionInBytes"	1:536,				
"CompressionLatencyUs":13180, "Comp	ressionOutBytes":519,"E	incodingCount":5,"	EncodingInBytes":9	96, "EncodingLatencyUs": 30, "EncodingOutBytes": 20, "IOBlockingLatencyUs	s":6,"IOCount":7,				
"WriterCount":1,"WriterInclLatency	Us":97}								
GlobalInit		458	0 {	<pre>{"LoadAndParseIR":{"T":3500},"OperatorCreation":{"T":455302}}</pre>					
TableFunctionScan1			14285						
TableScan1			54 {	{"DecodingCount":2, "DecodingInBytes":4, "DecodingLatencyUs":1, "Decoding	ingOutBytes":12,				
"DecompLatencyUs":2,"IOBlockingLatencyUs":1255,"IOCount":3, "ReadRowCount":3, "ReaderCount":1, "ReaderInclLatencyUs":3570, "RequestedRowCount":3, "VectorBatchConversionLatencyUs":25}									
[2021-07-12 15:35:16]	End Of Task: M1#0 -								

产生原因

- 现象一: 实际业务处理逻辑导致出现数据膨胀, 需要确认业务逻辑是否存在问题。
- 现象二: UDF代码逻辑不合理, 需要调整代码逻辑。

解决措施

- 现象一的解决措施:确认业务逻辑是否存在问题,如果存在问题请修改;如果不存在问题,请通
- 过odps.stage.mapper.split.size、odps.stage.reducer.num、odps.stage.joiner.num或odps.stage.num参数调整并行度。更多参数信息,请参 见SET操作。
- 现象二的解决措施:检查并修改UDF代码逻辑,优先使用内建函数实现,如果内建函数无法满足业务需求,再考虑使用UDF。更多内建函数信息,请参见内建函数。

6.成本优化6.1.成本优化概述

本文介绍了成本优化的流程。

MaxCompute的成本优化是一个持续不断的过程。由于大数据的动态性和不断变化的性质,企业用户成本优化的活动应该持续不断的进行。您可以参考以下流程进行优化:

- 1. 在使用MaxCompute之前,建议您详细了解付费策略以及预估自己需要使用的资源,选择适合您的付费方式。详情请参见选择付费方式。
- 在使用过程中,可以从计算、存储、上传和下载这几个方面进行优化,以减少成本。详情请参见计算成本优化、存储成本优化、数据上传下载成本优化。
- 3. 及时查看账单, 对账单中的异常点进行分析和优化。详情请参见成本追踪。

6.2. 选择付费方式

本文为您介绍如何根据实际情况选择付费方式以降低使用成本。

MaxCompute的计费策略

MaxCompute提供了两种计费方式:

- 包年包月: 计算资源是包月或者包年的, 存储和下载资源是按实际使用量计费。
- 按量计费:存储、计算和下载资源都是按实际使用量计费。

详细的计费策略请参见计费项与计费方式概述。MaxCompute提供了TCO工具和成本预估实践两种选型工具帮助您分析如何选择付费方式。

TCO工具

您可以使用如下TCO工具进行费用预估:

- MaxCompute报价速算器:适用包年包月方式。您可通过输入上传和下载的数据大小以及需要的计算资源自动地计算月成本。
- CostSQL方法:适用按量计费方式。
 - 您在实际生产环境中,即正式上线一个分析SQL前,可以通过Cost SQL命令估算该SQL作业的费用。详情请参见计量预估。
 - 如果您使用的开发工具为Intellij IDEA,可以在提交SQL脚本时自动估算费用。详情请参见开发及提交SQL脚本。
 - 如果您使用的开发工具为DataWorks,也可以进行费用预估。

? 说明

- 部分SQL运算不支持费用预估。例如,外部表参与计算的SQL。
- 此功能仅为费用预估,实际费用以最终出账为准。

成本预估实践

成本预估实践为您提供了一些成本预估的案例和技巧,您可以根据实际情况,参考案例选择最经济的付费方式。

● 处理1 TB数据的付费方式。

经过相关测试,对于费用我们给出如下预估信息供您参考。

付费方式	用户类型	响应速度	每月预估费用
勾在勾日	密集型计算	分钟级别	3768 USD
민두민거	密集型存储	小时级别	1177.5 USD
按量计费			

包年包月方式下根据您对CPU资源的要求,有如下两种建议:

- 密集型计算:适用于CPU资源要求比较高的场景。使用160 CU资源运行1 TB数据,响应速度为分钟级别,资源费用为每月3768 USD左右。
- 密集型存储:如果对于计算的响应时间要求不高,推荐您使用包年包月密集型存储,使用的计算资源为50 CU左右,响应速度为小时级别,费用为 每月1177.5 USD左右。

如果您选择按量计费,按照基础的复杂度1来计算,对于1 TB的数据的单次计算资源费用大约为47.1 USD/天,一个月为1413 USD。按量计费是按照 次数计费的,如果多次进行1 TB数据的计算,其费用也会成倍增加。

对于刚开始上云的企业,建议先开通按量计费,然后将数据进行POC测试(即针对客户具体应用的验证性测试),计算自己的任务大概需要消耗多少 Worker,通过Worker数推算CU数量,这样就能大概估算出最终需要购买资源的数量。

• Hadoop用户上云迁移的付费方式。

某个Hadoop集群可能有1个管控节点以及5台计算节点,每台机器32核,相当于是32个CPU,5台计算节点就是160个CPU,对应标准的官方报价是每 个月3768 USD(此价格未包含折扣或者优惠)。 MaxCompute无需考虑管控节点,比Hive性能快80%,且免运维,为您节省成本。

- 混合使用的付费方式。
 - 包年包月模式进行生产业务(小时级ETL)+按量计费模式进行非周期任务或即席查询。

对周期性高密度计算作业使用包年包月模式,对非周期性的大规模数据处理作业使用按量计费模式。按量计费模式下可以不存储数据,通过读取 其它账号下的表获取数据,从而可以节省数据存储费用。不同账号下跨表计算需要通过授权来实现,详细请参见创建角色(项目级别)。

○ 包年包月模式进行非周期任务或即席查询+按量计费模式进行生产业务(天级别ETL)。

企业为了解决因为日常数据测试引起的费用不可控的问题,可以把数据测试和非周期任务放在固定资源组,通过MaxCompute管家为开发组和B/组 配置不同的二级资源。生产作业如果只是每天处理一次,可以放在按量计费资源组。

调整付费方式

如果您选择了包年包月模式的服务,但因为业务变化导致数据量急剧变化,资源不够使用或者空余,您可以进行升配或者降配,详情请参见<mark>升级和降</mark> 配。

