PG外部表应用

mysql_fdw, oss_fdw

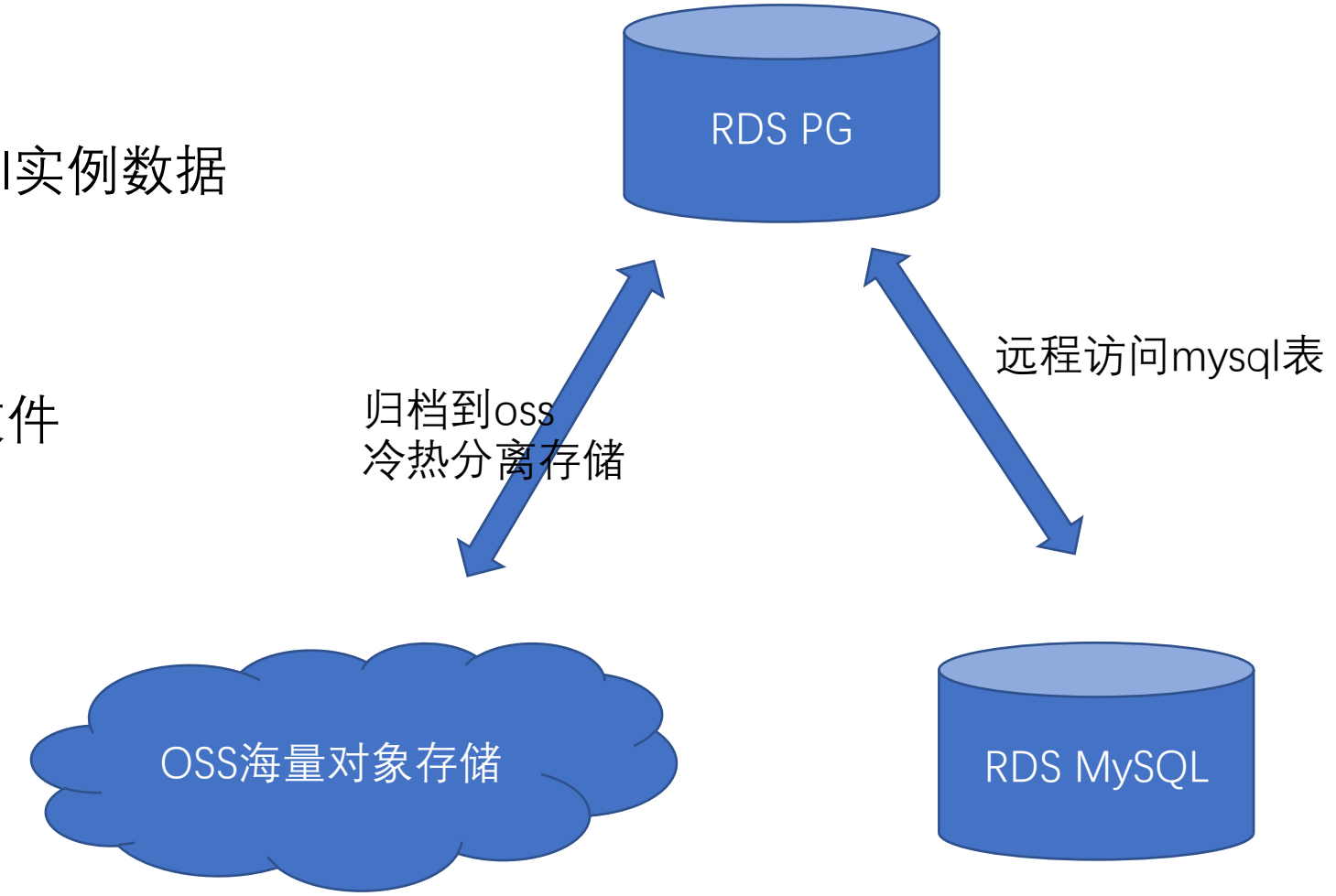
（直接读写MySQL数据，冷热分离、外部归档表）

阿里云

digoal

目录

- mysql_fdw
  - 从外部表直接读写mysql实例数据
- oss_fdw
  - 冷热分离
  - 归档历史数据到oss
  - 从外部表直接读写oss文件



mysql_fdw

- server(host:port)
  - mysql的网络配置
- user mapping(user:pwd)
  - pg数据库user如何连接到以上mysql server（mysql的用户密码）
- foreign table(dbname,tablename,column define)
  - mysql表结构如何映射到pg外部表

