阿里云 表格存储

计算与分析

文档版本: 20190805

为了无法计算的价值 | []阿里云

<u>法律声明</u>

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。
- 3. 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
•	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。	禁止: 重置操作将丢失用户配置数据。
A	该类警示信息可能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,恢复业务所需 时间约10分钟。
	用于补充说明、最佳实践、窍门等,不 是用户必须了解的内容。	道 说明: 您也可以通过按Ctrl + A选中全部文件。
>	多级菜单递进。	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。	单击 确定。
courier 字体	命令。	执行 cd /d C:/windows 命令,进 入Windows系统文件夹。
##	表示参数、变量。	bae log listinstanceid Instance_ID
[]或者[a b]	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。	<pre>swich {stand slave}</pre>

目录

法律声明	I
通用约定	I
1 函数触发器	1
1.1 使用函数计算	1
1.2 使用函数计算做数据清洗	11
2 数据展现	22
3 MaxCompute	27
- 3.1 使用MaxCompute访问表格存储	27
3.2 跨账号授权	33
3.3 使用AccessKey访问表格存储	37
3.4 使用 UDF 处理数据	38
3.5 常见问题	40
4 Data Lake Analytics	41
4.1 背景信息	41
4.2 准备工作	
4.3 使用DLA服务	
5 Hive/HadoopMR	52
5.1 环境准备	52
5.2 使用教程	54
6 Spark/SparkSQL	
6.1 环境准备	59
6.2 使用教程	60

1函数触发器

1.1 使用函数计算

函数计算(Function Compute)是一个事件驱动的服务,通过函数计算,用户无需管理服务器等运行情况,只需编写代码并上传。

函数计算准备计算资源,并以弹性伸缩的方式运行用户代码,而用户只需根据实际代码运行所消耗 的资源进行付费。详细信息请参见函数计算帮助文档。

函数计算示例请参见表格存储函数计算示例。

Table Store Stream是用于获取Table Store表中增量数据的一个数据通道,通过创建Table Store触发器,能够实现Table Store Stream和函数计算的自动对接,让计算函数中自定义的程序 逻辑自动处理Table Store表中发生的数据修改。详细信息请参见Table Store Stream。

本节介绍如何使用函数计算对表格存储增量数据进行实时计算。



使用场景:

·数据同步:同步表格存储中的实时数据到数据缓存、搜索引擎或者其他数据库实例中。

·数据归档:增量归档表格存储中的数据到 OSS 等做冷备份。

・事件驱动:利用触发器触发函数调用 IOT 套件、云应用 API 或者做消息通知等。

触发器详细信息请参见Table Store 触发器。

配置Table Store触发器

您可以通过控制台创建Table Store触发器来处理Table Store数据表的实时数据流。

- 1. 创建开通了Stream流的数据表。
 - a. 在Table Store控制台创建实例。

创建实例		\times
* 实例名称:	testInstance	
* 实例规格:	容量型实例 ▼	
* 实例注释:	对接函数计算的数据实例	
提示: 1.创建实例需要几 2.创建实例后,实例	,秒钟的时间,创建成功请按刷新按钮刷新实例列表. 则域名会在1分钟内生效.	



b. 在该实例下创建数据表,并且开启数据表的Stream功能。

	创建数据表	×
🛧 testInstance		
实例访问地址	■ 数据表名称:	streamDataTable
公网: http://testInstance.cn-shanghai.ots.aliyuncs.com	* 数据生命周期:	-1 秒
版网: http://testInstance.cn-shanghai.ots-internal.aliyuncs.com		注:数据生命周期最低为86400秒(一天)或者-1(永不过期)
实例网络类型 更改	*最大数据版本:	1
允许任意网络访问 💿		注:数据版本需要为非0值
VPC列表	* 数据有效版本偏差:	86400 眇
		注:写人数据所有列的版本号与写入时时间的整值需要在数据有效版本偏差范围之内,否者将会写 入失败
	* 表主键:	
 数据表列表		id 字符串 ▼ 分片鍵
		注:表主键最多4个,您已创建1个
		注:为了使得数据在表上分布均均,避免读写热点问题,充分利用预留读写吞吐量,我们建议您 使用哈希或者类似方式使得分片键的值均匀分布,详见最佳实践。
		+ 添加表主鍵
	✔ 开启Stream功能	注:Stream资费详见价格总宽
	* Stream记录过期时长:	12
		确 定 取消

2. 创建函数计算的函数。

a. 在函数计算控制台创建服务(Service)。

新建服务		\times
服务名称:	testService 命名规范: » 1. 只能包含字母,数字、下划线和中划线 » 2. 不能以数字、中划线开头 » 3. 长度限制在1-128之间	
所属区域:	华东 2 🔻 相同区域内的产品内网可以互通 ; 订购后不支持更换区域 , 请谨慎选择	
功能描述:	用户处理table store数据表中增量数据的函数集	

- 提交 取消
- b. 在Service的高级配置中,您可以配置服务的角色,用于授权函数进行日志收集,以及在计算 函数中继续访问用户的其他资源。详细信息请参见函数计算权限模型。

	帮助文档 服务实时监控
基本信息	修改 🗸
服务名称: testService	所在区域: 华东 2
创建时间: 2017-09-26 16:20:57	修改时间: 2017-09-26 16:21:11
描述信息: 用户处理table store数据表中增量数据的函数集	
高级配置	取消 保存 イ
LogProject:	LogStore:
角色: 已经存在的角色 🔻 🞯	现有角色: 🔹 🗸 🥥

- c. 在新创建的Service下单击新建函数。
- d. 在函数模板页面选择 空白函数。
- e. 在触发器配置页面选择不创建任何触发器, 然后单击下一步。
- f. 配置函数信息。

Table Store触发器会使用CBOR格式将增量数据编码为函数计算的Event,并调用用户函数。下图示例函数就是将编码后的Event重新解码和打印到日志中心,您可以在解码数据后进行任意数据处理。

函数模版	>	触发器配置		基础管理配置		信息	岐対
基础信息	* 函数名称:	streamProcessFu 命名规范: » 1. 只能包含字母 和中划线 » 2. 不能以数字. » 3. 长度限制在1-	nction ,数字、下划 中划线开头 128之间				
	函数描述:	用于处理tablesto	18 数据表增量	数据的函数			
	* 运行环境:	python2.7 🔻					
代码配置	* 代码上传方 式:	在线编辑(DSS <u>上</u> 传	本地上传			
	4 import 5 import 6 7 def har 8 logge 9 recor 10 logge 11 retur	cbor logging mdler(event, con rr = logging.ge ds = cbor.load r.info('Tables n 'ok'	ntext): tLogger() s(event) tore strea	um records sample	e %s', rec	ords)	
环境配置	* 函数入口: * 函数内存:	index.handler 格式为"[文件名].[i 定了函数的调用入 数。如果是代码直 hello_world.js,只 128 ▼ MB	函数名]"。例如 口为hello_wor 接上传,代码 需关注函数名	Ihello_world.handler指 ld.js文件中的handler函 将保存为对应的文件名 填写。			
	* 超时时间:	3	秒				

3. 创建和测试Table Store触发器。

- a. 在Table Store控制台新创建的数据表下,选择使用已有函数计算来创建触发器。
- b. 创建过程中需要授权Table Store发送事件通知所需的权限。

勾选完毕后,实际可以在RAM控制台查看到自动创建的授权角 色AliyunTableStoreStreamNotificationRole。

创建触发器(使用现有函数)



数据处理

・数据格式

Table Store触发器使用CBOR格式对增量数据进行编码构成函数计算的Event。增量数据的具体数据格式如下:

确

"ColumnName": "string", "Value": formated_value], "Columns": [{ "Type": "string", "ColumnName": "string", "Value": formated_value, "Timestamp": int64 }] }

	计算与分析 /
}	
成员定义	
- Version	
■ 今以·navload版本是 日前为Sync v1	
■ 各文: payloau/成本与, 日前分5yne-v1	
- Records	
■ 含义: 数据表中的增量数据行数组	
■ 内部成员:	
■ Туре	
■ 含义: 数据行类型,包含PutRow、UpdateRow和Delete	Row
■ 类型: string	
■ Info	
■ 含义: 数据行基本信息	
■ 内部成员:	
■ Timestamp	
■ 含义: 该行的最后修改UTC时间	
■ 类型: int64	
PrimaryKey	
■ 含义: 主键列数组	
■ ColumnName	
 ■ 含义: 主键列名称 ■ ***** 	
■	
■ value	
■ 古文, 土硬列内存 ■ 类型: formated value, 支持integer, string和b	lob
Columns	~
■ 含义: 属性列数组	
■ 内部成员	
■ Туре	

■ 含义: 属性列类型, 包含Put、DeleteOneVersion和DeleteAllVersions

■ 类型: string

- ColumnName
 - 含义: 属性列名称
 - 类型: string
- Value
 - 含义: 属性列内容
 - 类型: formated_value,支持integer、boolean、double、string和 blob
- Timestamp
 - 含义: 属性列最后修改UTC时间
 - 类型: int64

```
    数据示例
```

```
{
    "Version": "Sync-v1",
    "Records": [
        {
             "Type": "PutRow",
             "Info": {
                  "Timestamp": 1506416585740836
             },
"PrimaryKey": [
                  {
                      "ColumnName": "pk_0",
                      "Value": 1506416585881590900
                  },
                  {
                      "ColumnName": "pk_1"
                      "Value": "2017-09-26 17:03:05.8815909 +0800 CST"
                  },
                  {
                      "ColumnName": "pk_2",
                      "Value": 1506416585741000
                  }
             ],
"Columns": [
                  {
                      "Type": "Put",
"ColumnName": "attr_0",
                      "Value": "hello_table_store",
                      "Timestamp": 1506416585741
                 },
{
                      "Type": "Put",
"ColumnName": "attr_1",
                      "Value": 1506416585881590900,
                      "Timestamp": 1506416585741
                 }
             ٦
```

]

}

在线调试

函数计算支持函数的在线调试功能,用户能够自己构建触发的Event,并测试代码逻辑是否符合期 望。

由于Table Store触发函数服务的Event是CBOR格式,该数据格式是一种类似JSON的二进制格式,可以按照如下方法进行在线调试:

- 1. 在代码中同时import cbor和json。
- 2. 单击触发事件,选择自定义,将上述数据示例的json文件粘贴至编辑框,根据实际情况修改,并 保存。
- 3. 代码中先使用 records = json.loads(event) 来处自定义的测试触发事件,单击执行,查 看是否符合期望。
- 当使用 records=json.loads(event) 测试OK之后,可以修改为 records = cbor.
 loads(event) 并保存,当Table Store中有数据写入时,相关的函数逻辑就会触发。



