目录

1. 概述

1.1. 目的

本文旨在指导如何安装、使用 chameleon 工具完成从 MySQL 数据库迁移到 openGauss。

1.2. chameleon 介绍

chameleon 是一个用 Python 3 编写的 MySQL 到 openGauss 的实时复制工具。工具使用 mysql-replication 库从 MySQL 中提取 row images，这些 row images 将以 jsonb 格式被存储到 openGauss 中。在 openGauss 中会执行一个 pl/pgsql 函数，解码 jsonb 并将更改重演到 openGauss。同时，工具通过一次初始化配置，使用只读模式，将 MySQL 的全量数据拉取到 openGauss，使得该工具提供了初始全量数据的复制以及后续增量数据的实时在线复制功能。

1.3. 注意事项

1.3.1. 一般性限制

- 根据 mysql-replication 的要求，Python 3 仅支持 3.5~3.7。
- detach 复制副本进程将重置 openGauss 中的序列，以使数据库独立工作。外键约束也会在 detach 过程中被创建和验证。
- 视图、自定义函数、存储过程、trigger、自定义 type 等不会被迁移。

- 列的 comment 不会被迁移。
- 对于分区表，openGauss 无法支持的分区表类型将被迁移成普通表。
- 配置文件中，schema mappings 中指定的 openGauss 侧的目的 schema 名称不能是 sch_chameleon。sch_chameleon 是工具将自行创建用于辅助复制的 schema 名称。
- 列默认值问题。由于列的默认值可以是表达式，部分 MySQL 的表达式若 openGauss 不支持的话，离线迁移过程中会报错，导致迁移失败。
- MySQL 的 unsigned 数据类型迁移时，会自动去掉 unsigned 属性，如 MySQL 的 unsigned int 迁移到 openGauss 时将变成 int，若 MySQL 中存储的数据超过了 int 的取值范围，迁移过程中会出错。
- 工具支持的 MySQL 版本为 5.5+，openGauss 的版本为 2.0.1+。
- 对于 float、double 等浮点数，迁移过程中可能由于精度误差，造成 MySQL 和 openGauss 中的值不完全一样。

1.3.2. 在线迁移限制

- 在线 DDL 仅支持部分语句，主要包括 CREATE/DROP/RENAME/TRUNCATE TABLE, ALTER TABLE DROP/ADD/CHANGE/MODIFY, DROP PRIMARY KEY。在线 CREATE TABLE 仅支持创建普通表且列的默认值不会被迁移。

2. chameleon 安装方法

可直接下载 whl 安装包或者通过源码安装。

2.1. whl 安装

安装包下载地址：
https://opengauss.obs.cn-south-1.myhuaweicloud.com/latest/chameleon/chameleon-1.0.0-py3-none-any.whl
下载完成后，通过 python virtual env 环境进行安装，首先创建 python 虚拟环境并激活：

```
python3 -m venv venv
source venv/bin/activate
```

然后通过 pip 安装即可：

```
pip3 install ./chameleon-1.0.0-py3-none-any.whl
```

注意：安装过程中，将自动安装该工具依赖的其他库，请确保本机的 pip 能正常下载安装相关依赖。相关依赖库及版本要求为：

```
PyMySQL>=0.10.0, <1.0.0
argparse>=1.2.1
mysql-replication>=0.22
py-opengauss>=1.3.1
PyYAML>=5.1.2
tabulate>=0.8.1
```

daemonize>=2.4.7

rollbar>=0.13.17

geomet>=0.3.0

2.2. 源码安装

通过 git 下载源码： ***git clone git@gitee.com:opengauss/openGauss-tools-chameleon.git***
下载完成后，同样需要先创建 python 虚拟环境并激活：

