# Alibaba Cloud Table Store

計算與分析

檔案版本:20180929



# 目錄

1 函數觸發器	1
1.1 使用Function Compute	1
1.2 使用Function Compute做資料清洗	9
2 MaxCompute	19
- 2.1 使用MaxCompute訪問Table Store	
2.2 跨帳號授權	25
2.3 使用AccessKey訪問Table Store	29
2.4 使用 UDF 處理資料	
2.5 常見問題	
3 Hive/HadoopMR	
- 3.1 環境準備	
3.2 使用教程	
4 Spark/SparkSQL	41
 4.1 環境準備	
4.2 使用教程	

# 1函數觸發器

# 1.1 使用Function Compute

Function Compute(Function Compute)是一個事件驅動的服務,通過Function Compute,使用者 無需管理伺服器等運行情況,只需編寫代碼並上傳。

Function Compute準備計算資源,並以Auto Scaling的方式運行使用者代碼,而使用者只需根據實 際代碼運行所消耗的資源進行付費。詳細資料請參見*Function Compute*####。

Table Store Stream是用於擷取Table Store表中增量資料的一個資料通道,通過建立Table Store觸發器,能夠實現Table Store Stream和Function Compute的自動對接,讓計算函數中自訂的程式邏輯自動處理Table Store表中發生的資料修改。詳細資料請參見*Table Store Stream*。

本節介紹如何使用Function Compute對錶格儲存增量資料進行即時計算。

#### 配置Table Store觸發器

您可以通過控制台建立Table Store觸發器來處理Table Store資料表的即時資料流。

- 1. 建立開通了Stream流的資料表。
  - a. 在Table Store###建立執行個體。

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. 列域名会在1分钟内生效.	

b. 在該執行個體下建立資料表,並且開啟資料表的Stream功能。

取消

角定

. רלאווי	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
私网: http://testInstance.cn-shanghai.ots-internal.aliyuncs.co		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	
允许任皇网络访问 🔘		注:数据版本需要为非0值	
VPC列表	* 数据有效版本偏差:	86400	秒
		注:写入数据所有列的版本号与写入时时间的差值需要 入失败	在数据有效版本偏差范围之内,否者将会写
	* 表主键:		
数据表列表		id 🗦	■符串 ▼ 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 使用哈希或者类似方式使得分片键的值均匀分布,详以	图,充分利用预留读写吞吐量,我们建议您 见最佳实践。
		+ 添加表主键	
	☑ 开启Stream功能	注:Stream资费详见价格总览	
	* Stream记录过期时长:	12	
			<del>确定</del> 取消

#### 2. 建立Function Compute的函數。

### a. 在Function Compute###建立服務(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的進階配置中,您可以佈建服務的角色,用於授權函數進行日誌收集,以及在計算 函數中繼續訪問使用者的其他資源。詳細資料請參見*Function Compute*######。

服务 > testService	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. 在新建立的Service下單擊建立函數。
- d. 在函數模板頁面選擇 空白函數。
- e. 在觸發器配置頁面選擇不建立任何觸發器 , 然後單擊下一步。
- f. 配置函數資訊。

Table Store觸發器會使用*CBOR##*將增量資料編碼為Function Compute的Event,並調用使 用者函數。下圖樣本函數就是將編碼後的Event重新解碼和列印到日誌中心,您可以在解碼資 料後進行任意資料處理。

四致限加	$\rightarrow$	触发器配置	基础管理配置	信息	刻
基础信息	* 函数名称:	streamProcessFunction           命名规范:           » 1. 只能包含字母,数字、           和中划线           » 2. 不能以数字、中划线开           » 3. 长度限制在1-128之间	下划线		
	函数描述:	用于处理tablestore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 OSS上传	本地上传		
	4 import 5 import 6 7 def han 8 logge 9 recor 10 logge 11 retur	cbor logging dler(event, context) r = logging.getLogge: ds = cbor.loads(even r.info(Tablestore s n ok	: r() t) tream records sample	%s', records)	
环境配置	* 函数入口:	index.handler 格式为"[文件名].[函数名]"。 定了函数的调用入口为hello 数。如果是代码直接上传, hello_world.js,只需关注函			
	*函数内存:	128 🔻 MB			

- 3. 建立和測試Table Store觸發器。
  - a. 在*Table Store*###新建立的資料表下,選擇使用已有Function Compute來建立觸發器。

b. 建立過程中需要授權Table Store發送事件通知所需的許可權。

勾選完畢後,實際可以在*RAM###*查看到自動建立的授權角色 AliyunTableStoreStreamNotificationRole。

创建触发器(使用现有函数)



資料處理

• 資料格式

Table Store觸發器使用*CBOR#*#對增量資料進行編碼構成Function Compute的Event。增量資料 的具體資料格式如下:

确於

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

- 成員定義
  - Version
    - 含義: payload版本號碼,目前為Sync-v1
    - 類型: string
  - Records
    - 含義: 資料表中的增量資料行數組
    - 內部成員:
      - ∎ Туре
        - 含義: 資料行類型,包含PutRow、UpdateRow和DeleteRow
        - 類型: string
      - Info
        - 含義: 資料行基本資料
        - 內部成員:
          - Timestamp
          - 含義: 該行的最後修改UTC時間
          - 類型: int64
      - PrimaryKey
        - 含義: 主鍵列數組
        - 內部成員
          - ColumnName
            - 含義: 主鍵列名稱
            - 類型: string
          - Value

- 含義: 主鍵列內容
- 類型: formated\_value,支援integer、string和blob
- Columns
  - 含義: 屬性列數組
  - 內部成員
    - ∎ Туре
      - 含義: 屬性列類型,包含Put、DeleteOneVersion和DeleteAllVersions
      - 類型: string
    - ColumnName
      - 含義: 屬性列名稱
      - 類型: string
    - Value
      - 含義: 屬性列內容
      - 類型: formated\_value,支援integer、boolean、double、string和blob
    - Timestamp
      - 含義: 屬性列最後修改UTC時間
      - 類型: int64
- 資料樣本

```
{
    "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

}

]

}
```