此外您还可以灵活地选择和转变付费方式。例如,从包年包月转换成按量计费,或从按量计费转换成包年包月,详细请参见<mark>转换计费方式</mark>。

⑦ 说明 请合理评估计算作业性能与时间的关系,避免转换成包年包月模式后,由于购买的CU数量少,导致延长作业计算周期,达不到预期后 又转回按量计费模式。

6.3. 计算成本优化

本文为您介绍如何通过对SQL作业和MapReduce作业的优化减少计算成本。

您可以在计算前对计算成本进行预估,控制计算成本。详细的预估方法,请参见TCO工具。也可以配置消费预警,预防意料之外的高额费用。如果计算 成本过高,您可以参考下面的方法进行优化,以控制计算成本。

SQL作业计算成本控制

对于SQL计算作业,大部分费用较高的SQL都是由全表扫描引起的。另外,调度频繁也会引起SQL作业费用的增加,调度频繁可能会产生任务的堆积, 在后付费的情况下会造成排队现象,如果任务多又出现了排队,那么第二天的账单就会异常。通过如下策略进行SQL作业计算成本控制:

- 避免频繁调度。MaxCompute是批量计算的服务,距离实时的计算服务还是存在一定距离的。如果间隔时间变短,计算频率增加,再加上使用SQL的不良习惯就会导致计算费用飙升,产生费用较高的账单。所以请尽量避免频繁调度,如果要进行频繁调度请通过CostSQL等方式预估一下SQL的开销 到底有多大,不然会造成较大预估外的开销。
- 控制全表扫描。您可以通过以下几种策略来控制全表扫描问题:
- 设置参数关闭全表扫描功能。目前支持Session级别和Project级别的控制。

--禁止session 级别全表扫描。 set odps.sql.allow.fullscan=false; --禁止project级别全表扫描。 SetProject odps.sql.allow.fullscan=false;

○ 使用列剪裁。在读数据的时候,只读取查询中需要用到的列,而忽略其他列,避免使用 SELECT ★ 引起全表扫描。

SELECT a, b FROM T WHERE e < 10;

其中,T包含5个列 (a,b,c,d,e) ,列c,d将会被忽略,只会读取a,b,e列。

○ 使用分区剪裁。分区剪裁是指对分区列指定过滤条件,使得只读取表的部分分区数据,避免全表扫描引起的错误及资源浪费。

SELECT a,b FROM T WHERE partitiondate='2017-10-01';

- SQL关键字的优化。计费的SQL关键字包括: JOIN、GROUP BY、ORDER BY、DIST INCT、INSERT INTO。您可以根据以下建议进行优化:
- 在进行JOIN的时候,一定要先进行分区剪裁再进行JOIN,不然的话就可能会先做全表扫描。分区裁剪失效请参考分区剪裁失效的场景分析。

■ 减少FULL OUT ER JOIN 的使用, 改为UNION ALL。

```
SELECT COALESCE(t1.id, t2.id) AS id, SUM(t1.col1) AS col1
, SUM(t2.col2) AS col2
FROM (
SELECT id, coll
FROM table1
) t1
FULL OUTER JOIN (
SELECT id, col2
FROM table2
) t2
ON t1.id = t2.id
GROUP BY COALESCE(t1.id, t2.id);
--可以优化为如下语句。
SELECT t.id, SUM(t.col1) AS col1, SUM(t.col2) AS col2
FROM (
SELECT id, coll, 0 AS col2
FROM table1
UNION ALL
SELECT id, 0 AS coll, col2
FROM table2
) t
GROUP BY t.id;
```

■ 在UNION ALL内部尽可能不使用GROUP BY, 改为在外层统一GROUP BY。

```
SELECT t.id, SUM(t.val) AS val
FROM (
SELECT id, SUM(col3) AS val
FROM table3
GROUP BY id
UNION ALL
SELECT id, SUM(col4) AS val
FROM table4
GROUP BY id
) t
GROUP BY t.id;
可以优化为----
SELECT t.id, SUM(t.val) AS val
FROM (
SELECT id, col3 AS val
FROM table3
UNION ALL
SELECT id, col4 AS val
FROM table4
) t
GROUP BY t.id;
```

- 临时导出的数据如果需要排序,尽量在导出后使用Excel等工具进行排序,避免使用ORDER BY。
- 尽量避免使用DIST INCT关键字, 改为多套一层GROUP BY。

- 尽量避免使用INSERT INTO方式写入数据,可以考虑增加一个分区字段。通过降低SQL复杂度,来节省SQL的费用。
- 避免使用运行查询的方式预览表数据。如果您想预览表数据,可以使用表预览的方式查看数据,而不会产生费用。如果您使用DataWorks,在数据 地图页面,可以预览表以及查看表的详情,具体方法请参见查看表详情。如果您使用MaxCompute Studio,双击表就可以进行表数据预览。
- 计算时合理的选择工具。由于MaxCompute的查询响应是分钟级,不适合直接用于前端查询,计算出的结果数据同步到外部存储中保存,对于大部分用户来说,关系型数据库是最优先的选择。轻度计算推荐使用MaxCompute,重度计算(即直接出最终结果。前端展示时,不做任何判断、聚合、关联字典表、甚至不带WHERE条件)推荐使用RDS等关系型数据库。

MapReduce作业计算成本控制

通过如下策略进行MapReduce作业计算成本控制:

• 设置合理的参数。

• split size

Map默认的split size是256 MB, split size的大小决定了Map个数多少,如果用户的代码逻辑比较耗时,Map需要较长时间结束,可以通过 JobCon f#setSplitSize 方法适当调小 split size 。然而 split size 也不宜设置太小,否则会占用过多的计算资源。

MapReduce Reduce Instance

单个job默认Reduce Instance个数为Map Instance个数的1/4,用户设置作为最终的Reduce Instance个数,范围[0, 2000],数量越多,计算时消耗 越多,成本越高,应合理设置。

• MapReduce减少中间环节

如果有多个MapReduce作业之间有关联关系,前一个作业的输出是后一个作业的输入,可以考虑采用Pipeline的模式,将多个串行的MapReduce作 业合并为一个,这样可以用更少的作业数量完成同样的任务。一方面减少中间表造成的多余磁盘IO,提升性能;另一方面减少作业数量使调度更加简 单,增强流程的可维护性,具体使用方法请参见Pipeline示例。

• 对输入表列裁剪

对于列数特别多的输入表,Map阶段处理只需要其中的某几列,可以通过在添加输入表时明确指定输入的列,减少输入量。例如只需要c1,c2列,可以参考如下设置。

InputUtils.addTable(TableInfo.builder().tableName("wc_in").cols(new String[]{"cl","c2"}).build(), job);

设置后,在Map中读取到的Record就只有c1,c2列,如果之前是使用列名获取Record数据,不会有影响,而用下标获取的需要注意这个变化。

• 避免资源重复读取

资源的读取尽量放置到Setup阶段读取,避免资源多次读取的性能损失,另外系统也有64次读取的限制,资源的读取请参见使用资源示例。

• 减少对象构造开销

对于Map、Reduce阶段每次都会用到的Java对象,避免在Map/Reduce函数里构造,可以放到Setup阶段,避免多次构造产生的开销。

```
{
    ...
    Record word;
    Record one;
    public void setup(TaskContext context) throws IOException {
        // 创建一次就可以,避免在map中每次重复创建。
        word = context.createMapOutputKeyRecord();
        one = context.createMapOutputValueRecord();
        one.set(new Object[]{1L});
    }
    ...
}
```