python3 -m venv venv

source venv/bin/activate

然后进入代码的目录，执行 python install 命令安装：

cd openGauss-tools-chameleon

python3 setup.py install

安装完成后，不要退出 python 虚拟环境，可以开始使用 chameleon 工具。

3. chameleon 配置文件说明

完整的配置文件如下所示：

```
---
# global settings
pid_dir: '~/.pg_chameleon/pid/'
log_dir: '~/.pg_chameleon/logs/'
log_dest: file
log_level: info
log_days_keep: 10
rollbar_key: ''
rollbar_env: ''

type_override:
    "tinyint(1)":
        override_to: boolean
        override_tables:
            - "*"
    "float(5,2)":
        override_to: float4
        override_tables:
            - "*"

# postgres  destination connection
pg_conn:
    host: "localhost"
```

```
    port: "5432"
    user: "usr_test"
    password: "test"
    database: "db_test"
    charset: "utf8"

sources:
    mysql:
        db_conn:
            host: "localhost"
            port: "3306"
            user: "usr_test"
            password: "test"
            charset: 'utf8'
            connect_timeout: 10
        schema_mappings:
            sakila: my_sakila
        limit_tables:
        skip_tables:
        grant_select_to:
        lock_timeout: "120s"
        my_server_id: 100
        replica_batch_size: 10000
        replay_max_rows: 10000
        batch_retention: '1 day'
        copy_max_memory: "300M"
        copy_mode: 'file'
        out_dir: /tmp
        sleep_loop: 1
        type: mysql
```

配置文件使用 yaml 文件规则配置，需要特别注意对齐，缩进表示层级关系，缩进时不允许使用 Tab 键，只允许使用空格，缩进的空格数目不重要，但相同层级的元素左侧需要对齐。

3.1. 全局设置

3.1.1. pid_dir

进程 pid 存储的路径。init_replica 和 start_replica 阶段，如果工具在后台运行，进程的 pid 将保存在改文件夹下。

3.1.2. log_dir

进程运行过程中，log 存储的路径。

3.1.3. log_dest

log 打印的目标。stdout 表示将 log 信息输出到屏幕, file 表示将 log 信息输出到文件，文件的位置由 log_dir 决定。

3.1.4. log_level

log 等级。有效值为 debug, info, warning, error, critical.

3.1.5. log_days_keep

log 文件保留时间，单位为天。

3.1.6. rollbar_key

可选项。工具可配合 rollbar(https://rollbar.com/)联合使用。如果在 rollbar 官网注册了账号，可将对应账号的 POST_SERVER_ITEM_ACCESS_TOKEN 填入 rollbar_key。

3.1.7. rollbar_env

可选项。用于表示 rollbar 环境，与 rollbar_key 配合使用。若同时配置了 rollbar_key 和 rollbar_env，工具执行阶段的部分消息将被发送到 rollbar，可在 rollbar 官网登录自己的账号后看到相关消息。

3.2. 类型重载规则

type_override，允许用户覆盖默认类型转换为不同的类型转换。每个类型键的命名应与要覆盖的 MySQL 类型完全相同，包括大小规模。每个类型键需要两个子键 override_to 和 override_tables。

示例：

```
type_override:
    "tinyint(1)":
        override_to: boolean
        override_tables:
            - "*"
    "float(5,2)":
```

```
        override_to: float4
        override_tables:
            - schema.table
```

3.2.1. override_to

指定目标类型，该类型必须是 openGauss 支持的类型，并且类型转换应该是可行的。

3.2.2. override_tables

一个 yaml 列表，它指定覆盖应用于哪些表。如果第一个列表项设置为“*”，则覆盖将应用于 schema 中的所有表。如果表名称与 override_tables 值匹配，则覆盖也会应用于每个匹配的 DDL（创建表/变更表）。