环境

- RDS MySQL
- RDS PG 12
  - 相同vpc , vswitch
- pg实例可以通过vpc内网访问mysql server
- mysql数据，使用第5课的测试表和数据

使用第5课生成数据

- test_mm
- test_innodb

创建插件、mysql server、映射用户密码

使用rds_superuser创建 extension

create extension mysql_fdw ;

创建 mysql server

CREATE SERVER **mysql_1**

  FOREIGN DATA WRAPPER mysql_fdw

  OPTIONS (host '**mysql网络地址**', port '**mysql端口**');

设置 pg数据库用户digoal的user mapping

CREATE USER MAPPING FOR **digoal**

  SERVER **mysql_1**

  OPTIONS (username **'mysqluser'**, password **'mysqluserpwd'**);

方法1，单独建外部表

```
CREATE FOREIGN TABLE t1(
id INT8 NOT NULL,
user_id VARCHAR (20) NOT NULL,
group_id INT8 NOT NULL,
create_time timestamp NOT NULL
)
SERVER mysql_1
    OPTIONS (dbname 'db1', table_name 'test_mm');

db1=> select count(*) from t1;
  count
---------
 1000000
(1 row)
```

方法2，一次性import所有外部表

```
db1=> create schema ft;
CREATE SCHEMA
db1=> import foreign schema db1 from server mysql_1 into ft;
IMPORT FOREIGN SCHEMA
db1=> \det ft.*
     List of foreign tables
 Schema |    Table    | Server
--------+-------------+---------
 ft     | test_innodb | mysql_1
 ft     | test_mm     | mysql_1
(2 rows)
db1=> select count(*) from ft.test_mm;
  count
---------
 1000000
(1 row)
```

查询、观察远程sql

```
db1=> select * from t1 where id=2;
 id |       user_id        | group_id |     create_time
----+----------------------+----------+---------------------
  2 | qGZKzux5FVrfN5RUN4QO |       19 | 2020-01-08 15:07:21
(1 row)

db1=> explain verbose select * from t1 where id=2;
                                  QUERY PLAN
-------------------------------------------------------------------------------------------------------
 Foreign Scan on digoal.t1  (cost=25.00..1025.00 rows=1000 width=82)
   Output: id, user_id, group_id, create_time
   Remote server startup cost: 25
   Remote query: SELECT `id`, `user_id`, `group_id`, `create_time` FROM `db1`.`test_mm` WHERE ((`id` = 2))
(4 rows)
```

条件没有下推，会导致传输所有记录

```
db1=> explain verbose select * from t1 limit 100;
                                QUERY PLAN
-------------------------------------------------------------------------------------------
 Limit  (cost=25.00..125.00 rows=100 width=82)
   Output: id, user_id, group_id, create_time
   ->  Foreign Scan on digoal.t1  (cost=25.00..1025.00 rows=1000 width=82)
         Output: id, user_id, group_id, create_time
         Remote server startup cost: 25
         Remote query: SELECT `id`, `user_id`, `group_id`, `create_time` FROM `db1`.`test_mm`
(6 rows)
```

插入远程mysql

```
db1=> explain verbose insert into t1 (user_id, group_id, create_time) values ('digoal', 123, now());
                                QUERY PLAN
-------------------------------------------------------------------------------
 Insert on digoal.t1  (cost=0.00..0.01 rows=1 width=82)
   ->  Result  (cost=0.00..0.01 rows=1 width=82)
         Output: NULL::bigint, 'digoal'::character varying(20), '123'::bigint, now()
(3 rows)

db1=> insert into t1 (user_id, group_id, create_time) values ('digoal', 123, now());
INSERT 0 1
db1=> select * from t1 where user_id='digoal';
   id    | user_id | group_id |     create_time
---------+---------+----------+---------------------
 1000003 | digoal  |      123 | 2020-01-08 16:11:05
(1 row)
```