合理使用Combiner

如果Map的输出结果中有很多重复的Key,可以合并后输出,Combiner后可以减少网络带宽传输和一定Shuffle的开销。如果Map输出本来就没有多少重复的,就不要用Combiner,用了反而可能会有一些额外的开销。Combiner实现的是和Reducer相同的接口,例如一个WordCount程序的Combiner可以定义如下。

```
^{\ast} A combiner class that combines map output by sum them.
 */
public static class SumCombiner extends ReducerBase {
 private Record count;
 @Override
 public void setup(TaskContext context) throws IOException {
   count = context.createMapOutputValueRecord();
  }
 @Override
 public void reduce(Record key, Iterator<Record> values, TaskContext context)
     throws IOException {
   long c = 0;
   while (values.hasNext()) {
     Record val = values next():
     c += (Long) val.get(0);
   }
   count.set(0, c);
   context.write(key, count);
  }
```

• 合理选择Partition Column或自定义Partitioner

合理选择Partition Columns,可以使用 JobConf#setPartitionColumns 这个方法进行设置(默认是Key Schema定义的Column),设置后数据将按照指定的列计算HASH值分发到Reduce中,避免数据倾斜导致作业长尾现象,如有必要也可以选择自定义Partitioner,自定义Partitioner的使用方法如下。

import com.aliyun.odps.mapred.Partitioner; public static class MyPartitioner extends Partitioner { @Override public int getPartition(Record key, Record value, int numPartitions) { // numPartitions即对应reducer的个数 // 通过该函数决定map输出的key value去往哪个reducer。 String k = key.get(0).toString(); return k.length() % numPartitions; } }

在 job conf 里进行设置如下。

jobconf.setPartitionerClass(MyPartitioner.class)

需要在jobconf里明确指定Reducer的个数。

jobconf.setNumReduceTasks(num)

● 合理使用JVM内存参数

过于追求调优,把MapReduce任务内存设置过大也会造成成本上升。标准配置是 1 Core 4G, odps.stage.reducer.jvm.mem=4006 , 当CPU与内存比超过 1:4 时,对应的费用也会大幅升高。

6.4. 存储成本优化

本文从数据分区、表生命周期和定期删除表3个方面为您介绍如何优化存储成本。

- 对于存储优化而言,有三个关键点:
- 合理地进行数据分区。
- 设置合理的表生命周期。
- 定期地删除废表。

.

合理设置数据分区

MaxCompute将分区列的每个值作为一个分区。您可以指定多级分区,即将表的多个字段作为表的分区,分区之间的关系类似多级目录的关系。在使 用数据时如果指定了需要访问的分区名称,则只会读取相应的分区,避免全表扫描,提高处理效率,降低费用。

- 假如最小统计周期为天,建议采用日期作为分区字段。每天将数据迁移到指定分区,再读取指定分区的数据进行下游统计。
- 假如最小统计周期为小时,建议采用日期+小时作为分区字段。每小时将数据迁移到指定分区,再读取指定分区的数据进行下游统计。如果小时调度 的统计任务也按天分区,数据每小时追加,则每小时将多读取大量的无用数据,增加不必要的费用。

您可以根据实际的业务情况选择分区字段,除了日期和时间,也可以使用其他的枚举值个数相对固定的字段,例如渠道、国家和省份地市。或者使用 时间和其他字段共同作为分区字段。一般而言,推荐使用二级分区,因为最大的单表最多只支持6万个分区。

合理设置表生命周期

您可以根据数据本身的使用情况,在创建表时对表设置生命周期,MaxCompute会及时删除超过生命周期的数据,达到节省存储空间的目的。

例如,创建一张生命周期为100天的表。如果这张表或者分区的最后修改时间超过了100天将会被删掉。

CREATE TABLE test3 (key boolean) PARTITIONED BY (pt string, ds string) LIFECYCLE 100;

生命周期最小单位是分区,所以一个分区表中,如果部分分区达到了生命周期的阈值,那么这些分区会被直接删掉,未达到生命周期阈值的分区不受 影响。

已经创建的表可以通过如下命令修改生命周期。详情请参见生命周期操作。

ALTER TABLE table_name SET lifecycle days;

删除废表

建议您定期地删除访问跨度大(即长期不会访问)的废表,因为这些表的意义并不大,会极大的浪费存储资源,例如:

- 3个月内没有被访问的表。
- 一张表是非分区表,同时最近1个月内没有被访问。
- 存储为0KB的表,即没有存储的表。

6.5. 数据上传下载成本优化

本文为您介绍如何优化数据上传和下载的同步成本。

```
• 尽可能使用经典网络和VPC网络
```

您可以使用内部网络(经典网络或VPC)实现零成本数据导入和导出。网络设置详情请参见Endpoint。

• 合理利用ECS的公共下载资源

如果您的ECS使用包月资源,可以使用Tunnel等数据同步工具,将MaxCompute数据同步到ECS,然后下载到本地。详情请参见<mark>导出SQL的运行结果</mark>。

● Tunnel文件上传优化

小文件会消耗更多计算资源,建议当文件量积累较大时一次性上传。例如,调用Tunnel SDK时,缓存达到64 MB提交一次。

● 合理预估VPC带宽

当数据在IDC机房时,如果您需要通过专线同步数据到MaxCompute,请预估带宽,平衡数据同步与带宽之间的成本。例如,50TB数据上云,同步1 天,预估需要5GB带宽。带宽的计算方式为 50(TB)×1024(GB)×8(bit)/(24(小时)×3600(秒))=4.7 GB/s 。

6.6. 成本追踪

本文介绍如何追踪成本的消耗,优化资源的使用以及减少费用。

阿里云为您提供了三个成本管理工具:

• 账单明细: 您可以在阿里云的费用中心看到。

• 使用记录: 它会记录每条SQL的使用, 复杂度、计量时间以及一天24小时的存储情况和下行流量等明细。

• 命令行工具: 您可以通过命令行工具还原之前操作, 查看如何产生了所谓的"贵SQL"。

账单明细

建议您定期查看账单,及时优化使用成本。您可以通过控制台查看账单明细。包年包月的出账时间是次日的12点,按量计费的出账时间是次日的9点。 账单明细的查看方法,请参见查看账单详情。

使用记录

当您在账单里面发现某一个项目的计费可能突然在某一天达到了几千元,是平常账单的多倍,类似这样的异常情况,需要查看它的明细。您可以下载 使用记录,查看异常记录的详细情况。使用记录的下载请参见查看账单详情。