3.3. openGauss 连接配置

pg_conn 配置部分，用于配置 openGauss 连接选项。

3.3.1. host

openGauss 所在机器的 IP 或者 hostname。

3.3.2. port

openGauss server 监听的端口号。

3.3.3. user

连接 openGauss 时使用的用户名。该用户需要对数据库有创建 schema、创建表的权限。

3.3.4. password

用户名对应的密码。

3.3.5. database

需要将 MySQL 的数据迁移到的目的数据库名称。

3.3.6. charset

database 所指定的数据库的编码格式。

3.4. MySQL 配置

sources 配置源数据库，主要是 MySQL 的连接配置及复制过程中用到的参数配置。

3.4.1. db_conn

MySQL 连接选项。

3.4.1.1. host

MySQL 所在机器的 IP 或者 hostname。

3.4.1.2. port

MySQL server 监听的端口号。

3.4.1.3. user

连接 MySQL 时使用的用户名。该用户至少需要对数据库有读取、REPLICATION CLIENT、REPLICATION SLAVE、RELOAD 的权限。

3.4.1.4. password

用户名对应的密码。

3.4.1.5. database

需要迁移的 MySQL 数据库名称。

3.4.1.6. charset

database 所指定的数据库的编码格式。

3.4.1.7. connect_timeout

连接 MySQL 时的超时时间。较大的值可以帮助工具在慢速网络中更好地工作。低值可能会导致连接在执行任何操作之前失败。

3.4.2. schema_mappings

```
        schema_mappings:
            my_sakila: pgsql_sakila
```

schema mappings 是一个字典。每个键都是一个需要在 openGauss 中被复制的 MySQL 数据库。在示例中提供了 MySQL 数据库 `my_sakila` 被复制到 schema `pgsql_sakila`，存储在 pg_conn（db_test）中指定的数据库中。

3.4.3. limit_tables

包含要复制的表。如果列表为空，则复制整个 MySQL 数据库。注意如果通过在线 DDL 更改了表名，limit_tables 并不会一同更新。比如配置 limit_tables 为 my_sakila.test_table，然后在线复制阶段，在 MySQL 侧通过 ***alter table test_table rename to test_table_rename***; 那么后续对于 test_table_rename 的 DML 操作无法被同步。因为 limit_tables 记录的仍是 rename 之前的 test_table，无法识别该表已经被 rename 成了 test_table_rename。

3.4.4. skip_tables

包含不被复制的表。同 limit_tables 一样，如果通过在线 DDL 更改了表名，skip_tables 并不会一同更新。

3.4.5. grant_select_to

在 openGauss 侧给指定的角色赋予对复制过来的表 select 权限。如果 keep_existing_schema 配置项设置为 Yes，grant_select_to 将不起作用。

3.4.6. lock_timeout

目前该配置项无任何作用。

3.4.7. my_server_id

MySQL 副本的服务器 ID，需要与 MySQL 服务器配置的 server_id 保持一致。

3.4.8. replica_batch_size

指在对 openGauss 数据库执行写入之前，从 MySQL 副本中提取的最大行数。如果为负数，效果相当于 0 或 1，即每从 mysql 副本中提取 1 行，就立即向 openGauss 执行写入。该参数主要影响大量数据待写入 openGauss 时的性能，比如从 mysql 副本中提取了 1W 行数据，如果 replica_batch_size <= 1，那么这 1W 行数据将分成 1W 次 copy 写入 openGauss，每次 copy 写入一条。如果 replica_batch_size >= 1W，那么这 1W 行数据将一次性通过 1 次 copy 批量写入。

3.4.9. batch_retention

指 t_replica_batch 中重演批处理行的最大保留时间。该数值 <= 0 意味着当该批次被重演过之后将即刻被删除，不会被保留。

3.4.10. copy_max_memory

在 openGauss 中复制表时要使用的最大内存量。可以指定以 KB、MB、GB 为单位的值，添加后缀（例如 300M）。

3.4.11. copy_mode

有效值为“file”和“direct”。“direct”会让复制实时进行。对于“file”，表首先转储到 csv 文件中，然后在 openGauss 中重新加载，加载完成后 csv 文件会被删除。csv 文件存储的位置由 out_dir 配置。

