数据仓库服务

快速入门

文档版本 09

发布日期 2021-08-11

华为技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或默示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

目 录

1 创建集群并连接集群

1.1 第 1 步：入门前准备

本指南是一个入门教程，向您演示如何创建示例GaussDB(DWS) 集群，连接示例GaussDB(DWS) 集群数据库、导入存储在OBS中的示例数据和分析示例数据的流程。您可以使用该入门教程评估GaussDB(DWS) 服务。

在开始创建GaussDB(DWS) 集群之前，请确保您已完成如下前提条件：

- **注册并实名认证公有云帐户**
- **确定集群端口**

注册并实名认证公有云帐户

如果您还没有公有云帐户，则必须先注册一个。如果您已有实名认证的帐户，则可以跳过此步骤，并使用您已有的帐户。

1. 打开公有云服务网址**http://www.huaweicloud.com/**，单击页面右上方的“注册”，进入注册页面。
2. 按照页面要求填写用户信息完成注册。
3. 单击右上角用户名，进入基本信息页面，单击“实名认证”，进入实名认证页面。
4. 按照页面提示完成实名认证。

说明

开通云服务需要先进行实名认证。

确定集群端口

- 在创建GaussDB(DWS) 集群时需要指定一个端口供SQL客户端或应用程序通过该端口访问集群。
- 如果您的客户端机器位于防火墙之后，则您需要有一个可用的开放端口，这样才能从SQL客户端工具连接到集群并进行查询分析。
- 如果您不了解可用的开放端口，则请联系网络管理员，在您的防火墙中确定一个开放端口。GaussDB(DWS) 支持的端口范围为8000～30000。

- 在集群创建之后无法更改集群的端口号，请务必确保在集群创建过程中指定的端口为可用的开放端口。

1.2 第 2 步：创建集群

在使用GaussDB(DWS) 执行数据分析任务前，您首先要创建一个集群，一个GaussDB(DWS) 集群由多个在相同子网中的节点组成，共同提供服务。请参考以下指导创建集群。

创建集群

步骤1 登录GaussDB(DWS) 管理控制台。

步骤2 单击左侧导航栏的“集群管理”。

步骤3 在“集群管理”页面，单击右上角“创建数据仓库集群”。

步骤4 选择待创建的集群所属的区域。

- **区域**：选择“华北-北京四”。
- **可用区**：默认即可。

步骤5 选择主机规格。

- **产品类型**：根据客户需求选择，例如“**标准数仓**”类型。
- **CPU架构**：根据客户需求选择，例如“**X86**”架构。
- **节点规格**：默认即可。
- **节点数量**：默认即可，至少3个。

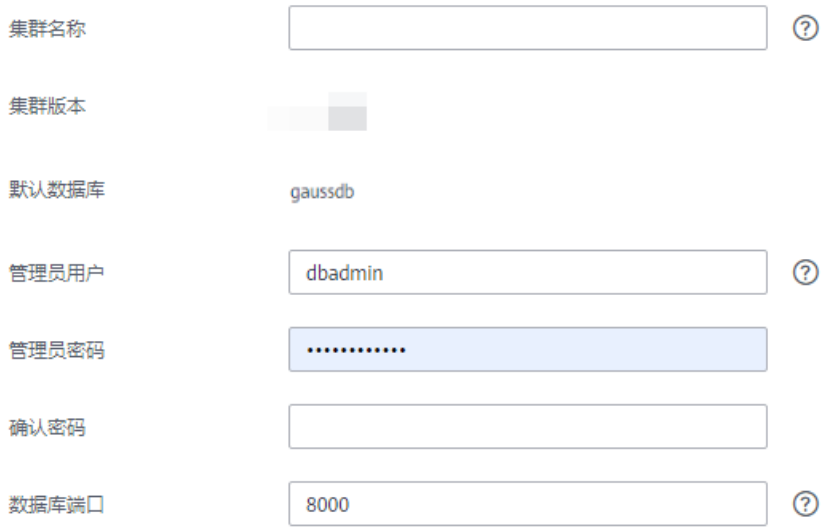
图 1-1 配置主机规格

步骤6 填写集群配置参数。

- **集群名称**：输入“dws-demo”。
- **集群版本**：显示为当前集群版本，暂不支持修改。
- **默认数据库**：显示为“gaussdb”。暂不支持修改。
- **管理员用户**：默认为“dbadmin”，使用默认值即可。集群创建成功后，客户端连接集群数据库时将使用该管理员用户及其密码。

- **管理员密码**：输入密码。
- **确认密码**：重复输入一次管理员密码。
- **数据库端口**：默认即可。客户端或应用程序将通过该端口连接集群中的数据库。

图 1-2 集群配置



步骤7 配置网络参数。

- **虚拟私有云**：可以在下拉框中选择已有的虚拟私有云，如果未配置过虚拟私有云，可以单击“查看虚拟私有云”进入虚拟私有云管理控制台，新创建一个虚拟私有云例如“vpc-dws”。然后回到GaussDB(DWS) 管理控制台的创建集群页面，单击“虚拟私有云”下拉框旁边的 进行刷新，再选择新创建虚拟私有云。
- **子网**：创建虚拟私有云时会默认创建一个子网，您可以选择对应的子网名。
- **安全组**：选择“自动创建安全组”。

  自动创建的安全组，将被命名为“GaussDB(DWS)-<*集群名称*>-<*GaussDB(DWS)集群的数据库端口*>”，出方向允许所有访问，入方向只开放“数据库端口”以允许来自客户端或应用程序的访问。

  如果您选择的是自定义创建的安全组，则需要在该安全组中添加一条入方向的规则，向访问GaussDB(DWS) 的客户端主机开放GaussDB(DWS) 集群的“数据库端口”，如**表1-1**所示。添加入规则的具体操作请参见《虚拟私有云用户指南》中的**添加安全组规则**章节。

  表 1-1 安全组入规则配置样例

参数名	样例值
协议/应用	TCP

参数名	样例值
端口	8000 **说明** 输入创建GaussDB(DWS) 集群时设置的“数据库端口”，这个端口是GaussDB(DWS) 用于接收客户端连接的端口，默认为8000。
源地址	选择“IP地址”，输入访问GaussDB(DWS) 的客户端主机的IP地址和子网掩码，例如“192.168.0.10/16”。

- **公网访问**：选择“现在购买”为集群购买一个弹性IP作为集群公网IP。并且，在“带宽”参数中设置弹性IP的带宽。

图 1-3 网络参数

步骤8 配置集群所属的“企业项目”。已开通企业项目管理服务的用户才可以配置该参数。默认值为default。

企业项目是一种云资源管理方式，企业项目管理服务提供统一的云资源按项目管理，以及项目内的资源管理、成员管理。

您可以选择默认的企业项目“default”或其他已有的企业项目。如果要创建新的企业项目，请登录企业管理控制台进行创建，详细操作请参考《企业管理用户指南》。

步骤9 高级配置，在本示例中，选择“默认配置”即可。

- **默认配置**：表示以下几项高级配置使用系统默认的配置。
  - 自动快照：默认开启自动生成集群快照的策略。
  - CN部署量：CN即协调节点，默认部署2个CN节点。
  - 参数模板：默认将系统默认的数据库参数模板与集群相关联。
  - 标签：默认未给集群添加标签。
  - 加密数据库：默认为关闭，表示不对数据库进行加密。
- **自定义**：选择该选项时页面上将显示自动快照、CN部署量、参数模板、标签、加密数据库、这几项高级配置，需要用户进行自定义设置。

步骤10 单击“立即购买”，进入“规格详情”页面。

步骤11 单击“提交”。

提交成功后开始创建。单击“返回集群列表”返回集群管理页面，所创集群的初始状态为“创建中”，集群创建需要时间，请等待一段时间。创建成功后状态更新为“可用”，用户可以开始使用集群。

----结束

1.3 第 3 步：连接集群

操作场景

您在创建好数据仓库集群，开始使用数据库服务前，需要使用数据库客户端连接到GaussDB(DWS) 集群中的数据库。本示例将使用Data Studio客户端工具通过公网地址连接GaussDB(DWS) 集群中的数据库。您也可以使用其他SQL客户端连接集群，更多连接方式请参见《数据仓库服务管理指南》中的“连接集群”章节。

1. 获取所要连接的数据库名称、用户名和密码。

   首次使用客户端连接集群时，您需使用**第2步：创建集群**时设置的管理员用户和密码连接到默认数据库“gaussdb”。
2. **获取集群公网访问地址**：通过集群公网访问地址连接数据库。
3. **使用Data Studio连接到集群数据库**：下载配置Data Studio客户端并连接集群数据库。

获取集群公网访问地址

步骤1 登录GaussDB(DWS) 管理控制台。

步骤2 在左侧导航栏中，单击“集群管理”。

步骤3 在集群列表中，选中已创建集群（如dws-demo），单击“集群名称”前面的向下展开按钮，获取并保存公网访问地址。

该公网访问地址将在**使用Data Studio连接到集群数据库**时使用。

图 1-4 集群管理

----结束

使用 Data Studio 连接到集群数据库

步骤1 GaussDB(DWS) 提供了基于Windows平台的Data Studio图形界面客户端，该工具依赖JDK，请先在客户端主机上安装Java 1.8.0_141或以上版本的JDK。

在Windows操作系统中，您可以访问**JDK官网**，下载符合操作系统版本的JDK，并根据指导进行安装。

步骤2 登录GaussDB(DWS) 管理控制台。

步骤3 单击“连接管理”。

步骤4 在“下载客户端和驱动”页面，下载“Data Studio图形界面客户端”。

- 请根据操作系统类型，选择“Windows x86”或“Windows x64”，再单击“下载”，可以下载与现有集群版本匹配的Data Studio工具。

  如果同时拥有不同版本的集群，单击“下载”时会下载与集群最低版本相对应的Data Studio工具。如果当前没有集群，单击“下载”时将下载到低版本的Data Studio工具。GaussDB(DWS) 集群可向下兼容低版本的Data Studio工具。
- 单击“历史版本”可根据集群版本下载相应版本的Data Studio工具，建议按集群版本下载配套的工具。

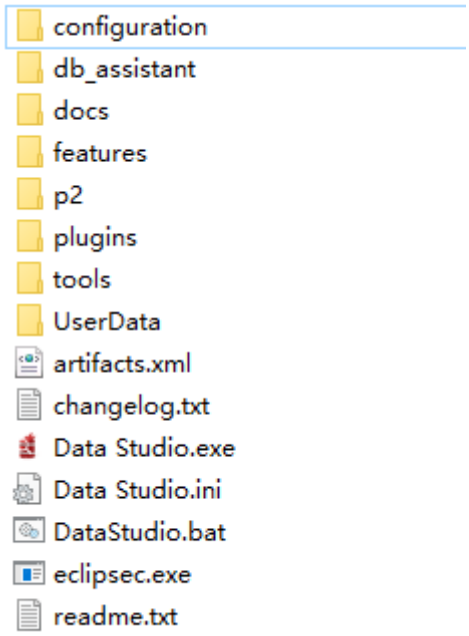
图 1-5 下载客户端

如果同时拥有不同版本的集群，系统会弹出对话框，提示您选择“集群版本”然后下载与集群版本相对应的客户端。在“集群管理”页面的集群列表中，单击指定集群的名称，再选择“基本信息”页签，可查看集群版本。

步骤5 解压下载的客户端软件包（32位或64位）到需要安装的路径。

步骤6 打开安装目录，双击Data Studio.exe，启动Data Studio客户端，如**图1-6**所示。

图 1-6 启动客户端



步骤7 在主菜单中选择 “文件>新建连接” ，如图1-7所示。

图 1-7 新建连接

步骤8 在弹出的“新建/选择数据库连接”页面中，如下图所示，输入连接参数。

图 1-8 配置连接参数

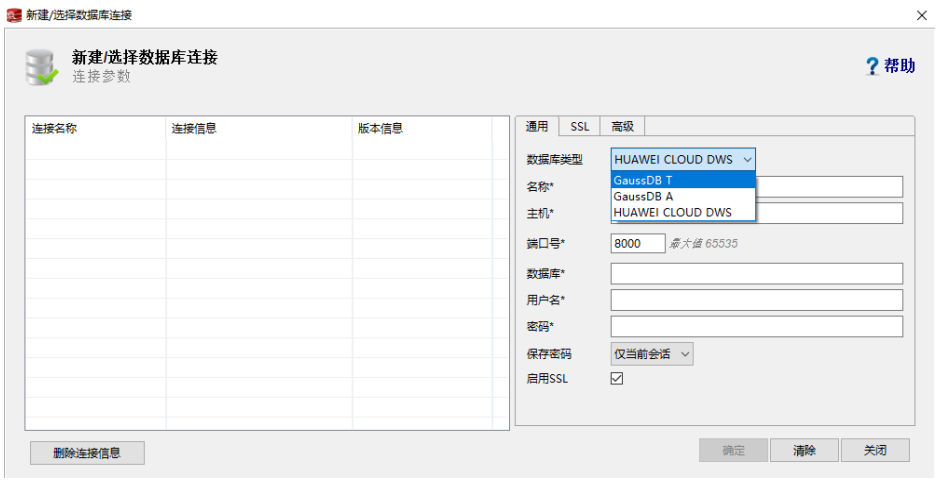


表 1-2 配置连接参数

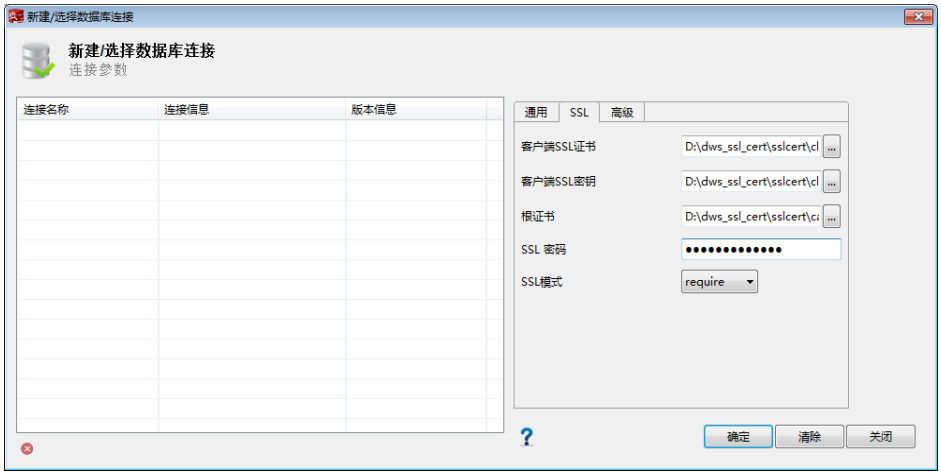
字段名称	说明	举例
数据库类型	选择“HUAWEI CLOUD DWS ”。	HUAWEI CLOUD DWS
名称	连接名称。	dws-demo
主机名	所要连接的集群IP地址（IPv4）或域名。	-
端口号	数据库端口。	8000
数据库	数据库名称。	gaussdb
用户名	所要连接数据库的用户名。	-
密码	所要连接数据库的登录密码。	-
保存密码	在下拉列表中选择： ● **“仅当前会话”**：仅在当前会话中保存密码。 ● **“不保存”**：不保存密码。	-
启用SSL	启用时，客户端将使用SSL加密连接方式。SSL连接方式安全性高于普通模式，建议开启。	-