对于存储费用,按照小时推送计量信息,1天24次。计算存储价格时需要将字节数相加并计算一个24小时的平均值,之后再按照阶梯定价的公式进行计 算最终得到存储价格。

计量信息是以每一条任务的结束时间为准,如果某条任务的结束时间是第二天凌晨,那么这条任务的计量时间就会计入第二天,不会计入第一天。

对于下载费用,内网也就是经典网络的下行流量是不收费的,上行流量也是不收费的。只有使用公网的时候,下行流量才会计费。

命令行

当发现异常的SQL时,可以通过命令行还原当时的情景。

• 通过使用记录或者 show p; 命令获取异常数据的InstanceId,使用 wait InstanceId 命令获取其Logview,即SQL的详细日志,并将logview打印 出来查看执行时的具体情况。

⑦ 说明 目前只能获取7天之内的logview信息,更早的信息无法获得。

• 使用 desc instance instid 将SQL显示在控制台里面。

6.7. 计费命令参考

本文介绍常用的计费命令。

语法表达式	用途	是否收费	样例
T UNNEL DOWNLOAD	下载数据(经典网络)	否	TUNNEL DOWNLOAD table_name e:/table_name.txt; 配置经典网络 Endpoint: http://dt.cn- shanghai.maxcompute.aliyun- inc.com Endpoint的配置,请参见Endpoint。
T UNNEL DOWNLOAD	下载数据(公网)	收费	TUNNEL DOWNLOAD table_name e:/table_name.txt; 配置外网网络Endpoint: http://dt.cn- shanghai.maxcompute.aliyun.co m Endpoint的配置,请参见Endpoint。

最佳实践·成本优化

大数据计算服务

TUNNEL UPLOAD	上传数据	否	TUNNEL UPLOAD e:/table_name.txt table_name;
COST SQL	费用预估	否	COST SQL SELECT * FROM table_name;
INSERT OVERWRITESELECT	数据更新	收费	INSERT OVERWRITE TABLE table_name PARTITION (sale_date='20180122') SELECT shop_name, customer_id, total_price FROM sale_detail;
DESC TABLE	查看表信息	否	<pre>DESC table_name;</pre>
DROP TABLE	删除表及表数据	否	DROP TABLE if exists table_name;
CREATE TABLE	创建表	否	CREATE TABLE if not exists table_name (key string ,value bigint) PARTITIONED BY(p string);
CREATE TABLESELECT	创建表	收费	CREATE TABLE if not exists table_name AS SELECT * FROM a_tab;
INSERT INTO TABLEVALUES	快速插入常量数据	否	<pre>INSERT INTO TABLE table_name partition (p) (key,p) VALUES ('d','20170101'), ('e','20170101'), ('f','20170101');</pre>
INSERT INTO TABLESELECT	插入数据	收费	<pre>INSERT INTO TABLE table_name SELECT shop_name, customer_id, total_price FROM sale_detail;</pre>
SELECT UDF [NOT COUNT or All] FROM TABLE	查询表数据	收费	SELECT sum(a) FROM table_name;
SET FLAG	会话设置	否	SET odps.sql.allow.fullscan=true;
JAR MR	运行MapReduce作业	收费	JAR -1 com.aliyun.odps.mapred.exampl e.WordCount wc_in wc_out
ADD JAR/FILE/ARCHIVE/TABLE	注册资源	否	ADD jar data\resources\mapreduce- examples.jar -f;
DROP JAR/FILE/ARCHIVE/TABLE	删除资源	否	DROP RESOURCE sale.res
LIST RESOURCES	查看资源列表	否	LIST RESOURCES;
GET RESOURCES	下载资源	否	GET RESOURCES odps-udf- examples.jar d:\;
CREATE FUNCTIONS	注册函数	否	CREATE FUNCTION test_lower;
DROP FUNCTIONS	删除函数	否	DROP FUNCTION test_lower;
LIST FUNCTIONS	查看函数列表	否	LIST FUNCTIONS;
ALT ER TABLEDROP PARTITION	删除表分区	否	ALTER TABLE user DROP if exists partition(region='hangzhou',d t='20150923');

TRUNCATE TABLE	删除非分区表数据	否	TRUNCATE TABLE table_name;
CREATE EXTERNAL TABLE	创建外表	否	CREATE EXTERNAL TABLE IF NOT EXISTS ambulance_data_csv_external LOCATION 'oss://oss-cn- shanghai- internal.aliyuncs.com/oss- odps-test/Demo/'
SELECT [EXTERNAL] TABLE	读取外表	收费	SELECT recordId, patientId, direction FROM ambulance_data_csv_external WHERE patientId > 25;
SHOW TBALES	列出当前项目空间下所有的表	否	SHOW TABLES;
SHOW PARTITIONS table_name	列出一张表中的所有分区	否	SHOW PARTITIONS <table_name></table_name>
SHOW INSTANCE/SHOW P	返回由当前用户创建的实例信息	否	SHOW INSTANCES/SHOW P
WAIT INSTANCE	返回指定实例Logview	否	WAIT 20131225123302267gk3u6k4y2
STATUS INSTANCE	返回指定实例的状态	否	STATUS 20131225123302267gk3u6k4y2
KILL INSTANCE	停止您指定的实例	否	KILL 20131225123302267gk3u6k4y2

6.8. MaxCompute账单分析最佳实践

如果您想了解费用的分布情况或确保在使用MaxCompute产品产生的费用不超出预期时,您可以通过获取MaxCompute账单进行分析,为资源使用率 最大化及降低成本提供有效支撑。本文为您介绍如何通过用量明细表分析MaxCompute的费用分布情况。

背景信息

MaxCompute是一款大数据分析平台,其计算资源的计费方式分为包年包月和按量计费两种。MaxCompute每天以项目为维度进行计费,账单会在第 二天06:00前生成。更多MaxCompute计量计费信息,请参见计费项与计费方式概述。

MaxCompute会在数据开发阶段或者产品上线前发布账单波动(通常情况下为消费增长)信息。您可以自助分析账单波动情况,再对MaxCompute项 目的作业进行优化。您可以在阿里云费用中心下载阿里云所有收费产品的用量明细。更多获取和下载账单操作,请参见查看账单详情。

上传用量明细数据至MaxCompute

1. 使用MaxCompute客户端(odpscmd)按如下示例语句创建表maxcomputefee。

```
DROP TABLE IF EXISTS maxcomputefee;
CREATE TABLE IF NOT EXISTS maxcomputefee
(
   projectid STRING COMMENT '项目编号'
   ,feeid STRING COMMENT '计费信息编号'
   ,type STRING COMMENT '数据分类,包括Storage、ComputationSQL、DownloadEx等'
   ,storage BIGINT COMMENT '存储 (Byte) '
   ,endtime STRING COMMENT '结束时间'
   , computationsqlinput BIGINT COMMENT 'SQL/交互式分析读取量(Byte)'
   ,computationsqlcomplexity DOUBLE COMMENT 'SQL复杂度'
   ,uploadex BIGINT COMMENT '公网上行流量Byte'
   , download BIGINT COMMENT '公网下行流量Byte'
   ,cu_usage DOUBLE COMMENT 'MR计算时*second'
   , input_oss BIGINT COMMENT '访问OSS的数据输入量'
   ,starttime STRING COMMENT '开始时间'
   ,source_type STRING COMMENT '计算资源'
   ,source_id STRING COMMENT 'DataWorks调度任务ID'
```

```
);
```