3.4.12. out_dir

如果 copy_mode 为 file，则在 init_replica 过程中转储 csv 文件的目录。

3.4.13. sleep_loop

两个副本批次之间的睡眠循环秒数。单位为秒。间隔 N 秒之后从 MySQL 读取下一批次数据。

3.4.14. on_error_replay

指在 start_replica 阶段，openGauss 侧重演失败时的动作。’exit’表示在 openGauss 侧重演失败时，退出复制进程。’continue’表示在 openGauss 侧重演失败时，从复制副本中删除失败的表，后续对于该表的改动不会再同步，同时继续其他表的复制。

3.4.15. on_error_read

指在 start_replica 阶段，连接 MySQL 失败时的动作。'exit'表示连接 MySQL 失败则退出进程。'continue'表示连接 MySQL 失败则一直重试。

3.4.16. auto_maintenance

指定触发自动维护（vaccum）后的超时时间。参数接受对 openGauss 间隔数据类型(例如 1 day)，如果设置为 disabled 自动维护不会运行。如果省略该参数，则默认值为 disabled。

3.4.17. gtid_enable

beta 参数，默认设置为 false。暂时不要启用。

3.4.18. type

指定源数据库类型。系统支持 mysql 或 pgsql。其中 pgsql 是实验特性，暂时不建议使用。

3.4.19. skip_events

skip_events 变量告诉 chameleon 跳过表或整个 schema 的特定事件。

3.4.19.1. insert

在线复制阶段，对于指定的表或 schema，跳过 insert 事件。

3.4.19.2. delete

在线复制阶段，对于指定的表或 schema，跳过 delete 事件。

3.4.19.3. update

在线复制阶段，对于指定的表或 schema，跳过 update 事件。

3.4.20. keep_existing_schema

当设置为 Yes 时，init_replica 时不会使用 MySQL 源中的数据重新创建受影响的表。相反，现有表将被截断并重新加载数据。执行 REINDEX 表，以便在重新加载后使索引处于良好状态。

这要求被复制的表已经在 openGauss 侧存在。
当设置为 No 时，init_replica 时会先删除 openGauss 测待复制的 schema 并重新创建。

4. 分区表迁移规则

分区表迁移的基本思想是对于 openGauss 支持的分区类型，迁移成对应的分区表即可。对于 openGauss 不支持的分区表类型，如二级分区、多列 list columns 分区等，将迁移成普通表。

类别	说明	MySQL	openGauss	迁移规则
range 分区	基于一个给定连续区间范围，把数据分配到不同的分区	1. range 分区要求分区键必须是 int 型。或者通过表达式返回 int 类型。 2. 若分区表中有主键/唯一建，则主键/唯一建必须包含分区键。 3. 仅支持 value less than 方式	value less than 语法：支持 4 列分区键，支持整数、浮点数、字符串、时间做分区键。 Start end 语法：支持 1 列分区键。支持整数、浮点数、时间做分区键。 Interval 语法：支持 1 列分区键。支持时间做分区键	openGauss 可完全兼容 MySQL。所有 MySQL 的 range 分区表均可迁移到 openGauss 的 range 分区表。
list 分区	类似 range 分区，区别在 list 分区是基于枚举出的值列表分区，range 是基于给定的连续区间范围分区	1. range 分区要求分区键必须是 int 型。或者通过表达式返回 int 类型。 2. 若分区表中有主键/唯一建，则主键/唯一建必须包含分区键。	1. 支持 1 列分区键。 2. 支持整数、字符串、时间做分区键。	openGauss 可完全兼容 MySQL。所有 MySQL 的 list 分区表均可迁移到 openGauss 的 list 分区表。
hash 分区	基于给定的分区个数，把数据分配到不同的分区	1. range 分区要求分区键必须是 int 型。或者通过表达式返回 int 类型。 2. 若分区表中有主键/唯一建，则主键/唯一建必须包含分区键。 3. 支持 linear 关键字，常规 hash 使用的是取模算法，线性 hash 使用的是一个线性的 2 的幂的运算法则	1. 支持 1 列分区键。 2. 支持整数、字符串、时间做分区键。	openGauss 不支持 linear 关键字，仅支持普通 hash 表。不过可以忽略，linear 影响的主要是 hash 分布，以及对 hash 分区增减时减少数据重分布的影响。所以对于 linear hash 分区表，将迁移成普通的 hash 表。
key 分区	类似于 hash 分区	1. blob 或 text 列类型除外可做分区键 2. 若分区表中有主键/唯一建，则主键/唯一建必须包含分	无对应类型，不过基本可以使用 hash 分区表替代。	1. 当只用一列做分区键且分区键类型是整数、字符串、时间时，可用 hash 分区替代。多列分区键不支持，其