当“启用SSL”设置为开启时，请先参见**下载SSL证书**下载SSL证书，并解压证书文件。然后，在如**图1-8**所示的窗口中单击“SSL”页签，设置如下参数：

表 1-3 配置 SSL 参数

字段名称	说明
客户端SSL证书	选择SSL证书解压目录下的“sslcert\client.crt”文件。
客户端SSL密钥	客户端SSL秘钥只支持PK8格式，请选择SSL证书解压目录下的“sslcert\client.key.pk8”文件。
根证书	当“SSL模式”设为“verify-ca”时，必须设置根证书，请选择SSL证书解压目录下的“sslcert\cacert.pem”文件。
SSL密码	客户端pk8格式SSL秘钥密码。
SSL模式	GaussDB(DWS) 支持的SSL模式有： ● require ● verify-ca GaussDB(DWS) 不支持“verify-full”模式。

图 1-9 配置 SSL 参数

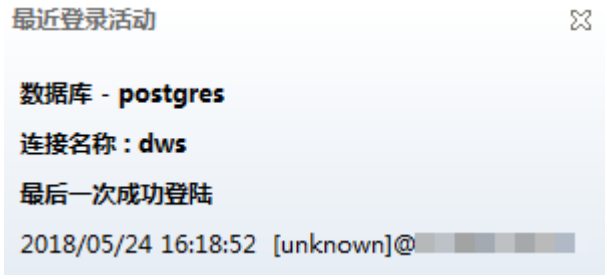


步骤9 单击“确定”建立数据库连接。

如果启用了SSL，在弹出的“连接安全告警”提示对话框中单击“继续”。

登录成功后，将弹出“最近登录活动”提示框，表示Data Studio已经连接到数据库。用户即可在Data Studio界面的“SQL终端”窗口中执行SQL语句。

图 1-10 登录成功



欲详细了解Data Studio其他功能的使用方法，请按“F1”查看Data Studio用户手册。

----结束

1.4 第 4 步：导入样例数据并查询

本文档为您提供了3个相互独立的入门样例，您可以根据需要选择其中一个或几个样例进行体验。

1.4.1 交通卡口通行车辆分析

本实践将演示交通卡口车辆通行分析，将加载8.9亿条交通卡口车辆通行模拟数据到数据仓库单个数据库表中，并进行车辆精确查询和车辆模糊查询，展示GaussDB(DWS)对于历史详单数据的高性能查询能力。

说明

- GaussDB(DWS) 已预先将样例数据上传到OBS桶的“traffic-data”文件夹中，并给所有华为云用户赋予了该OBS桶的只读访问权限。
- 当前“北京一”区域的OBS桶暂无该样例数据，推荐使用“北京四”或其他区域的集群进行体验。

本实践预计时长40分钟，基本流程如下：

1. **准备工作**
2. **步骤一：创建集群**
3. **步骤二：使用Data Studio连接集群**
4. **步骤三：导入交通卡口样例数据**
5. **步骤四：车辆分析**

准备工作

- 已注册公有云帐号，且在使用GaussDB(DWS) 前检查帐号状态，帐号不能处于欠费或冻结状态。
- 获取此帐号的“AK/SK”。

步骤一：创建集群

步骤1 登录华为云管理控制台。

步骤2 在“服务列表”中，选择“EI企业智能 > 数据仓库服务”。

步骤3 左侧导航栏单击“集群管理”，进入页面后，单击右上角的“购买数据仓库集群”按钮。

步骤4 参见**表1-4**进行参数配置。

表 1-4 软件配置

参数名称	配置方式
区域	选择“华北-北京四”。 **说明** 本指导以“华北-北京四”为例进行介绍，如果您需要选择其他区域进行操作，请确保所有操作均在同一区域进行。
可用区	可用区2
产品类型	标准数仓
CPU架构	X86
节点规格	dws2.m6.4xlarge.8（16 vCPU | 128GB | 2000GB SSD） **说明** 如规格售罄，可选择其他可用区或规格。
节点数量	3
集群名称	dws-demo

参数名称	配置方式
管理员用户	dbadmin
管理员密码	-
确认密码	-
数据库端口	8000
虚拟私有云	vpc-default
子网	subnet-default(192.168.0.0/24)
安全组	自动创建安全组
公网访问	现在购买
宽带	1Mbit/s
高级配置	默认配置

步骤5 信息核对无误，单击“立即购买”，单击“提交”。

步骤6 等待约6分钟，待集群创建成功后，单击集群名称前面的，弹出集群信息，记录下“公网访问地址”，例如dws-demov.dws.huaweicloud.com。

----结束

步骤二：使用 Data Studio 连接集群

步骤1 请确保客户端主机已安装JDK 1.8.0以上版本，并进入“此电脑 > 属性 > 高级系统设置 > 环境变量”设置JAVA_HOME（例如C:\Program Files\Java\jdk1.8.0_191），并在变量path中添加“;%JAVA_HOME%\bin”。

步骤2 在GaussDB(DWS) 控制台的“连接管理”页面，下载Data Studio客户端。

步骤3 解压下载的Data Studio软件包，进入解压目录后，双击Data Studio.exe启动客户端。

步骤4 在Data Studio主菜单中选择“文件 > 新建连接”，并在弹出框中参照表1-5所示配置。

表 1-5 Data Studio 软件配置

参数名称	配置方式
数据库类型	GaussDB(DWS)
名称	dws-demo
主机	dws-demov.dws.huaweicloud.com 与**步骤一：创建集群**查询到的“公网访问地址”一致。
端口	8000
数据库	gaussdb
用户名	dbadmin
密码	-
启用SSL	不启用

步骤5 单击“确定”。

----结束

步骤三：导入交通卡口样例数据

使用SQL客户端工具连接到集群后，就可以在SQL客户端工具中，执行以下步骤导入交通卡口车辆通行的样例数据并执行查询。

步骤1 执行以下语句，创建traffic数据库。

```
create database traffic encoding 'utf8' template template0;
```

步骤2 执行以下步骤切换为连接新建的数据库。

1. 在Data Studio客户端的**“对象浏览器”**窗口，右键单击数据库连接名称，在弹出菜单中单击“刷新”，刷新后就可以看到新建的数据库。
2. 右键单击“traffic”数据库名称，在弹出菜单中单击“打开连接”。
3. 右键单击“traffic”数据库名称，在弹出菜单中单击“打开新的终端”，即可打开连接到指定数据库的SQL命令窗口，后面的步骤，请全部在该命令窗口中执行。

步骤3 执行以下语句，创建用于存储卡口车辆信息的数据库表。

```
create schema traffic_data;
set current_schema= traffic_data;
drop table if exists GCJL;
CREATE TABLE GCJL
(
    kkbh   VARCHAR(20),
    hphm   VARCHAR(20),
    gcsj   DATE ,
    cplx   VARCHAR(8),
    cllx   VARCHAR(8),
    csys   VARCHAR(8)
)
with (orientation = column, COMPRESSION=MIDDLE)
distribute by hash(hphm);
```

步骤4 创建外表。外表用于识别和关联OBS上的源数据。

须知

其中，*<obs_bucket_name>*代表OBS桶名。本实践以“华北-北京四”地区为例，可填入dws-demo-cn-north-4，<Access_Key_Id>和<Secret_Access_Key>替换为实际值，在**准备工作**获取。

```
create schema tpchobs;
set current_schema = 'tpchobs';
drop FOREIGN table if exists GCJL_OBS;
CREATE FOREIGN TABLE GCJL_OBS
(
    like traffic_data.GCJL
)
SERVER gsmpp_server
OPTIONS (
    encoding 'utf8',
    location 'obs://<obs_bucket_name>/traffic-data/gcxx',
    format 'text',
    delimiter ',',
    access_key '<Access_Key_Id>',
    secret_access_key '<Secret_Access_Key>',
    chunksize '64',
    IGNORE_EXTRA_DATA 'on'
);
```

步骤5 执行以下语句，将数据从外表导入到数据库表中。

```
insert into traffic_data.GCJL select * from tpchobs.GCJL_OBS;
```

导入数据需要一些时间，请耐心等待。

----结束

步骤四：车辆分析

1. **执行Analyze**

   用于收集与数据库中普通表内容相关的统计信息，统计结果存储在系统表PG_STATISTIC中。执行计划生成器会使用这些统计数据，以生成最有效的查询执行计划。

   执行以下语句生成表统计信息：

   ```
   Analyze;
   ```

2. **查询数据表中的数据量**

   执行如下语句，可以查看已加载的数据条数。

   ```
   set current_schema= traffic_data;
   Select count(*) from traffic_data.gcjl;
   ```

3. **车辆精确查询**

   执行以下语句，指定车牌号码和时间段查询车辆行驶路线。GaussDB(DWS) 在应对点查时秒级响应。

   ```
   set current_schema= traffic_data;
   select hphm, kkbh, gcsj
   from traffic_data.gcjl
   where hphm =  'YD38641'
   and gcsj between '2016-01-06' and '2016-01-07'
   order by gcsj desc;
   ```

4. **车辆模糊查询**

   执行以下语句，指定车牌号码和时间段查询车辆行驶路线，GaussDB(DWS) 在应对模糊查询时秒级响应。


```
set current_schema= traffic_data;
select hphm, kkbh, gcsj
from traffic_data.gcjl
where hphm like  'YA23F%'
and kkbh in('508', '1125', '2120')
and gcsj between '2016-01-01' and '2016-01-07'
order by hphm,gcsj desc;
```

1.4.2 某公司供应链需求分析

本实践将演示从OBS加载样例数据集到GaussDB(DWS) 集群中并查询数据的流程，从而向您展示GaussDB(DWS) 在数据分析场景中的多表分析与主题分析。

说明

- GaussDB(DWS) 已经预先生成了1GB的TPC-H-1x的标准数据集，已将数据集上传到了OBS桶的tpch文件夹中，并且已赋予所有华为云用户该OBS桶的只读访问权限，用户可以方便的进行导入。
- 当前“北京一”区域的OBS桶暂无该样例数据，推荐使用“北京四”或其他区域的集群进行体验。

本实践预计时长60分钟，基本流程如下：

1. **准备工作**
2. **步骤一：导入公司样例数据**
3. **步骤二：多表分析与主题分析**

场景描述

目的：了解GaussDB(DWS)的基本功能和数据导入，对某公司与供应商的订单数据分析，分析维度如下：

1. 分析某地区供应商为公司带来的收入，通过该统计信息可用于决策在给定的区域是否需要建立一个当地分配中心。
2. 分析零件/供货商关系，可以获得能够以指定的贡献条件供应零件的供货商数量，通过该统计信息可用于决策在订单量大，任务紧急时，是否有充足的供货商。
3. 分析小订单收入损失，通过查询得知如果没有小量订单，平均年收入将损失多少。筛选出比平均供货量的20%还低的小批量订单，如果这些订单不再对外供货，由此计算平均一年的损失。

准备工作

- 已注册公有云帐号，且在使用GaussDB(DWS) 前检查帐号状态，帐号不能处于欠费或冻结状态。
- 获取此帐号的“AK/SK”。
- 已创建集群，并已使用Data Studio连接集群，参见**交通卡口通行车辆分析**。