更新远程mysql

```
db1=> explain verbose update t1 set group_id=321 where id=1000003;
                                        QUERY PLAN
-----------------------------------------------------------------------------------------------------------------
 Update on digoal.t1  (cost=25.00..1025.00 rows=1000 width=90)
   ->  Foreign Scan on digoal.t1  (cost=25.00..1025.00 rows=1000 width=90)
         Output: id, user_id, '321'::bigint, create_time, id
         Remote server startup cost: 25
         Remote query: SELECT `id`, `user_id`, `create_time` FROM `db1`.`test_mm` WHERE ((`id` = 1000003)) FOR UPDATE
(5 rows)

db1=> update t1 set group_id=321 where id=1000003;
UPDATE 1
db1=> select * from t1 where user_id='digoal';
   id    | user_id | group_id |     create_time
---------+---------+----------+---------------------
 1000003 | digoal  |      321 | 2020-01-08 16:11:05
(1 row)
```

删除远程mysql记录

```
db1=> explain verbose delete from t1 where id=1000003;
                                QUERY PLAN
-------------------------------------------------------------------------------------------
 Delete on digoal.t1  (cost=25.00..1025.00 rows=1000 width=8)
   ->  Foreign Scan on digoal.t1  (cost=25.00..1025.00 rows=1000 width=8)
         Output: id
         Remote server startup cost: 25
         Remote query: SELECT `id` FROM `db1`.`test_mm` WHERE ((`id` = 1000003)) FOR UPDATE
(5 rows)

db1=> delete from t1 where id=1000003;
DELETE 1
db1=> select * from t1 where user_id='digoal';
 id | user_id | group_id | create_time
----+---------+----------+-------------
(0 rows)
```

注意

- 检查explain verbose，如果没有push down，需要返回全表数据到pg

查询有哪些外部表、server、用户映射

```
db1=> select * from pg_foreign_server ;
  oid  | srvname | srvowner | srvfdw | srvtype | srvversion | srvacl |                        srvoptions
-------+---------+----------+--------+---------+------------+--------+-----------------------------------------------------------
 16561 | mysql_1 |    16385 |  16559 |         |            |        | {host=rm-xxxx.mysql.rds.aliyuncs.com,port=3306}
(1 row)

db1=> \h alter server
Command:     ALTER SERVER
Description: change the definition of a foreign server
Syntax:
ALTER SERVER name [ VERSION 'new_version' ]
    [ OPTIONS ( [ ADD | SET | DROP ] option ['value'] [, ... ] ) ]
ALTER SERVER name OWNER TO { new_owner | CURRENT_USER | SESSION_USER }
ALTER SERVER name RENAME TO new_name

URL: https://www.postgresql.org/docs/12/sql-alterserver.html
```

查询有哪些外部表、server、用户映射

```
db1=> \des
        List of foreign servers
   Name    | Owner  | Foreign-data wrapper
-----------+--------+----------------------
 mysql_1    | digoal | mysql_fdw
 ossserver1 | digoal | oss_fdw
(2 rows)

db1=> \deu
List of user mappings
 Server  | User name
---------+-----------
 mysql_1 | digoal
(1 row)
```

```
db1=> \det
    List of foreign tables
 Schema |  Table  |   Server
--------+---------+------------
 public | oss_tb1 | ossserver1
 public | oss_tb2 | ossserver1
 public | test_mm | mysql_1
(3 rows)
```