账单明细字段说明如下:

○ 项目编号:当前账号或RAM用户对应的阿里云账号的MaxCompute项目列表。

○ 计量信息编号:以存储、计算、上传和下载的任务ID为计费信息编号,SQL为Inst anceID,上传和下载为T unnel SessionId。

。 数据分类: Storage(存储)、ComputationSql(计算)、UploadIn(内网上传)、UploadEx(外网上传)、DownloadIn(内网下载)、

DownloadEx (外网下载)。按照计费规则只有红色部分为实际计费项目。

- 存储 (Byte): 每小时读取的存储量, 单位为Byte。
- 开始时间或结束时间:按照实际作业执行时间进行计量,只有存储是按照每个小时取一次数据。
- SQL/交互式分析读取量(Byte): SQL计算项,每一次SQL执行时SQL的Input数据量,单位为Byte。
- SQL复杂度: SQL的复杂度,为SQL计费因子之一。
- 公网上行流量(Byte)或公网下行流量(Byte):分别为外网上传或下载的数据量,单位为Byte。
- MR/Spark作业计算(Core*Second): MapReduce或Spark作业的计算时单位为 Core*Second ,需要转换为计算时Hour。
- SQL读取量_访问OTS(Byte)、SQL读取量_访问OSS(Byte):外部表实施收费后的读取数据量,单位为Byte。

2. 使用Tunnel上传数据。

在上传CSV文件时,您需确保CSV文件中的列数、数据类型必须与表maxcomputefee的列数、数据类型保持一致,否则会导入失败。

tunnel upload ODPS_2019-01-12_2019-01-14.csv maxcomputefee -c "UTF-8" -h "true" -dfp "yyyy-MM-dd HH:mm:ss";

Dende Lastron, 70180115109000420-0000000560 Dende Lastron, 70180115109000420-000000560 Sang Un ta plat: Arecend Dende Lastron, Sang Unit and Sang Unit 2019-01-22,003-01-44. Sang Unit applit: Facerd Dende Lastron, Sang Unit Topot for 1 blocks 0018-01-51 50-300 store block complete, Bockid-1 002-01-51 50-300 store block complete, Bockid-1 004-01-51 50-300 store block complete, Bockid-1 04-51 50-300 store block complete, Bockid-1 04-51 50-500 store block complete, Bockid-1 50 50-5

? 说明

- Tunnel的配置详情请参见Tunnel命令。
- 。 您也可以通过DataWorks的数据导入功能来执行此操作,具体请参见数据集成导入数据。

3. 执行如下语句验证数据。

SELECT * FROM maxcomputefee limit 10;



1. 分析SQL费用。云上用户使用MaxCompute, 95%的用户通过SQL即可满足需求, SQL也在消费增长中占很大比例。

```
⑦ 说明 一次SQL计算费用 = 计算输入数据量×SQL复杂度×单价(0.0438 USD/GB)
```

```
--分析SQL消费,按照sqlmoney排行。

SELECT to_char(endtime,'yyyymmdd') as ds,feeid as instanceid

,projectid

,computationsqlcomplexity --复杂度

,SUM((computationsqlinput / 1024 / 1024 / 1024)) as computationsqlinput --数据输入量(GB)

,SUM((computationsqlinput / 1024 / 1024 / 1024)) * computationsqlcomplexity * 0.0438 AS sqlmoney

FROM maxcomputefee

WHERE TYPE = 'ComputationSql'

AND to_char(endtime,'yyyymmdd') >= '20190112'

GROUP BY to_char(endtime,'yyyymmdd'),feeid

,projectid

,computationsqlcomplexity

ORDER BY sqlmoney DESC

LIMIT 10000

;
```

根据查询结果可以得到以下结论:

- 大作业可以减小数据读取量、降低复杂度、优化费用成本。
- 可以按照ds字段(按照天)进行汇总,分析某个时间段内的SQL消费金额走势。例如利用本地Excel或Quick Bl等工具绘制折线图等方式,更直观 的反应作业的趋势。
- 根据执行结果可以定位到需要优化的点,方法如下:

- a. 通过查询的instanceid,获取目标实例运行日志的Logview地址。
 - 在MaxCompute客户端(odpscmd)或DataWorks中执行 wait <instanceid>; 命令, 查看instanceid的运行日志。



b. 在浏览器中打开Logview的URL地址,在Logview页面的SourceXML页签,获取该实例的SKYNET_NODENAME。



? 说明

- 关于Logview的介绍,详情请参见使用Logview 2.0查看Job运行信息。
- 如果获取不到SKYNET_NODENAME或SKYNET_NODENAME无值,您可以在SQL Script页签获取代码片段后,在DataWorks上通过 搜索代码片段,获取目标节点进行优化。详情请参见DataWorks代码搜索。
- c. 在DataWorks中,搜索查询到的SKYNET_NODENAME,对目标节点进行优化。

2. 分析作业增长趋势。通常费用的增长是由于重复执行或调度属性配置不合理造成作业量暴涨。

--分析作业增长趋势。

SELECT	TO_CHAR(endtime,'yyyymmdd') AS ds		
	,projectid		
	,COUNT(*) AS tasknum		
FROM	maxcomputefee		
WHERE	TYPE = 'ComputationSql'		
AND	<pre>TO_CHAR(endtime,'yyyymmdd') >= '20190112'</pre>		
GROUP BY TO_CHAR(endtime, 'yyyymmdd')			
	,projectid		
ORDER B	Y tasknum DESC		
LIMIT	10000		
;			

执行结果如下。

	A		В		C	
1	ds	~	projectid	~	tasknum	~
2	20190112				311	
3	20190113				304	
4	20190114		All states and states and		282	

从执行结果可以看出12~14日提交到MaxCompute且执行成功的作业数的波动趋势。

3. 分析存储费用。

```
--分析存储费用。
SELECT t.ds
       ,t.projectid
       ,t.storage
       ,CASE WHEN t.storage < 0.5 THEN 0.01
               WHEN t.storage >= 0.5 AND t.storage <= 10240 THEN t.storage*0.0072
                WHEN t.storage > 10240 AND t.storage <= 102400 THEN (10240*0.0072+(t.storage-10240)*0.006)
               WHEN t.storage > 102400 THEN (10240*0.0072+(102400-10240)*0.006+(t.storage=102400)*0.004)
       END storage_fee
FROM
       (
           SELECT to_char(starttime,'yyyymmdd') as ds
                  ,projectid
                   ,SUM(storage/1024/1024/1024)/24 AS storage
           FROM maxcomputefee
WHERE TYPE = 'Storage'
           and to_char(starttime,'yyyymmdd') >= '20190112'
           GROUP BY to_char(starttime,'yyyymmdd')
                    ,projectid
       ) t
ORDER BY storage_fee DESC
;
```

