**2024实验三：数据库SQL对象迁移**

**背景与定义**

* 数据库对象迁移：
* 将源数据库 A 的存储的数据库对象，迁移到 openGauss 数据库中。
* 源数据库 A：
* 可以是除了 openGauss 之外的其它关系型数据库，例如 MySQL 数据库。
* 数据库对象，可以简单包括为：
* 视图
* 触发器
* 自定义函数
* 存储过程
* 系统元数据(系统表)：
* 存储关于数据库的元数据信息
* 例如 MySQL 数据库的INFORMATION\_SCHEMA数据表，请自行使用搜索引擎了解。以下是整理出的要点：

对于函数和存储过程，可以通过数据表 INFORMATION\_SCHEMA的[ROUTINES](https://dev.mysql.com/doc/refman/5.6/en/information-schema-routines-table.html)表获得元数据SPECIFIC\_NAME、ROUTINE\_DEFINITION、ROUTINE\_BODY等信息，也可以通过mysql.proc系统表获取元数据。此外，可使用[SHOW CREATE PROCEDURE](https://dev.mysql.com/doc/refman/5.6/en/show-create-procedure.html) *proc\_name*语句查看存储过程定义、使用[SHOW CREATE FUNCTION](https://dev.mysql.com/doc/refman/5.6/en/show-create-function.html) *func\_name*语句查看函数定义。

对于触发器，可以查询系统数据表INFORMATION\_SCHEMA的[TRIGGERS](https://dev.mysql.com/doc/refman/8.0/en/information-schema-triggers-table.html)表获得元数据，也可以使用[SHOW CREATE TRIGGER](https://dev.mysql.com/doc/refman/8.0/en/show-create-trigger.html) *trigger\_name*语句查看触发器定义。

对*于视图，可以查询系统数据表INFORMATION\_SCHEMA的*[*VIEWS*](https://dev.mysql.com/doc/refman/8.0/en/information-schema-views-table.html)*表获得视图名（字段"TABLE\_NAME")，使用*[*SHOW CREATE VIEW*](https://dev.mysql.com/doc/refman/8.0/en/show-create-view.html) *view\_name语句查看视图定义（字段"Create View"）。*

* 例如openGauss以pg\_为开头的数据表

**实验任务**

你需要**编写一个简单的小程序**，能够从源数据库 A 中，**迁移四种数据库对象**到 openGauss 中，你需要

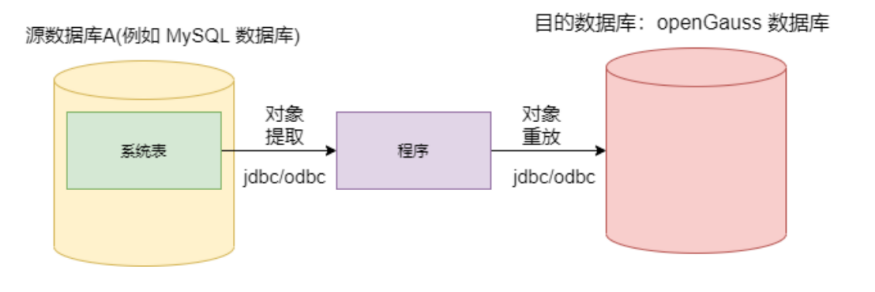
做具体的任务大致可以分为如下：

1.安装其它类型的关系型数据库(下面称为数据库 A)，例如 MySQL 数据库

2.在数据库 A 中创建视图、存储过程、函数、触发器四种数据库对象

3.编写迁移程序：使用上次实验学到的 jdbc/odbc 知识，在程序中读取数据库 A 的系统表，提取出你定义的存储在数据库 A 的数据库对象，然后将提取的对象“重放”到 openGauss 端，即使用 jdbc/odbc 在 openGauss 一侧执行对象定义SQL语句。如果源数据库和 openGauss 相应的 SQL 语句语法存储差异，请在程序中进行转化，再“重放”执行。

4.测试：对象迁移重放成功，可以正常使用



**具体要求**

1.使用JDBC或者ODBC从源数据库A迁移**视图**到 openGauss 中（源数据库A和openGauss相应的创建对象的SQL语法不存在差异）将实验过程及相关截图记录下来。

2.使用JDBC或者ODBC从源数据库A迁移**存储过程、函数、触发器三**种数据库对象到 openGauss 中（源数据库A和openGauss相应的创建对象的SQL语法存在差异），将实验过程及相关截图记录下来。

3.请简述一下源数据库A和openGauss这四种数据库对象定义的语法差异。

4.在实验过程中你遇到什么困难和问题，你怎么解决的？（没有则可以不写）

**实验步骤**

**MySQL端**

1.在安装有opengauss的虚拟机上安装mysql

|  |
| --- |
| 以5.7为例，我们需要从mysql.proc中提取函数的参数，如果是高一点的版本比如8.0.15以上的不存在mysql.proc这个系统表，此时可以从 [INFORMATION\_SCHEMA PARAMETERS](https://dev.mysql.com/doc/refman/5.7/en/information-schema-parameters-table.html)中提取函数的参数部分 |