归档数据、冷热分离

- oss_fdw
  - 目前仅支持PG V10，未来会覆盖所有主流版本（11,12等）

申请有oss访问权限的AK

申请有oss访问权限的AK

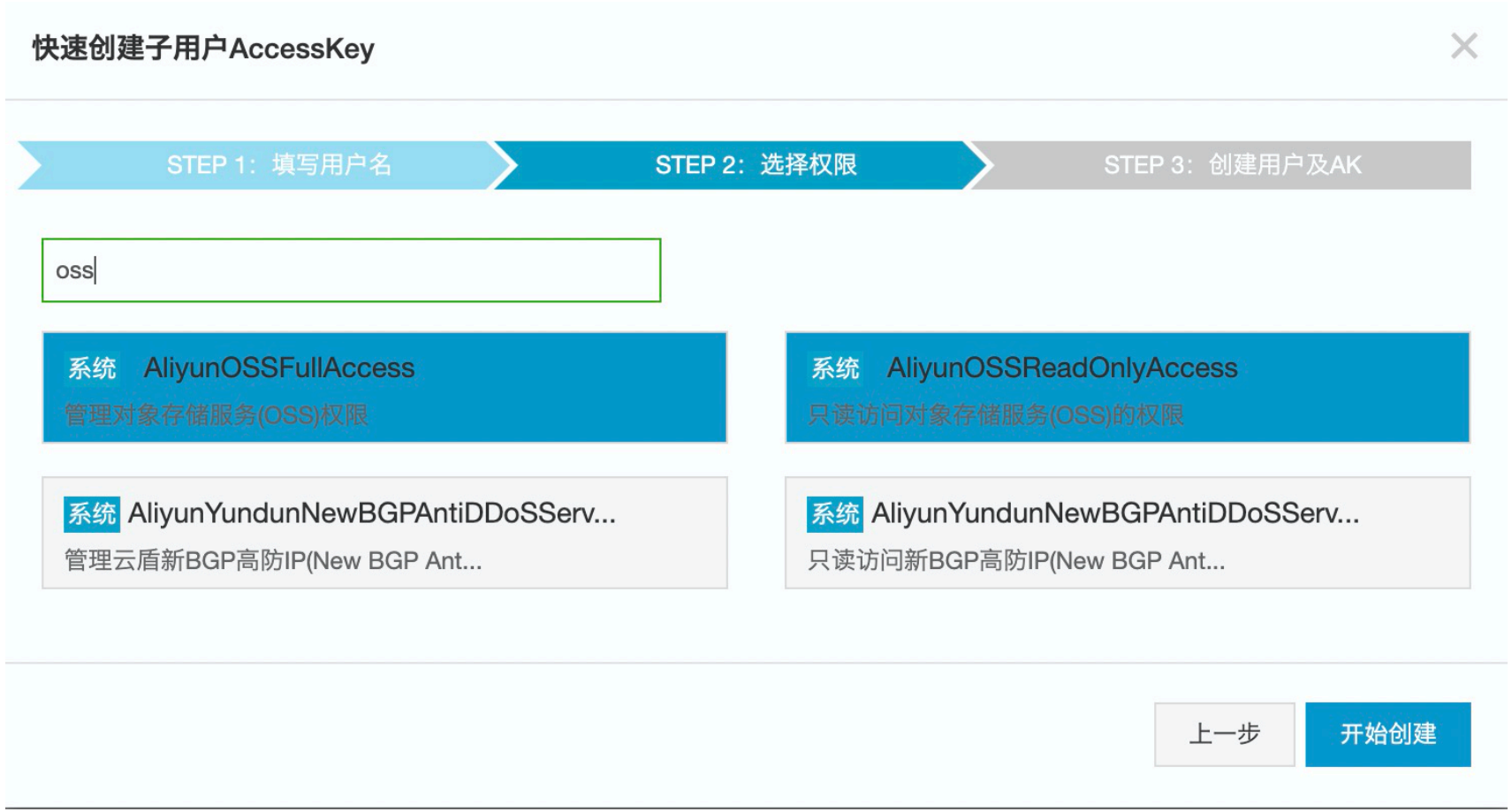
快速创建子用户AccessKey

STEP 1：填写用户名 | STEP 2：选择权限 | STEP 3：创建用户及AK

* 用户名：pg_oss_fdw_user

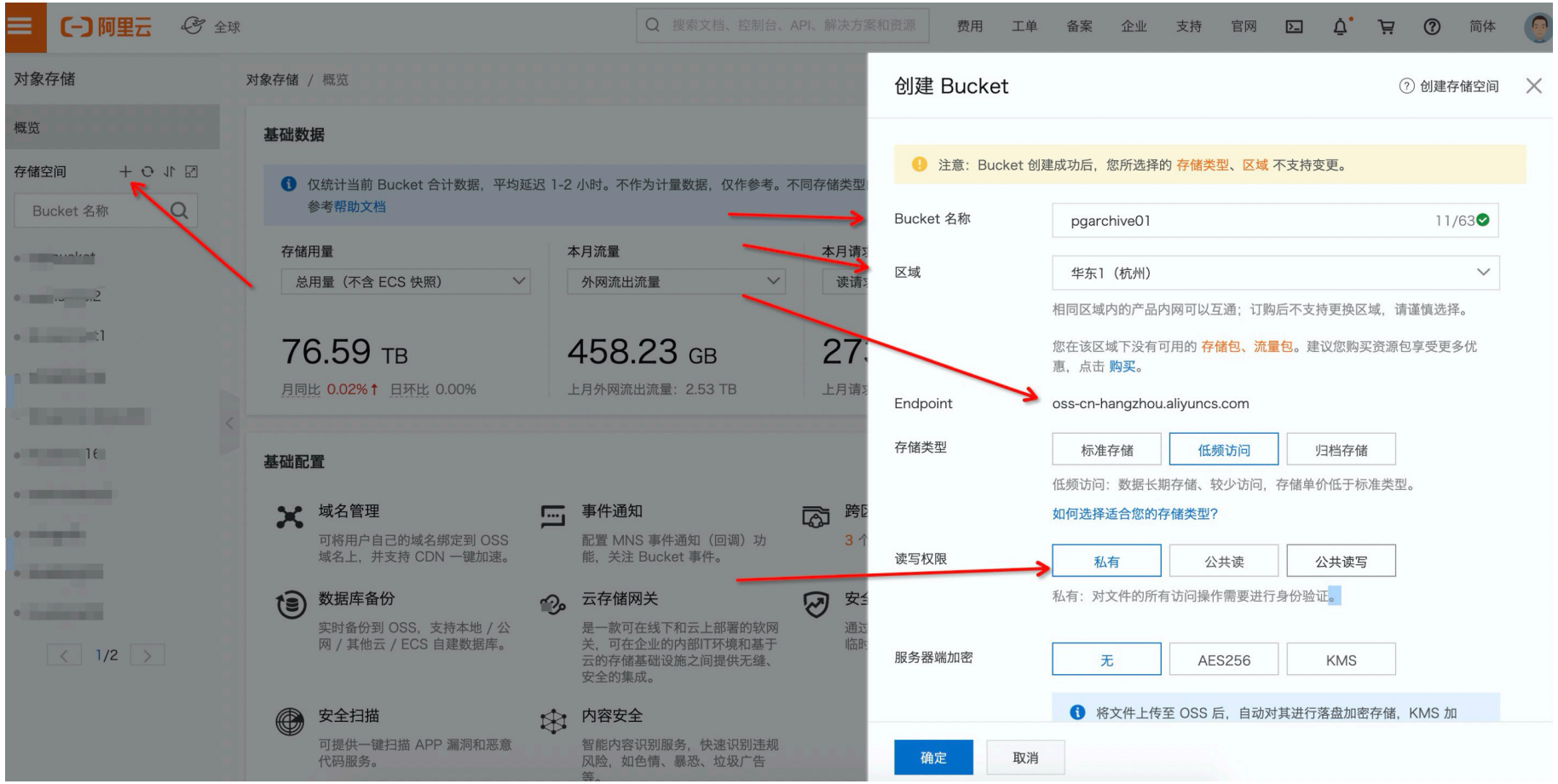
长度1-64个字符，允许输入大小写英文字母、数字、"."、"_"或"-"

下一步


快速创建子用户AccessKey
STEP 1：填写用户名
STEP 2：选择权限
STEP 3：创建用户及AK
oss
系统 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
系统 AliyunYundunNewBGPAntiDDoSServ...
管理云盾新BGP高防IP(New BGP Ant...
系统 AliyunYundunNewBGPAntiDDoSServ...
只读访问新BGP高防IP(New BGP Ant...
上一步
开始创建


快速创建子用户AccessKey
STEP 1：填写用户名
STEP 2：选择权限
STEP 3：创建用户及AK
这是AccessKey可供下载的唯一机会，请及时保存！
用户 pg_oss_fdw_user 创建成功
您已成功新建用户和AccessKey，并且添加完成授权。
AccessKey详情
AccessKeyID:
AccessKeySecret:
查看用户详情
下载AccessKey