#### 線上調試

Function Compute支援函數的線上調試功能,使用者能夠自己構建觸發的Event,並測試代碼邏輯 是否符合期望。

由於Table Store觸發函數服務的Event是CBOR格式,該資料格式是一種類似JSON的二進位格式,可以按照如下方法進行線上調試:

- 1. 在代碼中同時import cbor和json。
- 2. 單擊觸發事件,選擇自訂,將上述資料樣本的json檔案粘貼至編輯框,根據實際情況修改,並儲存。
- **3.** 代碼中先使用 records = json.loads(event) 來處自訂的測試觸發事件,單擊執行,查看 是否符合期望。
- 4. 當使用 records=json.loads(event) 測試OK之後,可以修改為 records = cbor.
   loads(event) 並儲存,當Table Store中有資料寫入時,相關的函數邏輯就會觸發。



#### 範例程式碼:

```
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 'OK'
```

### 1.2 使用Function Compute做資料清洗

經過了上一節的介紹,下面我們來完成一個使用Function Compute對錶格儲存中的資料做簡單清洗 的情境。

Table Store高並發的寫入效能以及低廉的儲存成本非常適合物聯網、日誌、監控資料的儲存,我們可以將資料寫入到Table Store中,同時在Function Compute中對新增的資料做簡單的清洗,並將清洗之後的資料寫回到Table Store的結果表中,並對未經處理資料及結果資料提供即時訪問。

#### 資料定義

我們假設寫入的為日誌資料,包括三個基礎欄位:

欄位名稱	類型	含義
id	整型	日誌id
level	整型	日誌的等級,越大表明等級越 高
message	字串	日誌的內容

我們需要將 level>1 的日誌寫入到另外一張資料表中,用作專門的查詢。

 $\times$ 

#### 建立執行個體及資料表

在*Table Store###*建立Table Store執行個體(本次以 華東2 distribute-test 為例),並建立源表 (source\_data)及結果表(result),主鍵為 id (整型),由於Table Store是 schemafree 結構,無 需預先定義其他屬性欄欄位。

以 source\_data 為例,建立如下圖:

创	建	数	据	表
69	Æ	80.	DD.	28

* 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 入失败	<sup>会</sup> 值需要在数据有效版本偏差范围之内,否者将会写
*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 使用哈希或者类似方式使得分片键的值均匀分7	热点问题,充分利用预留读写吞吐量,我们建议您 布,详见最佳实践。
	十 添加表主键	

#### 開啟資料來源表的Stream功能

觸發器功能需要先開啟資料表的*Stream##*,才能在Function Compute中處理寫入Table Store中的 增量資料。

 数据表列表
 文本
 発表

 数据表名称 
 文本
 発表

 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

 Source\_data
 -1
 1
 86400
 关闭

Stream記錄到期時間長度是指通過 StreamAPI 能夠讀取到的增量資料的最長時間。

由於觸發器只能綁定現有的函數,故先到Function Compute的控制台上在同region建立服務及函

數。

 $\times$ 

#### • 建立Function Compute服務

在Function Compute####上建立服務及處理函數,我們繼續使用華東2節點。

1. 在華東2節點建立服務。

新建服务

* 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. 建立函數依次選擇空白函數 > 不建立觸發器。

* 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	~
代码上传方式	● 在线编辑 OSS上传 (代码包上传	○ 文件夹上传



- 函數名稱為:etl\_test,選擇 python2.7 環境,線上編輯代碼
- 函數入口為: etl\_test.handler
- 代碼稍後編輯,點擊下一步。
- 3. 進行服務授權

由於Function Compute需要將運行中的日誌寫入到Log Service中,同時,需要對錶格儲存的 表進行讀寫,故需要對Function Compute進行授權,為方便起見,我們先添加 AliyunOTSF ullAccess 與 AliyunLogFullAccess 許可權,實際生產中,建議根據許可權最小原則來添加許 可權。

权限配置			
	(1) 您可以重新进行	授权操作。	×
	系统角色授权		Ś
		系统模版授权 AliyunOTSFullAccess × AliyunLogFullAccess × ✓	
(	点击授权		

- 4. 單擊授權完成,並建立函數。
- 5. 修改函數代碼。

#### 建立好函數之後,點擊對應的函數一代碼執行,編輯代碼並儲存,其

中,INSTANCE\_NAME(Table Store的執行個體名稱)、REGION(使用的地區)需要根據情況 進行修改:

<	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 十 🖸	代码执行管理
• etl_test	执行 触发事件 ?
< 1/1 >	<ul> <li>● 在线编辑</li> <li>○ OSS上传</li> <li>○ 代码包上传</li> <li>○ 文件夹上传</li> </ul>
	<pre>1 #!/usr/bin/env python 2 # -*- coding: utf-8 -*- 3 import cbor 4 import json 5 import tablestore as ots 配置根据实际情况进行修改 6 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>

使用範例程式碼如下:

```
#!/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']
#由於已經授權了AliyunOTSFullAccess許可權,此處擷取的credentials具有訪問
Table Store的許可權
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>
```

#### 綁定觸發器

 回到Table Store的執行個體管理頁面,單擊表 source\_data 後的使用觸發器按鈕,進入觸發器 綁定介面,單擊使用已有Function Compute,選擇剛建立的服務及函數,勾選Table Store發 送事件通知的許可權,進行確定。

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

创建触发		
* - 华 塚	e_data	Source
ノビゴキ		创建触发器
* 选择	使用已有	新建函数计算
		触发器列表
	長只支持创建-	● 目前一张数据表
		函数计算服务名