		区键。 3. 支持 linear 关键字，常规 hash 使用的是取模算法，线性 hash 使用的是一个线性的 2 的幂的运算法则 4. 不能使用表达式作为分区键 5. 可使用多个列做分区键		他数据类型不支持。对于不支持的分区表，将迁移成普通表。 2.不支持 linear 关键字，没有类似的处理。不过可以忽略，linear 影响的主要是 hash 分布，以及对 hash 分区增减时减少数据重分布的影响。所以对于 linear key 分区表，将迁移成普通的 hash 表。
columns 分区	是 range 分区和 list 分区的扩展，分为 range columns 分区和 list columns 分区	1. 支持支持整数，日期时间数据类型做分区键。 2. 可使用多个列做分区键 3. 若分区表中有主键/唯一建，则主键/唯一建必须包含分区键。	无对应类型，部分可用 range 或 list 分区替代。	1.range columns 分区，支持不超过 4 列。 2.list columns 分区，不支持多列分区键。
二级分区	分区表中对每个分区再次分割		N/A	不支持

5. 默认的类型转换规则

需要注意的是，下面列出来的 MySQL 数据类型指的是 information_schema.COLUMNS 中的 data_type 而非 column_type。比如对于 int unsigned 的 column_type，其 data_type 实际为 int。对于 float(5,2)的 column_type，其 data_type 实际为 float。

MySQL	openGauss	备注
integer	integer	
mediumint	integer	
tinyint	integer	
smallint	integer	
int	integer	
bigint	bigint	
varchar	character varying	支持迁移最大存储长度（character_maximum_length）
character varying	character varying	支持迁移最大存储长度（character_maximum_length）
text	text	
char	character	支持迁移最大存储长度（character_maximum_length）
datetime	timestamp without time zone	
date	date	

time	time without time zone	
timestamp	timestamp without time zone	
tinytext	text	
mediumtext	text	
longtext	text	
tinyblob	blob	
mediumblob	blob	
longblob	blob	
blob	blob	
binary	bytea	
varbinary	bytea	
decimal	numeric	支持迁移精度（numeric_precision 和 numeric_scale）
dec	numeric	支持迁移精度（numeric_precision 和 numeric_scale）
numeric	numeric	支持迁移精度（numeric_precision 和 numeric_scale）
double	double precision	由于精度误差，移过到 openGauss 的值和 MySQL 侧可能不完全一致
double precision	double precision	由于精度误差，移过到 openGauss 的值和 MySQL 侧可能不完全一致
float	double precision	由于精度误差，移过到 openGauss 的值和 MySQL 侧可能不完全一致
bit	integer	
year	integer	
enum	enum	openGauss 不直接支持 enum 类型。对于 MySQL 的 enum 类型，openGauss 将通过自定义类型创建枚举类型，例如 CREATE TYPE enumtype AS ENUM('a','b')创建 ENUM 类型，然后使用 enumtype 去定义列类型。
set	text	
json	json	
bool	boolean	
boolean	boolean	
geometry	point	若 openGauss 侧安装有 postgis，将迁移成 geometry 类型
point	point	若 openGauss 侧安装有 postgis，将迁移成 geometry 类型
linestring	path	若 openGauss 侧安装有 postgis，将迁移成 geometry 类型
polygon	polygon	若 openGauss 侧安装有 postgis，将迁移成 geometry 类型