示例代码:

```
import logging
import cbor
import json
def get_attrbute_value(record, column):
    attrs = record[u'Columns']
    for x in attrs:
        if x[u'ColumnName'] == column:
            return x['Value']
def get_pk_value(record, column):
    attrs = record[u'PrimaryKey']
    for x in attrs:
        if x['ColumnName'] == column:
            return x['Value']
```

```
def handler(event, context):
    logger = logging.getLogger()
    logger.info("Begin to handle event")
    #records = cbor.loads(event)
    records = json.loads(event)
    for record in records['Records']:
        logger.info("Handle record: %s", record)
        pk_0 = get_pk_value(record, "pk_0")
        attr_0 = get_attrbute_value(record, "attr_0")
    return '0K'
```

1.2 使用函数计算做数据清洗

本文主要为您介绍函数计算对表格存储中的数据进行简单清洗的场景。

表格存储高并发的写入性能以及低廉的存储成本非常适合物联网、日志、监控数据的存储,我们可 以将数据写入到表格存储中,同时在函数计算中对新增的数据做简单的清洗,并将清洗之后的数据 写回到表格存储的结果表中,并对原始数据及结果数据提供实时访问。

数据定义

我们假设写入的为日志数据,包括三个基础字段:

字段名称	类型	含义
id	整型	日志id
level	整型	日志的等级,越大表明等级越 高
message	字符串	日志的内容

我们需要将 level>1 的日志写入到另外一张数据表中,用作专门的查询。

创建实例及数据表

在表格存储控制台创建表格存储实例(本次以 华东2 distribute-test 为例),并创建源

表(source_data)及结果表(result), 主键为 id (整型), 由于表格存储是 schemafree 结

构,无需预先定义其他属性列字段。

以 source_data 为例, 创建如下图:

创建数据表	×
* 数据表名称:	source_data
*数据生命周期:	-1 秒
	注:数据生命周期最低为86400秒(一天)或者-1(永不过期)
*最大数据版本:	1
	注:数据版本需要为非0值
* 数据有效版本偏差:	86400 秒
	注:写入数据所有列的版本号与写入时时间的差值需要在数据有效版本偏差范围之内,否者将会写 入失败
*表主键:	
	id 整型 🛟 分片键
	注:表主键最多4个,您已创建1个
	注:为了使得数据在表上分布均匀,避免读写热点问题,充分利用预留读写吞吐量,我们建议您 使用哈希或者类似方式使得分片键的值均匀分布,详见 <mark>最佳实践。</mark>
	十 添加表主键

开启数据源表的Stream功能

触发器功能需要先开启数据表的Stream功能,才能在函数计算中处理写入表格存储中的增量数据。

数据表列表					
数据表名称 🕏	文本		搜索		
数据表名称	数据生命周期	最大数据版本	数据有效版本偏差	Stream状态	操作
source_data	-1	1	86400	关闭	数据管理 开启Stream 使用触发器 调整生命周期与最大版本 删除

Stream记录过期时长是指通过 StreamAPI 能够读取到的增量数据的最长时间。

由于触发器只能绑定现有的函数,故先到函数计算的控制台上在同region创建服务及函数。

创建函数计算服务

在函数计算的控制台上创建服务及处理函数,我们继续使用华东2节点。

1. 在华东2节点创建服务。

新建服务

 \times

* 服务名称	transform_test
	1. 只能包含字母、数字、下划线和中划线 2. 不能以数字、中划线开头 3. 长度限制在1-128之间
所属区域	华东 2(上海)
	相同区域内的产品内网可以互通,创建服务后无法更换区域, 请谨慎选择。
功能描述	使用函数计算对表格存储的数据做简单的清洗
高级配置	

2. 创建函数依次选择空白函数 > 不创建触发器。

* 所在服务	transform_test ~	新建服务
* 函数名称	etl_test	
	1. 只能包含字母、数字、下划线和中划线 2. 不能以数字、中划线开头 3. 长度限制在1-128之间	
描述信息	对表格存储中的数据做简单的清洗	
* 运行环境	python2.7 \checkmark]



* 函数入口	etl_test.handler
	"Handler"的格式: 那么文件名为inde
* 函数执行内存	256MB
* 超时时间	30

- · 函数名称为: etl_test, 选择 python2.7 环境, 在线编辑代码
- · 函数入口为: etl_test.handler
- ・代码稍后编辑,点击下一步。
- 3. 进行服务授权

由于函数计算需要将运行中的日志写入到日志服务中,同时,需要对表格存储的表进行读写,故 需要对函数计算进行授权,为方便起见,我们先添加 AliyunOTSFullAccess 与 AliyunLogF ullAccess 权限,实际生产中,建议根据权限最小原则来添加权限。

系统角色授权	5
系统模版授权 AliyunOTSFullAccess × AliyunLogFullAccess × ✓	

- 4. 单击授权完成,并创建函数。
- 5. 修改函数代码。

创建好函数之后,点击对应的函数一代码执行,编辑代码并保存,其

中, INSTANCE_NAME(表格存储的实例名称)、REGION(使用的区域)需要根据情况进行修改:

<	华东 2(上海) > transform_test > etl_test					
服务概览	概览 代码执行 触发器					
函数列表 + 27	代码执行管理					
• etl_test	执行 触发事件 ⑦					
< 1/1 >	 ● 在线编辑 ○ OSS上传 ○ 代码包上传 ○ 文件夹上传 					
	1					
	<pre>7 INSTANCE_NAME = 'distribute-test' 8 REGION = 'cn-shanghai' 9 ENDPOINT = 'http://%s.%s.ots-internal.aliyuncs.com'%(INSTANCE_NAME, REGION) 10 RESULT_TABLENAME = 'result' 11</pre>					

使用示例代码如下:

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import cbor
import json
import tablestore as ots
INSTANCE_NAME = 'distribute-test'
REGION = 'cn-shanghai'
ENDPOINT = 'http://%s.%s.ots-internal.aliyuncs.com'%(INSTANCE_NAME,
REGION)
RESULT_TABLENAME = 'result'
def _utf8(input):
    return str(bytearray(input, "utf-8"))
def get_attrbute_value(record, column):
    attrs = record[u'Columns']
    for x in attrs:
        if x[u'ColumnName'] == column:
            return x['Value']
def get_pk_value(record, column):
    attrs = record[u'PrimaryKey']
    for x in attrs:
        if x['ColumnName'] == column:
            return x['Value']
#由于已经授权了AliyunOTSFullAccess权限,此处获取的credentials具有访问表格存
储的权限
def get_ots_client(context):
    creds = context.credentials
    client = ots.OTSClient(ENDPOINT, creds.accessKeyId, creds.
accessKeySecret, INSTANCE_NAME, sts_token = creds.securityToken)
    return client
def save_to_ots(client, record):
    id = int(get_pk_value(record, 'id'))
```

```
level = int(get_attrbute_value(record, 'level'))
msg = get_attrbute_value(record, 'message')
pk = [(_utf8('id'), id),]
attr = [(_utf8('level'), level), (_utf8('message'), _utf8(msg
)),]
row = ots.Row(pk, attr)
client.put_row(RESULT_TABLENAME, row)
def handler(event, context):
records = cbor.loads(event)
#records = json.loads(event)
client = get_ots_client(context)
for record in records['Records']:
    level = int(get_attrbute_value(record, 'level'))
    if level > 1:
        save_to_ots(client, record)
    else:
```

print "Level <= 1, ignore."</pre>

绑定触发器

 回到表格存储的实例管理页面,单击表 source_data 后的使用触发器按钮,进入触发器绑定界 面,单击使用已有函数计算,选择刚创建的服务及函数,勾选表格存储发送事件通知的权限, 进行确定。

数据表列表					
数据表名称 💲	文本		搜索		
数据表名称	数据生命周期	最大数据版本	数据有效版本偏差	Stream状态	操作
result	-1	1	86400	关闭	数据管理 开启Stream 使用触发器 调整生命周期与最大版本 删除
source_data	-1	1	86400	开启	数据管理 关闭Stream 使用触发器 调整生命周期与最大版本 删除



2. 绑定成功之后,能够看到如下的信息:

transform_test	etl_test	tablestore_etl	编辑/测试 删除
函数计算服务名	函数计算函数名	触发器名称	操作
目前一张数据表只支持创建一个触发器			
触发器列表			
创建触发器 新建函数计算 使用已有函数计算			〇 刷新
source_data			

运行验证

1. 向 source_data 表中写入数据。

单击source_data的数据管理页面,然后单击插入数据,如图依次填入id、level及message信息。

1		插入数据					
	source_data						
基本详情	表格数据	主键名称	主键类型	主键值			插入数据
数据管理	●表格数据最多显示50行	id	INTEGER	1			
■友器官理▶ 监控指标		× 移除所有属性列	J				
		属性列名称	属性列类型	属性值	数据版本号	操作	
		level	INTEGER \$	2	或 🔽 使用系统时间	■删除	
		message	STRING \$	Test data	或 🗹 使用系统时间	茴 删除	
		+ 增加属性列 🏾 🎗	主: 属性列最多只能自	1定义插入20列,您已经创]建 <mark>2</mark> 列		
					确定插入	取消	

2. 在 result 表中查询清洗后的数据

单击result表的数据管理页面, 会查询到刚写入到 source_data 中的数据。

向 soure_data 写入level <=1的数据将不会同步到 result 表中。

<	春 result						
基本详情	表格数据			插入数据	查询数据	更新数据	删除数据
数据管理 触发器管理	● 表格数据最多	·显示50行。					
▶ 监控指标	0	id(主键)	level		messa	ge	
		1	2		Test da	ata	
	0		版本 值 类型	共有1条,名	要页显示:10条	ec c	1 > »
			1525319981646 2 INTEGER				

2 数据展现

本文主要为您介绍如何使用DataV将表格存储的数据可视化。

表格存储的表数据支持接入 DataV 数据可视化。DataV 可以将数据由单一的数字转化为各种动态的可视化图表,根据表数据生成数据看板,实时地将数据展示给需要的用户。

操作步骤

- 1. 创建并保存 AccessKey。
- 2. 获取表格存储实例的公网访问地址。
 - a. 登录表格存储控制台。
 - b. 单击实例名称,进入实例详情页面。
 - c. 保存该实例的公网访问地址。

<	
实例详情	实例访问地址
	公网: http:// .cn-hangzhou.ots.aliyuncs.com 私网: http:// cn-hangzhou.ots-internal.aliyuncs.com
	实例网络 <u>类型</u> 更改

3. 在 DataV 控制台配置表格存储数据源。

- a. 登录 DataV 控制台。
- b. 选择我的数据 > 添加数据 。

7			企业版⑦			V	~	帮助~
	我的可视化	公告	rome 浏览器版本 56 以上。	2. 用户使用中如遇到问题,	推荐直接提交工单。	也可前往钉钉群提问,	钉钉群号	: 1172093
	我的数据	2	+添加数据					
6	优秀案例&教程		名称	数据类型	修改时间			操作
			(Contract)	1.00x Harris		10	编辑	删除
			() and	10.000			编辑	删除

c. 添加表格存储数据源。

新建数据	×	
类型*	TableStore -	
名称*	test manufacture and and	
AK ID*	21402017644060	
AK Secret*	•••••	
公网地址*	http://.ots.aliyuncs.com	
0	测试连接 ① 连接成功 请确保数据库可以被公网访问(不支持IP白名单,详见:数程 请确保数据库没有被防火墙禁止 请确保数据库域名能够被解析 请确保数据库已经启动)
	完成	

- 4. 创建可视化应用。
 - a. 选择我的可视化 > 新建可视化。

	✔ ♥ 企业版 ⑦	
副 我的可视化	公告: 1. DataV 仅支持谷歌 Chrome 浏览器版	瓦本 56 以上。2. 用户使用中如遇到问题,
日 我的数据	我的可视化 8个还可创建12个 ??	
⑦ 优秀案例&教程	2 — 一 一 新建可视化	

b. 选择空白模板或现有模板来创建大屏。

您可以在大屏上随意添加、拖曳、删除组件,并设置组件的样式和数据。组件的详细设置请 参见 DataV 组件。

5. 在组件中使用表格存储数据源。



目前支持两种查询: getRow 和 getRange。查询的 JSON 参数如下所示:

```
// getRow
{
    "table_name": "test", //表格名
    "rows": { //表字段名, 用作过滤条件
    "id": 2
},
    "columns": ["test"] // 需要查询的 column 的名称
}
// getRange
{
    "table_name": "test", //表格名
    "direction": "FORWARD", //可以是 FORWARD或者 BACKWARD
    "columns": ["id", "test"], // 需要查询的 column 的名称
    "range": { // range参数定义查询的访问
    "limit": 4, //limit字段设置查询的记录数的限制
    "start": { //start 设置查询的起点
        "id": 1
    },
    "end": { //end 设置查询的截止
    "table_name": "test", //end 设置查询的截止
    "table_name": "test", //end 设置查询的表达
```

"id": 3 } } //注意查询的访问包含 start 的记录,不包含 end 的记录,比如上面的查询会查出id 为1、2的记录,不包含id 为3的记录

3 MaxCompute

3.1 使用MaxCompute访问表格存储

本文主要为您介绍如何在同一个云账号下实现表格存储和 MaxCompute 之间的无缝连接。

背景信息

MaxCompute是一项大数据计算服务,它能提供快速、完全托管的 PB 级数据仓库解决方 案,使您可以经济并高效地分析处理海量数据。您只需通过一条简单的 DDL 语句,即可在 MaxCompute 上创建一张外部表,建立 MaxCompute 表与外部数据源的关联,提供各种数据的 接入和输出能力。MaxCompute 表是结构化的数据,而外部表可以不限于结构化数据。

表格存储与 MaxCompute 都有其自身的类型系统,两者之间的类型对应关系如下表所示。

Table Store	MaxCompute
STRING	STRING
INTEGER	BIGINT
DOUBLE	DOUBLE
BOOLEAN	BOOLEAN
BINARY	BINARY

准备工作

使用 MaxCompute 访问表格存储前, 您需要完成以下准备工作:

- 1. 开通 MaxCompute 服务。
- 2. 创建 创建工作空间。
- 3. 创建 AccessKey。

4. 在 RAM 控制台授权 MaxCompute 访问表格存储的权限。

- ·方法一:登录阿里云账号单击此处完成一键授权。
- ・方法二:进行手动授权。步骤如下:
 - a. 登录 RAM 控制台。
 - b. 在角色管理页面,创建用户角色 AliyunODPSDefaultRole。

访问控制 RAM	角色管理	2 新建角色 ♀ 刷新
概览	角色名 ▼ 请输入角色名进行模	期查询 搜索
用戶管理	角色名称	创建时间 操作
策略管理	AliyunBastionHostDefaultRole	2017-08-21 17:45:59 管理 授权 删除
角色管理	AliyunCloudMonitorDefaultRole	2017-06-14 19:54:11 管理 授权 删除
设置	AliyunCodePipelineDefaultRole	2017-08-23 15:47:20 管理 授权 删除

c. 在角色详情页面,设置策略内容。

<	AliyunODPSDefaultRole	
角色详情		2
<u> </u>	基本信息	编辑基本信息
用巴拉仪來酯	角色名称 AliyunODPSDefaultRole	备注
	创建时间 2017-09-18 14:53:39	Arn a ino

策略内容如下:

```
{
"Statement": [
{
"Action": "sts:AssumeRole",
"Effect": "Allow",
"Principal": {
   "Service": [
      "odps.aliyuncs.com"
  ]
}
],
"Version": "1"
```

}

d. 在策略管理页面,新建授权策略 AliyunODPSRolePolicy。

管理控制台	产品与	服务▼	Q 搜索	图 消息 416 费用 工单	支持		简体中文
访问控制 RAM		创建授权策略 STEP 1:选择权限策略模板	STEP 2 : 编辑	報限 井提 交 STEP 3 : 新	X 建成功	2 新建授权策略	€刷新
概览		全部模板 ▼ 请输入关键词在下方模板•	中动态筛选				
用户管理		3 空白模板	>	系统 AdministratorAccess 管理所有阿里云资源的权限	>		
策略管理 角色管理		 AliyunOSSFullAccess 管理对象存储服务(OSS)权限 	>	系统 AliyunOSSReadOnlyAccess 只读访问对象存储服务(OSS)的权限	>	被引用次数	操作
设置	ų.	 AliyunECSFullAccess 管理云服务器服务(ECS)的权限 	>	系统 AliyunECSReadOnlyAccess 只读访问云服务器服务(ECS)的权限	>	19 4	查看
		 AliyunRDSFullAccess 管理云数据库服务(RDS)的权限 	>	AliyunRDSReadOnlyAccess 只读访问云数据库服务(RDS)的权限	>	2	查看
						1	查看

策略内容如下:

```
{
   "Version": "1",
   "Statement": [
        "Action": [
            "ots:ListTable",
            "ots:DescribeTable",
            "ots:GetRow",
            "ots:UpdateRow",
            "ots:DeleteRow",
            "ots:BatchGetRow",
            "ots:BatchWriteRow",
            "ots:ComputeSplitPointsBySize"
        ],
        "Resource": "*",
        "Effect": "Allow"
    }
}
```

--还可自定义其他权限

e. 在角色管理页面,将 AliyunODPSRolePolicy 权限授权给 AliyunODPSDefaultRole 角色。

访问控制 RAM	-	2017-06-21 16:06:06	管理 授权 删除
概览	Automotive to the test	2017-07-20 16:42:24	管理 授权 删除
用户管理	0	2017-05-25 17:52:47	管理 授权 删除
群组管理	durrent date	2017-07-26 14:57:50	管理 授权 删除
策略管理 1	durante da la la	2017-03-14 17:42:08	管理 授权 删除
角色管理	AliyunODPSDefaultRole	2017-09-19 14:10:04	管理 授权 删除
设置		2017-07-31 15:33:24	管理 授权 删除

5. 在表格存储控制台创建实例和创建数据表。

在本示例中, 创建的表格存储实例和数据表信息如下:

- · 实例名称: cap1
- ·数据表名称:vehicle_track
- · 主键信息: vid (integer), gt (integer)
- ・访问域名: https://cap1.cn-hangzhou.ots-internal.aliyuncs.com

📋 说明:	
使用 MaxCompute 访问表格存储时,	建议使用表格存储的私网地址。

· 设置实例网络类型为允许任意网络访问。

<	🛧 cap1
实例详情	 实例访问地址 公网: http://cap1.cn-hangzhou.ots.aliyuncs.com 私网: http://cap1.cn-hangzhou.ots-internal.aliyuncs.com 实例网络类型 更改 介许任意网络访问 ② VPC列表

步骤 1. 安装并配置客户端

1. 下载 MaxCompute 客户端并解压。



确保您的机器上已安装 JRE 1.7或以上版本。

2. 编辑 conf/odps_config.ini 文件,对客户端进行配置,如下所示:

📔 说明:

odps_config.ini 文件中使用#作为注释, MaxCompute 客户端内使用--作为注释。

3. 运行 bin/odpscmd.bat, 输入 show tables; 。

如果显示当前 MaxCompute 项目中的表,则表述上述配置正确。



步骤 2. 创建外部表

创建一张 MaxCompute 的数据表(ots_vehicle_track)关联到 Table Store 的某一张表(vehicle_track)。

关联的数据表信息如下:

- · 实例名称: cap1
- · 数据表名称: vehicle_track
- ・主键信息: vid (int); gt (int)

·访问域名: https://cap1.cn-hangzhou.ots-internal.aliyuncs.com