没有安装mysql数据库的同学,请自行安装下载再进行以下实验

|  |
| --- |
| 成功安装后下载mysql jar包  注:此链接不需要关注Maven相关内容 |

* [下载mysql jar包](https://blog.csdn.net/weixin_51909882/article/details/124645571)

|  |
| --- |
| Mysql 启动命令  默认安装可执行以下命令,若自定义安装则找到包含mysqld可执行文件,进入到此文件下存在的路径  cd /usr/local/mysql/bin  启动方式多种,选择其中一种即可./使用service启动进入数据库  service mysql start |

2.连接mysqlmysql -u root -p，创建新的数据库create database test;切换数据库USE test;

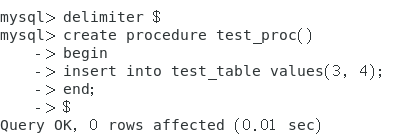
|  |
| --- |
| 如果你想进入你已经创建好的数据库，可以用mysql -u 用户名 - D 数据库名 -p  查看当前使用的数据库SELECT DATABASE();  查看当前使用的用户SELECT CURRENT\_USER();  查看数据库所有用户SELECT DISTINCT CONCAT('User: ''',user,'''@''',host,''';') AS query FROM mysql.user; |

3.创建视图

|  |
| --- |
| 注意： 迁移视图需要提前在openGauss端创建好一模一样的表，并插入相同的数据（示例中是test\_table) |

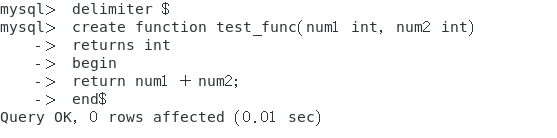
|  |
| --- |
| SQL -- mysql端创建表（opengauss也要执行，见“openGauss端”部分） CREATE TABLE test\_table(aaa int, bbb int);  --mysql端插入数据 mysql（opengauss也要执行，见“openGauss端”部分） insert into test\_table values(1, 2);  -- mysql端创建视图 CREATE VIEW test\_view AS select test\_table.aaa AS aaa,test\_table.bbb AS bbb from test\_table; |

4.创建存储过程



|  |
| --- |
| SQL --mysql侧创建存储过程 delimiter $ create procedure test\_proc() begin insert into test\_table values(3, 4); end; $ delimiter ; |

5.创建函数



|  |
| --- |
| SQL --设置mysql的参数，否则会报错ERROR 1418 (HY000): This function has none of DETERMINISTIC, NO SQL, or READS SQL DATA in its declaration and binary logging is enabled (you \*might\* want to use the less safe log\_bin\_trust\_function\_creators variable) SET GLOBAL log\_bin\_trust\_function\_creators=1;  --mysql侧创建自定义函数  delimiter $  create function test\_func(num1 int, num2 int)  returns int  begin  return num1 + num2;  end$  delimiter ; |

6.创建触发器

|  |
| --- |
| SQL --创建表，mysql和opengauss都要执行 create table test\_table2(id int,name varchar(25),deptid int not null,salary float not null); create table test\_table3(id int,name varchar(25),deptid int not null,salary float not null);  --创建触发器，触发条件是向表test\_table2插入数据库之后，再向表test\_table2插入相同数据，并且salary为test\_table3中新插入的salary字段值的2倍（字段同源表字段名一致） create trigger test\_trigger after insert on test\_table2 for each row insert into test\_table3 values(NEW.id,NEW.name,NEW.deptid,2\*NEW.salary); |

**openGauss端**

1.连接openGauss：gsql -d postgres -p 端口号

|  |
| --- |
| 查看数据库所有用户：select usename from pg\_user; |

创建'B'兼容性（MySQL)数据库：CREATE DATABASE opengauss\_database dbcompatibility='B';

2.使用\c opengauss\_database切换成数据库opengauss\_database;

创建与MySQL相同的表，否则无法迁移视图和触发器

|  |
| --- |
| SQL --openGauss端创建表 CREATE TABLE test\_table(aaa int, bbb int); insert into test\_table values(1, 2); create table test\_table2(id int,name varchar(25),deptid int not null,salary float not null); create table test\_table3(id int,name varchar(25),deptid int not null,salary float not null); |

**编写迁移程序**

* 编译命令

|  |
| --- |
| SQL javac -encoding utf-8 -cp /home/omm/t\_jdbc/postgresql.jar:/home/omm/t\_jdbc/mysql-connector-java-8.0.29.jar:/home/omm/t\_jdbc/postgresql.jar transfer.java  java -cp .:/home/omm/t\_jdbc/postgresql.jar:/home/omm/t\_jdbc/mysql-connector-java-8.0.29.jar transfer |