申请oss bucket



https://oss.console.aliyun.com/

服务器端加密

无 | AES256 | KMS

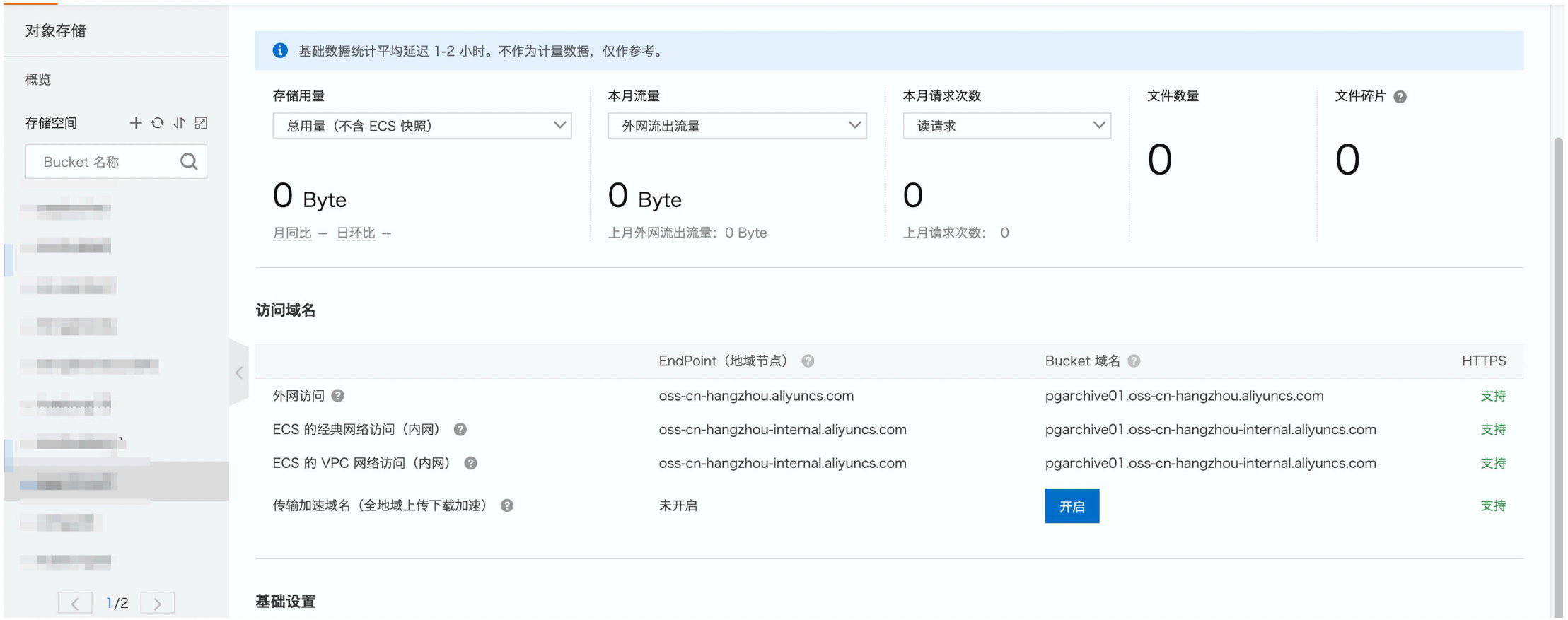
将文件上传至 OSS 后，自动对其进行落盘加密存储，KMS 加密方式需要进行权限设置，当前 KMS 仅支持 OSS 默认托管的 CMK，如需试用 KMS 单独创建的 CMK 加密（BYOK），请和我们联系，了解 更多服务器端加密指南。

实时日志查询

开通 | 不开通

OSS 与日志服务深度结合，免费提供最近 7 天内的 OSS 实时日志查询。开通该功能后，您可对 Bucket 的访问记录进行实时查询分析， 了解详情。

确定 取消



oss_fdw 采用ecs的vpc网络访问（内网） EndPoint

创建oss_fdw， 创建server

PostgreSQL 创建插件

create extension oss_fdw;