```
CREATE EXTERNAL TABLE IF NOT EXISTS ots_vehicle_track
(
vid bigint,
gt bigint,
longitude double,
latitude double,
distance double ,
speed double,
oil_consumption double
)
STORED BY 'com.aliyun.odps.TableStoreStorageHandler' -- (1)
WITH SERDEPROPERTIES ( -- (2)
'tablestore.columns.mapping'=':vid, :gt, longitude, latitude, distance
, speed, oil_consumption', -- (3)
'tablestore.table.name'='vehicle_track' -- (4)
)
LOCATION 'tablestore://cap1.cn-hangzhou.ots-internal.aliyuncs.com'; --
(5)
```

参数说明如下:

标号	参数	说明
(1)	com.aliyun.odps. TableStoreStorageHandler	MaxCompute 内置的处 理 Table Store 数据的 StorageHandler,定义了 MaxCompute 和 Table Store 的交互,相关逻辑由 MaxCompute 实现。
(2)	SERDEPROPERITES	可以理解为提供参数选项的 接口,在使用 TableStore StorageHandler 时,有 两个必须指定的选项,分别 是 tablestore.columns. mapping 和 tablestore. table.name。
(3)	tablestore.columns. mapping	必填选项。MaxCompute 将 要访问的 Table Store 表的 列,包括主键和属性列。其 中,带:的表示 Table Store 主键,例如本示例中的:vid 与:gt,其他均为属性列。在 指定映射的时候,用户必须 提供指定 Table Store 表的 所有主键,属性列无需全部 提供,可以只提供需要通过 MaxCompute 来访问的属性 列。

标号	参数	说明
(4)	tablestore.table.name	需要访问的 Table Store 表 名。如果指定的 Table Store 表名错误(不存在),则会报 错。MaxCompute 不会主动 创建 Table Store 表。
(5)	LOCATION	指定访问的 Table Store 的 实例信息,包括实例名和 endpoint 等。

步骤 3. 通过外部表访问 Table Store 数据

创建外部表后,Table Store 的数据便引入到了 MaxCompute 生态中,您可通过 MaxCompute SQL 命令来访问 Table Store 数据。

// 统计编号 4 以下的车辆在时间戳 1469171387 以前的平均速度和平均油耗
select vid,count(*),avg(speed),avg(oil_consumption) from ots_vehicl
e_track where vid <4 and gt<1469171387 group by vid;</pre>

返回类似如下结果:

odps@ ana	alysis_vehicle>se	lect vid,count(*), avg(:	speed),avg(oil_cons	umption)	from ots_	vehicle_track where vid <4 and gt<1469171387 group by vid;
ID = 2017 Log view: http://lo 09CTzoxNj aGljbGUva Job Queue	70306160538185gsv : ogview.odps.aliyu jA0MzM3MzgzMTcONj aN5zdGFuY2VzLzIwM eing.	lupk2 n.com/logview/3 E5LDE000k0MjExh TcwMzA2MTYwNTM4	h=http:// lzgsey3Tdi MTg1Z3N2I	/service.oc GF0ZW1lbnQ3 bHVwazI1XX1	ips.aliyu iOlt7IkFji idLCJWZXJ:	n.com∕api MGlvbiIGW ZƏW9uIjoi	šp=analys yJvZHBzO1 MSJ9	Ls, vehici eki - 2017036656833185gs viljek 2016en - sijoukznaams medio sobosta ontrinations venem 19 vegi Skulinkanami folgen bici Li CI Sznivedo (2016) in vankazine zageni jeza firski zir vehici z takazi
	STAGE	S STATUS	TOTAL	COMPLETED	RUNNING	PENDING	BACKUP	
M1_job_0		. TERMINATED						
82_1_job_	_0	. TERMINATED						
STAGES: 0								
Summary:								
vid		_c2						
0		0.1162258979	6672624	6.5155061	80530814			
1		0.1120064978		6.5673983	00182988			
2	47	0.0989717631	46085	6.738738527	883797			
	47	0.1150391655	1605494	+	109275189			

3.2 跨账号授权

本文主要为您介绍不同账号之间如何实现表格存储和 MaxCompute 之间的无缝连接。

如需了解同账号下的表格存储与 Maxcompute 对接操作,请参考同账号下使用 MaxCompute 访问表格存储。

准备工作

跨云账号需要两个主账号,账号 A 将访问权限授予账号 B,则运行 MaxCompute 时,账号 B 可 以访问账号 A 下的表数据。基本信息如下:

📋 说明:

以下信息仅为示例,在操作时请替换为实际使用的信息。

项目	表格存储	MaxCompute
主账号名	账号 A	账号 B
UserId	12345	56789

使用 MaxCompute 跨云账号访问表格存储前,您需要完成以下准备工作:

- 1. 账号 B 开通MaxCompute 服务,并创建工作空间。
- 2. 账号 A 和 B 分别创建 AccessKey。
- 3. 使用账号 A 登录 RAM 控制台,并在RAM角色管理页面,新建RAM角色。

在本示例中,假设创建的角色名称为 AliyunODPSRoleForOtherUser。

4. 在RAM角色列表中,找到AliyunODPSRoleForOtherUser角色,然后单击RAM角色名称,设置策略内容。策略内容设置如下:

```
{
   "Statement": [
      {
        "Action": "sts:AssumeRole",
        "Effect": "Allow",
        "Principal": {
            "Service": [
              "1xxxx@odps.aliyuncs.com"
        ]
        }
      }
   ],
   "Version": "1"
}
```

│ _ _ _ 说明:

请将上述策略内容中的 1xxxx 替换成您的 UID 即可。

5. RAM角色创建后,您可以在基本信息页面查看该角色的ARN:

人员管理	~					
用户组		基本信息 RAM角色名称	AlijunConsumeDump205SRole		创建时间	
用户		备注	面目中心(Consume)期以使用此角色中的问题的OSS规律		ARN	
设置						
SSO 管理		权限管理	信任策略管理			
权限管理	^ <	添加权限	精确授权			
授权		权限应用范围	权限策略名称	权限策略类型		备注
权限策略管理		全局	AlgunConsumeDumg2055RolePaticy	系统策略		用于意用中心的思想
RAM角色管理						

6. 返回 RAM 控制台首页,进入权限策略管理页面,单击新建权限策略。

RAM访问控制	← 新建自定义权限策略
概览	
人员管理へ	策略名称
用户组	AliyunODPSRolePolicyForOtherUser
用户	备注
设置	
SSO 管理	配置模式
权限管理へ	 ○ 一可视化配置 ● 脚本配置
授权	
权限策略管理	策略内容
	导人已有系统策略
KAM用巴官埋	1 {
OAuth应用管理	<pre> 2 "Statement": [3 { 4 {</pre>

在本示例中, 假设授权策略名称为AliyunODPSRolePolicyForOtherUser。

选择脚本配置,然后输入策略内容:

```
"Effect": "Allow"
}
]
}
```

蕢 说明:

您也可以自定义其他权限,如 CreateTable 等。

- 7. 在RAM角色管理页面,找到AliyunODPSRoleForOtherUser角色,然后单击添加权限。
- 8. 在添加权限页面,选择AliyunODPSRolePolicyForOtherUse权限,然后单击确定。
- 9. 在表格存储控制台创建实例和创建数据表。

在本示例中, 创建的表格存储实例和数据表信息如下:

- · 实例名称: cap1
- ·数据表名称:vehicle_track
- · 主键信息: vid (integer), gt (integer)
- ・访问域名: https://cap1.cn-hangzhou.ots-internal.aliyuncs.com

📕 说明:

使用 MaxCompute 访问表格存储时,建议使用表格存储的私网地址。

· 设置实例网络类型为允许任意网络访问。

使用 MaxCompute 访问表格存储

跨账号访问的操作与同账号下的访问一样,只是在创建外部表时使用 roleArn。

账号 B 通过 MaxCompute 创建外部表,指定准备工作中创建出来的 roleArn 来访问表格存储。

具体操作步骤请参考同账号授权访问。其中,在步骤2创建外部表时,使用如下代码:

```
CREATE EXTERNAL TABLE ads_log_ots_pt_external
vid bigint,
gt bigint,
longitude double,
latitude double,
distance double ,
speed double,
oil_consumption double
STORED BY 'com.aliyun.odps.TableStoreStorageHandler'
WITH SERDEPROPERTIES (
'tablestore.columns.mapping'=':vid, :gt, longitude, latitude, distance
 speed, oil_consumption',
'tablestore.table.name'='vehicle_track'
'odps.properties.rolearn'='acs:ram::12345:role/aliyunodpsroleforoth
eruser
)
```

```
LOCATION 'tablestore://cap1.cn-hangzhou.ots-internal.aliyuncs.com'
USING 'odps-udf-example.jar'
```

3.3 使用AccessKey访问表格存储

除了授权方式外,您还可以在 MaxCompute 中使用 AccessKey 访问表格存储的数据。

准备工作

获取表格存储资源所属账号的AccessKeyId 和 AccessKeySecret,如果该 AK 是资源所属账号的 子账号,那么该子账号至少需要对表格存储相关的资源具有以下权限:

```
{
  "Version": "1",
  "Statement": [
    {
      "Action": [
        "ots:ListTable",
        "ots:DescribeTable",
        "ots:GetRow",
"ots:PutRow",
        "ots:UpdateRow",
        "ots:DeleteRow",
        "ots:GetRange",
        "ots:BatchGetRow",
        "ots:BatchWriteRow"
        "ots:ComputeSplitPointsBySize"
      ],
      "Resource": "*",
      "Effect": "Allow"
    }
  ]
}
--您也可以自定义其他权限
```

在 MaxCompute 中使用 AccessKey 访问表格存储

同授权方式不同的是,需要在创建外表时在LOCATION中显示写入AK信息,其格式为:

LOCATION 'tablestore://\${AccessKeyId}:\${AccessKeySecret}@\${InstanceNa me}.\${Region}.ots-internal.aliyuncs.com'

假设需要访问的表格存储资源的信息为:

AccessKeyId	AccessKeyS ecret	实例名称	区域	网络模式
abcd	1234	cap1	cn-hangzhou	内网访问

创建外表的语句为:

```
CREATE EXTERNAL TABLE ads_log_ots_pt_external
(
vid bigint,
gt bigint,
longitude double,
latitude double,
distance double ,
speed double,
oil_consumption double
)
STORED BY 'com.aliyun.odps.TableStoreStorageHandler'
WITH SERDEPROPERTIES (
'tablestore.columns.mapping'=':vid, :gt, longitude, latitude, distance
, speed, oil_consumption',
'tablestore.table.name'='vehicle_track'
)
LOCATION 'tablestore://abcd:1234@cap1.cn-hangzhou.ots-internal.
aliyuncs.com'
```

对数据访问的操作步骤请参考使用MaxCompute访问表格存储中的步骤3.通过外部表访问 Table Store 数据。

3.4 使用 UDF 处理数据

如果您在表格存储里面的数据有着独特的结构,希望自定义开发逻辑来处理每一行数据,比如解析 特定的 JSON 字符串,可以使用 UDF(User Defined Function,即用户自定义函数)来处理。

操作步骤

1. 参考 MaxCompute Studio 文档, 在 IntelliJ 中安装 MaxCompute-Java/MaxCompute-Studio 插件。插件安装完毕, 就可以直接开发。

下图是一个简单的 UDF 定义,将两个字符串连接。MaxCompute 支持更复杂的 UDF,包括 自定义窗口执行逻辑等,更多信息请参考开发和调试 UDF。

 idea META-INF mr_ut_local_jobs out src com.aliyun.tablestore.sql com.aliyun.tablestore.sql a cloud_metric_udf.iml cloud_metric_udf.properties cloud_metric_udf.xml External Libraries 	5 public 6 // 7 pub 8 9 0 1 2 2 3 ⊖ }	<pre>class ExtractBusID extends UDF { TODO define parameters and return type, e.g: pi lic String evaluate(String a, String b) { StringBuilder sb = new StringBuilder(); sb.append(a); sb.append(":"); sb.append(b); return sb.toString();</pre>
 In activation-1.1.jar library root In antlr4-4.3-complete.jar library root In antlr4-annotations-4.3.jar library root 	4 }	

2. 打包之后可以上传到 MaxCompute。

选择 File > Project Structure > Artifacts, 输入Name 和 Output directory 后,单击+ 选择输出模块。打包后通过 ODPS Project Explorer 来上传资源、创建函数,然后就可以在 SQL 中调用。

Project Structure				
 ☑ Project Structure ♀ ⇒ Project Settings Project Modules Libraries Facets Artifacts Platform Settings SDKs Global Libraries Problems 	+ − ≇ cloud_metric_extract_r	Name: cloud_metric_extract_md5 Output directory: D:\temp\cloud_m Build on make Output Layout Pre-processing Pos Output Layout Pre-processing Pos Car III +, Module Output Cloud_me File Cloud_me File Cloud_me Extracted Directory	etric_udf\out\artifacts\cloud_metric_u t-processing Available Elem t t t t t t t t t t t t t t t t t t t	Type: AR Typ
		Show content of elements	i com i com i com i com i com i com i com i com i com i fastj i fluer fluer i fluer	mons-cli-1.2.jar mons-cli-1.3.1.jar mons-codec-1.4.jar mons-compress-1.4.jar mons-io-2.4.jar mons-lang-2.5.jar mons-lang-2.5.jar mons-logging-1.1.1.jar son-1.2.8.jar nt-core-0.2.0-SNAPSHOT.jar nt-sdk-0.2.0-SNAPSHOT.jar

3. 运行bin/odpscmd.bat。

// 我们选出来1行数据,并将name/name传入UDF, 返回两个string的累加
select cloud_metric_extract_md5(name, name) as udf_test from
test_table limit 1;

返回结果如下:

odps@ table_store_sql_en	gine_dev>select o	cloud_me	etric_extrac	t_md5(c,	c) as udf __	_test from	cloud_metr	ic_stable	limit 1;
ID = 20170302055324953gq .og view: http://logview.odps.aliy 955324953gq1tsau1&token= -jdGlvbiI6WyJvZHBz0IJ1YW c3RhbmNlcy8yMDE3MDMwMjA1 Job Queueing QuotaCPUUsage: 99.99%	ltsau1 un-inc.com:8080/J d214cGJkSk9VRW1G(QiXSwiRWZmZWN0Ijc NTMyNDk1M2dxMXRzY QuotaMemUsage: 7	logview/ QkNmNXZC DiQWxsb3 YXUxI119 79.36%	??h=http://s CV0J0ZWQ4T21 3ciLCJSZXNvd 9XSwiVmVyc21	ervice-co zPSxPRFBT IXJjZSI6Wy vbiI6IjEi	orp.odps.a TX09CTzoxI rJhY3M6b2F ifQ==	aliyun-inc NDE0MDcwMj RwczoqOnByl	.com/api&p= YwNjg3NzQ1L o2p1Y3RzL3R	table_sto DE00DkwMz hYmx1X3N0	re_sql_eng g4MDUseyJ b3J1X3NxbF
STAG	ES STATUS	TOTAL	COMPLETED	RUNNING	PENDING	BACKUP			
11_job_0	TERMINATED	1	1	0	0	0			
R2_1_job_0	TERMINATED	1	1	0	0	0			
STAGES: 02/02 [=====		====>>]	100% ELAPS	SED TIME:	350.08 s				
Summary:									
+ udf_test + code4xx1,0.00,netflow,	2512570.00,qps,29	989.00,p	999RT,95607.	60,code5>	(x,0.00,Ma	axRT,43255	3.00,MinRT,	0.00,AvgR	, 9940.51

3.5 常见问题

本文主要为您介绍使用MaxCompute访问表格存储的相关常见问题。

• FAILED: ODPS-0010000:System internal error - fuxi job failed, WorkerPack ageNotExist

原因: 需要设置 set odps.task.major.version=unstructured_data。

• FAILED: ODPS-0010000:System internal error - std::exception:Message: a timeout was reached

原因:一般情况下是表格存储的 endpoint 填写错误,导致 MaxCompute 无法访问。

logview invalid end_point

原因:在执行过程中,会返回 logview URL 地址,如果使用浏览器访问该地址返回错误,可能 是配置不对,请检查 MaxCompute 配置。

如果仍未解决问题,请提交工单。

4 Data Lake Analytics

4.1 背景信息

Table Store 中接入 Data Lake Analytics(简称 DLA)服务的方式,为您提供一种快速的 OLAP(On-Line Analytical Processing) 解决方案。DLA 是阿里云一款通用的 SQL 查询引 擎,使用通用的 SQL 语言(兼容 MySQL 5.7 绝大部分查询语法)可在 Table Store 中进行灵活 的数据分析。



如架构图所示, OLAP查询架构涉及阿里云DLA、Table Store 和 OSS 三款产品。

- · DLA:负责分布式 SQL 查询计算。在实际运行过程中将 SQL 查询请求进行任务拆解,产生若 干可并行化的子任务,提升数据计算和查询能力。
- · Table Store:数据存储层,用于接收 DLA 的各类子查询任务。如果您在 Table Store 中已经 有存量数据,可以直接在 DLA 上建立映射视图,从而体验 SQL 计算的便捷服务。
- · OSS: 分布式对象存储系统, 主要用于保存查询结果。

如果您要在 Table Store 中体验分布式 SQL 计算,须开通 Table Store、DLA 和 OSS 服务。



- ・ 开通OSS服务的主要原因是 DLA 默认将查询结果集数据写入 OSS,因此需要引入一个额外的 存储依赖。您仅需开通 OSS 服务,无需预先创建 OSS 存储实例。
- ・目前开服公测的区域是华东2(上海),对应的实例是该 region 内所有的容量型实例。在开通 DLA 服务时,需要先填写公测申请。详情请参考准备工作文档。

4.2 准备工作

如果您要在 Table Store 中体验分布式 SQL 计算,须开通 Table Store、Data Lake Analytics 和 OSS 服务。本文主要为您介绍如何开通这些服务。

道 说明:

完成以上三种服务接入后,实际查询将会产生相应的费用。在实际查询过程中如果您的账号欠 费,则查询失败。

开通 Table Store 服务

如果您已经开通 Table Store 服务,并且已创建实例和数据表,则忽略该步骤。

如果您首次使用 Table Store,可按照如下步骤开通 Table Store并创建实例和数据表:

- 1. 登录表格存储(Table Store)产品详情页。
- 2. 单击立即开通。

开通 Table Store 服务后,登录Table Store控制台完成实例和表的创建。

1. 单击右上角的创建实例,如下图所示:

创建实例		\times
* 实例名称:		
* 实例规格:	容量型实例 ▼	
* 实例注释:	最多256个字	
提示: 1.创建实例需要 2.创建实例后,	更几秒钟的时间,创建成功请按刷新按钮刷新实例列表. 实例域名会在1分钟内生效.	
	确定	取消

2. 完成实例创建后,单击右上角创建数据

*数据表名称:	test001
*数据生命周期:	-1
	注:数据生命周期最低为86400秒(一天)或者-1(永
* 最大数据版本:	1
	注:数据版本需要为非0值
* 数据有效版本偏差:	86400
	注:写入数据所有列的版本号与写入时时间的差值 入失败
* 表主键:	
	pk0
	注:表主键最多4个,您已创建1个
	注:为了使得数据在表上分布均匀,避免读写热点 使用哈希或者类似方式使得分片键的值均匀分布,
	+ 添加表主键
□ 开启Stream功能	注:Stream资费详见价格总览

3. 完成数据表创建后,单击右上角的插入数据。

插	入数据						×
	主键名称		主键类型		主键值		
	pk0		INTEGER		001		
	× 移除所有属性列						
	属性列名称	属性	主列类型	属性值		数据版本号	操作
	col0001	IN	TEGER 🔻	100		或 ☑ 使用系统时间	☆●●●
	col002	IN	TEGER 🔻	50		或 🗹 使用系统时间	^面 删除
	+ 増加属性列 注	È: 扂	性列最多只能自	定义插入江	20列,您已经创]建 2 列	
						确定插入	取消

开通 OSS 服务

如果您首次使用OSS,可按照如下步骤开通OSS:

- 1. 登录对象存储 OSS 产品详情页。
- 2. 单击立即开通。

开通 Data Lake Analytics 服务

如果您首次使用Data Lake Analytics,可按照如下步骤开通DLA:

1. 登录Data Lake Analytics 产品详情页。

2. 单击立即开通。

ľ	说明:		
---	-----	--	--

处于公测阶段时,开通服务需要做公测申请,填写好相关信息即可。

开通 DLA 服务后,按照如下步骤申请 DLA 账号:

1. 登录 DLA 管理控制台,选择开通对应 region 的 DLA 服务实例(如华东 2 上海区域),然后 单击初始化服务。

说明:

不同的region对应不同的账号,且不同region之间的DLA账号不能混用。

6)	管理控制台	产品与服务	•	1	🛀 华	东2(上海)	•						1	皮索		Q	A 🔽	费用	工单	备案	企业		支持与服务	简体中文	: 📀
=	Data Lake An	alytics	D	Data	a Lake	Analy	tics 服	务集群多	例													初始化	服务	TableSto	ire 授权	〇 刷新
8	服务集群实例	1	涟	连接信	信息										网络类	き型	区域		操作							
¥			se	servic	ce.cn-s	hangha	.datalak	eanalytic	s.aliyunc	s.com:1	0000				经典网	网络	cn-sha	nghai	登录	数据库	MyS	L连接串	J	JDBC连接串		
*			se	servic	ce-any	pc.cn-	hanghai	datalak	analytics	s.aliyunc	s.com:1	10000			VPC		cn-sha	nghai	登录	数据库	MyS	L连接串	J	JDBC连接串		
o																										
*																										
\$																										
ග																										

2. 在云产品开通页面,单击立即开通。

_	
	计分口目
_	「尻明

账号创建完成之后会收到相关邮件(邮箱为阿里云的注册邮箱),内含该region的DLA账号和 密码,注意查收。

3. 选择region,单击右上角的Table Store授权。

4. 在云资源访问授权页面,单击同意授权,授权 DLA 访问 Table Store 中的用户实例数据,如下 图所示:

提示:如需修改角色权附	艮,请前往RAM控制台角色管理中设置,需要注意的是,错误的配置可能导致OpenAnalytics无法获取到必要的权限。
penAnalytics请求获	丧取访问您云资源的权限
方是系统创建的可供Ope	nAnalytics使用的角色,授权后,OpenAnalytics拥有对您云资源相应的访问权限。
AliyunOpenAnaly	ticsAccessingOTSRole
AliyunOpenAnaly 描述: OpenAnalytics累	ticsAccessingOTSRole 默认使用此角色来访问OTS
AliyunOpenAnaly 描述: OpenAnalytics 权限描述: 用于开放分	ticsAccessingOTSRole 默认使用此角色来访问OTS 济服务的授权策略,包括OTS的部分访问权限
AliyunOpenAnalyi 描述: OpenAnalytics 权限描述: 用于开放分	ticsAccessingOTSRole 就认使用此角色来访问OTS 计服务的授权策略,包括OTS的部分访问权限

4.3 使用DLA服务

开通服务后,可通过控制台、MySQL Client 以及 JDBC 这 3 种方式接入 DLA 服务并进行 SQL 查询。

Table Store 和 DLA 元信息映射逻辑

 ・ 库和表等概念映射

Table Store	DLA
实例 (instance)	schema 或 database
表 (table)	table
主键列 (pk)	column, isPrimaryKey=true, isNullable=false
非主键列(column)	column, isPrimaryKey=false, isNullable=<用户的DDL定义>

· 字段的映射关系

Table Store	DLA
INTEGER(8bytes)	bigint(8bytes)
STRING	varchar
BINARY	varbinary(目前主键中不支持)
DOUBLE	double

Table Store	DLA
BOOLEAN	byte

控制台访问 DLA

控制台访问 DLA 步骤如下:

- 1. 使用邮件中随附的该 region 的用户名和密码登录数据库。
- 2. 为 Table Store 中的实例表格数据建立映射。

假设您在上海 region 已创建一个名为 sh_tpch 的实例,该实例包含表格 test001,表格内包含 2 行测试数据。该实例建立映射的步骤如下:

 将 Table Store 的实例映射成 DLA 的一个 DataBase 实例。建立 DLA 的 Database 映 射前,首先需要在 Table Store 中创建实例,如创建一个名为 sh-tpch 的实例,对应的 endpoint 为 sh-tpch.cn-shanghai.ots.aliyuncs.com。

完成测试实例创建后,执行下列语句建立 Database 映射:

```
CREATE SCHEMA sh_tpch001 with DBPROPERTIES(LOCATION ='https://sh-
tpch.cn-shanghai.ots.aliyuncs.com', catalog='ots', instance ='sh-
tpch');
```



使用 MySQL Client 时,可以使用 create database 或 create schema 语句创建 database 映射。但是控制台目前只支持 create schema 语句创建 database 映射。

DMS for Data Lake Analytics	SQL窗口					
对象列表 🔇	首页 SQL查询 ×					
🛓 🚞 sh_tpch	💞 同步执行(F8) 🥑 异步执行 🛄 单行详情 🔤 格式化(F9) 数据库: sh_tpch 🔹 3					
	1 CREATE SCHEW sh_tpch001 with DBPROPERTIES(LOCATION ='https://sh-tpch.cn-shanghai.ots.aliyuncs.com', catalog='ots', instance ='sh-tpch');					
	执行历史 执行状态 执行结果					
	浏览器能展示的数据量有限,同步执行最大返回 10000 行数据,如果您需要查询超过 10000 行的数据,请使用「异步执行」					
	序号 RESULT ▼					
	1 命令已成功完成					

2. 在 tp_tpch001 的 Database 下建立表格的映射。在建立 DLA 的表格映射前,首先需要在 Table Store 中创建数据表。

数据表创建完成后,执行下列语句建立表格映射:

CREATE EXTERNAL TABLE test001 (pk0 int NOT NULL , primary key(pk0));

📕 说明:

建立 DLA 映射表时,指定的 Primary Key 必须与 Table Store 表格定义 Primary Key 列 表一致。Primary Key 用于唯一确定一行的数据,一旦映射表的 Primary Key 列表与 Table Store 表格的 PK 不一致,可能导致 SQL 查询结果出现非预期错误。

DMS for Data Lake Analytics	SQL窗口							
对象列表	首页 SQL查询 ×							
sh_tpch (1)	🥐 同步执行(F8) 🛷 异步执行 🛄 单行详情 🛄 格式化(F9) 数据库: sh_tpch001 🔽 📿							
	CREATE TABLE test001 (pk0 int , primary key(pk0));							
	选择DB,在该DB下,创建表格test001							
	执行历史 执行状态 执行结果 初览器能展示的数据量有限,同步执行最大返回 10000 行数据,如果您需要查询超过 10000 行的数据,请使用 [异步执行]							
	序号 RESULT [▼]							
	1 命令已成功完成							

例如,您的 Table Store 实例 sh_tpch 中包含 test001 表格,其中只有一列 pk0。使用 show 命令可查看该表已创建成功:

DMS for Data Lake Analytics	SQL窗口
对象列表 🕜	首页 SQL查询 ×
+ _ sh_tpch (1)	🦸 同步执行(F8) 🛷 异步执行 🔤 单行详情 🔤 格式化(F9) 数据库: sh_tpch001 🔹 🗲
	1 show tables;
	外门加更 外门状态 外门结果
	浏览器能展示的数据量有限,同步执行最大返回 10000 行数据,如果您需要查询超过 10000 行的数据,请使用「异步执行」
	序号 I Table_Name *
	1 test001

- 3. 使用select语句执行SQL查询:
 - ・ 査出所有数据:

select * from test001;

DMS for Data Lake Analytics	SQL窗口					
对象列表	首页 SQL查询 ×					
sh_tpch (1)	🦸 同步执行(F8) 🦸 异步执行 🛄 单行详情 🔤 格式化(F9) 数据库: sh_tpch001 🔹 🗲					
	1 select * from test001; 2					
	执行历史 执行状态 执行结果					
	浏览器能展示的数据量有限,同步执行最大返回 10000 行数据,如果您需要查询超过 10000 行的数据,请使用「异步执行」					
	序号 iz pk0 ▼					
	1 100					
	2 120					

·执行count统计:

select count(*) from test001;

DMS for Data Lake Analytics	SQL窗口				
对象列表 ③	首页 SQL查询 ×				
sh_tpch(1)	🦸 同步执行(F8) 🛷 异步执行 🔄 单行详情 🔄 格式化(F9) 数据库: sh_tpch001 💌 🗢				
	<pre>select count(*) from test001;</pre>				
	执行历史 执行状态 执行结果				
	浏览器能展示的数据量有限,同步执行最大返回 10000 行数据,如果您需要查询超过 10000 行的数据,请使用「异步执行」				
	序号 12 *				
	1 2				

・执行sum统计:

select sum(pk0) from test001;

DMS for Data Lake Analytics	SQL窗口
对象列表 🔇	首页 SQL查询 ×
sh_tpch (1)	🦸 同步执行(F8) 🛷 异步执行 🔤 单行详情 🤤 格式化(F9) 数据库: sh_tpch001 🔹 🗲
	1 select sum(pk0) from test001;
	执行历史 执行状态 执行结果
	浏览器能展示的数据量有限,同步执行最大返回 10000 行数据,如果您需要查询超过 10000 行的数据,请使用「异步执行」
	序号
	1 220

执行SQL查询时,可以选择同步执行结果,返回满足条件的前 10,000 条记录;如要获取大结果集数据,请选择异步执行,并使用show query_id的方式异步获取结果:

show query_task where id = '59a05af7_1531893489231';

DMS for Data Lake Analytics	SQL窗口
对象列表	首页 SQL查询 × ☞ 同步执行(F8) ● 异步执行 ● 单行详情 ● 格式化(F9) 数据库: sh_tpch001 ▼ 2 1 select * from test001; 异步执行
DMS for Data Lake Analytics	<u>执行历史</u> <u>执行状态</u> <u>执行结果</u> 用于查询结果的任务ID 浏览器能展示的数据量有限,同步执行最大返回 10000 行数据,如果您需要查询超过 10000 行的数据,请使用「异步执行」 序号 I ASYNC_TASK_ID 59a05af7_1531893489231 SQL窗口
对象列表	G 首页 SQL查询 ×
 → □ sh_tpch → □ sh_tpch001 	伊 · · · · · · · · · · · · · · · · ·

其他执行语句,请查看如下说明文档:

- · create schema语句
- · create table语句
- select语句
- show语句
- · drop table语句
- · drop schema语句

MySQL Client 访问 DLA

您可以使用标准的 MySQL Client 快速接入 DLA 的数据实例,其连接语句为:

```
mysql -h service.cn-shanghai.datalakeanalytics.aliyuncs.com -P 10000 -
u <username> -p <password> -c -A
```



其他操作语句与控制台访问一致。

JDBC 访问 DLA

您还可以使用标准的 Java API 访问 DLA,其连接语句为:

jdbc:mysql://service.cn-shanghai.datalakeanalytics.aliyuncs.com:10000/



其他操作语句与控制台访问一致。

5 Hive/HadoopMR

5.1 环境准备

本文主要为您介绍使用 Hive/HadoopMR 来访问表格存储中的表前的环境准备。

使用 Hive/HadoopMR 来访问表格存储中的表

通过表格存储及 E-MapReduce 官方团队发布的依赖包,可以直接使用 Hive 及 HadoopMR 来 访问表格存储中的数据并进行数据分析。

安装 JDK-7+

- 1. 下载并安装 JDK-7+ 安装包。
 - · Linux/MacOS 系统:使用系统自带的包管理器安装
 - · Windows 系统: 点此下载
- 2. 按照以下示例进行安装检查。

```
$ java -version
java version "1.8.0_77"
Java(TM) SE Runtime Environment (build 1.8.0_77-b03)
Java HotSpot(TM) 64-Bit Server VM (build 25.77-b03, mixed mode)
```

安装并启动 Hadoop 环境

- 1. 下载 2.6.0 版本以上的 Hadoop 安装包。(点此下载)
- 2. 解压并安装,根据实际集群情况安装 Hadoop 服务。
- 3. 按照如下示例启动 Hadoop 环境。

```
$ bin/start-all.sh
# 检查服务是否成功启动
$ jps
24017 NameNode
24835 Jps
24131 DataNode
24438 ResourceManager
5114 HMaster
24287 SecondaryNameNode
24527 NodeManager
```

4. 在 /etc/profile 中添加 Hadoop 路径,并执行 source /etc/profile 的命令使配置生

效。