2. 綁定成功之後,能夠看到如下的資訊:

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
n source_data			

#### 運行驗證

1. 向 source\_data 表中寫入資料。

單擊source\_data的資料管理頁面,然後單擊插入資料,如圖依次填入id、level及message資 訊。

	插入数据			
< source_	data			
基本详情	主键名称	主键类型	主键值	括入粉捉
数据管理	id	INTEGER	1	
● 表格数据最多	<mark>5显示50</mark> 行			
▶ 监控指标	× 移除所有属性列			
	属性列名称  属	性列类型属性值	数据版本号	操作
	level	NTEGER 🛊 🛛 2		■删除
			或 🗹 使用系统时间	
E	message	TRING 🛊 Test da	ata	■刪除
	+ 増加属性列 注: 園	(性列最多只能自定义插λ;	20列 你已经创建2列	
	<ul> <li>All VH (200 LT &gt; 2)</li> </ul>			
			确定插入	取消

2. 在 result 表中查詢清洗後的資料

單擊result表的資料管理頁面,會查詢到剛寫入到 source\_data 中的資料。

向 soure\_data 寫入level <=1的資料將不會同步到 result 表中。

<	result								
基本详情 数据管理	表格数据					插入数据	查询数据	更新数据	删除数据
触发器管理	●表格数据最多显示	<del>示50行。</del>							
▶ 监控指标		id(主键)		level		message			
		1		2			Test d	ata	
			15	反本 值 类型	型	共有1条, 钅	每页显示: 10条	- «	1 > »
			1	525319981646 2 INT	EGER				

# 2 MaxCompute

# 2.1 使用MaxCompute訪問Table Store

#### 背景資訊

本文介紹如何在同一個雲帳號下實現Table Store和 MaxCompute 之間的無縫串連。

MaxCompute是一項大資料計算服務,它能提供快速、完全託管的 PB 級資料倉儲解決方案,使 您可以經濟並高效地分析處理海量資料。您只需通過一條簡單的 DDL 語句,即可在 MaxCompute 上建立一張外部表格,建立 MaxCompute 表與外部資料源的關聯,提供各種資料的接入和輸出能 力。MaxCompute 表是結構化的資料,而外部表格可以不限於結構化資料。

Table Store與 MaxCompute 都有其自身的類型系統,兩者之間的類型對應關係如下表所示。

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

#### 準備工作

使用 MaxCompute 訪問Table Store前,您需要完成以下準備工作:

- 1. 開通MaxCompute##。
- 2. 建立AccessKey。
- 3. 在 RAM 控制台授權 MaxCompute 訪問Table Store的許可權。

  - 方法二:進行手動授權。步驟如下:
    - **1.** 登入 RAM ###。
    - 2. 在角色管理頁面,建立使用者角色 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拟	期查询 <b>搜索</b>
用户管理		
群组管理	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

3. 在角色詳情頁面,設定策略內容。

<	AliyunODPSDefaultRole				
角色详情		2			
A.4.₩17在10	基本信息	编辑基本信息			
用巴拉仪未晒	角色名称 AliyunODPSDefaultRole	备注			
	创建时间 2017-09-18 14:53:39	Arn a ino			

```
策略內容如下:
```

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

4. 在策略管理頁面,建立授權策略 AliyunODPSRolePolicy。

管理控制台 产品与服务 → Q 搜索 消息 416 费用 T单 支持	简体中文
创建授权策略         2           訪问控制 RAM         STEP 1: 选择权限策器使板           STEP 2: 编辑权限并提交         STEP 3: 新建成功	₿ 刷新
概范 全部模板 ▼ 请输入关键词在下方模板中动态筛选	
田戸管理 空白模板 第 個 AdministratorAccess 管理所有阿里云液源的权限	
策部管理         減約 AliyunOSSFulAccess         減約 AliyunOSSReadOnlyAccess         減約 IliyunOSSReadOnlyAccess         減約 IliyunOSSReadOnlyAccess         減約 IliyunOSSReadOnlyAccess         減約 IliyunOSSReadOnlyAccess         減約 IliyunOSSReadOnlyAccess         10	操作
设置	查看
新設 AliyunRDSFullAccess                和             AliyunRDSReadOnlyAccess                 管理云数超库服务(RDS)的权限	查看
	直看

策略內容如下:

```
{
"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"
}
]
}
--還可自訂其他許可權
```

5. 在角色管理頁面,將 AliyunODPSRolePolicy 許可權授權給 AliyunODPSDefaultRole 角色。

访问控制 RAM	-	2017-06-21 16:06:06	管理   授权   删除
概览	0.000	2017-07-20 16:42:24	管理   授权   删除
用户管理	0.0000000000000000000000000000000000000	2017-05-25 17:52:47	管理   授权   删除
群组管理	diameter in their	2017-07-26 14:57:50	管理   授权   删除
策略管理	-	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

4. 在Table Store控制台#####和#####。

在本樣本中,建立的Table Store執行個體和資料表資訊如下:

- 執行個體名稱: cap1
- 資料表名稱:vehicle\_track
- 主鍵資訊:vid (integer),gt (integer)
- 訪問網域名稱:https://capl.cn-hangzhou.ots-internal.aliyuncs.com



使用 MaxCompute 訪問Table Store時,建議使用Table Store的私網地址。

• 設定執行個體網路類型為允許任意網路訪問。

<	🛧 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 允许任意网络访问 2
	VPC列表

#### 步驟 1. 安裝並配置用戶端

1. 下載 MaxCompute ###並解壓。

<b>C</b>		
	说明	:

確保您的機器上已安裝 JRE 1.7或以上版本。

2. 編輯 conf/odps\_config.ini 檔案,對用戶端進行配置,如下所示:

📔 说明 :

odps\_config.ini 檔案中使用#作為注釋, MaxCompute 用戶端內使用--作為注釋。

3. 運行 bin/odpscmd.bat, 輸入 show tables:。

如果顯示當前 MaxCompute 項目中的表,則表述上述配置正確。

odps@	MCOTStest>show	tables;
ALIYU ALIYU	N\$document@aliyu N\$document@aliyu	un-test.com:bank_data un-test.com:result_table
ок		

步驟 2. 建立外部表格

建立一張 MaxCompute 的資料表(ots\_vehicle\_track) 關聯到 Table Store 的某一張表(vehicle\_track)。

關聯的資料表資訊如下:

- 執行個體名稱: cap1
- 資料表名稱:vehicle\_track
- 主鍵資訊:vid (int); gt (int)

• 訪問網域名稱:https://capl.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://capl.cn-hangzhou.ots-internal.aliyuncs.com'; --
(5)
```

參數說明如下:

標號	參數	說明
(1)	com.aliyun.odps.TableStore StorageHandler	MaxCompute 內建的處 理 Table Store 資料的 StorageHandler,定義了 MaxCompute 和 Table Store 的互動,相關邏輯由 MaxCompute 實現。
(2)	SERDEPROPERITES	可以理解為提供參數選項的 介面,在使用 TableStore StorageHandler 時,有兩 個必須指定的選項,分別是 tablestore.columns.mapping 和 tablestore.table.name。
	tablestore.columns.mapping	必填選項。MaxCompute 將要 訪問的 Table Store 表的列, 包括主鍵和屬性列。 其中, 帶:的表示 Table Store 主鍵, 例如本樣本中的 :vid 與 :gt, 其他均為屬性列。在指定映射 的時候,使用者必須提供指定 Table Store 表的所有主鍵,屬 性列無需全部提供,可以只提 供需要通過 MaxCompute 來訪 問的屬性列。

標號	參數	說明
(4)	tablestore.table.name	需要訪問的 Table Store 表名。 如果指定的 Table Store 表名 錯誤(不存在),則會報錯。 MaxCompute 不會主動建立 Table Store 表。
(5)	LOCATION	指定訪問的 Table Store 的執行 個體資訊,包括執行個體名和 endpoint 等。

#### 步驟 3. 通過外部表格訪問 Table Store 資料

建立外部表格後,Table Store 的資料便引入到了 MaxCompute 生態中,您可通過 MaxCompute SQL 命令來訪問 Table Store 資料。

// 統計編號 4 以下的車輛在時間戳記 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>

返回類似如下結果:

odps@ an ID = 201 Log view http://l 09CTzoxN aGljbGUv Job Queu	Halysis_vehicle>se 170306160538185gsv 7: Logview.odps.aliyu 13A0HZM3M2g2HTC0Nj YaW5zdGFuY2VzLzIwM Heing.	lect vid,count( lupk2 n.com/logview/? ESLDE00Dk0MjExM TcwMzA2MTYwNTM4	*), avg h=http:. izgsey]Ti MTg1Z3N	(speed),avg //service.o dGF0ZW1lbnQ 2bHVwazIiXX	(oil_consi dps.aliyu iOlt7IkFji IdLCJWZXJ:	mption) f 1.com/api8 1GlvbiI6Wy 2aW9uIjoi/	From ots Mp=analys VJVZHB201 MSJ9	vehicle_track where vid 44 and gtt346937387 group by vid; 15. vehicleki-2017036036938385gs/lupi24token-sjhod/2000/7cmink/roshopSrtartiv6Eve5x84887 j1NgUSK-B&Radowifs(tolQeckSrtLCSSZ04007)25369y3hr34682haczodnity62p178kL27v0465212327
	STAGE	S STATUS	TOTAL	COMPLETED	RUNNING	PENDING	BACKUP	
M1_job_0 R2 1 job	э Э. ө	<ul> <li>TERMINATED</li> <li>TERMINATED</li> </ul>			9 8	9 9	0 0	
STAGES: Summary:								
vid								
0   1   2   3	47   47   47   47   47	0.1162258979 0.1120064978 0.0989717631 0.1150391653	6672624 9552555 46085   1605494	+   6.515506   6.567398 6.73873852   6.603890	180530814 300182988 7883797   269275189	5   5   5		

### 2.2 跨帳號授權

本文檔介紹不同帳號之間如何?Table Store和 MaxCompute 之間的無縫串連。如需瞭解同帳號下的 Table Store與 Maxcompute 對接操作,請參考####### MaxCompute ##Table Store。

準備工作

跨雲帳號需要兩個主帳號,帳號 A 將存取權限授予帳號 B,則運行 MaxCompute 時,帳號 B 可以 訪問帳號 A 下的表資料。基本資料如下:

道 说明:

以下資訊僅為樣本,在操作時請替換為實際使用的資訊。

項目	Table Store	MaxCompute
主帳號名	帳號 A	帳號 B

項目	Table Store	MaxCompute
Userld	12345	56789

使用 MaxCompute 跨雲帳號訪問Table Store前,您需要完成以下準備工作:

- 1. 帳號 B 開通MaxCompute ##, 並建立MaxCompute ##。
- 2. 帳號 A 和 B 分別 ## AccessKey。
- 3. 使用帳號 A 登入 RAM ###, 並在角色管理頁面, 建立使用者角色。

在本樣本中,假設建立的角色名稱為 AliyunODPSRoleForOtherUser。

4. 進入該角色, 在角色詳情頁面單擊編輯基本資料, 設定策略內容。 策略內容設定如下:

1	
	"Statement": [
	<pre>"Action": "sts:AssumeRole", "Effect": "Allow", "Principal": {     "Sorwige": [</pre>
	"Service". [ "1xxxx@odps.aliyuncs.com"
}	} } ], "Version": "1"

▋ 说明:

*c* 

請將上述策略內容中的 1xxxx 替換成您的 UID 即可。

5. 角色建立後,您可以在角色詳情頁面查看該角色的 roleArn:

<		
角色详情		
角色授权策略	基本信息	编辑基本信息
	角色名称	备注 人使用此角色来访问您在其他云产品 中的资源
	创建时间 2017-03-30 14:45:00	Arn

6. 返回 RAM 控制台首頁,進入策略管理頁面,建立授權策略。

在本樣本中,假設授權策略名稱為AliyunODPSRolePolicyForOtherUser。

管理控制台	产品与	漏务▼
访问控制 RAM		创建授权策略
		STEP 1:选择权限策略模板
概览		全部模板 ▼ 请输入关键词在
用户管理		3
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1		空白模板
策略管理		系统 AliyunOSSFullAccess
角色管理		管理对象存储服务(OSS)权限
设置	Ę	系统 AliyunECSFullAccess 管理云服务器服务(ECS)的权限
		系统 AliyunRDSFullAccess 管理云数据库服务(RDS)的权限

策略內容如下:

```
"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"
 }
]
}
```

#### 说明:

您也可以自訂其他許可權,如 CreateTable 等。

7. 在角色管理頁面,將 AliyunODPSRolePolicyForOtherUser 許可權授權給

AliyunODPSRoleForOtherUser 角色。

访问控制 RAM	August 100 100 100 100 100 100	2017-06-21 16:06:06	管理   授权   删除
概览	No. and Arch	2017-07-20 16:42:24	管理   授权   删除
用户管理	and the second sec	2017-05-25 17:52:47	管理   授权   删除
群组管理	and the second state	2017-07-26 14:57:50	管理   授权   删除
策略管理	durantic inter-	2017-03-14 17:42:08	管理   授权   删除
角色管理	Algun00PSDefealRole	2017-09-19 14:10:04	管理 授权 删除
设置	No standard and state	2017-07-31 15:33:24	管理   授权   删除

8. 在Table Store控制台#####和#####。

在本樣本中,建立的Table Store執行個體和資料表資訊如下:

- 執行個體名稱: cap1
- 資料表名稱: vehicle\_track
- 主鍵資訊:vid (integer),gt (integer)

• 訪問網域名稱:https://cap1.cn-hangzhou.ots-internal.aliyuncs.com

📕 说明:

使用 MaxCompute 訪問Table Store時,建議使用Table Store的私網地址。

• 設定執行個體網路類型為允許任意網路訪問。

#### 使用 MaxCompute 訪問Table Store

跨帳號訪問的操作與同帳號下的訪問一樣,只是在建立外部表格時使用 roleArn。

帳號 B 通過 MaxCompute 建立外部表格,指定####中建立出來的 roleArn 來訪問Table Store。

具體操作步驟請參考######。其中,在步驟 2 建立外部表格時,使用如下代碼:

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

### 2.3 使用AccessKey訪問Table Store

除了授權方式外,您還可以在 MaxCompute 中使用 AccessKey 訪問Table Store的資料。

準備工作

擷取Table Store資源所屬帳號的AccessKeyId # AccessKeySecret,如果該 AK 是資源所屬帳號的 子帳號,那麼該子帳號至少需要對錶格儲存相關的資源具有以下許可權:

```
{
   "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"
}
]
}
--您也可以自訂其他許可權
```

在 MaxCompute 中使用 AccessKey 訪問Table Store

同授權方式不同的是,需要在建立外表時在LOCATION中顯示寫入 AK 資訊,其格式為:

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

假設需要訪問的Table Store資源的資訊為:

AccessKeyId	AccessKeyS ecret	執行個體名稱	地區	網路模式
abcd	1234	cap1	cn-hangzhou	內網訪問

建立外表的語句為:

```
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@capl.cn-hangzhou.ots-internal.
aliyuncs.com'
```

對資料訪問的操作步驟請參考##MaxCompute##Table Store中的步驟3.通過外部表格訪問 Table Store 資料。

# 2.4 使用 UDF 處理資料

如果您在Table Store裡面的資料有著獨特的結構,希望自訂開發邏輯來處理每一行資料,比如解析 特定的 json 字串,可以使用 UDF(User Defined Function,即使用者自訂函數)來處理。

操作步驟

**1.** 參考*MaxCompute Studio* 文檔,在 IntelliJ 中安裝 MaxCompute-Java/MaxCompute-Studio 外掛 程式。外掛程式安裝完畢,就可以直接開發。

下圖是一個簡單的 UDF 定義,將兩個字串串連。MaxCompute 支援更複雜的 UDF,包括自訂視 窗執行邏輯等,更多資訊請參考##### UDF。

Image:	5	<pre>public class ExtractBusID extends UDF {</pre>
Imr_ut_local_jobs	6	// TODO define parameters and return type, e.g: p
out	7	<b>public</b> String evaluate(String a, String b) {
<ul> <li>src</li> <li>com.aliyun.tablestore.sql</li> </ul>	8	<pre>StringBuilder sb = new StringBuilder();</pre>
C 🔓 ExtractBusID	9	sb. append (a) ;
tail warehouse	10	sb. append (":");
Gloud_metric_udf.iml	11	sb. append(b);
lall cloud_metric_udf.properties	12	<pre>return sb. toString();</pre>
💼 External Libraries	13	}
activation-1.1.jar library root	14	}
antlr4-4.3-complete.jar library root		
antlr4-annotations-4.3.jar library root		

2. 打包之後可以上傳到 MaxCompute。

選擇 File > Project Structure > Artifacts,輸入Name 和 Output directory 後,單擊 + 選擇輸 出模組。打包後通過 ODPS Project Explorer 來上傳資源、建立函數,然後就可以在 SQL 中調 用。

·				
Project Structure				×
(† †	+ -			
Designed Coddinana	# cloud_metric_extract_r	Name: cloud_metric_extract_md5		Type: 🕋 JAR
Project Settings				
Project		Output directory: D:\temp\cloud_n	netric_udf\out\artifacts	\cloud_metric_udf_jar
Modules		Build on make		
Libraries				
Facets		Output Layout Pre-processing Po	st-processing	
Artifacts		🕞 🏢 🕂 📴 Module Output		Available Elements ?
Platform Settings		📕 cloud_me 🖹 File		▼ 🛅 cloud_metric_udf
SDKs		🔁 'cloud 🏝 Directory Content	t	activation-1.1.jar
Global Libraries		🖆 Extracted Directory		antlr-runtime-3.5.2.jar
				antlr4-4.3-complete.jar
Problems				antlr4-annotations-4.3.jar
				antlr4-runtime-4.3.jar
				asm-4.2.jar
				aspectjrt-1.8.2.jar
				bouncycastle.provider-1.38-jdk15.jar
				commons-cli-1.2.jar
				commons-cli-1.3.1.jar
				commons-codec-1.4.jar
				commons-collections-3.2.1.jar
				commons-compress-1.4.jar
				commons-io-2.4.jar
				commons-lang-2.5.jar
				commons-logging-1.1.1.jar
				fastjson-1.2.8.jar
				fluent-core-0.2.0-SNAPSHOT.jar
				fluent-sdk-0.2.0-SNAPSHOT.jar
				III gson-2.2.4 iar
		Show content of elements		

3. 運行 bin/odpscmd.bat。