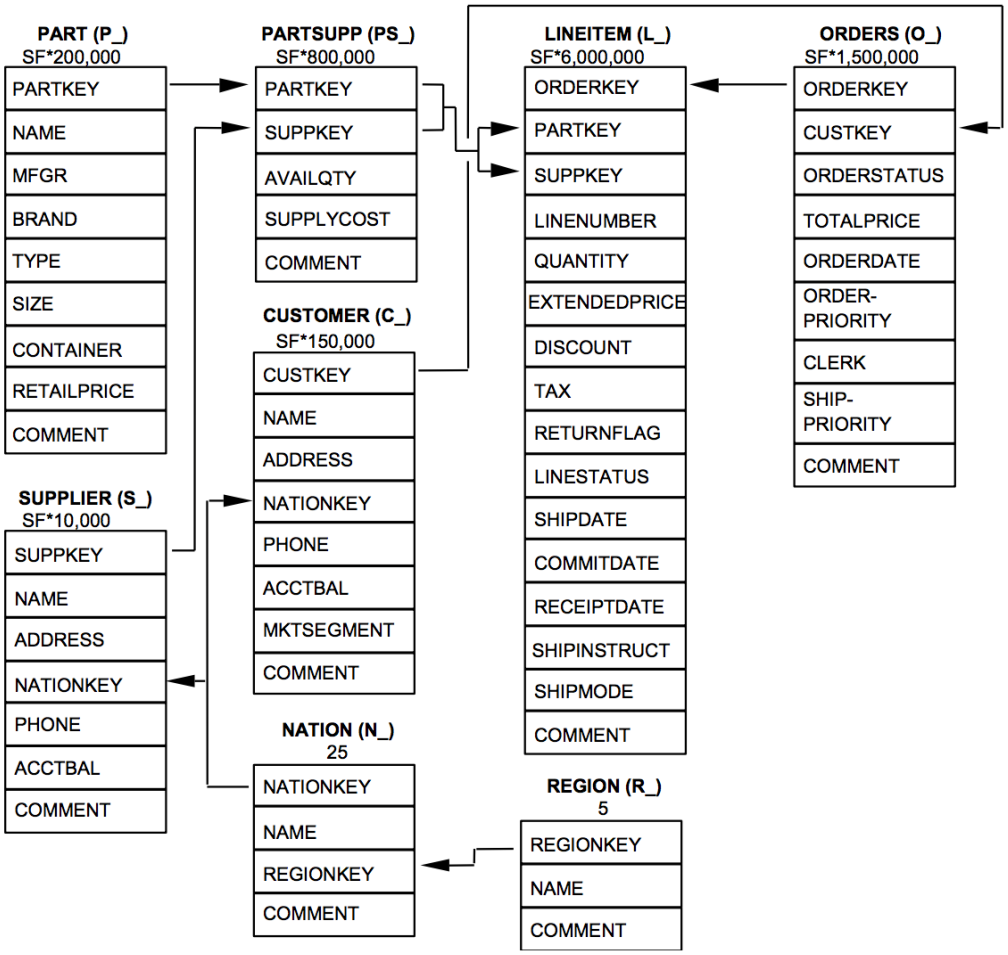
步骤一：导入公司样例数据

使用SQL客户端工具连接到集群后，就可以在SQL客户端工具中，执行以下步骤导入TPC-H样例数据并执行查询。

步骤1 创建数据库表。

TPC-H样例包含8张数据库表，其关联关系如**图1-11**所示。

图 1-11 TPC-H 数据表



复制并执行下列表创建语句，在gaussdb数据库中创建对应的数据表。

```
CREATE schema tpch;
set current_schema = tpch;

drop table if exists region;
CREATE TABLE REGION
(
        R_REGIONKEY  INT NOT NULL ,
        R_NAME       CHAR(25) NOT NULL ,
        R_COMMENT    VARCHAR(152)
)
with (orientation = column, COMPRESSION=MIDDLE)
distribute by replication;

drop table if exists nation;
CREATE TABLE NATION
(
        N_NATIONKEY  INT NOT NULL,
        N_NAME       CHAR(25) NOT NULL,
        N_REGIONKEY  INT NOT NULL,
        N_COMMENT    VARCHAR(152)
)
with (orientation = column,COMPRESSION=MIDDLE)
distribute by replication;

drop table if exists supplier;
CREATE TABLE SUPPLIER
```



```
(
    S_SUPPKEY     BIGINT NOT NULL,
    S_NAME        CHAR(25) NOT NULL,
    S_ADDRESS     VARCHAR(40) NOT NULL,
    S_NATIONKEY   INT NOT NULL,
    S_PHONE       CHAR(15) NOT NULL,
    S_ACCTBAL     DECIMAL(15,2) NOT NULL,
    S_COMMENT     VARCHAR(101) NOT NULL
)
with (orientation = column,COMPRESSION=MIDDLE)
distribute by hash(S_SUPPKEY);

drop table if exists customer;
CREATE TABLE CUSTOMER
(
    C_CUSTKEY     BIGINT NOT NULL,
    C_NAME        VARCHAR(25) NOT NULL,
    C_ADDRESS     VARCHAR(40) NOT NULL,
    C_NATIONKEY   INT NOT NULL,
    C_PHONE       CHAR(15) NOT NULL,
    C_ACCTBAL     DECIMAL(15,2)   NOT NULL,
    C_MKTSEGMENT  CHAR(10) NOT NULL,
    C_COMMENT     VARCHAR(117) NOT NULL
)
with (orientation = column,COMPRESSION=MIDDLE)
distribute by hash(C_CUSTKEY);

drop table if exists part;
CREATE TABLE PART
(
    P_PARTKEY     BIGINT NOT NULL,
    P_NAME        VARCHAR(55) NOT NULL,
    P_MFGR        CHAR(25) NOT NULL,
    P_BRAND       CHAR(10) NOT NULL,
    P_TYPE        VARCHAR(25) NOT NULL,
    P_SIZE        BIGINT NOT NULL,
    P_CONTAINER   CHAR(10) NOT NULL,
    P_RETAILPRICE DECIMAL(15,2) NOT NULL,
    P_COMMENT     VARCHAR(23) NOT NULL
)
with (orientation = column,COMPRESSION=MIDDLE)
distribute by hash(P_PARTKEY);

drop table if exists partsupp;
CREATE TABLE PARTSUPP
(
    PS_PARTKEY     BIGINT NOT NULL,
    PS_SUPPKEY     BIGINT NOT NULL,
    PS_AVAILQTY    BIGINT NOT NULL,
    PS_SUPPLYCOST  DECIMAL(15,2)  NOT NULL,
    PS_COMMENT     VARCHAR(199) NOT NULL
)
with (orientation = column,COMPRESSION=MIDDLE)
distribute by hash(PS_PARTKEY);

drop table if exists orders;
CREATE TABLE ORDERS
(
    O_ORDERKEY       BIGINT NOT NULL,
    O_CUSTKEY        BIGINT NOT NULL,
    O_ORDERSTATUS    CHAR(1) NOT NULL,
    O_TOTALPRICE     DECIMAL(15,2) NOT NULL,
    O_ORDERDATE      DATE NOT NULL ,
    O_ORDERPRIORITY  CHAR(15) NOT NULL,
    O_CLERK          CHAR(15) NOT NULL ,
    O_SHIPPRIORITY   BIGINT NOT NULL,
    O_COMMENT        VARCHAR(79) NOT NULL
)
with (orientation = column,COMPRESSION=MIDDLE)
```



```
distribute by hash(O_ORDERKEY);

drop table if exists lineitem;
CREATE TABLE LINEITEM
(
        L_ORDERKEY    BIGINT NOT NULL,
        L_PARTKEY     BIGINT NOT NULL,
        L_SUPPKEY     BIGINT NOT NULL,
        L_LINENUMBER  BIGINT NOT NULL,
        L_QUANTITY    DECIMAL(15,2) NOT NULL,
        L_EXTENDEDPRICE  DECIMAL(15,2) NOT NULL,
        L_DISCOUNT    DECIMAL(15,2) NOT NULL,
        L_TAX         DECIMAL(15,2) NOT NULL,
        L_RETURNFLAG  CHAR(1) NOT NULL,
        L_LINESTATUS  CHAR(1) NOT NULL,
        L_SHIPDATE    DATE NOT NULL,
        L_COMMITDATE  DATE NOT NULL ,
        L_RECEIPTDATE DATE NOT NULL,
        L_SHIPINSTRUCT CHAR(25) NOT NULL,
        L_SHIPMODE     CHAR(10) NOT NULL,
        L_COMMENT      VARCHAR(44) NOT NULL
)
with (orientation = column,COMPRESSION=MIDDLE)
distribute by hash(L_ORDERKEY);
```

步骤2 创建外表。外表用于识别和关联OBS上的源数据。

须知

其中，*<obs_bucket_name>*代表OBS桶名。本实践以“华北-北京四”地区为例，可填入dws-demo-cn-north-4，<Access_Key_Id>和<Secret_Access_Key>替换为实际值，在**准备工作**获取。

```
CREATE schema tpchobs;
set current_schema='tpchobs';
drop FOREIGN table if exists region;
CREATE FOREIGN TABLE REGION
(
        like tpch.region
)
SERVER gsmpp_server
OPTIONS (
        encoding 'utf8',
        location 'obs://<obs_bucket_name>/tpch/region.tbl',
        format 'text',
        delimiter '|',
        access_key '<Access_Key_Id>',
        secret_access_key '<Secret_Access_Key>',
        chunksize '64',
        IGNORE_EXTRA_DATA 'on'
);

drop FOREIGN table if exists nation;
CREATE FOREIGN TABLE NATION
(
        like tpch.nation
)
SERVER gsmpp_server
OPTIONS (
         encoding 'utf8',
         location 'obs://<obs_bucket_name>/tpch/nation.tbl',
         format 'text',
         delimiter '|',
         access_key '<Access_Key_Id>',
         secret_access_key '<Secret_Access_Key>',
         chunksize '64',
```



```
        IGNORE_EXTRA_DATA 'on'
);

drop FOREIGN table if exists supplier;
CREATE FOREIGN TABLE SUPPLIER
(
        like tpch.supplier
)
SERVER gsmpp_server
OPTIONS (
        encoding 'utf8',
        location 'obs://<obs_bucket_name>/tpch/supplier.tbl',
        format 'text',
        delimiter '|',
        access_key '<Access_Key_Id>',
        secret_access_key '<Secret_Access_Key>',
        chunksize '64',
        IGNORE_EXTRA_DATA 'on'
);

drop FOREIGN table if exists customer;
CREATE FOREIGN TABLE CUSTOMER
(
        like tpch.customer
)
SERVER gsmpp_server
OPTIONS (
        encoding 'utf8',
        location 'obs://<obs_bucket_name>/tpch/customer.tbl',
        format 'text',
        delimiter '|',
        access_key '<Access_Key_Id>',
        secret_access_key '<Secret_Access_Key>',
        chunksize '64',
        IGNORE_EXTRA_DATA 'on'
);
drop FOREIGN table if exists part;
CREATE FOREIGN TABLE PART
(
        like tpch.part

)
SERVER gsmpp_server
OPTIONS (
        encoding 'utf8',
        location 'obs://<obs_bucket_name>/tpch/part.tbl',
        format 'text',
        delimiter '|',
        access_key '<Access_Key_Id>',
        secret_access_key '<Secret_Access_Key>',
        chunksize '64',
        IGNORE_EXTRA_DATA 'on'
);
drop FOREIGN table if exists partsupp;
CREATE FOREIGN TABLE PARTSUPP
(
        like tpch.partsupp
)
SERVER gsmpp_server
OPTIONS (
        encoding 'utf8',
        location 'obs://<obs_bucket_name>/tpch/partsupp.tbl',
        format 'text',
        delimiter '|',
        access_key '<Access_Key_Id>',
        secret_access_key '<Secret_Access_Key>',
        chunksize '64',
        IGNORE_EXTRA_DATA 'on'
);
```



```
drop FOREIGN table if exists orders;
CREATE FOREIGN TABLE ORDERS
(
    like tpch.orders
)
SERVER gsmpp_server
OPTIONS (
    encoding 'utf8',
    location 'obs://<obs_bucket_name>/tpch/orders.tbl',
    format 'text',
    delimiter '|',
    access_key '<Access_Key_Id>',
    secret_access_key '<Secret_Access_Key>',
    chunksize '64',
    IGNORE_EXTRA_DATA 'on'
);
drop FOREIGN table if exists lineitem;
CREATE FOREIGN TABLE LINEITEM
(
    like tpch.lineitem
)
SERVER gsmpp_server
OPTIONS (
    encoding 'utf8',
    location 'obs://<obs_bucket_name>/tpch/lineitem.tbl',
    format 'text',
    delimiter '|',
    access_key '<Access_Key_Id>',
    secret_access_key '<Secret_Access_Key>',
    chunksize '64',
    IGNORE_EXTRA_DATA 'on'
);
```

步骤3 复制并执行以下语句，将外表数据导入到对应的数据库表中。

将OBS外表的数据通过insert命令导入GaussDB(DWS) 的数据库表中，数据库内核对应的操作为OBS数据高速并发导入GaussDB(DWS) 。

```
insert into tpch.lineitem select * from tpchobs.lineitem;
insert into tpch.part select * from tpchobs.part;
insert into tpch.partsupp select * from tpchobs.partsupp;
insert into tpch.customer select * from tpchobs.customer;
insert into tpch.supplier select * from tpchobs.supplier;
insert into tpch.nation select * from tpchobs.nation;
insert into tpch.region select * from tpchobs.region;
insert into tpch.orders select * from tpchobs.orders;
```

导入数据需要约10分钟，请耐心等待。

----结束

步骤二：多表分析与主题分析

以下以TPC-H标准查询为例，演示在GaussDB(DWS) 中进行的基本数据查询。

在进行数据查询之前，请先执行“Analyze”命令生成与数据库表相关的统计信息。统计信息存储在系统表PG_STATISTIC中，执行计划生成器会使用这些统计数据，以生成最有效的查询执行计划。

查询示例如下：

- **某地区供货商为公司带来的收入查询（TPCH-Q5）**

  通过执行TPCH-Q5查询语句，可以查询到通过某个地区零件供货商获得的收入（收入按sum(l_extendedprice * (1 - l_discount))计算）统计信息。该统计信息可用于决策在给定的区域是否需要建立一个当地分配中心。

复制并执行以下TPCH-Q5语句进行查询。该语句的特点是：带有分组、排序、聚集操作并存的多表连接查询操作。

```
set current_schema='tpch';
Select
n_name,
sum(l_extendedprice * (1 - l_discount)) as revenue
from
customer,
orders,
lineitem,
supplier,
nation,
region
where
c_custkey = o_custkey
and l_orderkey = o_orderkey
and l_suppkey = s_suppkey
and c_nationkey = s_nationkey
and s_nationkey = n_nationkey
and n_regionkey = r_regionkey
and r_name = 'ASIA'
and o_orderdate >= '1994-01-01'::date
and o_orderdate < '1994-01-01'::date + interval '1 year'
group by
n_name
order by
revenue desc;
```

- **零件/供货商关系查询（TPCH-Q16）**

  通过执行TPCH-Q16查询语句，可以获得能够以指定的贡献条件供应零件的供货商数量。该信息可用于决策在订单量大，任务紧急时，是否有充足的供货商。

  复制并执行以下TPCH-Q16语句进行查询，该语句的特点是：带有分组、排序、聚集、去重、NOT IN子查询操作并存的多表连接操作。

```
set current_schema='tpch';
select
p_brand,
p_type,
p_size,
count(distinct ps_suppkey) as supplier_cnt
from
partsupp,
part
where
p_partkey = ps_partkey
and p_brand <> 'Brand#45'
and p_type not like 'MEDIUM POLISHED%'
and p_size in (49, 14, 23, 45, 19, 3, 36, 9)
and ps_suppkey not in (
        select
        s_suppkey
        from
        supplier
        where
        s_comment like '%Customer%Complaints%'
)
group by
p_brand,
p_type,
p_size
order by
supplier_cnt desc,
p_brand,
p_type,
p_size
limit 100;
```


- **小订单收入损失查询（TPCH-Q17）**

  通过查询得知如果没有小量订单，平均年收入将损失多少。筛选出比平均供货量的20%还低的小批量订单，如果这些订单不再对外供货，由此计算平均一年的损失。

  复制并执行以下TPCH-Q17语句进行查询，该语句的特点是：带有聚集、聚集子查询操作并存的两表连接操作。

```
set current_schema='tpch';
select
sum(l_extendedprice) / 7.0 as avg_yearly
from
lineitem,
part
where
p_partkey = l_partkey
and p_brand = 'Brand#23'
and p_container = 'MED BOX'
and l_quantity < (
        select 0.2 * avg(l_quantity)
        from lineitem
        where l_partkey = p_partkey
);
```

1.4.3 零售业百货公司经营状况分析

零售业百货公司样例简介

本实践将演示以下场景：从OBS加载各个零售商场每日经营的业务数据到数据仓库对应的表中，然后对商铺营业额、客流信息、月度销售排行、月度客流转化率、月度租售比、销售坪效等KPI信息进行汇总和查询。本示例旨在展示在零售业场景中GaussDB(DWS) 数据仓库的多维度查询分析的能力。