执行结果如下。
 A
 B
 C
 D

 ds
 projectid
 storage
 storage_fee
 3 20190112 4 20190114 根据执行结果可以分析得出如下结论: 。存储在12日有一个增长的过程,但在14日出现降低。 ○ 存储优化,建议为表设置生命周期,删除长期不使用的临时表等。 4. 分析下载费用。 对于外网或者跨区域的数据下载, MaxCompute将按照下载的数据量进行计费。 ⑦ 说明 一次下载费用=下载数据量×单价(0.1166USD/GB) --分析下载消费明细。 SELECT TO_CHAR(starttime,'yyyymmdd') AS ds ,projectid ,SUM((download/1024/1024)*0.1166) AS download_fee FROM maxcomputefee WHERE type = 'DownloadEx' TO_CHAR(starttime,'yyyymmdd') >= '20190112' AND GROUP BY TO_CHAR(starttime, 'yyyymmdd') ,projectid ORDER BY download_fee DESC ; 5. 分析MapReduce作业消费。

② 说明 MapReduce作业当日计算费用 = 当日总计算时×单价 (0.0690 USD/Hour/Task)

--分析MapReduce作业消费。 SELECT TO_CHAR(starttime,'yyyymmdd') AS ds ,projectid ,(cu_usage/3600)*0.0690 AS mr_fee FROM maxcomputefee WHERE type = 'MapReduce' AND TO_CHAR(starttime,'yyyymmdd') >= '20190112' GROUP BY TO_CHAR(starttime,'yyyymmdd') ,projectid ,cu_usage ORDER BY mr_fee DESC ;

6. 分析外部表作业(OTS和OSS)。

⑦ 说明 一次SQL外部表计算费用 = 计算输入数据量×单价(0.0044 USD/GB)

--分析OTS外部表SQL作业消费。

SELECT TO_CHAR(starttime,'yyyymmdd') AS ds ,projectid , (computationsqlinput/1024/1024/1024)*1*0.03 AS ots_fee FROM maxcomputefee WHERE type = 'ComputationSqlOTS' AND TO_CHAR(starttime,'yyyymmdd') >= '20190112' GROUP BY TO_CHAR(starttime,'yyyymmdd') ,projectid ,computationsqlinput ORDER BY ots_fee DESC ; --分析OSS外部表SQL作业消费。 SELECT TO_CHAR(starttime,'yyyymmdd') AS ds ,projectid ,(computationsqlinput/1024/1024/1024)*1*0.03 AS oss_fee FROM maxcomputefee WHERE type = 'ComputationSqlOSS' AND TO_CHAR(starttime,'yyyymmdd') >= '20190112' GROUP BY TO_CHAR(starttime,'yyyymmdd') ,projectid ,computationsqlinput ORDER BY oss_fee DESC ;

7. 分析Spark计算费用。

⑦ 说明 Spark作业当日计算费用 = 当日总计算时×单价 (0.1041 USD/Hour/Task)

```
--分析Spark作业消费。
SELECT TO_CHAR(starttime,'yyyymmdd') AS ds
        ,projectid
        ,(cu_usage/3600)*0.1041 AS mr_fee
FROM maxcomputefee
WHERE type = 'spark'
AND TO_CHAR(starttime,'yyyymmdd') >= '20190112'
GROUP BY TO_CHAR(starttime,'yyyymmdd')
        ,projectid
        ,cu_usage
ORDER BY mr_fee DESC
;
```

7.安全管理 7.1. MaxCompute项目设置RAM子账号为超级管理员

本文为您介绍在MaxCompute项目中如何将RAM子账号设置为超级管理员,并提供了超级管理员在成员管理、权限管理等方面的使用建议。

背景信息

日常工作中,为了保障数据安全,通常主账号为特定人员管理,使用MaxCompute的大部分用户都只持有RAM子账号。但是项目的所有者(Owner)只 能为主账号,且MaxCompute的很多权限管理需要项目所有者才可以操作(例如项目级别Flag的设置、Package跨项目资源共享配置等),因此您需要 一个拥有超级管理员权限的RAM子账号。

MaxCompute新增了内置的管理角色Super_Administrator,拥有项目内所有类型资源的全部权限以及项目的管理类权限,具体权限说明请参见角色规划 与管理。

Super_Administrator角色可以由项目所有者授权给RAM子账号。RAM子账号获得该角色后,即可替代项目所有者对项目执行各种管理操作,包括常用 的项目级别Flag设置以及所有资源权限的管理。

设置方法

建议您为有权限创建项目的RAM子账号指派Super_Administrator角色,这样该账号在管理DataWorks工作空间的同时,也可以管理DataWorks工作空间对应的MaxCompute项目。

? 说明

- 为RAM子账号授予创建项目的权限方法请参见给RAM子账号授权DataWorks相关管理权限。
- 建议明确该RAM子账号持有人的责任,一个RAM子账号对应一个开发者,避免账号共用,以便更好地保障数据安全。
- 一个项目中只能为一个RAM子账号指派Super_Administrator角色。您可以为其他需要有基本管理权限的账号可以赋予Admin角色权限。

选定RAM子账号后,使用该RAM子账号创建项目。此时项目所有者依然是主账号,主账号可以通过以下方式将Super_Administrator角色授权给RAM子 账号:

● 通过MaxCompute客户端授权。

假设主账号用户bob@aliyun.com是项目project_a的所有者,Allen是bob@aliyun.com中的RAM子账号。

i. 使用主账号执行如下命令进行授权。

```
--打开项目project_a。
```

- use project_a;
- --为项目project_a添加RAM子账号Allen。
- add user ram\$bob@aliyun.com:Allen;
- --为RAM子账号Allen授权Super_Administrator角色权限。
- grant super_administrator TO ram\$bob@aliyun.com:Allen; --为RAM子账号Allen授权Admin角色权限。
- grant admin TO ram\$bob@aliyun.com:Allen;
- grant admin to ramspob@allyun.com:Allen;
- ii. 使用被授权RAM子账号执行如下命令查看账号自身权限。如果返回值中有Super_Administrator角色则说明赋权成功。

show grants;

● 通过DataWorks授权。

- i. 登录DataWorks, 进入工作空间配置。
- ii. (可选)添加RAM子账号为项目成员。如果RAM子账号已经为项目成员,可以忽略此步骤。
 - a. 在工作空间配置页面左侧导航栏上,单击成员管理,进入成员管理页面。
 - b. 单击右上角的**添加成员**。
 - c. 在添加成员页面,从待添加账号列表中选择需要添加的组织成员显示在已添加账号列表中。