multipoint	bytea	若 openGauss 侧安装有 postgis，将迁移成 geometry 类型
geometrycollection	bytea	若 openGauss 侧安装有 postgis，将迁移成 geometry 类型
multilinestring	bytea	若 openGauss 侧安装有 postgis，将迁移成 geometry 类型
multipolygon	bytea	若 openGauss 侧安装有 postgis，将迁移成 geometry 类型

6. 实例

6.1. 创建配置文件目录

参考第 2 节，进入 python 虚拟环境安装好 chameleon 工具。
首先创建 chameleon 配置文件目录：

chameleon set_configuration_files

执行该命令后，将在 ***~/.pg_chameleon/configuration*** 目录下创建默认的配置文件模板。然后复制一份作为实际的配置文件。

cd ~/.pg_chameleon/configuration

cp config-example.yml default.yml

参考第 3 节的配置文件说明，按照实际情况修改 default.yml 配置文件。
如下是一份配置文件示例：

```
# global settings
pid_dir: '~/.pg_chameleon/pid/'
log_dir: '~/.pg_chameleon/logs/'
log_dest: file
log_level: info
log_days_keep: 10
rollbar_key: ''
rollbar_env: ''

# type_override allows the user to override the default type conversion
# into a different one.
type_override:
"tinyint(1)":
    override_to: boolean
    override_tables:
    - "*"

# postgres  destination connection
pg_conn:
  host: "1.1.1.1"
```

```
  port: "5432"
  user: "opengauss_test"
  password: "password_123"
  database: "opengauss_database"
  charset: "utf8"

sources:
  mysql:
    db_conn:
      host: "1.1.1.1"
      port: "3306"
      user: "mysql_test"
      password: "password123"
      charset: 'utf8'
      connect_timeout: 10
    schema_mappings:
      mysql_database:sch_mysql_database
    limit_tables:
    skip_tables:
    grant_select_to:
      - usr_migration
    lock_timeout: "120s"
    my_server_id: 1
    replica_batch_size: 10000
    replay_max_rows: 10000
    batch_retention: '1 day'
    copy_max_memory: "300M"
    copy_mode: 'file'
    out_dir: /tmp
    sleep_loop: 1
    on_error_replay: continue
    on_error_read: continue
    auto_maintenance: "disabled"
    gtid_enable: false
    type: mysql
    keep_existing_schema: No
```

以上配置文件的含义是，迁移数据时，MySQL 侧使用的用户名密码分别是 ***mysql_test*** 和 ***password123***。MySQL 服务器的 IP 和 port 分别是 ***1.1.1.1*** 和 ***3306***，待迁移的数据库是 ***mysql_database***。

openGauss 侧使用的用户名密码分别是 ***opengauss_test*** 和 ***password_123***。openGauss 服务器的 IP 和 port 分别是 ***1.1.1.1*** 和 ***5432***，目标数据库是 ***opengauss_database***，同时会在 ***opengauss_database*** 下创建 **sch_mysql_database** schema，迁移的表都将位于该 schema 下。

需要注意的是，这里使用的用户需要有远程连接 MySQL 和 openGauss 的权限，以及对对应数据库的读写权限。同时对于 openGauss，运行 chameleon 所在的机器需要在 openGauss

的远程访问白名单中。对于 MySQL，用户还需要有 RELOAD、REPLICATION CLIENT、REPLICATION SLAVE 的权限。

6.2. 创建用户及修改 database 配置

准备好配置文件后，在 openGauss 侧创建迁移时需要用到的用户以及 database。

CREATE USER opengauss_test WITH PASSWORD 'password123';

CREATE DATABASE opengauss_database WITH OWNER opengauss_test dbcompatibility='B' ;