说明

- GaussDB(DWS) 已预先将样例数据上传到OBS桶的“retail-data”文件夹中，并给所有华为云用户赋予了该OBS桶的只读访问权限。
- 当前“北京一”区域的OBS桶暂无该样例数据，推荐使用“北京四”或其他区域的集群进行体验。

本实践预计时长60分钟，基本流程如下：

1. **准备工作**
2. **步骤一：导入零售业百货公司样例数据**
3. **步骤二：经营状况分析**

准备工作

- 已注册公有云帐号，帐号不能处于欠费或冻结状态。
- 获取此帐号的“AK/SK”。
- 已创建集群，并已使用Data Studio连接集群，参见**步骤一：创建集群**和**步骤二：使用Data Studio连接集群**。

步骤一：导入零售业百货公司样例数据

使用SQL客户端工具连接到集群后，就可以在SQL客户端工具中，执行以下步骤导入零售业百货公司样例数据并执行查询。

步骤1 执行以下语句，创建retail数据库。

```
create database retail encoding 'utf8' template template0;
```

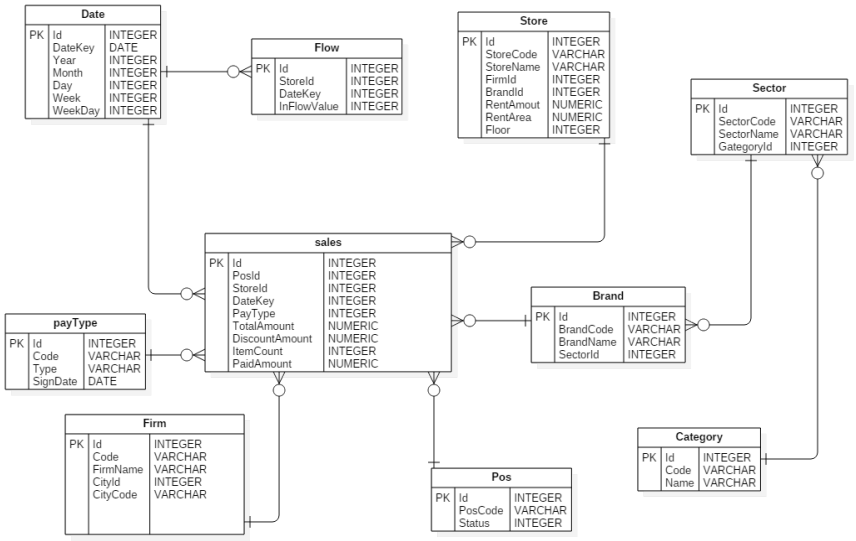
步骤2 执行以下步骤切换为连接新建的数据库。

1. 在Data Studio客户端的**“对象浏览器”**窗口，右键单击数据库连接名称，在弹出菜单中单击“刷新”，刷新后就可以看到新建的数据库。
2. 右键单击“retail”数据库名称，在弹出菜单中单击“打开连接”。
3. 右键单击“retail”数据库名称，在弹出菜单中单击“打开新的终端”，即可打开连接到指定数据库的SQL命令窗口，后面的步骤，请全部在该命令窗口中执行。

步骤3 创建数据库表。

样例数据包含10张数据库表，其关联关系如图1-12所示。

图 1-12 百货公司样例数据表



复制并执行以下语句，创建零售业百货公司信息数据库表。

```
create schema retail_data;
set current_schema='retail_data';

DROP TABLE IF EXISTS STORE;
CREATE TABLE STORE (
        ID INT,
        STORECODE VARCHAR(10),
        STORENAME VARCHAR(100),
        FIRMID INT,
        FLOOR INT,
        BRANDID INT,
        RENTAMOUNT NUMERIC(18,2),
        RENTAREA NUMERIC(18,2)
)
WITH (ORIENTATION = COLUMN, COMPRESSION=MIDDLE) DISTRIBUTE BY REPLICATION;

DROP TABLE IF EXISTS POS;
CREATE TABLE POS(
        ID INT,
        POSCODE VARCHAR(20),
        STATUS INT,
        MODIFICATIONDATE DATE
)
WITH (ORIENTATION = COLUMN, COMPRESSION=MIDDLE) DISTRIBUTE BY REPLICATION;
```



```
DROP TABLE IF EXISTS BRAND;
CREATE TABLE BRAND (
    ID INT,
    BRANDCODE VARCHAR(10),
    BRANDNAME VARCHAR(100),
    SECTORID INT
)
WITH (ORIENTATION = COLUMN, COMPRESSION=MIDDLE) DISTRIBUTE BY REPLICATION;

DROP TABLE IF EXISTS SECTOR;
CREATE TABLE SECTOR(
    ID INT,
    SECTORCODE VARCHAR(10),
    SECTORNAME VARCHAR(20),
    CATEGORYID INT
)
WITH (ORIENTATION = COLUMN, COMPRESSION=MIDDLE) DISTRIBUTE BY REPLICATION;

DROP TABLE IF EXISTS CATEGORY;
CREATE TABLE CATEGORY(
    ID INT,
    CODE VARCHAR(10),
    NAME VARCHAR(20)
)
WITH (ORIENTATION = COLUMN, COMPRESSION=MIDDLE) DISTRIBUTE BY REPLICATION;

DROP TABLE IF EXISTS FIRM;
CREATE TABLE FIRM(
    ID INT,
    CODE VARCHAR(4),
    NAME VARCHAR(40),
    CITYID INT,
    CITYNAME VARCHAR(10),
    CITYCODE VARCHAR(20)
)
WITH (ORIENTATION = COLUMN, COMPRESSION=MIDDLE) DISTRIBUTE BY REPLICATION;

DROP TABLE IF EXISTS DATE;
CREATE TABLE DATE(
    ID INT,
    DATEKEY DATE,
    YEAR INT,
    MONTH INT,
    DAY INT,
    WEEK INT,
    WEEKDAY INT
)
WITH (ORIENTATION = COLUMN, COMPRESSION=MIDDLE) DISTRIBUTE BY REPLICATION;

DROP TABLE IF EXISTS PAYTYPE;
CREATE TABLE PAYTYPE(
    ID INT,
    CODE VARCHAR(10),
    TYPE VARCHAR(10),
    SIGNDATE DATE
)
WITH (ORIENTATION = COLUMN, COMPRESSION=MIDDLE) DISTRIBUTE BY REPLICATION;

DROP TABLE IF EXISTS SALES;
CREATE TABLE SALES(
     ID INT,
     POSID INT,
     STOREID INT,
     DATEKEY INT,
     PAYTYPE INT,
     TOTALAMOUNT NUMERIC(18,2),
     DISCOUNTAMOUNT NUMERIC(18,2),
     ITEMCOUNT INT,
```



```
        PAIDAMOUNT NUMERIC(18,2)
)
WITH (ORIENTATION = COLUMN, COMPRESSION=MIDDLE) DISTRIBUTE BY HASH(ID);

DROP TABLE IF EXISTS FLOW;
CREATE TABLE FLOW (
        ID INT,
        STOREID INT,
        DATEKEY INT,
        INFLOWVALUE INT
)
WITH (ORIENTATION = COLUMN, COMPRESSION=MIDDLE) DISTRIBUTE BY HASH(ID);
```

步骤4 创建外表。外表用于识别和关联OBS上的源数据。

须知

其中，*<obs_bucket_name>*代表OBS桶名。本实践以“华北-北京四”地区为例，可填入dws-demo-cn-north-4，<Access_Key_Id>和<Secret_Access_Key>替换为实际值，在**准备工作**获取。

```
create schema retail_obs_data;
set current_schema='retail_obs_data';
drop FOREIGN table if exists SALES_OBS;
CREATE FOREIGN TABLE SALES_OBS
(
        like retail_data.SALES
)
SERVER gsmpp_server
OPTIONS (
        encoding 'utf8',
        location 'obs://<obs_bucket_name>/retail-data/sales',
        format 'csv',
        delimiter ',',
        access_key '<Access_Key_Id>',
        secret_access_key '<Secret_Access_Key>',
        chunksize '64',
        IGNORE_EXTRA_DATA 'on',
        header 'on'
);

drop FOREIGN table if exists FLOW_OBS;
CREATE FOREIGN TABLE FLOW_OBS
(
        like retail_data.flow
)
SERVER gsmpp_server
OPTIONS (
        encoding 'utf8',
        location 'obs://<obs_bucket_name>/retail-data/flow',
        format 'csv',
        delimiter ',',
        access_key '<Access_Key_Id>',
        secret_access_key '<Secret_Access_Key>',
        chunksize '64',
        IGNORE_EXTRA_DATA 'on',
        header 'on'
);

drop FOREIGN table if exists BRAND_OBS;
CREATE FOREIGN TABLE BRAND_OBS
(
        like retail_data.brand
)
SERVER gsmpp_server
OPTIONS (
```



```
        encoding 'utf8',
        location 'obs://<obs_bucket_name>/retail-data/brand',
        format 'csv',
        delimiter ',',
        access_key '<Access_Key_Id>',
        secret_access_key '<Secret_Access_Key>',
        chunksize '64',
        IGNORE_EXTRA_DATA 'on',
        header 'on'
);


drop FOREIGN table if exists CATEGORY_OBS;
CREATE FOREIGN TABLE CATEGORY_OBS
(
      like retail_data.category
)
SERVER gsmpp_server
OPTIONS (
       encoding 'utf8',
       location 'obs://<obs_bucket_name>/retail-data/category',
       format 'csv',
       delimiter ',',
       access_key '<Access_Key_Id>',
       secret_access_key '<Secret_Access_Key>',
       chunksize '64',
       IGNORE_EXTRA_DATA 'on',
       header 'on'
);

drop FOREIGN table if exists DATE_OBS;
CREATE FOREIGN TABLE DATE_OBS
(
       like retail_data.date
)
SERVER gsmpp_server
OPTIONS (
        encoding 'utf8',
        location 'obs://<obs_bucket_name>/retail-data/date',
        format 'csv',
        delimiter ',',
        access_key '<Access_Key_Id>',
        secret_access_key '<Secret_Access_Key>',
        chunksize '64',
        IGNORE_EXTRA_DATA 'on',
        header 'on'
);

drop FOREIGN table if exists FIRM_OBS;
CREATE FOREIGN TABLE FIRM_OBS
(
       like retail_data.firm
)
SERVER gsmpp_server
OPTIONS (
        encoding 'utf8',
        location 'obs://<obs_bucket_name>/retail-data/firm',
        format 'csv',
        delimiter ',',
        access_key '<Access_Key_Id>',
        secret_access_key '<Secret_Access_Key>',
        chunksize '64',
        IGNORE_EXTRA_DATA 'on',
        header 'on'
);


drop FOREIGN table if exists PAYTYPE_OBS;
CREATE FOREIGN TABLE PAYTYPE_OBS
```



```
(
	like retail_data.paytype
)
SERVER gsmpp_server
OPTIONS (
	encoding 'utf8',
	location 'obs://<obs_bucket_name>/retail-data/paytype',
	format 'csv',
	delimiter ',',
	access_key '<Access_Key_Id>',
	secret_access_key '<Secret_Access_Key>',
	chunksize '64',
	IGNORE_EXTRA_DATA 'on',
	header 'on'
);


drop FOREIGN table if exists POS_OBS;
CREATE FOREIGN TABLE POS_OBS
(
	like retail_data.pos
)
SERVER gsmpp_server
OPTIONS (
	encoding 'utf8',
	location 'obs://<obs_bucket_name>/retail-data/pos',
	format 'csv',
	delimiter ',',
	access_key '<Access_Key_Id>',
	secret_access_key '<Secret_Access_Key>',
	chunksize '64',
	IGNORE_EXTRA_DATA 'on',
	header 'on'
);

drop FOREIGN table if exists SECTOR_OBS;
CREATE FOREIGN TABLE SECTOR_OBS
(
	like retail_data.sector
)
SERVER gsmpp_server
OPTIONS (
	encoding 'utf8',
	location 'obs://<obs_bucket_name>/retail-data/sector',
	format 'csv',
	delimiter ',',
	access_key '<Access_Key_Id>',
	secret_access_key '<Secret_Access_Key>',
	chunksize '64',
	IGNORE_EXTRA_DATA 'on',
	header 'on'
);


drop FOREIGN table if exists STORE_OBS;
CREATE FOREIGN TABLE STORE_OBS
(
	 like retail_data.store
)
SERVER gsmpp_server
OPTIONS (
	 encoding 'utf8',
	 location 'obs://<obs_bucket_name>/retail-data/store',
	 format 'csv',
	 delimiter ',',
	 access_key '<Access_Key_Id>',
	 secret_access_key '<Secret_Access_Key>',
	 chunksize '64',
	 IGNORE_EXTRA_DATA 'on',
```



```
    header 'on'
);
```

步骤5 复制并执行以下语句，导入外表数据到集群。

```
insert into retail_data.store select * from retail_obs_data.STORE_OBS;
insert into retail_data.sector select * from retail_obs_data.SECTOR_OBS;
insert into retail_data.paytype select * from retail_obs_data.PAYTYPE_OBS;
insert into retail_data.firm select * from retail_obs_data.FIRM_OBS;
insert into retail_data.flow select * from retail_obs_data.FLOW_OBS;
insert into retail_data.category select * from retail_obs_data.CATEGORY_OBS;
insert into retail_data.date select * from retail_obs_data.DATE_OBS;
insert into retail_data.pos select * from retail_obs_data.POS_OBS;
insert into retail_data.brand select * from retail_obs_data.BRAND_OBS;
insert into retail_data.sales select * from retail_obs_data.SALES_OBS;
```

导入数据需要一些时间，请耐心等待。

步骤6 复制并执行以下语句，创建视图v_sales_flow_details。

```
set current_schema='retail_data';
CREATE VIEW v_sales_flow_details AS
SELECT
FIRM.ID FIRMID, FIRM.NAME FIRNAME, FIRM. CITYCODE,
CATEGORY.ID CATEGORYID, CATEGORY.NAME CATEGORYNAME,
SECTOR.ID SECTORID, SECTOR.SECTORNAME,
BRAND.ID BRANDID, BRAND.BRANDNAME,
STORE.ID STOREID, STORE.STORENAME, STORE.RENTAMOUNT, STORE.RENTAREA,
DATE.DATEKEY, SALES.TOTALAMOUNT, DISCOUNTAMOUNT, ITEMCOUNT, PAIDAMOUNT, INFLOWVALUE
FROM SALES
INNER JOIN STORE ON SALES.STOREID = STORE.ID
INNER JOIN FIRM ON STORE.FIRMID = FIRM.ID
INNER JOIN BRAND ON STORE.BRANDID = BRAND.ID
INNER JOIN SECTOR ON BRAND.SECTORID = SECTOR.ID
INNER JOIN CATEGORY ON SECTOR.CATEGORYID = CATEGORY.ID
INNER JOIN DATE ON SALES.DATEKEY = DATE.ID
INNER JOIN FLOW ON FLOW.DATEKEY = DATE.ID AND FLOW.STOREID = STORE.ID;
```