```
export HAD00P_HOME=/data/hadoop/hadoop-2.6.0
export PATH=$PATH:$HAD00P_HOME/bin
```

下载及安装 Hive 环境

- 1. 下载类型为 bin.tar.gz 的 Hive 安装包。(点此下载)
- 2. 按照如下示例解压安装包。
 - \$ mkdir /home/admin/hive-2.1.0
 - \$ tar -zxvf apache-hive-2.1.0-bin.tar.gz -C /home/admin/
 - \$ mv /home/admin/apache-hive-2.1.0-bin /home/admin/hive-2.1.0/
- 3. 按照如下示例初始化 schema。
 - # 进入指定的目录
 - \$ cd /home/admin/hive-2.1.0/
 - # 初始化,如果是mysql则derby可以直接替换成mysql
 - # 如果执行出错可以删除rm -rf metastore_db/之后重新执行
 - \$./bin/schematool -initSchema -dbType derby
- 4. 按照如下示例启动 Hive 环境。

```
$ ./bin/hive
# 检查服务是否成功启动
hive> show databases;
OK
default
Time taken: 0.207 seconds, Fetched: 1 row(s)
```

下载表格存储的 Java SDK

1. 在 Maven 库中下载 4.1.0 版本以上的 Java SDK 相关依赖包。(点此下载)



该依赖包会随最新的 Java SDK 发布,请根据最新的 Java SDK 版本下载相关依赖包。

2. 按照如下示例将 SDK 拷贝到 Hive 目录下。

```
$ mv tablestore-4.1.0-jar-with-dependencies.jar /home/admin/hive-2.1
.0/
```

下载阿里云 EMR SDK

点此下载 EMR SDK 依赖包。



了解更多 EMR 信息请参考<mark>这里</mark>。

5.2 使用教程

本文主要为您介绍如何使用 Hive/HadoopMR 来访问表格存储中的表。

数据准备

在表格存储中准备一张数据表 pet, name 是唯一的一列主键, 数据示例如下:

name	owner	species	sex	birth	death
Fluffy	Harold	cat	f	1993-02-04	
Claws	Gwen	cat	m	1994-03-17	
Buffy	Harold	dog	f	1989-05-13	
Fang	Benny	dog	m	1990-08-27	
Bowser	Diane	dog	m	1979-08-31	1995-07-29
Chirpy	Gwen	bird	f	1998-09-11	
Whistler	Gwen	bird		1997-12-09	
Slim	Benny	snake	m	1996-04-29	
Puffball	Diane	hamster	f	1999-03-30	

📕 说明:

表格中空白的部分不需要写入,因为表格存储是一个 schema-free 的存储结构(数据模型),没 有值也不需要写入NULL。

Hive 访问示例

前提条件

按照准备工作准备好 Hadoop、Hive、JDK 环境以及表格存储 JAVA SDK 和 EMR SDK 依赖包。

示例

```
# HADOOP_HOME 及 HADOOP_CLASSPATH 可以添加到 /etc/profile 中
$ export HADOOP_HOME=${你的 Hadoop 安装目录}
$ export HADOOP_CLASSPATH=emr-tablestore-1.4.2.jar:tablestore-4.3.1-
jar-with-dependencies.jar:joda-time-2.9.4.jar
$ bin/hive
hive> CREATE EXTERNAL TABLE pet
(name STRING, owner STRING, species STRING, sex STRING, birth STRING
, death STRING)
STORED BY 'com.aliyun.openservices.tablestore.hive.TableStore
StorageHandler'
WITH SERDEPROPERTIES(
"tablestore.columns.mapping"="name,owner,species,sex,birth,death")
TBLPROPERTIES (
"tablestore.endpoint"="YourEndpoint",
```

"tab	lestore.	access_k	ey_id"="	YourAcce	essKeyId"	, 	
"tac	lestore.	access_k	ey_secre	t^{-} = "Your	Accessive	ysecret"	,
"tab	lestore.	table.na	ıme"="pet	:");			
hive> SE	ELECT * F	ROM pet;					
Bowser	Diane	dog	m	1979-08-	-31	1995-07-	29
Buffy	Harold	dog	f	1989-05-	-13	NULL	
Chirpy	Gwen	bird	f	1998-09-	·11	NULL	
Claws	Gwen	cat	m	1994-03-	-17	NULL	
Fang	Benny	dog	m	1990-08-	-27	NULL	
Fluffy	Harold	cat	f	1993-02-	-04	NULL	
Puffball	-	Diane	hamster	f	1999-03-	30	NULL
Slim	Benny	snake	m	1996-04-	-29	NULL	
Whistler		Gwen	bird	NULL	1997-12-	09	NULL
Time tak	ken: 5.04	5 second	ls, Fetch	ed 9 rov	ı(s)		
hive> SE	ELECT * F	ROM pet	WHERE bi	rth > "1	995-01-0	1";	
Chirpy	Gwen	bird	f	1998-09-	-11	NULL	
Puffball	-	Diane	hamster	f	1999-03-	30	NULL
Slim	Benny	snake	m	1996-04-	-29	NULL	
Whistler	-	Gwen	bird	NULL	1997-12-	09	NULL
Time tak	ken: 1.41	_ seconds	, Fetche	d 4 row((s)		

参数说明如下:

• WITH SERDEPROPERTIES

tablestore.columns.mapping(可选):在默认的情况下,外表的字段名即为表格存储上表的列名(主键列名或属性列名)。但有时外表的字段名和表上列名并不一致(比如处理大小写或字符集相关的问题),这时候就需要指定 tablestore.columns.mapping。该参数为一个英文 逗号分隔的字符串,每一项都是表上列名,顺序与外表字段一致。

空白也会被认为是表上列名的一部分。

TBLPROPERTIES

- tablestore.endpoint(必填):访问表格存储的服务地址,也可以在表格存储控制台上查 看这个实例的 endpoint 信息。
- tablestore.instance(可选):表格存储的实例名。若不填,则为 tablestore.endpoint 的第一段。
- tablestore.table.name(必填):表格存储上对应的表名。
- tablestore.access_key_id、tablestore.access_key_secret(必填),请参见访问控制。
- tablestore.sts_token (可选),请参见授权管理。

HadoopMR 访问示例

以下示例介绍如何使用 HadoopMR 程序统计数据表 pet 的行数。

示例代码

· 构建 Mappers 和 Reducers

```
public class RowCounter {
public static class RowCounterMapper
extends Mapper<PrimaryKeyWritable, RowWritable, Text, LongWritable>
{
    private final static Text agg = new Text("TOTAL");
    private final static LongWritable one = new LongWritable(1);
    @Override
    public void map(
        PrimaryKeyWritable key, RowWritable value, Context context)
        throws IOException, InterruptedException {
        context.write(agg, one);
    }
}
public static class IntSumReducer
extends Reducer<Text,LongWritable,Text,LongWritable> {
    @Override
    public void reduce(
        Text key, Iterable<LongWritable> values, Context context)
        throws IOException, InterruptedException {
        long sum = 0;
        for (LongWritable val : values) {
            sum += val.get();
        }
        context.write(key, new LongWritable(sum));
    }
}
}
```

数据源每从表格存储上读出一行,都会调用一次 mapper 的 map()。前两个参数 PrimaryKey Writable 和 RowWritable 分别对应这行的主键以及这行的内容。可以通过调用 PrimaryKey Writable.getPrimaryKey() 和 RowWritable.getRow() 取得表格存储 JAVA SDK 定义的主 键对象及行对象。

配置表格存储作为 mapper 的数据源。

```
private static RangeRowQueryCriteria fetchCriteria() {
        RangeRowQueryCriteria res = new
                                             RangeRowQueryCriteria("
YourTableName");
        res.setMaxVersions(1);
        List<PrimaryKeyColumn> lower = new ArrayList<PrimaryKey
Column>();
List<PrimaryKeyColumn> upper = new ArrayList<PrimaryKey
Column>();
        lower.add(new PrimaryKeyColumn("YourPkeyName", PrimaryKey
Value.INF_MIN));
        upper.add(new PrimaryKeyColumn("YourPkeyName", PrimaryKey
Value.INF_MAX));
        res.setInclusiveStartPrimaryKey(new PrimaryKey(lower));
        res.setExclusiveEndPrimaryKey(new PrimaryKey(upper));
        return res;
    }
    public static void main(String[] args) throws Exception {
        Configuration conf = new Configuration();
```

```
Job job = Job.getInstance(conf, "row count");
job.addFileToClassPath(new Path("hadoop-connector.jar"));
job.setJarByClass(RowCounter.class);
job.setMapperClass(RowCounterMapper.class);
job.setCombinerClass(IntSumReducer.class);
job.setOutputKeyClass(IntSumReducer.class);
job.setOutputKeyClass(Text.class);
job.setOutputValueClass(LongWritable.class);
job.setInputFormatClass(TableStoreInputFormat.class);
TableStoreInputFormat.setEndpoint(job, "https://YourInstance
.Region.ots.aliyuncs.com/");
TableStoreInputFormat.setCredential(job, "YourAccessKeyId",
"YourAccessKeySecret");
TableStoreInputFormat.addCriteria(job, fetchCriteria());
FileOutputFormat.setOutputPath(job, new Path("output"));
System.exit(job.waitForCompletion(true) ? 0 : 1);
}
```

示例代码中使用 job.setInputFormatClass(TableStoreInputFormat.class) 把表格存储设 为数据源,除此之外,还需要:

- 把 hadoop-connector.jar 部署到集群上并添加到 classpath 里面。路径为 addFileToC lassPath() 指定 hadoop-connector.jar 的本地路径。代码中假定 hadoop-connector.jar 在当前路径。
- 访问表格存储需要指定入口和身份。通过 TableStoreInputFormat.setEndpoint()和 TableStoreInputFormat.setCredential()设置访问表格存储需要指定的 endpoint 和 access key 信息。
- 指定一张表用来计数。

📕 说明:

- 每调用一次 addCriteria()可以在数据源里添加一个 JAVA SDK 定义的 RangeRowQu eryCriteria 对象。可以多次调用addCriteria()。RangeRowQueryCriteria 对象与 表格存储 JAVA SDK GetRange 接口所用的 RangeRowQueryCriteria 对象具有相同 的限制条件。
- 可以利用 RangeRowQueryCriteria 的 setFilter() 和 addColumnsToGet() 在表格 存储的服务器端过滤掉不必要的行和列,减少访问数据的大小,降低成本,提高性能。
- 通过添加对应多张表的多个 RangeRowQueryCriteria,可以实现多表的 union。
- 通过添加同一张表的多个 RangeRowQueryCriteria,可以做到更均匀的切分。 TableStore-Hadoop Connector 会根据一些策略将用户传入的范围切细。

程序运行示例