* 获取已存在的数据库对象的元数据，例如数据库对象的NAME

|  |
| --- |
| SQL --VIEW:  SELECT TABLE\_NAME AS OBJECT\_NAME, VIEW\_DEFINITION FROM information\_schema.VIEWS WHERE TABLE\_SCHEMA = "test"; --TRIGGER:  SELECT TRIGGER\_NAME AS OBJECT\_NAME FROM information\_schema.TRIGGERS WHERE TRIGGER\_SCHEMA = "test"; --PROC:  SELECT NAME AS OBJECT\_NAME FROM mysql.proc WHERE type = 'PROCEDURE' and db ='test';  --或者 SELECT SPECIFIC\_NAME FROM information\_schema.ROUTINES WHERE ROUTINE\_SCHEMA = "test" and ROUTINE\_TYPE = "PROCEDURE"; --FUNC:  SELECT NAME AS OBJECT\_NAME FROM mysql.proc WHERE type = 'FUNCTION' and db = 'test';   --或者 SELECT SPECIFIC\_NAME FROM information\_schema.ROUTINES WHERE ROUTINE\_SCHEMA = "test"and ROUTINE\_TYPE = "FUNCTION"; |

* mysql与opengauss的语法差别

|  |
| --- |
| 请大家通过opengauss官网查看创建四种对象的sql语法 |

* 函数

|  |
| --- |
| 点击查看opengauss官网 [opengauss创建自定义函数语法](https://docs.opengauss.org/zh/docs/5.0.0/docs/SQLReference/CREATE-FUNCTION.html) |

|  |
| --- |
| SQL --opengauss 兼容O风格的创建自定义函数的语法 CREATE FUNCTION test\_func(num1 int, num2 int) RETURN int AS BEGIN  RETURN num1 + num2; END / --或兼容PostgreSQL风格的创建自定义函数语法 CREATE FUNCTION test\_func(num1 int, num2 int) returns int as $$   begin  return num1 + num2;  end;  $$LANGUAGE plpgsql; --mysql见上面  delimiter $  create function test\_func(num1 int, num2 int)  returns int  begin  return num1 + num2;  end$  delimiter ; |

* 存储过程

|  |
| --- |
| 点击查看opengauss官网 [opengauss创建存储过程语法](https://docs.opengauss.org/zh/docs/5.0.0/docs/SQLReference/CREATE-PROCEDURE.html) |

|  |
| --- |
| SQL --opengauss CREATE PROCEDURE test\_proc() AS BEGIN  insert into test\_table values(3, 4); END / --mysql见上面 delimiter $ create procedure test\_proc() begin insert into test\_table values(3, 4); end; $ delimiter ; |

* 触发器

|  |
| --- |
| 点击查看opengauss官网[opengauss创建触发器语法](https://docs.opengauss.org/zh/docs/5.0.0/docs/SQLReference/CREATE-TRIGGER.html) |

|  |
| --- |
| SQL --opengauss mysql风格的创建触发器的语法 create trigger test\_trigger  after insert on test\_table2 for each row begin insert into test\_table3 values(NEW.id,NEW.name,NEW.deptid,2\*NEW.salary); end; / --opengauss o风格创建触发器的语法 CREATE OR REPLACE FUNCTION createFunction\_fa8a8a92867a4a5c9dcd0d1a5b0f0697() RETURNS TRIGGER AS $$ DECLARE BEGIN insert into test\_table3 values(NEW.id,NEW.name,NEW.deptid,2\*NEW.salary); RETURN NEW; END $$ LANGUAGE PLPGSQL; CREATE TRIGGER test\_trigger AFTER INSERT on test\_table2 for each ROW EXECUTE PROCEDURE createFunction\_fa8a8a92867a4a5c9dcd0d1a5b0f0697()  --mysql见上面 create trigger test\_trigger  after insert on test\_table2  for each row  insert into test\_table3 values(NEW.id,NEW.name,NEW.deptid,2\*NEW.salary); |