----结束

步骤二：经营状况分析

以下以零售百货公司标准查询为例，演示在GaussDB(DWS) 中进行的基本数据查询。

在进行数据查询之前，请先执行“Analyze”命令生成与数据库表相关的统计信息。统计信息存储在系统表PG_STATISTIC中，执行计划生成器会使用这些统计数据，以生成最有效的查询执行计划。

查询示例如下：

- **查询各商铺的总营业额**

  复制并执行以下语句查询各商铺的总营业额。

  ```
  set current_schema='retail_data';
  SELECT DATE_TRUNC('month',datekey)
  AT TIME ZONE 'UTC' AS __timestamp,
  SUM(paidamount)
  AS sum__paidamount
  FROM v_sales_flow_details
  GROUP BY DATE_TRUNC('month',datekey) AT TIME ZONE 'UTC'
  ORDER BY SUM(paidamount) DESC;
  ```

- **查询各门店营收及租售比状况**

  复制并执行以下语句进行营收及租售比状况查询。

  ```
  set current_schema='retail_data';
  SELECT firname AS firname,
  storename AS storename,
  ```



```
SUM(paidamount)
AS sum__paidamount,
AVG(RENTAMOUNT)/SUM(PAIDAMOUNT)
AS rentamount_sales_rate
FROM v_sales_flow_details
GROUP BY firname, storename
ORDER BY SUM(paidamount) DESC;
```

- **各省营业汇总分析**

  复制并执行以下语句进行汇总分析查询。

```
set current_schema='retail_data';
SELECT citycode AS citycode,
SUM(paidamount)
AS sum__paidamount
FROM v_sales_flow_details
GROUP BY citycode
ORDER BY SUM(paidamount) DESC;
```

- **各门店租售比和客流转化率对比分析**

```
set current_schema='retail_data';
SELECT brandname AS brandname,
firname AS firname,
SUM(PAIDAMOUNT)/AVG(RENTAREA) AS sales_rentarea_rate,
SUM(ITEMCOUNT)/SUM(INFLOWVALUE) AS poscount_flow_rate,
AVG(RENTAMOUNT)/SUM(PAIDAMOUNT) AS rentamount_sales_rate
FROM v_sales_flow_details
GROUP BY brandname,  firname
ORDER BY sales_rentarea_rate DESC;
```

- **品牌业态分析**

```
set current_schema='retail_data';
SELECT categoryname AS categoryname,
brandname AS brandname,
SUM(paidamount) AS sum__paidamount
FROM v_sales_flow_details
GROUP BY categoryname,
brandname
ORDER BY sum__paidamount DESC;
```

- **查询各品牌每日营业状况**

```
set current_schema='retail_data';
SELECT brandname AS brandname,
DATE_TRUNC('day', datekey) AT TIME ZONE 'UTC' AS __timestamp,
SUM(paidamount) AS sum__paidamount
FROM v_sales_flow_details
WHERE datekey >= '2016-01-01 00:00:00'
AND datekey <= '2016-01-30 00:00:00'
GROUP BY brandname,
DATE_TRUNC('day', datekey) AT TIME ZONE 'UTC'
ORDER BY sum__paidamount ASC
LIMIT 50000;
```

1.5 第 5 步：查看其它资料并清理资源

查看其它资料

完成如上操作步骤后，我们推荐您可以参考如下资料继续对数据仓库服务进行更详细深入的了解：

- **《数据仓库服务管理指南》**：本指南在此入门的基础上，对创建、管理、监控以及连接集群的概念和相关操作提供全面详细的信息。
- **《数据仓库服务数据库开发指南》**：本指南在此入门的基础上，为数据库开发人员提供全面详细的信息，帮助他们了解如何构建、管理和查询GaussDB(DWS) 数据库，包括SQL语法、用户管理、数据导入导出等指导。

清理资源

当完成快速入门的样例后，如果您不再需要使用本样例创建的样例数据、集群、ECS以及VPC时，您可以删除这些资源，以免资源浪费或占用您的配额。

步骤1 删除GaussDB(DWS) 集群。

在GaussDB(DWS) 管理控制台，单击“集群管理”，在集群列表中集群“dws-demo”所在行，单击“更多 > 删除”。然后在弹出对话框中勾选“释放与集群绑定的弹性IP”，单击“确定”。

如果待删除集群使用了自动创建的安全组，且该自动创建的安全组没有被别的集群使用，删除集群时，该安全组也会一起被自动删除。

步骤2 删除子网。删除前请先确保该子网未被其他资源绑定。

登录虚拟私有云管理控制台，在左侧导航树单击“虚拟私有云”，在虚拟私有云列表中，单击名称“vpc-dws”，然后在子网列表中“subnet-dws”所在行单击“删除”。

步骤3 删除虚拟私有云。删除前请先确保该虚拟私有云未被其他资源绑定。

登录虚拟私有云管理控制台，在虚拟私有云列表中，找到虚拟私有云“vpc-dws)”，单击其所在行的“删除”。

具体步骤，请参见《虚拟私有云用户指南》中“虚拟私有云和子网 > 删除虚拟私有云”章节。

----结束

2 数据库使用入门

2.1 从这里开始

通过本小节您可以快速完成创建数据库、创建表及向表中插入数据和查询表中数据。本章的子节更详细地讲解数据库使用的一些常用步骤。

数据库基本操作

步骤1 创建数据库用户。

默认只有创建集群时生成的管理员用户可以访问初始数据库。要向其他用户授予访问权限，必须创建新的用户帐户。

```
CREATE USER joe WITH PASSWORD "Bigdata@123";
```

当结果显示为如下信息，则表示创建成功。

```
CREATE USER
```

上面，创建了一个用户名为joe，密码为Bigdata@123的用户。

新创建的用户帐户默认具有所有数据库的登录权限和创建表、视图、索引等的权限及对这些自己所建对象的操作权限。更多信息请参见**用户**。

步骤2 创建数据库。

```
CREATE DATABASE mytpcds;
CREATE DATABASE
```

有关数据库管理的更多操作指导，请参考**创建和管理数据库**。

步骤3 （可选）创建schema。

schema又称作模式。通过schema，允许多个用户使用同一数据库而不相互干扰。

执行如下命令来创建一个schema。

```
CREATE SCHEMA myschema;
```

当结果显示如下信息，则表示成功创建一个名为myschema的schema。

```
CREATE SCHEMA
```

schema创建成功后，就可以在该schema下创建对象了。但是，请确保在创建对象前使用如下两种方式之一将对象创建到对应的schema下。

先将数据库的search_path设成对应schema，然后再创建对象。

```
SET SEARCH_PATH TO myschema;
CREATE TABLE mytable (firstcol int);
```

在创建对象时指定由“模式名称+对象名称”组成的完整对象名称，中间由符号“.”隔开。例如：

```
CREATE TABLE myschema.mytable (firstcol int);
```

如果在创建对象时不指定schema，则会将对象创建在当前的schema下。查询当前schema的办法为：

```
 show search_path;
  search_path
----------------
 "$user",public
(1 row)
```

创建完mytpcds数据库后，就可以按如下方法退出gaussdb数据库。

```
\q
```

有关schema的更多信息请参考**创建和管理schema**。

步骤4 创建表。

- 创建一个名称为mytable，只有一列的表。字段名为firstcol，字段类型为integer。

  ```
  mytpcds=>  CREATE TABLE mytable (firstcol int);
  ```

  未使用“DISTRIBUTE BY”指定分布列时，系统默认会使用第一个符合分布列数据类型要求的列为分布列，且给出提示。系统返回信息以“CREATE TABLE”结束，表示创建表成功。

  ```
  NOTICE:  The 'DISTRIBUTE BY' clause is not specified. Using 'firstcol' as the distribution column by default.
  HINT:  Please use 'DISTRIBUTE BY' clause to specify suitable data distribution column.
  CREATE TABLE
  ```

  PG_TABLES系统表包含集群中所有表的有关信息。通过SELECT 命令可以在此系统表中查看表的属性。

  ```
  SELECT * FROM PG_TABLES WHERE TABLENAME = 'mytable';
  ```

- 向表中插入数据：

  ```
  mytpcds=> INSERT INTO mytable values (100);
  INSERT 0 1
  ```

  INSERT 命令可向数据库表插入各个行。要进行标准的批量加载，请参阅**关于OBS并行导入**。

- 查看表中数据：

  ```
  mytpcds=> SELECT * from mytable;
   firstcol
  ----------
        100
  (1 row)
  ```


说明

- 默认情况下，新的数据库对象是创建在“public”模式下的，例如刚刚新建的表。关于模式的更多信息请参考**创建和管理schema**。
- 关于创建表的更多信息请参见**创建表**。
- 除了创建的表以外，数据库还包含很多系统表。这些系统表包含集群安装信息以及GaussDB(DWS)上运行的各种查询和进程的信息。可以通过查询系统表来收集有关数据库的信息。请参见**查看系统表**。
- GaussDB(DWS)支持行列混合存储，为各种复杂场景下的交互分析提供更好的查询性能，关于存储模型的选择，请参考**规划存储模型**。

----结束

加载示例数据

本手册中大部分示例均使用在gaussdb数据库中创建TPC-DS示例表。如果希望通过您的SQL查询工具按照示例操作，请先创建TPC-DS示例表，然后为示例表加载示例数据。

OBS桶中提供了示例数据，该存储桶向所有经过身份验证的云用户提供了读取权限。

有关创建表和加载示例数据的步骤，请参考**加载示例数据**。

释放资源

如果为完成本节的练习部署了集群，那么应在完成练习后删除该集群。

要删除集群，请按照**删除集群**进行操作。

如果要保留集群但清除db_tpcds数据库，请运行如下命令：

```
DROP DATABASE mytpcds;
```

如果要保留集群和数据库，只清空数据库中的表，请运行如下命令：

```
DROP TABLE mytable;
```

2.2 创建和管理数据库

前提条件

用户必须拥有数据库创建的权限或者是数据库的系统管理员权限才能创建数据库（如何赋予创建数据库的权限请参见**用户**）。

背景信息

- 初始时，GaussDB(DWS)包含两个模板数据库template0、template1，以及一个默认的用户数据库gaussdb。
- CREATE DATABASE实际上通过拷贝模板数据库来创建新数据库。默认情况下，拷贝template1。请避免使用客户端或其他手段连接及操作两个模板数据库。
- GaussDB(DWS)允许创建的数据库总数目上限为128个。
- 数据库系统中会有多个数据库，但是客户端程序一次只能连接一个数据库。也不能在不同的数据库之间相互查询。一个数据库集群中存在多个数据库时，需要通过-d参数指定相应的数据库实例进行连接。

操作步骤

步骤1 创建一个新的数据库db_tpcds。

```
CREATE DATABASE db_tpcds;
```

当结果显示如下信息，则表示创建成功。

```
CREATE DATABASE
```

正如**背景信息**中所说，创建数据库时默认拷贝模板数据库template1。template1的编码格式为SQL_ASCII。对于这种编码格式，在创建数据库对象时，如果对象名中含有多字节字符（例如中文），超过数据库对象名长度限制（63字节）的时候，系统会从最后一个字节（而不是字符）截断，可能造成出现半个字符的情况。

针对这种情况，请遵循以下条件：

- 保证数据对象的名称不超过限定长度。
- 不要使用多字节字符做为对象名。

如果出现因为误操作导致在多字节字符的中间截断进而无法删除数据库对象的现象，请使用截断前的数据库对象名进行删除操作，或将该对象从各个数据库节点的相应系统表中依次删掉。

您也可以通过指示CREATE DATABASE使用template0取代template1进行拷贝，在复制tmplate0时指定新的编码和区域设置。例如使用utf-8编码集做为数据库的默认存储编码集（server_encoding）。详细请参见CREATE DATABASE的语法指导。

通过“show server_encoding”命令可以查看当前数据库存储编码。

说明

- 数据库名称遵循SQL标识符的一般规则。当前用户自动成为此新数据库的所有者。
- 如果一个数据库系统用于承载相互独立的用户和项目，建议把它们放在不同的数据库里。
- 如果项目或者用户是相互关联的，并且可以相互使用对方的资源，则应该把它们放在同一个数据库里，但可以规划在不同的Schema中。Schema只是一个纯粹的逻辑结构，某个Schema的访问权限由**三权分立**的 “表1 默认的用户权限”控制。

步骤2 查看数据库

- 使用\l元命令查看数据库系统的数据库列表。
  ```
  \l
  ```
- 使用以下命令通过系统表pg_database查询数据库列表：
  ```
  SELECT datname FROM pg_database;
  ```

步骤3 修改数据库

用户可以使用ALTER DATABASE命令修改数据库属性（比如：owner、名称和默认的配置属性）。

- 使用以下命令为数据库设置默认的模式搜索路径：
  ```
  ALTER DATABASE db_tpcds SET search_path TO pa_catalog,public;
  ALTER DATABASE
  ```
- 使用以下命令修改数据库表空间：
  ```
  ALTER DATABASE db_tpcds SET TABLESPACE tpcds;
  ALTER DATABASE
  ```
- 使用以下命令为数据库重新命名：
  ```
  ALTER DATABASE db_tpcds RENAME TO human_tpcds;
  ALTER DATABASE
  ```

步骤4 删除数据库

用户可以使用DROP DATABASE命令删除数据库。这个命令删除了数据库中的系统目录，并且删除了带有数据的磁盘上的数据库目录。用户必须是数据库的owner或者系统管理员才能删除数据库。当有人连接数据库时，删除操作会失败。删除数据库时请先连接到其他的数据库。