```
// 我們選出來1行資料,並將name/name傳入UDF,返回兩個string的累加
select cloud_metric_extract_md5(name, name) as udf_test from
test_table limit 1;
```

返回結果如下:

odps@ table_store_sql_engine	e_dev≻select c	loud_me	tric_extrac	t_md5(c,c	;) as udf_	test from	<pre>n cloud_metric_stab</pre>	le limit 1;
ID = 20170302055324953gq1tsa .og view: http://logview.odps.aliyun- 55324953gq1tsau1&token=d214 FjdG1vbi16WyJvZHBzOlJ1VWQiX c3RhbmNlcy8yMDE3MDMwMjAINTMy Dob Queueing QuotaCPUUsage: 99.99% Quo	aul inc.com:8080/l 4cGJKSk9VRWIGQ wwiRWZmZWN0Ijo vNDk1M2dxMXRzY otaMemUsage: 7	ogview/ kNmNXZC iQWxsb3 XUxI119 9.36%	?h=http://s VØJOZWQ4T21 ciLCJSZXNvd XSwiVmVyc21	ervice-co zPSxPRFBT XJjZSI6Wy vbiI6IjEi	orp.odps.a X09CTzoxN JhY3M6b2F .fQ==	aliyun-ino IDEØMDcwMj WwczoqOnBy	c.com/api&p=table_s jYwNjg3NzQ1LDE00Dkw/ yb2plY3RzL3RhYmx1X3I	tore_sql_en Mzg4MDUseyJ NØb3JlX3NxbH
STAGES	STATUS	TOTAL	COMPLETED	RUNNING	PENDING	BACKUP		
11_job_0 R2_1_job_0	TERMINATED TERMINATED	1 1	1 1	0 0	0 0	0 0		
STAGES: 02/02 [======		===>>]	100% ELAPS	ED TIME:	350.08 s			
Summary:								
+   udf_test   +   code4xx1,0.00,netflow,2512	2570.00,qps,29	89.00,p	99RT,95607.	60,code5x	x,0.00,Ma	ixRT,43255	53.00, MinRt, 8.00, Av	gRT,9940.51.

# 2.5 常見問題

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

原因:需要設定 set odps.task.major.version=unstructured\_data。

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

原因:一般情況下是Table Store的 endpoint 填寫錯誤,導致 MaxCompute 無法訪問。

• logview invalid end\_point

原因:在執行過程中,會返回 logview URL 地址,如果使用瀏覽器訪問該地址返回錯誤,可能 是配置不對,請檢查 MaxCompute 配置。

如果仍未解決問題,請####。

# 3 Hive/HadoopMR

### 3.1 環境準備

使用 Hive/HadoopMR 來訪問Table Store中的表

通過*Table Store*及 *E-MapReduce* 官方團隊發布的依賴包,可以直接使用 Hive 及 HadoopMR 來訪 問Table Store中的資料並進行資料分析。

#### 安裝 JDK-7+

- 1. 下載並安裝 JDK-7+ 安裝包。
  - Linux/MacOS 系統:使用系統內建的包管理器安裝
  - Windows 系統:####
- 2. 按照以下樣本進行安裝檢查。

```
$ 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)
```

#### 安裝並啟動 Hadoop 環境

- 1. 下載 2.6.0 版本以上的 Hadoop 安裝包。(####)
- 2. 解壓並安裝,根據實際叢集情況安裝 Hadoop 服務。
- 3. 按照如下樣本啟動 Hadoop 環境。

```
$ bin/start-all.sh
# 檢查服務是否成功啟動
$ jps
24017 NameNode
24835 Jps
24131 DataNode
24438 ResourceManager
5114 HMaster
24287 SecondaryNameNode
24527 NodeManager
```

4. 在 /etc/profile 中添加 Hadoop 路徑, 並執行 source /etc/profile 的命令使配置生

效。

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

#### 下載及安裝 Hive 環境

1. 下載類型為 bin.tar.gz 的 Hive 安裝包。(####)

#### 2. 按照如下樣本解壓安裝包。