⑦ 说明 单击添加成员对话框中的刷新,即可将RAM中当前阿里云主账号下存在的RAM子账号同步至待添加账号列表中。

d. 勾选角色并单击确定,完成成员添加。

- iii. 为RAM子账号授权Super_Administrator角色。
 - a. 在工作空间配置页面左侧导航栏上,单击MaxCompute高级配置。
 - b. 在MaxCompute高级配置页面的左侧导航栏上,单击自定义用户角色。

c. 单击需要授权角色后的成员管理,从待添加账号列表中选择需要添加的组织成员显示在已添加账号列表中。

基本设置	角色名称	角色类型	操作
自定义用户角色	admin	project	查看详情 成员管理
	delete_test	project	查看详情 成员管理 权限管理
	role123	project	查看详情 成员管理 权限管理
	role_project_admin	project	查看详情 成员管理 权限管理
	role_project_deploy	project	查看详情 成员管理 权限管理
	role_project_dev	project	查若详情 成员管理 权限管理
	role_project_guest	project	查看详情 成员管理 权限管理
	role_project_pe	project	查看详情 成员管理 权限管理
	role_project_scheduler	project	查若详情 成员管理 权限管理
	role_project_security	project	查看详情 成员管理 权限管理
	super_administrator	project	查翻详情 成员管理 权限管理

d. 单击确定,完成账号授权。

Ⅳ. 使用被授权RAM子账号执行如下命令查看账号自身权限。如果返回值中有Super_Administrator角色则说明赋权成功。

show grants;

使用说明

- 成员管理
 - MaxCompute支持云账号(主账号)和RAM子账号。为了更好的保障数据安全,建议项目中添加的成员均为此项目所有者的RAM子账号。
 主账号可以控制RAM子账号,在人员转岗离职等场景下主账号可以注销或更新对应的RAM子账号以保证数据安全。

⑦ 说明 如果通过DataWorks进行项目成员管理,只能添加项目所有者的RAM子账号作为项目成员。

- RAM子账号只能通过主账号添加。即使拥有Super_Administrator角色的超级管理员,也需要主账号先创建RAM子账号后,才可以将创建成功的 RAM子账号添加到项目中。
- 建议您只添加需要在当前项目进行数据开发(即会在当前项目执行作业)的用户为项目成员。对于有数据交互业务需求的用户,通过Package方式 进行跨项目资源共享,避免将过多用户添加至项目内而增加了成员管理的复杂度。
- 员工转岗或离职,需要先将其持有的RAM子账号在项目中移除,然后再通知项目所有者注销RAM子账号。如果是拥有Super_Administrator角色的 RAM子账号持有者转岗或离职,则需要由主账号执行移除、注销账号操作。
- 权限管理
 - 建议通过角色进行权限管理,即权限和角色(Role)关联,角色(Role)和用户(User)关联。
 - 。 建议实施最小够用原则,避免权限过大造成安全隐患。
 - 跨项目使用数据时,建议通过Package方式实现,避免资源提供方增加成员管理成本,只需要管理Package即可。

⑦ 说明 拥有Super_Administrator角色的RAM子账号本身已经拥有项目所有资源的查询和操作权限,所以无须再给自身授权。

• 权限审计

可以通过MaxCompute的元数据服务提供的视图进行权限审计。详情请参见元数据视图列表。

● 成本管理

关于成本管理,请参见查看账单详情。对于RAM子账号,需要主账号给RAM子账号赋予费用中心相关权限才可以进行账单数据的查询。授权方法请参 见为RAM角色授权,所需权限如下:

- AliyunBSSFullAccess: 管理费用中心的权限。
- 。 AliyunBSSReadOnlyAccess:费用中心的访问、只读权限。
- 。 AliyunBSSOrderAccess:费用中心查看订单、支付订单及取消订单的权限。

⑦ 说明 费用中心相关权限与MaxCompute项目的Super_Administrator角色无关联,需要单独授予给用户。

- 资源使用管理
 - 如果您使用MaxCompute包年包月计算资源,则可以通过MaxCompute管家对已经使用的计算资源进行查看、对所有计算资源进行管理。详情请 参见MaxCompute管家。

○ 如果您使用MaxCompute按量计费计算资源,则可以通过MaxCompute的元数据服务提供的相关视图对已经使用的计算资源进行查看。例如, TASKS_HISTORY可以查看详细的审计作业执行情况,包括时间、Job内容、资源消耗等信息,详情请参见TASKS_HISTORY。

⑦ 说明 元数据服务提供的视图只保留最近15天的数据。如果需要更长时间的数据,建议您定期自行读取数据并保存。

7.2. 基于Policy对具备内置角色的用户进行权限管理

用户被赋予MaxCompute内置的角色后,会具备内置角色相应的权限,例如用户被赋予开发角色则具备表、资源等的操作权限。但实际业务场景中, 需要对此类用户的操作权限进行更精细化的管理,例如不允许删除重要表。本文基于案例为您介绍如何通过Policy对具备内置角色权限的用户进行权限 管理。

前提条件

已安装MaxCompute客户端。更多安装MaxCompute客户端操作,请参见安装并配置MaxCompute客户端。

背景信息

当用户已经被赋予<mark>內置角色</mark>,且需要对用户的权限进行更精细化管理时,推荐您使用Policy权限管控机制,来解决ACL权限管控机制无法精细化管理此类 权限的问题。

Policy权限管控机制是一种基于角色的权限管理方式,允许或禁止角色操作(例如读、写)项目中的对象(例如表),将角色赋予用户后,即可管控用 户权限。更多Policy授权或撤销授权语法信息,请参见Policy权限管控。

Policy授权

假设RAM用户Alice已具备MaxCompute项目的开发角色,现需要禁止Alice删除以tb_开头的所有表。Alice账号名为Alice,所属阿里云账号为 Bob@aliyun.com。

Policy授权操作需要由项目所有者、具备Super_Administrator或Admin角色的用户执行。Policy授权操作流程如下:

- 1. 登录MaxCompute客户端。
- 2. 执行 create role 命令创建角色delete_test。
 - 命令示例如下。

create role delete_test;

更多创建角色信息,请参见角色规划与管理。

3. 基于Policy权限管控机制,执行 grant 命令为角色delete_test授权,禁止删除以tb_开头的所有表。

命令示例如下。

grant drop on table tb_* to role delete_test privilegeproperties("policy" = "true", "allow"="false");

更多Policy授权语法格式信息,请参见Policy授权(Grant)。

4. 执行 grant 命令将delete_test角色赋予Alice。

命令示例如下。

grant delete_test to ram\$bob@aliyun.com:Alice;

如果不清楚账号名称,可以在MaxCompute客户端执行 list users; 命令获取。更多将角色赋予用户信息,请参见角色规划与管理。

5. 执行 show grants 命令查看Alice的权限信息。

命令示例如下。

show grants for ram\$bob@aliyun.com:Alice;