使用如下命令删除数据库：

```
DROP DATABASE human_tpcds;
DROP DATABASE
```

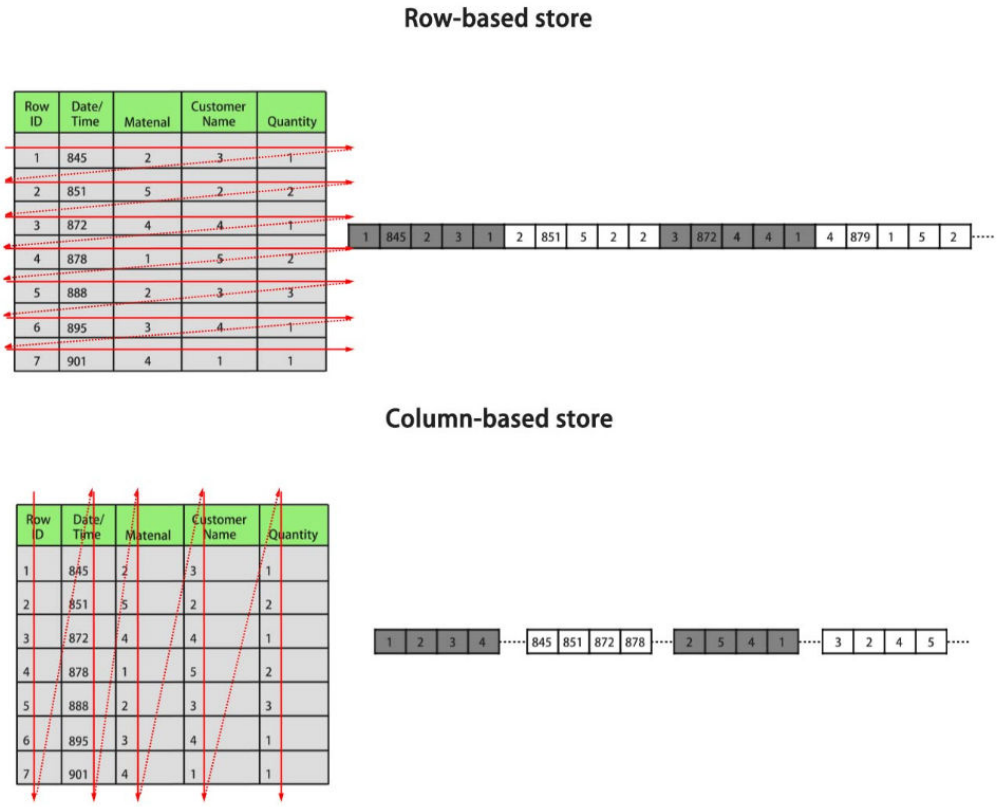
----结束

2.3 规划存储模型

GaussDB(DWS)支持行列混合存储。行、列存储模型各有优劣，建议根据实际情况选择。

行存储是指将表按行存储到硬盘分区上，列存储是指将表按列存储到硬盘分区上。默认情况下，创建的表为行存储。行存储和列存储的差异请参见图2-1。

图 2-1 行存储和列存储的差异



上图中，左上为行存表，右上为行存表在硬盘上的存储方式。左下为列存表，右下为列存表在硬盘上的存储方式。

行、列存储有如下优缺点：

存储模型	优点	缺点
行存	数据被保存在一起。INSERT/UPDATE容易。	选择(Selection)时即使只涉及某几列，所有数据也都会被读取。
列存	● 查询时只有涉及到的列会被读取。 ● 投影(Projection)很高效。 ● 任何列都能作为索引。	● 选择完成时，被选择的列要重新组装。 ● INSERT/UPDATE比较麻烦。

一般情况下，如果表的字段比较多（大宽表），查询中涉及到的列不多的情况下，适合列存储。如果表的字段个数比较少，查询大部分字段，那么选择行存储比较好。

存储类型	适用场景
行存	● 点查询(返回记录少，基于索引的简单查询)。 ● 增、删、改操作较多的场景。
列存	● 统计分析类查询 (关联、分组操作较多的场景)。 ● 即席查询（查询条件不确定，行存表扫描难以使用索引）。

行存表

默认创建表的类型。数据按行进行存储，即一行数据是连续存储。适用于对数据需要经常更新的场景。

```
CREATE TABLE customer_t1
(
  state_ID   CHAR(2),
  state_NAME VARCHAR2(40),
  area_ID    NUMBER
);

--删除表
DROP TABLE customer_t1;
```

列存表

数据按列进行存储，即一列所有数据是连续存储的。单列查询IO小，比行存表占用更少的存储空间。适合数据批量插入、更新较少和以查询为主统计分析类的场景。列存表不适合点查询。

```
CREATE TABLE customer_t2
(
  state_ID   CHAR(2),
  state_NAME VARCHAR2(40),
  area_ID    NUMBER
)
WITH (ORIENTATION = COLUMN);

--删除表
DROP TABLE customer_t2;
```

2.4 创建和管理表

2.4.1 创建表

背景信息

表是建立在数据库中的，在不同的数据库中可以存放相同的表。甚至可以通过使用模式在同一个数据库中创建相同名称的表。创建表前请先**规划存储模型**。

创建表

执行如下命令创建表。

```
CREATE TABLE customer_t1
(
    c_customer_sk             integer,
    c_customer_id             char(5),
    c_first_name              char(6),
    c_last_name               char(8)
)
with (orientation = column,compression=middle)
distribute by hash (c_last_name);
```

当结果显示为如下信息，则表示创建成功。

```
CREATE TABLE
```

其中c_customer_sk 、c_customer_id、c_first_name和c_last_name是表的字段名，integer、char(5)、char(6)和char(8)分别是这四字段名称的类型。

2.4.2 向表中插入数据

在创建一个表后，表中并没有数据，在使用这个表之前，需要向表中插入数据。本小节介绍如何使用**INSERT**命令插入一行或多行数据，及从指定表插入数据。如果有大量数据需要批量导入表中，请参考**导入方式说明**。

背景信息

服务端与客户端使用不同的字符集时，两者字符集中单个字符的长度也会不同，客户端输入的字符串会以服务端字符集的格式进行处理，所以产生的最终结果可能会与预期不一致。

表 2-1 客户端和服务端设置字符集的输出结果对比

操作过程	服务端和客户端编码一致	服务端和客户端编码不一致
存入和取出过程中没有对字符串进行操作	输出预期结果	输出预期结果（输入与显示的客户端编码必须一致）。
存入取出过程对字符串有做一定的操作（如字符串函数操作）	输出预期结果	根据对字符串具体操作可能产生非预期结果。

操作过程	服务端和客户端编码一致	服务端和客户端编码不一致
存入过程中对超长字符串有截断处理	输出预期结果	字符集中字符编码长度是否一致，如果不一致可能会产生非预期的结果。

上述字符串函数操作和自动截断产生的效果会有叠加效果，例如：在客户端与服务端字符集不一致的场景下，如果既有字符串操作，又有字符串截断，在字符串被处理完以后的情况下继续截断，这样也会产生非预期的效果。详细的示例请参见**表2-2**。

说明

数据库DBCOMPATIBILITY设为兼容TD（Teradata）模式，且**td_compatible_truncation**参数设置为on的情况下，才会对超长字符串进行截断。

执行如下命令建立示例中需要使用的表table1、table2。

```
CREATE TABLE table1(id int, a char(6), b varchar(6),c varchar(6)) ;
CREATE TABLE table2(id int, a char(20), b varchar(20),c varchar(20)) ;
```

表 2-2 示例

编号	服务端字符集	客户端字符集	是否启用自动截断	示例	结果	说明
1	SQL_ASCII	UTF8	是	INSERT INTO table1 VALUES(1,reverse('123A A 78'),reverse('123A A 78'),reverse('123A A 78'));	id |a|b|c ----+------+------+------ 1 | 87| 87| 87	字符串在服务端翻转后，并进行截断，由于服务端和客户端的字符集不一致，字符A在客户端由多个字节表示，结果产生异常。
2	SQL_ASCII	UTF8	是	INSERT INTO table1 VALUES(2,reverse('123A 78'),reverse('123A 78'),reverse('123A 78'));	id |a|b|c ----+------+------+------ 2 | 873| 873| 873	字符串翻转后，又进行了自动截断，所以产生了非预期的效果。
3	SQL_ASCII	UTF8	是	INSERT INTO table1 VALUES(3,'87A 123','87A 123','87A 123');	id | a | b | c ----+-------+-------+------- 3 | 87A1 | 87A1 | 87A1	字符串类型的字段长度是客户端字符编码长度的整数倍，所以截断后产生结果正常。

编号	服务端字符集	客户端字符集	是否启用自动截断	示例	结果	说明
4	SQL_ASCII	UTF8	否	**INSERT INTO** *table2* **VALUES**(*1,reverse('123A A 78'),reverse('123A A 78'),reverse('123A A 78')*); **INSERT INTO** *table2* **VALUES**(*2,reverse('123A 78'),reverse('123 A 78'),reverse('123A 78')*);	id |a|b|c ---- +---------- --+-------- +-------- 1 | 87 321| 87 321 | 87 321 2 | 87321| 87321| 87321	与示例1类似，多字节字符翻转之后不再表示原来的字符。

常见操作

向表中插入数据前，意味着表已创建成功。创建表的步骤请参考**创建和管理表**。

- 向表customer_t1中插入一行：

  数据值是按照这些字段在表中出现的顺序列出的，并且用逗号分隔。通常数据值是文本（常量），但也允许使用标量表达式。

  ```
  INSERT INTO customer_t1(c_customer_sk, c_customer_id, c_first_name) VALUES (3769, 'hello', 'Grace');
  ```

  如果用户已经知道表中字段的顺序，也可无需列出表中的字段。例如以下命令与上面的命令效果相同。

  ```
  INSERT INTO customer_t1 VALUES (3769, 'hello', 'Grace');
  ```

  如果用户不知道所有字段的数值，可以忽略其中的一些。没有数值的字段将被填充为字段的缺省值。例如：

  ```
  INSERT INTO customer_t1 (c_customer_sk, c_first_name) VALUES (3769, 'Grace');

  INSERT INTO customer_t1 VALUES (3769, 'hello');
  ```

  用户也可以对独立的字段或者整个行明确缺省值：

  ```
  INSERT INTO customer_t1 (c_customer_sk, c_customer_id, c_first_name) VALUES (3769, 'hello', DEFAULT);

  INSERT INTO customer_t1 DEFAULT VALUES;
  ```

- 如果需要在表中插入多行，请使用以下命令：

  ```
  INSERT INTO customer_t1 (c_customer_sk, c_customer_id, c_first_name) VALUES
      (6885, 'maps', 'Joes'),
      (4321, 'tpcds', 'Lily'),
      (9527, 'world', 'James');
  ```

  如果需要向表中插入多条数据，除此命令外，也可以多次执行插入一行数据命令实现。但是建议使用此命令可以提升效率。

- 如果从指定表插入数据到当前表，例如在数据库中创建了一个表customer_t1的备份表customer_t2，现在需要将表customer_t1中的数据插入到表customer_t2中，则可以执行如下命令。

  ```
  CREATE TABLE customer_t2
  (
      c_customer_sk             integer,
      c_customer_id             char(5),
      c_first_name              char(6),
      c_last_name               char(8)
  ) ;

  INSERT INTO customer_t2 SELECT * FROM customer_t1;
  ```

说明

从指定表插入数据到当前表时，若指定表与当前表对应的字段数据类型之间不存在隐式转换，则这两种数据类型必须相同。

- 删除备份表
  DROP TABLE *customer_t2* **CASCADE;**

说明

在删除表的时候，若当前需删除的表与其他表有依赖关系，需先删除关联的表，然后再删除当前表。

2.4.3 更新表中数据

修改已经存储在数据库中数据的行为叫做更新。用户可以更新单独一行，所有行或者指定的部分行。还可以独立更新每个字段，而其他字段则不受影响。

使用UPDATE命令更新现有行，需要提供以下三种信息：

- 表的名字和要更新的字段名
- 字段的新值
- 要更新哪些行

说明

- 表名字也可以使用模式名修饰，否则会从默认的模式路径找到这个表。
- SET后面紧跟字段和新的字段值。新的字段值不仅可以是常量，也可以是变量表达式。
- 表中可以含WHERE子句，且是等值测试。
  - 如果省略了WHERE子句，表示表中的所有行都要被更新。
  - 如果出现了WHERE子句，只有匹配其条件的行才会被更新。

在SET子句中的等号是一个赋值，而在WHERE子句中的等号是比较。WHERE条件也可以是不相等测试，许多其他的操作符也可以使用。

SQL通常不会为数据行提供唯一标识，因此无法直接声明需要更新哪一行。但是可以通过声明一个被更新的行必须满足的条件。只有在表里存在主键的时候，才可以通过主键指定一个独立的行。

建立表和插入数据的步骤请参考**创建表**和**向表中插入数据**。

更新表中数据具体示例如下：

- 需要将表customer_t1中c_customer_sk为9527的地域重新定义为9876：
  UPDATE *customer_t1* **SET** *c_customer_sk = 9876* **WHERE** *c_customer_sk = 9527*;
- 需要将表customer_t1中c_customer_sk为9527的地域重新定义为c_customer_sk + 100：
  UPDATE *customer_t1* **SET** *c_customer_sk= c_customer_sk + 100* **WHERE** *c_customer_sk= 9527*;
- 需要将Public模式下表customer_t1中c_customer_sk为9527的地域重新定义为9876：
  UPDATE *public.customer_t1* **SET** *c_customer_sk= 9876* **WHERE** *c_customer_sk= 9527*;
- 不含WHERE子句表示把所有c_customer_sk的值增加100：
  UPDATE *customer_t1* **SET** c_customer_sk *= c_customer_sk + 100*;
- 需要将表customer_t1中c_customer_sk大于9527的地域全部重新定义为9876：
  UPDATE *customer_t1* **SET** *c_customer_sk = 9876* **WHERE** *c_customer_sk > 9527*;

- 用户可以在一个UPDATE命令中更新更多的字段，方法是在SET子句中列出更多赋值，比如：
  ```
  UPDATE customer_t1 SET  c_customer_id = 'Admin', c_first_name = 'Local' WHERE c_customer_sk = 4421;
  ```

批量更新或删除数据后，会在数据文件中产生大量的删除标记，查询过程中标记删除的数据也是需要扫描的。故多次批量更新/删除后，标记删除的数据量过大会严重影响查询的性能。建议在批量更新/删除业务会反复执行的场景下，定期执行VACUUM FULL以保持查询性能。