注意：创建'B'兼容性的数据库。

修改 openGauss 配置文件。将 chameleon 工具所在机器 IP 地址加入白名单，修改 openGauss 监听地址。

gs_guc set -D {DATADIR} -c " listen_addresses = '*'"

gs_guc set -D {DATADIR} -h "host all all x.x.x.x/32 sha256"

修改完毕后重启 openGauss。

gs_ctl restart -D {DATADIR}

然后再 MySQL 侧创建用于复制的用户。

CREATE USER mysql_test;

SET PASSWORD FOR mysql_test=PASSWORD('password123');

GRANT ALL ON *.* TO 'mysql_test';

GRANT RELOAD ON *.* to 'mysql_test';

GRANT REPLICATION CLIENT ON *.* to 'mysql_test';

GRANT REPLICATION SLAVE ON *.* to 'mysql_test';

FLUSH PRIVILEGES;

创建用户后，修改 MySQL 的配置文件，开启 MySQL 的复制功能。（一般是/etc/my.cnf 或者/etc/my.cnf.d/mariadb-server.cnf）

在 [mysqld] 配置块下修改如下配置（若没有[mysqld] 配置块新增即可）

```
[mysqld]
binlog_format= ROW
log_bin = mysql-bin
server_id = 1
binlog_row_image=FULL
```

修改完毕后重启 MySQL。

6.3. 初始化迁移过程

开始使用工具做迁移过程。先初始化复制流。

chameleon create_replica_schema --config default

chameleon add_source --config default --source mysql

此步骤将在 openGauss 侧创建用于复制过程的辅助 schema 和表。

6.4. 复制基础数据

接下来开始复制基础数据。

chameleon init_replica --config default --source mysql

做完此步骤后，将把 MySQL 当前的全量数据复制到 openGauss。可以在 openGauss 侧查看全量数据复制后的情况。

```
opengauss_database=> set current_schema to sch_mysql_database;
SET
opengauss_database=> \d
                              List of relations
       Schema        |     Name     | Type  |     Owner      |             Storage
---------------------+--------------+-------+----------------+---------------------------------
 sch_mysql_database | test_decimal | table | opengauss_test | {orientation=row,compression=no}
 sch_mysql_database | test_float   | table | opengauss_test | {orientation=row,compression=no}
 sch_mysql_database | test_int     | table | opengauss_test | {orientation=row,compression=no}
 sch_mysql_database | test_numeric | table | opengauss_test | {orientation=row,compression=no}
 sch_mysql_database | test_str     | table | opengauss_test | {orientation=row,compression=no}
 sch_mysql_database | test_time    | table | opengauss_test | {orientation=row,compression=no}
(6 rows)

opengauss_database=> select * from test_decimal;
   a
-------
  1.11
 12.22
(2 rows)
```

6.5. 开启在线复制

接下来可以开启在线实时复制。

chameleon start_replica --config default --source mysql

开启实时复制后，在 MySQL 侧插入一条数据：

```
MariaDB [mysql_database]> select * from test_decimal;
+-------+
| a     |
+-------+
|  1.11 |
| 12.22 |
+-------+
2 rows in set (0.000 sec)

MariaDB [mysql_database]> insert into test_decimal values('3.14');
Query OK, 1 row affected (0.001 sec)
```

在 openGauss 侧查看 test_decimal 表的数据：

```
opengauss_database=> select * from test_decimal;
   a
-------
  1.11
 12.22
(2 rows)

opengauss_database=> select * from test_decimal;
   a
-------
  1.11
 12.22
  3.14
(3 rows)
```

可以看到新插入的数据在 openGauss 侧成功被复制过来了。

6.6. 结束复制过程及清理资源

chameleon stop_replica --config default --source mysql

chameleon detach_replica --config default --source mysql

chameleon drop_replica_schema --config default

外键的创建和验证将在 detach_replica 阶段进行。