返回结果如下。

[roles]				
role_project_admin, delete_test		Alice 已 被赋予delete_test角色。 授权方式为Policy。		
Authorization Type: Policy				
[role/delete_test]				
D	projects/mcproject_name/tables/tb_*: Drop	不允许	(D表示Deny)	删除项目中以tb_开头的所有表。
[role/role_project_admin]				
A	projects/mcproject_name: *			
A	<pre>projects/mcproject_name/instances/*: *</pre>			
A	<pre>projects/mcproject_name/jobs/*: *</pre>			
A	<pre>projects/mcproject_name/offlinemodels/*: *</pre>			
A	<pre>projects/mcproject_name/packages/*: *</pre>			
A	<pre>projects/mcproject_name/registration/functions/*: *</pre>			
A	<pre>projects/mcproject_name/resources/*: *</pre>			
A	<pre>projects/mcproject_name/tables/*: *</pre>			
A	<pre>projects/mcproject_name/volumes/*: *</pre>			
Authori	zation Type: ObjectCreator			
AG	projects/mcproject_name/tables/local_test: All			
AG	<pre>projects/mcproject_name/tables/mr_multiinout_out1: All</pre>			
AG	<pre>projects/mcproject_name/tables/mr_multiinout_out2: All</pre>			
AG	projects/mcproject_name/tables/ramtest: All			
AG	projects/mcproject_name/tables/wc_in: All			
AG	projects/mcproject_name/tables/wc_in1: All			
AG	projects/mcproject_name/tables/wc_in2: All			
AG	projects/mcproject name/tables/wc out: All			

更多查看用户权限信息,请参见通过MaxCompute SQL查询权限信息。

6. 以Alice身份登录MaxCompute客户端,执行 drop table 命令尝试删除以tb_开头的表。

命令示例如下。

drop table tb_test;

返回结果如下。表明权限控制生效。如果删除成功,表明权限控制不生效,请仔细确认是否已经正确执行上述步骤。

FAILED: Catalog Service Failed, ErrorCode: 50, Error Message: ODPS-0130013:Authorization exception - Authorization Failed [4
011],
You have NO privilege 'odps:Drop' on {acs:odps:*:projects/mcproject_name/tables/tb_test}.
Explicitly denied by policy.

Context ID:85efa8e9-40da-4660-bbfd-b503dfa64c0a. --->Tips: Pricipal:RAM\$bob@aliyun.com:Alice; Deny by policy

撤销Policy授权

基于Policy授权中的案例,假设以tb_开头的所有表在实际业务中已不再需要,允许Alice执行删除操作。

撤销Policy授权操作需要由项目所有者、具备Super_Administrator或Admin角色的用户执行。撤销Policy授权方式如下:

• 撤销为角色授予的权限,保留角色

操作流程如下:

i. 登录MaxCompute客户端。

ii. 基于Policy权限管控机制,执行 revoke 命令撤销Policy授权,允许delete_test角色删除以tb_开头的所有表。

命令示例如下。

revoke drop on table tb_* from role delete_test privilegeproperties("policy" = "true", "allow"="false");

更多撤销Policy授权语法格式信息,请参见撤销Policy授权(Revoke)。

iii. 执行 show grants 命令查看Alice的权限信息。命令示例如下。

show grants for ram\$bob@aliyun.com:Alice;

返回结果如下。

[roles] --保留delete test角色。 role_project_admin, delete_test ___Policy**授权信息已撤销。** Authorization Type: Policy [role/role_project_admin] projects/mcproject_name: *
projects/mcproject_name/instances/*: * А A projects/mcproject_name/instances/*: *
projects/mcproject_name/jobs/*: *
projects/mcproject_name/offlinemodels/*: *
projects/mcproject_name/packages/*: *
projects/mcproject_name/registration/functions/*: *
projects/mcproject_name/resources/*: * A А А A А A projects/mcproject_name/tables/*: * projects/mcproject_name/volumes/*: * A Authorization Type: ObjectCreator AG projects/mcproject_name/tables/local_test: All AG projects/mcproject_name/tables/mr_multiinout_out1: All AG projects/mcproject_name/tables/mr_multiinout_out2: All projects/mcproject_name/tables/ramtest: All
projects/mcproject_name/tables/tb_test: All AG AG AG projects/mcproject_name/tables/wc_in: All AG projects/mcproject_name/tables/wc_in1: All AG projects/mcproject_name/tables/wc_in2: All

更多查看用户权限信息,请参见通过MaxCompute SQL查询权限信息。

AG projects/mcproject_name/tables/wc_out: All

iv. 以Alice身份登录MaxCompute客户端,执行 drop table 命令尝试删除以tb_开头的表。

命令示例如下。

drop table tb_test;

- 返回OK, 表明撤销Policy授权成功。
- 收回赋予用户的角色,如果不再需要角色,可删除角色

操作流程如下:

- i. 登录MaxCompute客户端。
- ii. 执行 revoke 命令收回授予Alice的delete_test角色。
 - 命令示例如下。

revoke delete_test from ram\$bob@aliyun.com:Alice;

更多收回赋予用户的角色信息,请参见角色规划与管理。

iii. 执行 show grants 命令查看Alice的权限信息。命令示例如下。

show grants for ram\$bob@aliyun.com:Alice;

返回结果如下。

pject_admin	已删除 delete_test 角色。
zation Type: Policy	
ple_project_admin]	
projects/mcproject_name: *	
<pre>projects/mcproject_name/instances/*: *</pre>	
<pre>projects/mcproject_name/jobs/*: *</pre>	
<pre>projects/mcproject_name/offlinemodels/*: *</pre>	
<pre>projects/mcproject_name/packages/*: *</pre>	
<pre>projects/mcproject_name/registration/functions/*: *</pre>	
<pre>projects/mcproject_name/resources/*: *</pre>	
<pre>projects/mcproject_name/tables/*: *</pre>	
projects/mcproject_name/volumes/*: *	
zation Type: ObjectCreator	
projects/mcproject_name/tables/local_test: All	
<pre>projects/mcproject_name/tables/mr_multiinout_out1: A</pre>	11
projects/mcproject_name/tables/mr_multiinout_out2: A	11
projects/mcproject_name/tables/ramtest: All	
projects/mcproject_name/tables/wc_in: All	
projects/mcproject_name/tables/wc_in1: All	
projects/mcproject_name/tables/wc_in2: All	
projects/mcproject_name/tables/wc_out: All	
	<pre>bject_admin zation Type: Policy ble_project_admin] projects/mcproject_name: * projects/mcproject_name/instances/*: * projects/mcproject_name/offlinemodels/*: * projects/mcproject_name/registration/functions/*: * projects/mcproject_name/registration/functions/*: * projects/mcproject_name/resources/*: * projects/mcproject_name/tables/*: * projects/mcproject_name/tables/*: * zation Type: ObjectCreator projects/mcproject_name/tables/mr_multiinout_out1: All projects/mcproject_name/tables/ramtest: All projects/mcproject_name/tables/rc_in2: All projects/mcpro</pre>

iv. 以Alice身份登录MaxCompute客户端,执行 drop table 命令尝试删除以tb_开头的表。

命令示例如下。

drop table tb_test;

返回OK,表明撤销Policy授权成功。

v. (可选)执行 drop role 命令删除delete_test角色。

命令示例如下。

drop role delete_test;

返回OK,表明角色删除成功。更多删除角色信息,请参见角色规划与管理。