2.4.4 查看数据

- 使用系统表pg_tables查询数据库所有表的信息。
  ```
  SELECT * FROM pg_tables;
  ```
- 使用gsql的\d+命令查询表的属性。
  ```
  \d+ customer_t1;
  ```
- 执行如下命令查询表customer_t1的数据量。
  ```
  SELECT count(*) FROM customer_t1;
  ```
- 执行如下命令查询表customer_t1的所有数据。
  ```
  SELECT * FROM customer_t1;
  ```
- 执行如下命令只查询字段c_customer_sk的数据。
  ```
  SELECT c_customer_sk FROM customer_t1;
  ```
- 执行如下命令过滤字段c_customer_sk的重复数据。
  ```
  SELECT DISTINCT( c_customer_sk ) FROM customer_t1;
  ```
- 执行如下命令查询字段c_customer_sk为3869的所有数据。
  ```
  SELECT * FROM customer_t1 WHERE c_customer_sk = 3869;
  ```
- 执行如下命令按照字段c_customer_sk进行排序。
  ```
  SELECT * FROM customer_t1 ORDER BY c_customer_sk;
  ```
  如果要取消运行时间过长的查询，请参考**查看系统表**的“查看和停止正在运行的查询语句”。

2.4.5 删除表中数据

在使用表的过程中，可能会需要删除已过期的数据，删除数据必须从表中整行的删除。

SQL不能直接访问独立的行，只能通过声明被删除行匹配的条件进行。如果表中有一个主键，用户可以指定准确的行。用户可以删除匹配条件的一组行或者一次删除表中的所有行。

使用DELETE命令删除行，如果删除表customer_t1中所有c_customer_sk为3869的记录：

```
DELETE FROM customer_t1 WHERE c_customer_sk = 3869;
```

如果执行如下命令，会删除表中所有的行。

```
DELETE FROM customer_t1;
TRUNCATE TABLE customer_t1;
```

说明

全表删除的场景下，建议使用truncate，不建议使用delete。

删除创建的表：

```
DROP TABLE customer_t1;
```


2.5 加载示例数据

在默认数据库gaussdb中加载示例数据。您将上传的示例数据位于OBS上。

说明

- 请确保您的SQL客户端已经连接至集群，然后再操作。
- TPC-DS的示例数据目前仅在**华北-北京一**区域使用。

1. 创建表。

   复制并执行下列创建表语句，以在gaussdb数据库中创建表。

```
DROP SCHEMA if exists tpcds cascade;
CREATE SCHEMA tpcds;
SET current_schema TO tpcds;
CREATE TABLE customer_address
(
    ca_address_sk             integer               not null,
    ca_address_id             char(16)              not null,
    ca_street_number          char(10)                      ,
    ca_street_name            varchar(60)                   ,
    ca_street_type            char(15)                      ,
    ca_suite_number           char(10)                      ,
    ca_city                   varchar(60)                   ,
    ca_county                 varchar(30)                   ,
    ca_state                  char(2)                       ,
    ca_zip                    char(10)                      ,
    ca_country                varchar(20)                   ,
    ca_gmt_offset             decimal(5,2)                  ,
    ca_location_type          char(20)
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE customer_demographics
(
    cd_demo_sk                integer               not null,
    cd_gender                 char(1)                       ,
    cd_marital_status         char(1)                       ,
    cd_education_status       char(20)                      ,
    cd_purchase_estimate      integer                       ,
    cd_credit_rating          char(10)                      ,
    cd_dep_count              integer                       ,
    cd_dep_employed_count     integer                       ,
    cd_dep_college_count      integer
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE date_dim
(
    d_date_sk                 integer               not null,
    d_date_id                 char(16)              not null,
    d_date                    date                          ,
    d_month_seq               integer                       ,
    d_week_seq                integer                       ,
    d_quarter_seq             integer                       ,
    d_year                    integer                       ,
    d_dow                     integer                       ,
    d_moy                     integer                       ,
    d_dom                     integer                       ,
    d_qoy                     integer                       ,
    d_fy_year                 integer                       ,
    d_fy_quarter_seq          integer                       ,
    d_fy_week_seq             integer                       ,
    d_day_name                char(9)                       ,
    d_quarter_name            char(6)                       ,
    d_holiday                 char(1)                       ,
```



```
    d_weekend                 char(1)                       ,
    d_following_holiday       char(1)                       ,
    d_first_dom               integer                       ,
    d_last_dom                integer                       ,
    d_same_day_ly             integer                       ,
    d_same_day_lq             integer                       ,
    d_current_day             char(1)                       ,
    d_current_week            char(1)                       ,
    d_current_month           char(1)                       ,
    d_current_quarter         char(1)                       ,
    d_current_year            char(1)
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE warehouse
(
    w_warehouse_sk            integer               not null,
    w_warehouse_id            char(16)              not null,
    w_warehouse_name          varchar(20)                   ,
    w_warehouse_sq_ft         integer                       ,
    w_street_number           char(10)                      ,
    w_street_name             varchar(60)                   ,
    w_street_type             char(15)                      ,
    w_suite_number            char(10)                      ,
    w_city                    varchar(60)                   ,
    w_county                  varchar(30)                   ,
    w_state                   char(2)                       ,
    w_zip                     char(10)                      ,
    w_country                 varchar(20)                   ,
    w_gmt_offset              decimal(5,2)
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE ship_mode
(
    sm_ship_mode_sk           integer               not null,
    sm_ship_mode_id           char(16)              not null,
    sm_type                   char(30)                      ,
    sm_code                   char(10)                      ,
    sm_carrier                char(20)                      ,
    sm_contract               char(20)
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE time_dim
(
    t_time_sk                 integer               not null,
    t_time_id                 char(16)              not null,
    t_time                    integer                       ,
    t_hour                    integer                       ,
    t_minute                  integer                       ,
    t_second                  integer                       ,
    t_am_pm                   char(2)                       ,
    t_shift                   char(20)                      ,
    t_sub_shift               char(20)                      ,
    t_meal_time               char(20)
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE reason
(
    r_reason_sk               integer               not null,
    r_reason_id               char(16)              not null,
    r_reason_desc             char(100)
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE income_band
(
    ib_income_band_sk         integer               not null,
    ib_lower_bound            integer                       ,
    ib_upper_bound            integer
)WITH (orientation = column, COMPRESSION = MIDDLE);
```

```
CREATE TABLE item
(
    i_item_sk               integer               not null,
    i_item_id               char(16)              not null,
    i_rec_start_date        date                          ,
    i_rec_end_date          date                          ,
    i_item_desc             varchar(200)                  ,
    i_current_price         decimal(7,2)                  ,
    i_wholesale_cost        decimal(7,2)                  ,
    i_brand_id              integer                       ,
    i_brand                 char(50)                      ,
    i_class_id              integer                       ,
    i_class                 char(50)                      ,
    i_category_id           integer                       ,
    i_category              char(50)                      ,
    i_manufact_id           integer                       ,
    i_manufact              char(50)                      ,
    i_size                  char(20)                      ,
    i_formulation           char(20)                      ,
    i_color                 char(20)                      ,
    i_units                 char(10)                      ,
    i_container             char(10)                      ,
    i_manager_id            integer                       ,
    i_product_name          char(50)
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE store
(
    s_store_sk              integer               not null,
    s_store_id              char(16)              not null,
    s_rec_start_date        date                          ,
    s_rec_end_date          date                          ,
    s_closed_date_sk        integer                       ,
    s_store_name            varchar(50)                   ,
    s_number_employees      integer                       ,
    s_floor_space           integer                       ,
    s_hours                 char(20)                      ,
    s_manager               varchar(40)                   ,
    s_market_id             integer                       ,
    s_geography_class       varchar(100)                  ,
    s_market_desc           varchar(100)                  ,
    s_market_manager        varchar(40)                   ,
    s_division_id           integer                       ,
    s_division_name         varchar(50)                   ,
    s_company_id            integer                       ,
    s_company_name          varchar(50)                   ,
    s_street_number         varchar(10)                   ,
    s_street_name           varchar(60)                   ,
    s_street_type           char(15)                      ,
    s_suite_number          char(10)                      ,
    s_city                  varchar(60)                   ,
    s_county                varchar(30)                   ,
    s_state                 char(2)                       ,
    s_zip                   char(10)                      ,
    s_country               varchar(20)                   ,
    s_gmt_offset            decimal(5,2)                  ,
    s_tax_percentage        decimal(5,2)
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE call_center
(
    cc_call_center_sk       integer               not null,
    cc_call_center_id       char(16)              not null,
    cc_rec_start_date       date                          ,
    cc_rec_end_date         date                          ,
    cc_closed_date_sk       integer                       ,
    cc_open_date_sk         integer                       ,
    cc_name                 varchar(50)                   ,
    cc_class                varchar(50)                   ,
```

```
    cc_employees              integer                   ,
    cc_sq_ft                  integer                   ,
    cc_hours                  char(20)                  ,
    cc_manager                varchar(40)               ,
    cc_mkt_id                 integer                   ,
    cc_mkt_class              char(50)                  ,
    cc_mkt_desc               varchar(100)              ,
    cc_market_manager         varchar(40)               ,
    cc_division               integer                   ,
    cc_division_name          varchar(50)               ,
    cc_company                integer                   ,
    cc_company_name           char(50)                  ,
    cc_street_number          char(10)                  ,
    cc_street_name            varchar(60)               ,
    cc_street_type            char(15)                  ,
    cc_suite_number           char(10)                  ,
    cc_city                   varchar(60)               ,
    cc_county                 varchar(30)               ,
    cc_state                  char(2)                   ,
    cc_zip                    char(10)                  ,
    cc_country                varchar(20)               ,
    cc_gmt_offset             decimal(5,2)              ,
    cc_tax_percentage         decimal(5,2)
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE customer
(
    c_customer_sk             integer               not null,
    c_customer_id             char(16)              not null,
    c_current_cdemo_sk        integer                   ,
    c_current_hdemo_sk        integer                   ,
    c_current_addr_sk         integer                   ,
    c_first_shipto_date_sk    integer                   ,
    c_first_sales_date_sk     integer                   ,
    c_salutation              char(10)                  ,
    c_first_name              char(20)                  ,
    c_last_name               char(30)                  ,
    c_preferred_cust_flag     char(1)                   ,
    c_birth_day               integer                   ,
    c_birth_month             integer                   ,
    c_birth_year              integer                   ,
    c_birth_country           varchar(20)               ,
    c_login                   char(13)                  ,
    c_email_address           char(50)                  ,
    c_last_review_date        char(10)
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE web_site
(
    web_site_sk               integer               not null,
    web_site_id               char(16)              not null,
    web_rec_start_date        date                      ,
    web_rec_end_date          date                      ,
    web_name                  varchar(50)               ,
    web_open_date_sk          integer                   ,
    web_close_date_sk         integer                   ,
    web_class                 varchar(50)               ,
    web_manager               varchar(40)               ,
    web_mkt_id                integer                   ,
    web_mkt_class             varchar(50)               ,
    web_mkt_desc              varchar(100)              ,
    web_market_manager        varchar(40)               ,
    web_company_id            integer                   ,
    web_company_name          char(50)                  ,
    web_street_number         char(10)                  ,
    web_street_name           varchar(60)               ,
    web_street_type           char(15)                  ,
    web_suite_number          char(10)                  ,
    web_city                  varchar(60)               ,
```

```
    web_county                varchar(30)                ,
    web_state                 char(2)                    ,
    web_zip                   char(10)                   ,
    web_country               varchar(20)                ,
    web_gmt_offset            decimal(5,2)               ,
    web_tax_percentage        decimal(5,2)
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE store_returns
(
    sr_returned_date_sk       integer                    ,
    sr_return_time_sk         integer                    ,
    sr_item_sk                integer             not null,
    sr_customer_sk            integer                    ,
    sr_cdemo_sk               integer                    ,
    sr_hdemo_sk               integer                    ,
    sr_addr_sk                integer                    ,
    sr_store_sk               integer                    ,
    sr_reason_sk              integer                    ,
    sr_ticket_number          integer             not null,
    sr_return_quantity        integer                    ,
    sr_return_amt             decimal(7,2)               ,
    sr_return_tax             decimal(7,2)               ,
    sr_return_amt_inc_tax     decimal(7,2)               ,
    sr_fee                    decimal(7,2)               ,
    sr_return_ship_cost       decimal(7,2)               ,
    sr_refunded_cash          decimal(7,2)               ,
    sr_reversed_charge        decimal(7,2)               ,
    sr_store_credit           decimal(7,2)               ,
    sr_net_loss               decimal(7,2)
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE household_demographics
(
    hd_demo_sk                integer             not null,
    hd_income_band_sk         integer                    ,
    hd_buy_potential          char(15)                   ,
    hd_dep_count              integer                    ,
    hd_vehicle_count          integer
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE web_page
(
    wp_web_page_sk            integer             not null,
    wp_web_page_id            char(16)            not null,
    wp_rec_start_date         date                       ,
    wp_rec_end_date           date                       ,
    wp_creation_date_sk       integer                    ,
    wp_access_date_sk         integer                    ,
    wp_autogen_flag           char(1)                    ,
    wp_customer_sk            integer                    ,
    wp_url                    varchar(100)               ,
    wp_type                   char(50)                   ,
    wp_char_count             integer                    ,
    wp_link_count             integer                    ,
    wp_image_count            integer                    ,
    wp_max_ad_count           integer
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE promotion
(
    p_promo_sk                integer             not null,
    p_promo_id                char(16)            not null,
    p_start_date_sk           integer                    ,
    p_end_date_sk             integer                    ,
    p_item_sk                 integer                    ,
    p_cost                    decimal(15,2)              ,
    p_response_target         integer                    ,
    p_promo_name              char(50)                   ,
```

```
    p_channel_dmail           char(1)                  ,
    p_channel_email           char(1)                  ,
    p_channel_catalog         char(1)                  ,
    p_channel_tv              char(1)                  ,
    p_channel_radio           char(1)                  ,
    p_channel_press           char(1)                  ,
    p_channel_event           char(1)                  ,
    p_channel_demo            char(1)                  ,
    p_channel_details         varchar(100)             ,
    p_purpose                 char(15)                 ,
    p_discount_active         char(1)
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE catalog_page
(
    cp_catalog_page_sk        integer               not null,
    cp_catalog_page_id        char(16)              not null,
    cp_start_date_sk          integer                       ,
    cp_end_date_sk            integer                       ,
    cp_department             varchar(50)                   ,
    cp_catalog_number         integer                       ,
    cp_catalog_page_number    integer                       ,
    cp_description            varchar(100)                  ,
    cp_type                   varchar(100)
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE inventory
(
    inv_date_sk               integer               not null,
    inv_item_sk               integer               not null,
    inv_warehouse_sk          integer               not null,
    inv_quantity_on_hand      integer
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE catalog_returns
(
    cr_returned_date_sk       integer                       ,
    cr_returned_time_sk       integer                       ,
    cr_item_sk                integer               not null,
    cr_refunded_customer_sk   integer                       ,
    cr_refunded_cdemo_sk      integer                       ,
    cr_refunded_hdemo_sk      integer                       ,
    cr_refunded_addr_sk       integer                       ,
    cr_returning_customer_sk  integer                       ,
    cr_returning_cdemo_sk     integer                       ,
    cr_returning_hdemo_sk     integer                       ,
    cr_returning_addr_sk      integer                       ,
    cr_call_center_sk         integer                       ,
    cr_catalog_page_sk        integer                       ,
    cr_ship_mode_sk           integer                       ,
    cr_warehouse_sk           integer                       ,
    cr_reason_sk              integer                       ,
    cr_order_number           integer               not null,
    cr_return_quantity        integer                       ,
    cr_return_amount          decimal(7,2)                  ,
    cr_return_tax             decimal(7,2)                  ,
    cr_return_amt_inc_tax     decimal(7,2)                  ,
    cr_fee                    decimal(7,2)                  ,
    cr_return_ship_cost       decimal(7,2)                  ,
    cr_refunded_cash          decimal(7,2)                  ,
    cr_reversed_charge        decimal(7,2)                  ,
    cr_store_credit           decimal(7,2)                  ,
    cr_net_loss               decimal(7,2)
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE web_returns
(
    wr_returned_date_sk       integer                       ,
    wr_returned_time_sk       integer                       ,
```

```
    wr_item_sk                integer               not null,
    wr_refunded_customer_sk   integer                       ,
    wr_refunded_cdemo_sk      integer                       ,
    wr_refunded_hdemo_sk      integer                       ,
    wr_refunded_addr_sk       integer                       ,
    wr_returning_customer_sk  integer                       ,
    wr_returning_cdemo_sk     integer                       ,
    wr_returning_hdemo_sk     integer                       ,
    wr_returning_addr_sk      integer                       ,
    wr_web_page_sk            integer                       ,
    wr_reason_sk              integer                       ,
    wr_order_number           integer               not null,
    wr_return_quantity        integer                       ,
    wr_return_amt             decimal(7,2)                  ,
    wr_return_tax             decimal(7,2)                  ,
    wr_return_amt_inc_tax     decimal(7,2)                  ,
    wr_fee                    decimal(7,2)                  ,
    wr_return_ship_cost       decimal(7,2)                  ,
    wr_refunded_cash          decimal(7,2)                  ,
    wr_reversed_charge        decimal(7,2)                  ,
    wr_account_credit         decimal(7,2)                  ,
    wr_net_loss               decimal(7,2)
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE web_sales
(
    ws_sold_date_sk           integer                       ,
    ws_sold_time_sk           integer                       ,
    ws_ship_date_sk           integer                       ,
    ws_item_sk                integer               not null,
    ws_bill_customer_sk       integer                       ,
    ws_bill_cdemo_sk          integer                       ,
    ws_bill_hdemo_sk          integer                       ,
    ws_bill_addr_sk           integer                       ,
    ws_ship_customer_sk       integer                       ,
    ws_ship_cdemo_sk          integer                       ,
    ws_ship_hdemo_sk          integer                       ,
    ws_ship_addr_sk           integer                       ,
    ws_web_page_sk            integer                       ,
    ws_web_site_sk            integer                       ,
    ws_ship_mode_sk           integer                       ,
    ws_warehouse_sk           integer                       ,
    ws_promo_sk               integer                       ,
    ws_order_number           integer               not null,
    ws_quantity               integer                       ,
    ws_wholesale_cost         decimal(7,2)                  ,
    ws_list_price             decimal(7,2)                  ,
    ws_sales_price            decimal(7,2)                  ,
    ws_ext_discount_amt       decimal(7,2)                  ,
    ws_ext_sales_price        decimal(7,2)                  ,
    ws_ext_wholesale_cost     decimal(7,2)                  ,
    ws_ext_list_price         decimal(7,2)                  ,
    ws_ext_tax                decimal(7,2)                  ,
    ws_coupon_amt             decimal(7,2)                  ,
    ws_ext_ship_cost          decimal(7,2)                  ,
    ws_net_paid               decimal(7,2)                  ,
    ws_net_paid_inc_tax       decimal(7,2)                  ,
    ws_net_paid_inc_ship      decimal(7,2)                  ,
    ws_net_paid_inc_ship_tax  decimal(7,2)                  ,
    ws_net_profit             decimal(7,2)
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE catalog_sales
(
    cs_sold_date_sk           integer                       ,
    cs_sold_time_sk           integer                       ,
    cs_ship_date_sk           integer                       ,
    cs_bill_customer_sk       integer                       ,
    cs_bill_cdemo_sk          integer                       ,
```

```
    cs_bill_hdemo_sk          integer                   ,
    cs_bill_addr_sk           integer                   ,
    cs_ship_customer_sk       integer                   ,
    cs_ship_cdemo_sk          integer                   ,
    cs_ship_hdemo_sk          integer                   ,
    cs_ship_addr_sk           integer                   ,
    cs_call_center_sk         integer                   ,
    cs_catalog_page_sk        integer                   ,
    cs_ship_mode_sk           integer                   ,
    cs_warehouse_sk           integer                   ,
    cs_item_sk                integer               not null,
    cs_promo_sk               integer                   ,
    cs_order_number           integer               not null,
    cs_quantity               integer                   ,
    cs_wholesale_cost         decimal(7,2)              ,
    cs_list_price             decimal(7,2)              ,
    cs_sales_price            decimal(7,2)              ,
    cs_ext_discount_amt       decimal(7,2)              ,
    cs_ext_sales_price        decimal(7,2)              ,
    cs_ext_wholesale_cost     decimal(7,2)              ,
    cs_ext_list_price         decimal(7,2)              ,
    cs_ext_tax                decimal(7,2)              ,
    cs_coupon_amt             decimal(7,2)              ,
    cs_ext_ship_cost          decimal(7,2)              ,
    cs_net_paid               decimal(7,2)              ,
    cs_net_paid_inc_tax       decimal(7,2)              ,
    cs_net_paid_inc_ship      decimal(7,2)              ,
    cs_net_paid_inc_ship_tax  decimal(7,2)              ,
    cs_net_profit             decimal(7,2)
)WITH (orientation = column, COMPRESSION = MIDDLE);

CREATE TABLE store_sales
(
    ss_sold_date_sk           integer                   ,
    ss_sold_time_sk           integer                   ,
    ss_item_sk                integer               not null,
    ss_customer_sk            integer                   ,
    ss_cdemo_sk               integer                   ,
    ss_hdemo_sk               integer                   ,
    ss_addr_sk                integer                   ,
    ss_store_sk               integer                   ,
    ss_promo_sk               integer                   ,
    ss_ticket_number          integer               not null,
    ss_quantity               integer                   ,
    ss_wholesale_cost         decimal(7,2)              ,
    ss_list_price             decimal(7,2)              ,
    ss_sales_price            decimal(7,2)              ,
    ss_ext_discount_amt       decimal(7,2)              ,
    ss_ext_sales_price        decimal(7,2)              ,
    ss_ext_wholesale_cost     decimal(7,2)              ,
    ss_ext_list_price         decimal(7,2)              ,
    ss_ext_tax                decimal(7,2)              ,
    ss_coupon_amt             decimal(7,2)              ,
    ss_net_paid               decimal(7,2)              ,
    ss_net_paid_inc_tax       decimal(7,2)              ,
    ss_net_profit             decimal(7,2)
)WITH (orientation = column, COMPRESSION = MIDDLE);
```