```
$ 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. 按照如下樣本初始化 schema。
  - # 進入指定的目錄
  - \$ cd /home/admin/hive-2.1.0/
  - # 初始化,如果是mysql則derby可以直接替換成mysql
  - # 如果執行出錯可以刪除rm -rf metastore\_db/之後重新執行
  - \$ ./bin/schematool -initSchema -dbType derby
- 4. 按照如下樣本啟動 Hive 環境。

```
$ ./bin/hive
# 檢查服務是否成功啟動
hive> show databases;
OK
default
Time taken: 0.207 seconds, Fetched: 1 row(s)
```

#### 下載Table Store的 Java SDK

1. 在 Maven 庫中下載 4.1.0 版本以上的 Java SDK 相關依賴包。(####)

说明:

該依賴包會隨最新的 Java SDK 發布,請根據最新的 Java SDK ##下載相關依賴包。

2. 按照如下樣本將 SDK 拷貝到 Hive 目錄下。

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

#### 下載阿里雲 EMR SDK

#### EMR SDK 依賴包。



# 3.2 使用教程

#### 資料準備

在Table Store中準備一張資料表 pet, name 是唯一的一列主鍵,資料樣本如下:

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	



表格中空白的部分不需要寫入,因為Table Store是一個 schema-free 的儲存結構(####),沒有 值也不需要寫入<sub>NULL</sub>。

#### Hive 訪問樣本

前提條件

按照####準備好 Hadoop、Hive、JDK 環境以及Table Store JAVA SDK 和 EMR SDK 依賴包。

樣本

```
# HADOOP_HOME 及 HADOOP_CLASSPATH 可以添加到 /etc/profile 中
$ export HADOOP_HOME=${你的 Hadoop 安裝目錄}
$ 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",
    "tablestore.access_key_id"="YourAccessKeyId",
    "tablestore.access_key_secret"="YourAccessKeySecret",
    "tablestore.table.name"="pet");
hive> SELECT * FROM pet;
                                                 1995-07-29
Bowser Diane
                                 1979-08-31
                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
                                 1990-08-27
Fanq
       Benny
                dog
                        m
                                                 NULL
Fluffy Harold cat
                        f
                                 1993-02-04
                                                 NULL
```

Diane hamster f 1999-03-30 Puffball NULT snake Slim Benny m 1996-04-29 NULL Whistler Gwen bird NULL 1997-12-09 NULL Time taken: 5.045 seconds, Fetched 9 row(s) hive> 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 m 1996-04-29 NULL Whistler Gwen bird NULL 1997-12-09 NULL Time taken: 1.41 seconds, Fetched 4 row(s)

參數說明如下:

WITH SERDEPROPERTIES

tablestore.columns.mapping(可選):在預設的情況下,外表的欄位名即為Table Store上表的 列名(主鍵列名或屬性列名)。但有時外表的欄位名和表上列名並不一致(比如處理大小寫或字 元集相關的問題),這時候就需要指定 tablestore.columns.mapping。該參數為一個英文逗號分 隔的字串,每一項都是表上列名,順序與外表欄位一致。

空白也會被認為是表上列名的一部分。

- TBLPROPERTIES
  - tablestore.endpoint(必填):訪問Table Store的####,也可以在Table Store控制台上查看 這個執行個體的 endpoint 資訊。
  - tablestore.instance(可選): Table Store的####名。若不填,則為 tablestore.endpoint 的 第一段。
  - tablestore.table.name(必填): Table Store上對應的表名。
  - tablestore.access\_key\_id、tablestore.access\_key\_secret(必填) ,請參見####。
  - tablestore.sts\_token(可選),請參見####。

#### HadoopMR 訪問樣本

以下樣本介紹如何使用 HadoopMR 程式統計資料表 pet 的行數。

#### 範例程式碼

• 構建 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));
    }
}
```

資料來源每從Table Store上讀出一行,都會調用一次 mapper 的 map()。前兩個參數 PrimaryKeyWritable 和 RowWritable 分別對應這行的主鍵以及這行的內容。可以通過調用 PrimaryKeyWritable.getPrimaryKey() 和 RowWritable.getRow() 取得Table Store JAVA SDK 定 義的主鍵對象及行對象。

• 配置Table Store作為 mapper 的資料來源。

```
private static RangeRowQueryCriteria fetchCriteria() {
     RangeRowQueryCriteria res = new RangeRowQueryCriteria("
YourTableName");
     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) 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.setReducerClass(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);
}
```

範例程式碼中使用 job.setInputFormatClass(TableStoreInputFormat.class) 把Table Store設為資 料來源,除此之外,還需要:

- 把 hadoop-connector.jar 部署到叢集上並添加到 classpath 裡面。路徑為 addFileToC lassPath() 指定 hadoop-connector.jar 的本地路徑。代碼中假定 hadoop-connector.jar 在當前 路徑。
- 訪問Table Store需要指定入口和身份。通過 TableStoreInputFormat.setEndpoint() 和 TableStoreInputFormat.setCredential() 設定訪問Table Store需要指定的 endpoint 和 access key 資訊。
- 指定一張表用來計數。

📕 说明 :

- 每調用一次 addCriteria() 可以在資料來源裡添加一個 JAVA SDK 定義的 RangeRowQu eryCriteria 對象。可以多次調用addCriteria()。RangeRowQueryCriteria 對象與Table Store JAVA SDK GetRange 介面所用的 RangeRowQueryCriteria 對象具有相同的限制條件。
- 可以利用 RangeRowQueryCriteria 的 setFilter() 和 addColumnsToGet() 在Table Store的 伺服器端過濾掉不必要的行和列,減少訪問資料的大小,降低成本,提高效能。
- 通過添加對應多張表的多個 RangeRowQueryCriteria,可以實現多表的 union。
- 通過添加同一張表的多個 RangeRowQueryCriteria,可以做到更均匀的切分。TableStore
   -Hadoop Connector 會根據一些策略將使用者傳入的範圍切細。

#### 程式運行樣本