* 视图

|  |
| --- |
| openGauss不支持`符号，如果提取出来的视图定义有该符号，需要替换成空 |

**迁移函数示例**

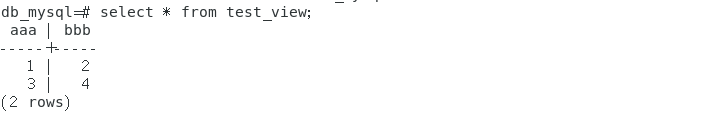
|  |
| --- |
| Java import java.sql.\*; public class func\_replica {  //opengauss端连接，按照你的信息修改相关值  static final String opengauss\_DRIVER = "org.postgresql.Driver";  static final String opengauss\_URL = "jdbc:postgresql://127.0.0.1:7777/db\_mysql";  static final String opengauss\_user = "omm";  static final String opengauss\_pwd = "yujian1211.";   //mysql端连接，按照你的信息修改相关值  static final String databaseName = "test";  //本机mysql  static final String mysql\_URL = "jdbc:mysql://localhost:3306/test";  static final String mysql\_user = "mysql\_test";  static final String mysql\_pwd = "yujian1211.";  static final String mysql\_DRIVER = "com.mysql.jdbc.Driver";      public static void main(String[]args)throws Exception {  Connection opengauss\_conn = null;  Statement opengauss\_stmt = null;  Connection mysql\_conn = null;  Statement mysql\_stmt = null;  try {  //opengauss侧  Class.forName(opengauss\_DRIVER);  System.out.println("连接opengauss数据库中..");  opengauss\_conn = DriverManager.getConnection(opengauss\_URL, opengauss\_user, opengauss\_pwd);  opengauss\_stmt=opengauss\_conn.createStatement();  //mysql侧  Class.forName(mysql\_DRIVER);  System.out.println("连接mysql数据库中");  mysql\_conn = DriverManager.getConnection(mysql\_URL, mysql\_user, mysql\_pwd);  mysql\_stmt = mysql\_conn.createStatement();    //迁移  System.out.println("函数迁移进行中...");    //从mysql.proc和 information\_schema表中获取函数定义   String Msql = "SELECT A.NAME, A.param\_list, B.data\_type, B.routine\_definition FROM mysql.proc A JOIN information\_schema.ROUTINES B ON A.NAME = B.specific\_name WHERE A.type = 'FUNCTION' and A.db=\'"+databaseName+"\';"; //改进语法  ResultSet Mrs = mysql\_stmt.executeQuery(Msql);   while (Mrs.next()) {  String name = Mrs.getString("NAME");  String param = Mrs.getString("param\_list");   String body = Mrs.getString("routine\_definition");   String returns=Mrs.getString("data\_type");     //封装，opengauss端复用  String Osql="CREATE OR REPLACE FUNCTION %s(%s) returns %s as $$ %s; $$LANGUAGE plpgsql;";  //或者用String Osql="CREATE OR REPLACE FUNCTION %s(%s) return %s AS %s\n /"; 因为在opengauss中创建函数有两种语法  Osql=String.format(Osql,name,param,returns,body);  opengauss\_stmt.execute(Osql);  System.out.println(Osql); //检验  System.out.println("函数迁移完成！");  }  mysql\_stmt.close();  mysql\_conn.close();  opengauss\_stmt.close();  opengauss\_conn.close();  } catch (SQLException e) {  e.printStackTrace();  } catch (Exception e1) {  e1.printStackTrace();  } finally {  try {  if (mysql\_stmt != null) mysql\_stmt.close();  } catch (SQLException e2) {  e2.printStackTrace();  }  }  try {  if (mysql\_conn != null) mysql\_conn.close();  } catch (SQLException e3) {  e3.printStackTrace();  }  System.out.println("goodbye");  } } |

**对象迁移测试**

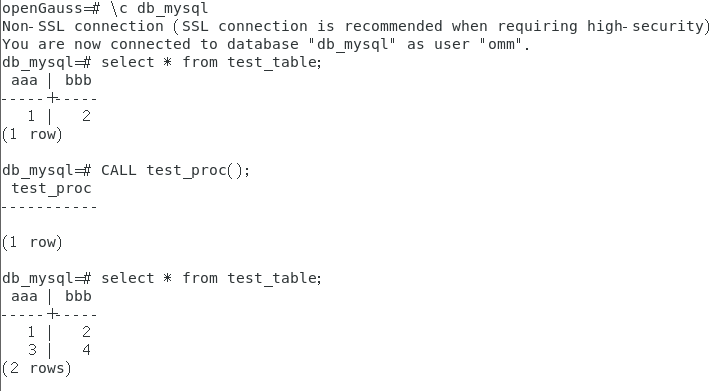
在opengauss侧测试是否迁移成功，并可正常使用

|  |
| --- |
| SQL \c opengauss\_database --查看视图 select \* from test\_view; --调用存储过程 CALL test\_proc();  --调用函数  select test\_func(1,2); --openGauss侧给源表插入数据，验证触发的表是否生效  insert into test\_table2 values(1,'A',1,1000),(2,'B',1,5000); |

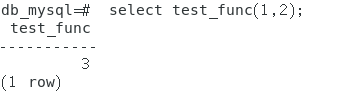
* 成功迁移视图：



* 成功迁移存储过程：



* 成功迁移函数：



* 成功迁移触发器：