创建 server

CREATE SERVER **ossserver1** FOREIGN DATA WRAPPER oss_fdw OPTIONS

(host '**oss-cn-hangzhou-internal.aliyuncs.com**' , id '**accessid内容**', key '**accesskey内容**', bucket '**pgarchive01**');

创建oss外部归档表

```
# 创建 oss 外部表，压缩
CREATE FOREIGN TABLE oss_tb1
    (date text, time text, open float,
     high float, low float, volume int)
     SERVER ossserver1
     OPTIONS ( dir 'oss_tb1/', delimiter ',' ,
         format 'csv', encoding 'utf8', compressiontype 'gzip' );

# 不压缩
CREATE FOREIGN TABLE oss_tb2
    (date text, time text, open float,
     high float, low float, volume int)
     SERVER ossserver1
     OPTIONS ( dir 'oss_tb2/', delimiter ',' ,
         format 'csv', encoding 'utf8', compressiontype 'none' );
```

创建本地表，写入测试数据

```
# 创建表，数据就装载到这张表中
create table tbl1
(date text, time text, open float,
    high float, low float, volume int) ;

insert into tbl1 select md5(random()::text), now()::text,
random()*10000, random()*10000000, random()*10000,
random()*100000000
from generate_series(1,1000000);
```

将本地表数据写入外部表

数据从 ossexample 装载到 example 中。

db1=> insert into oss_tb1 select * from tbl1;

WARNING: oss compress process does not close

WARNING: oss compres thread does not close

NOTICE: begin writiing data to oss location oss_tb1/, with block size 10 MB and oss file size 1024 MB

INSERT 0 1000000

Time: 8899.910 ms (00:08.900)

查询外部表

```
db1=> select * from oss_tb1 limit 1;
NOTICE:  a total of 1 files will be loaded, begin oss_tb1/_oss_tb1_631787927722928, end oss_tb1/_oss_tb1_631787927722928, file format gzip
                date                |            time            |      open      |      high      |      low       | volume
------------------------------------+----------------------------+----------------+----------------+----------------+---------
-
 ee421482662f0424dfb337925b35f029 | 2020-01-08 16:35:07.795619+08 | 4588.77929486334 | 3359252.90361047 | 3289.49342481792 | 47414201
(1 row)
Time: 198.820 ms

db1=> select count(*) from oss_tb1;
NOTICE:  a total of 1 files will be loaded, begin oss_tb1/_oss_tb1_631787927722928, end oss_tb1/_oss_tb1_631787927722928, file format gzip
  count
---------
 1000000
(1 row)
Time: 5873.628 ms (00:05.874)
```

查询外部表

db1=> insert into oss_tb1 select * from tbl1;

NOTICE: begin writiing data to oss location oss_tb1/, with block size 10 MB and oss file size 1024 MB

INSERT 0 1000000

Time: 8720.626 ms (00:08.721)

db1=> select count(*) from oss_tb1;

NOTICE: a total of 2 files will be loaded, begin oss_tb1/_oss_tb1_631787927722928, end oss_tb1/_oss_tb1_631788054925016, file format gzip

  count

 2000000

(1 row)

Time: 12469.130 ms (00:12.469)

查询外部表

```
db1=> insert into oss_tb2 select * from tbl1;
NOTICE:  begin writiing data to oss location oss_tb2/, with block size 10 MB and oss file size 1024 MB
INSERT 0 1000000
Time: 7120.737 ms (00:07.121)
db1=> insert into oss_tb2 select * from tbl1;
NOTICE:  begin writiing data to oss location oss_tb2/, with block size 10 MB and oss file size 1024 MB
INSERT 0 1000000
Time: 7004.067 ms (00:07.004)
db1=> select count(*) from oss_tb2;
NOTICE:  a total of 2 files will be loaded, begin oss_tb2/_oss_tb2_631788259206276, end oss_tb2/_oss_tb2_631788267353423, file format text
  count
---------
 2000000
(1 row)

Time: 21546.740 ms (00:21.547)
```