```
$ HADOOP_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

#### 類型轉換說明

Table Store支援的資料類型和 Hive/Spark 支援的資料類型不完全相同。

下表列出了從Table Store的資料類型(行)轉換到 Hive/Spark 資料類型(列)的支援情況。

	TINYINT	SMALLI	INT	BIGINT	FLOAT	DOUBLE	BOOLEA	STRING	BINARY
INTEGE	₹可,損 失精度	可,損 失精度	可, 損失精 度	न्	可,損 失精度	可,損 失精度			
DOUBLE	可,損 失精度	可,損 失精度	可,損 失精度	可,損 失精度	可,損 失精度	可			
BOOLEA	N						可		
STRING								可	
BINARY									可

# 4 Spark/SparkSQL

### 4.1 環境準備

使用 Saprk/Spark SQL 來查詢和連結資料表格儲存中的表

通過*Table Store*及 *E-MapReduce* 官方團隊發布的依賴包,可以直接使用 Spark 及 Spark SQL 來 訪問Table Store中的資料並進行資料的查詢分析。

#### 下載及安裝 Spark/Spark SQL

- 1. 下載版本號碼為 1.6.2 的 Spark 安裝包,安裝包類型為 Pre-built for Hadoop 2.6。(####)
- 2. 按照如下樣本解壓安裝包。

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

#### 安裝 JDK-7+

- 1. 下載並安裝 JDK-7+ 安裝包。
  - Linux/MacOS 系統:請用系統內建的包管理器進行安裝
  - Windows 系統:####
- 2. 按照如下樣本進行安裝檢查。

```
$ 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)
```

#### 下載Table Store的 Java SDK

1. 在 Maven 庫中下載 4.1.0 版本以上的 Java SDK 相關依賴包。(####)

```
📕 说明 :
```

該依賴包會隨最新的 Java SDK 發布,請根據最新的 Java SDK ##下載相關依賴包。

2. 按照如下樣本將 SDK 拷貝到 Spark 目錄下。

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

#### 下載阿里雲 EMR SDK

```
下載 EMR SDK 相關的依賴包。(####)
```



瞭解更多 EMR 資訊請參見##。

#### 啟動 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
```

# 4.2 使用教程

#### 資料準備

在Table Store中準備一張資料表 pet,其中name是唯一的一列主鍵。資料樣本如下:

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寫入,因為Table Store是一個 schema-free 的儲存結構(####),沒有 值也不需要寫入NULL。

#### Spark SQL 訪問樣本

前提條件

按照####中的步驟準備好 Spark、JDK 環境以及Table Store Java SDK 和 EMR SDK 的依賴包。

樣本

```
$ 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
               dog
                     m
                              1979-08-31
                                             1995-07-29
       Harold dog
                              1989-05-13
Buffy
                      f
                                             NULL
                      f
Chirpy Gwen
              bird
                              1998-09-11
                                             NULL
Claws
                              1994-03-17
       Gwen
               cat
                      m
                                             NULL
       Benny
                              1990-08-27
Fang
               dog
                      m
                                             NULL
Fluffy Harold cat
                      f
                              1993-02-04
                                             NULL
Puffball
                      hamster f
                                      1999-03-30
                                                     NULL
              Diane
                              1996-04-29
                                             NULL
Slim Benny
              snake
                      m
                             NULL 1997-12-09
Whistler
              Gwen
                      bird
                                                     NULL
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
                                      1999-03-30
Puffball
               Diane
                      hamster f
                                                     NULL
Slim Benny
                              1996-04-29
              snake
                                             NULL
                      m
Whistler
               Gwen
                      bird
                             NULL
                                     1997-12-09
                                                     NULL
Time taken: 1.41 seconds, Fetched 4 row(s)
```

參數說明如下:

- WITH SERDEPROPERTIES
  - tablestore.columns.mapping(可選):在預設情況下,外表的欄位名即為Table Store上表的 列名(主鍵列名或屬性列名)。但有時外表的欄位名和表上列名並不一致(比如處理大小寫 或字元集相關的問題),這時候就需要指定 tablestore.columns.mapping。該參數為一個英 文逗號分隔的字串,每個分隔之間不能添加空格,每一項都是表上列名,順序與外表欄位一 致。

```
📋 说明 :
```

Table Store的列名支援空白字元,所以空白也會被認為是表上列名的一部分。

- TBLPROPERTIES
  - tablestore.endpoint(必填):訪問Table Store的####,也可以在Table Store控制台上查看
     這個執行個體的 endpoint 資訊。
  - tablestore.instance(可選): Table Store的####名。若不填,則為 tablestore.endpoint 的 第一段。
  - tablestore.table.name(必填):Table Store上對應的表名。
  - tablestore.access\_key\_id、tablestore.access\_key\_secret(必填) ,請參見####。

- tablestore.sts\_token(可選),請參見####。

#### Spark 訪問樣本

以下樣本介紹如何使用 Spark 程式統計資料表 pet 的行數。

```
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 {
        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();
    }
}
```

如果使用 scala,只需把 JavaSparkContext 換成 SparkContext,JavaPairRDD 換成 PairRDD 即

#### 可。或者更簡單,交給編譯器自行做類型推斷

#### 運行程式

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

#### 類型轉換說明

Table Store支援的資料類型和 Hive/Spark 支援的資料類型不完全相同。

下表列出了從Table Store的資料類型(行)轉換到 Hive/Spark 資料類型(列)時所支援的情況。

	TINYINT	SMALLI	INT	BIGINT	FLOAT	DOUBLE	BOOLEA	STRING	BINARY
INTEGE	R可,損 失精度	可,損 失精度	可, 損失精 度	可	可,損 失精度	可,損 失精度			
DOUBLE	可,損 失精度	可,損 失精度	可,損 失精度	可,損 失精度	可,損 失精度	可			
BOOLEA	N						可		
STRING								可	
BINARY									可