```
$ HAD00P_CLASSPATH=hadoop-connector.jar bin/hadoop jar row-counter.jar
```

```
$ find output -type f
output/_SUCCESS
```

```
output/part-r-00000
output/._SUCCESS.crc
output/.part-r-00000.crc
$ cat out/part-r-00000
TOTAL 9
```

类型转换说明

表格存储支持的数据类型和 Hive/Spark 支持的数据类型不完全相同。

下表列出了从表格存储的数据类型(行)转换到 Hive/Spark 数据类型(列)的支持情况。

	TINYIN	SMALLI	INT	BIGINT	FLOAT	DOUBLI	BOOLEA	STRING	BINARY
INTEGE	편,损 失精度	可,损 失精度	可,损 失精度	ग	可,损 失精度	可,损 失精度			
DOUBLI	三 可,损 失精度	可,损 失精度	可,损 失精度	可,损 失精度	可,损 失精度	म्			
BOOLEA	N						म		
STRING								म्	
BINARY									ग

6 Spark/SparkSQL

6.1 环境准备

本文主要为您介绍使用 Saprk/Spark SQL 来查询和链接表格存储中的表需要的环境准备。

使用 Saprk/Spark SQL 来查询和链接表格存储中的表

通过表格存储及 E-MapReduce 官方团队发布的依赖包,可以直接使用 Spark 及 Spark SQL 来 访问表格存储中的数据并进行数据的查询分析。

- 下载及安装 Spark/Spark SQL
 - 下载版本号为 1.6.2 的 Spark 安装包,安装包类型为 Pre-built for Hadoop 2.6。(点此下 载)
 - 2. 按照如下示例解压安装包。

```
$ cd /home/admin/spark-1.6.2
$ tar -zxvf spark-1.6.2-bin-hadoop2.6.tgz
```

安装 JDK-7+

- 1. 下载并安装 JDK-7+ 安装包。
 - · Linux/MacOS系统:请用系统自带的包管理器进行安装
 - · Windows 系统: 点此下载
- 2. 按照如下示例进行安装检查。

```
$ java -version
java version "1.8.0_77"
Java(TM) SE Runtime Environment (build 1.8.0_77-b03)
Java HotSpot(TM) 64-Bit Server VM (build 25.77-b03, mixed mode)
```

下载表格存储的 Java SDK

1. 在 Maven 库中下载 4.1.0 版本以上的 Java SDK 相关依赖包。(点此下载)



该依赖包会随最新的 Java SDK 发布,请根据最新的 Java SDK 版本下载相关依赖包。

2. 按照如下示例将 SDK 拷贝到 Spark 目录下。

```
$ mv tablestore-4.1.0-jar-with-dependencies.jar /home/admin/spark-1.
6.2/
```

下载阿里云 EMR SDK

下载 EMR SDK 相关的依赖包。(点此下载)

📕 说明:

了解更多 EMR 信息请参见<mark>这里</mark>。

启动 Spark SQL

```
$ cd /home/admin/spark-1.6.2/
$ bin/spark-sql --master local --jars tablestore-4.3.1-jar-with-
dependencies.jar,emr-tablestore-1.4.2.jar
```

6.2 使用教程

本文主要为您介绍如何使用 Saprk/Spark SQL 来查询和链接表格存储中的表。

数据准备

在表格存储中准备一张数据表 pet, 其中name是唯一的一列主键。数据示例如下:

name	owner	species	sex	birth	death
Fluffy	Harold	cat	f	1993-02-04	
Claws	Gwen	cat	m	1994-03-17	
Buffy	Harold	dog	f	1989-05-13	
Fang	Benny	dog	m	1990-08-27	
Bowser	Diane	dog	m	1979-08-31	1995-07-29
Chirpy	Gwen	bird	f	1998-09-11	
Whistler	Gwen	bird		1997-12-09	
Slim	Benny	snake	m	1996-04-29	
Puffball	Diane	hamster	f	1999-03-30	

📕 说明:

表格中空白的部分不需要写入,因为表格存储是一个 schema-free 的存储结构(数据模型),没 有值也不需要写入NULL。

Spark SQL 访问示例

前提条件

按照准备工作中的步骤准备好 Spark、JDK 环境以及表格存储 Java SDK 和 EMR SDK 的依赖

包。

示例

```
$ bin/spark-sql --master local --jars tablestore-4.3.1-jar-with-
dependencies.jar,emr-tablestore-1.4.2.jar
spark-sql> CREATE EXTERNAL TABLE pet
  (name STRING, owner STRING, species STRING, sex STRING, birth STRING
 death STRING)
  STORED BY 'com.aliyun.openservices.tablestore.hive.TableStore
StorageHandler'
  WITH SERDEPROPERTIES(
    "tablestore.columns.mapping"="name,owner,species,sex,birth,death")
  TBLPROPERTIES (
    "tablestore.endpoint"="YourEndpoint",
    "tablestore.access_key_id"="YourAccessKeyId",
    "tablestore.access_key_secret"="YourAccessKeySecret",
"tablestore.table.name"="pet");
spark-sql> SELECT * FROM pet;
Bowser
        Diane
                                  1979-08-31
                                                   1995-07-29
                 dog
                         m
Buffy
        Harold
                 dog
                         f
                                  1989-05-13
                                                   NULL
Chirpy
        Gwen
                 bird
                         f
                                  1998-09-11
                                                   NULL
Claws
        Gwen
                 cat
                         m
                                  1994-03-17
                                                   NULL
Fang
        Benny
                 dog
                         m
                                  1990-08-27
                                                   NULL
Fluffy
        Harold
                 cat
                         f
                                  1993-02-04
                                                   NULL
                                                           NULL
Puffball
                 Diane
                         hamster f
                                          1999-03-30
Slim
        Benny
                 snake
                         m
                                  1996-04-29
                                                   NULL
Whistler
                                 NULL
                                                           NULL
                 Gwen
                         bird
                                          1997-12-09
Time taken: 5.045 seconds, Fetched 9 row(s)
spark-sql> SELECT * FROM pet WHERE birth > "1995-01-01";
Chirpy Gwen
                 bird
                         f
                                  1998-09-11
                                                   NULL
Puffball
                 Diane
                         hamster f
                                          1999-03-30
                                                           NULL
Slim
        Benny
                 snake
                                  1996-04-29
                                                   NULL
                         m
                                  NULL
Whistler
                 Gwen
                         bird
                                          1997-12-09
                                                           NULL
Time taken: 1.41 seconds, Fetched 4 row(s)
```

参数说明如下:

- · WITH SERDEPROPERTIES
 - tablestore.columns.mapping(可选):在默认情况下,外表的字段名即为表格存储上表的列名(主键列名或属性列名)。但有时外表的字段名和表上列名并不一致(比如处理大小写或字符集相关的问题),这时候就需要指定 tablestore.columns.mapping。该参数为一个英文逗号分隔的字符串,每个分隔之间不能添加空格,每一项都是表上列名,顺序与外表字段一致。

📕 说明:

表格存储的列名支持空白字符,所以空白也会被认为是表上列名的一部分。

• TBLPROPERTIES

- tablestore.endpoint(必填):访问表格存储的服务地址,也可以在表格存储控制台上查 看这个实例的 endpoint 信息。
- tablestore.instance(可选):表格存储的实例名。若不填,则为 tablestore.endpoint 的第一段。
- tablestore.table.name(必填):表格存储上对应的表名。
- tablestore.access_key_id、tablestore.access_key_secret(必填),请参见访问控制。
- tablestore.sts_token (可选),请参见授权管理。

Spark 访问示例

以下示例介绍如何使用 Spark 程序统计数据表 pet 的行数。

```
private static RangeRowQueryCriteria fetchCriteria() {
    RangeRowQueryCriteria res = new RangeRowQueryCriteria("YourTableN
ame");
    res.setMaxVersions(1);
    List<PrimaryKeyColumn> lower = new ArrayList<PrimaryKeyColumn>();
    List<PrimaryKeyColumn> upper = new ArrayList<PrimaryKeyColumn>();
    lower.add(new PrimaryKeyColumn("YourPkeyName", PrimaryKeyValue.
INF_MIN));
    upper.add(new PrimaryKeyColumn("YourPkeyName", PrimaryKeyValue.
INF_MAX));
    res.setInclusiveStartPrimaryKey(new PrimaryKey(lower));
    res.setExclusiveEndPrimaryKey(new PrimaryKey(upper));
    return res;
}
public static void main(String[] args) {
    SparkConf sparkConf = new SparkConf().setAppName("RowCounter");
    JavaSparkContext sc = new JavaSparkContext(sparkConf);
    Configuration hadoopConf = new Configuration();
    TableStoreInputFormat.setCredential(
        hadoopConf,
        new Credential("YourAccessKeyId", "YourAccessKeySecret"));
    TableStoreInputFormat.setEndpoint(
        hadoopConf,
        new Endpoint("https://YourInstance.Region.ots.aliyuncs.com
/"));
    TableStoreInputFormat.addCriteria(hadoopConf, fetchCriteria());
    try
        ł
        JavaPairRDD<PrimaryKeyWritable, RowWritable> rdd = sc.
newAPIHadoopRDD(
            hadoopConf,
            TableStoreInputFormat.class,
            PrimaryKeyWritable.class,
            RowWritable.class);
        System.out.println(
            new Formatter().format("TOTAL: %d", rdd.count()).toString
());
} finally {
        sc.close();
```

}
}
〕
说明:
如果使用 scala,只需把 JavaSparkContext 换成 SparkContext, JavaPairRDD 换成
PairRDD 即可。或者更简单,交给编译器自行做类型推断

运行程序

```
$ bin/spark-submit --master local --jars hadoop-connector.jar row-
counter.jar
TOTAL: 9
```

类型转换说明

表格存储支持的数据类型和 Hive/Spark 支持的数据类型不完全相同。

下表列出了从表格存储的数据类型(行)转换到 Hive/Spark 数据类型(列)时所支持的情况。

	TINYIN	SMALLI	INT	BIGINT	FLOAT	DOUBLI	BOOLEA	STRING	BINARY
INTEGE	편,损 失精度	可,损 失精度	可,损 失精度	ग	可,损 失精度	可,损 失精度			
DOUBLI	三 可,损 失精度	可,损 失精度	可,损 失精度	可,损 失精度	可,损 失精度	म			
BOOLEA	N						म्		
STRING								म्	
BINARY									म्