压缩与不压缩的存储空间区别

```
db1=> select * from oss_fdw_list_file('oss_tb1','public');
              name               |   size
---------------------------------+----------
 oss_tb1/_oss_tb1_631787927722928 | 53922166
 oss_tb1/_oss_tb1_631788054925016 | 53922166
(2 rows)

db1=> select * from oss_fdw_list_file('oss_tb2','public');
              name               |   size
---------------------------------+-----------
 oss_tb2/_oss_tb2_631788259206276 | 122555480
 oss_tb2/_oss_tb2_631788267353423 | 122555480
(2 rows)
```

归档数据不支持记录级别的更新和删除

```
db1=> update oss_tb1 set date=now() where volume=1;
ERROR:  cannot update foreign table "oss_tb1"
db1=> delete from oss_tb1 where volume=1;
ERROR:  cannot delete from foreign table "oss_tb1"
db1=> delete from oss_tb2 where volume=1;
ERROR:  cannot delete from foreign table "oss_tb2"
db1=> update oss_tb2 set date=now() where volume=1;
ERROR:  cannot update foreign table "oss_tb2"
```

扩展资料

- 阿里云RDS PostgreSQL OSS 外部表实践 - (dblink异步调用封装并行) 从OSS并行导入数据
- https://github.com/digoal/blog/blob/master/201804/20180427_01.md
- 阿里云RDS PostgreSQL OSS 外部表实践 - (dblink异步调用封装并行) 数据并行导出到OSS
- https://github.com/digoal/blog/blob/master/201709/20170906_01.md
- 强制数据分布与导出prefix - 阿里云pg, hdb pg oss快速数据规整外部表导出实践案例
- https://github.com/digoal/blog/blob/master/201801/20180109_01.md

注意事项

- https://help.aliyun.com/document_detail/44461.html
  - filepath, dir, prefix 参数互斥
  - 有oss写入要求时，必须使用dir或prefix
  - 有oss写入要求时，不能使用parse_errors

参考资料

- mysql_fdw手册
  - https://github.com/EnterpriseDB/mysql_fdw
- oss_fdw手册
  - https://help.aliyun.com/document_detail/44461.html
- MySQL手册
  - https://www.mysqltutorial.org/
  - https://dev.mysql.com/doc/refman/8.0/en/
- PG 管理、开发规范
  - https://github.com/digoal/blog/blob/master/201609/20160926_01.md
- PG手册
  - https://www.postgresql.org/docs/current/index.html
  - https://www.postgresqltutorial.com/postgresql-tutorial/postgresql-vs-mysql/
- GIS手册
  - http://postgis.net/docs/manual-3.0/

一期开课计划(PG+MySQL联合方案)

- - 2019.12.30 19:30 RDS PG产品概览，如何与MySQL结合使用
- - 2019.12.31 19:30 如何连接PG，GUI，CLI的使用
- - 2020.1.3 19:30 如何压测PG数据库、如何瞬间构造海量测试数据
- - 2020.1.6 19:30 MySQL与PG对比学习(面向开发者)
- - 2020.1.7 19:30 如何将MySQL数据同步到PG（DTS）
- - 2020.1.8 19:30 PG外部表妙用 - mysql_fdw, oss_fdw（直接读写MySQL数据、冷热分离）
- - 2020.1.9 19:30 PG应用场景介绍 - 并行计算，实时分析
- - 2020.1.10 19:30 PG应用场景介绍 - GIS
- - 2020.1.13 19:30 PG应用场景介绍 - 用户画像、实时营销系统
- - 2020.1.14 19:30 PG应用场景介绍 - 多维搜索
- - 2020.1.15 19:30 PG应用场景介绍 - 向量计算、图像搜索
- - 2020.1.16 19:30 PG应用场景介绍 - 全文检索、模糊查询
- - 2020.1.17 19:30 PG 数据分析语法介绍
- - 2020.1.18 19:30 PG 更多功能了解：扩展语法、索引、类型、存储过程与函数。如何加入PG技术社群

本课程习题

- 如何一次性创建所有mysql的表作为pg的外部表
- 可以删除mysql外部表的数据吗
- 如何查看访问外部表的远程SQL
- 访问频次非常非常低的归档数据，使用什么方法存储更加廉价
- oss外部归档数据支持记录级别的删除和更新吗
- 如何提升归档数据写入oss的速度
- 如何提升查询的oss归档表速度

技术社群

PG技术交流钉钉群(3500+人)