2. 创建OBS外表。

   复制并执行下列创建OBS外表语句，以在gaussdb数据库中创建OBS外表，用于识别数据格式及设置导入容错性。

   说明

   将创建外表语句中的参数ACCESS_KEY和SECRET_ACCESS_KEY替换为实际值，然后在客户端工具中执行替换后的语句创建外表。

   ACCESS_KEY和SECRET_ACCESS_KEY的值，请参见**创建访问密钥（AK和SK）**章节进行获取，然后将获取到的值替换到创建外表语句中。


```
CREATE FOREIGN TABLE obs_from_customer_address_001
(
ca_address_sk integer not null,
ca_address_id char(16) not null,
ca_street_number char(10) ,
ca_street_name varchar(60) ,
ca_street_type char(15) ,
ca_suite_number char(10) ,
ca_city varchar(60) ,
ca_county varchar(30) ,
ca_state char(2) ,
ca_zip char(10) ,
ca_country varchar(20) ,
ca_gmt_offset float4 ,
ca_location_type char(20)
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/customer_address/customer_address',
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_customer_address_001;

CREATE FOREIGN TABLE obs_from_customer_demographics_001
(
    cd_demo_sk                integer               not null,
    cd_gender                 char(1)                       ,
    cd_marital_status         char(1)                       ,
    cd_education_status       char(20)                      ,
    cd_purchase_estimate      integer                       ,
    cd_credit_rating          char(10)                      ,
    cd_dep_count              integer                       ,
    cd_dep_employed_count     integer                       ,
    cd_dep_college_count      integer
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/customer_demographics/
customer_demographics',
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_customer_demographics_001;

CREATE FOREIGN TABLE obs_from_date_dim_001
(
    d_date_sk                 integer               not null,
    d_date_id                 char(16)              not null,
    d_date                    date                          ,
    d_month_seq               integer                       ,
    d_week_seq                integer                       ,
    d_quarter_seq             integer                       ,
    d_year                    integer                       ,
    d_dow                     integer                       ,
    d_moy                     integer                       ,
    d_dom                     integer                       ,
```

```
    d_qoy                     integer                   ,
    d_fy_year                 integer                   ,
    d_fy_quarter_seq          integer                   ,
    d_fy_week_seq             integer                   ,
    d_day_name                char(9)                   ,
    d_quarter_name            char(6)                   ,
    d_holiday                 char(1)                   ,
    d_weekend                 char(1)                   ,
    d_following_holiday       char(1)                   ,
    d_first_dom               integer                   ,
    d_last_dom                integer                   ,
    d_same_day_ly             integer                   ,
    d_same_day_lq             integer                   ,
    d_current_day             char(1)                   ,
    d_current_week            char(1)                   ,
    d_current_month           char(1)                   ,
    d_current_quarter         char(1)                   ,
    d_current_year            char(1)
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/ date_dim/date_dim',
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_date_dim_001;

CREATE FOREIGN TABLE obs_from_warehouse_001
(
    w_warehouse_sk            integer               not null,
    w_warehouse_id            char(16)              not null,
    w_warehouse_name          varchar(20)                   ,
    w_warehouse_sq_ft         integer                       ,
    w_street_number           char(10)                      ,
    w_street_name             varchar(60)                   ,
    w_street_type             char(15)                      ,
    w_suite_number            char(10)                      ,
    w_city                    varchar(60)                   ,
    w_county                  varchar(30)                   ,
    w_state                   char(2)                       ,
    w_zip                     char(10)                      ,
    w_country                 varchar(20)                   ,
    w_gmt_offset              decimal(5,2)
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/ warehouse/warehouse',
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_warehouse_001;

CREATE FOREIGN TABLE obs_from_ship_mode_001
(
    sm_ship_mode_sk           integer               not null,
    sm_ship_mode_id           char(16)              not null,
    sm_type                   char(30)                      ,
```

```
    sm_code                 char(10)                  ,
    sm_carrier              char(20)                  ,
    sm_contract              char(20)
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/ ship_mode/ship_mode' ,
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_ship_mode_001;

CREATE FOREIGN TABLE obs_from_time_dim_001
(
    t_time_sk               integer             not null,
    t_time_id               char(16)             not null,
    t_time                  integer                   ,
    t_hour                  integer                   ,
    t_minute                 integer                   ,
    t_second                 integer                   ,
    t_am_pm                   char(2)                   ,
    t_shift                 char(20)                  ,
    t_sub_shift              char(20)                   ,
    t_meal_time               char(20)
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/time_dim/time_dim',
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_time_dim_001;

CREATE FOREIGN TABLE obs_from_reason_001
(
    r_reason_sk             integer             not null,
    r_reason_id             char(16)             not null,
    r_reason_desc            char(100)
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/reason/reason' ,
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_reason_001;

CREATE FOREIGN TABLE obs_from_income_band_001
(
    ib_income_band_sk       integer             not null,
    ib_lower_bound          integer                   ,
```

```
    ib_upper_bound            integer
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/income_band/income_band',
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_income_band_001;

CREATE FOREIGN TABLE obs_from_item_001
(
    i_item_sk                integer               not null,
    i_item_id                char(16)              not null,
    i_rec_start_date         date                          ,
    i_rec_end_date           date                          ,
    i_item_desc              varchar(200)                  ,
    i_current_price          decimal(7,2)                  ,
    i_wholesale_cost         decimal(7,2)                  ,
    i_brand_id               integer                       ,
    i_brand                  char(50)                      ,
    i_class_id               integer                       ,
    i_class                  char(50)                      ,
    i_category_id            integer                       ,
    i_category               char(50)                      ,
    i_manufact_id            integer                       ,
    i_manufact               char(50)                      ,
    i_size                   char(20)                      ,
    i_formulation            char(20)                      ,
    i_color                  char(20)                      ,
    i_units                  char(10)                      ,
    i_container              char(10)                      ,
    i_manager_id             integer                       ,
    i_product_name           char(50)
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/item/item',
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_item_001;

CREATE FOREIGN TABLE obs_from_store_001
(
    s_store_sk               integer               not null,
    s_store_id               char(16)              not null,
    s_rec_start_date         date                          ,
    s_rec_end_date           date                          ,
    s_closed_date_sk         integer                       ,
    s_store_name             varchar(50)                   ,
    s_number_employees       integer                       ,
    s_floor_space            integer                       ,
    s_hours                  char(20)                      ,
    s_manager                varchar(40)                   ,
    s_market_id              integer                       ,
    s_geography_class        varchar(100)                  ,
```

```
    s_market_desc             varchar(100)                ,
    s_market_manager          varchar(40)                 ,
    s_division_id             integer                     ,
    s_division_name           varchar(50)                 ,
    s_company_id              integer                     ,
    s_company_name            varchar(50)                 ,
    s_street_number           varchar(10)                 ,
    s_street_name             varchar(60)                 ,
    s_street_type             char(15)                    ,
    s_suite_number            char(10)                    ,
    s_city                    varchar(60)                 ,
    s_county                  varchar(30)                 ,
    s_state                   char(2)                     ,
    s_zip                     char(10)                    ,
    s_country                 varchar(20)                 ,
    s_gmt_offset              decimal(5,2)                ,
    s_tax_percentage          decimal(5,2)
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/store/store',
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_store_001;

CREATE FOREIGN TABLE obs_from_call_center_001
(
    cc_call_center_sk         integer               not null,
    cc_call_center_id         char(16)              not null,
    cc_rec_start_date         date                        ,
    cc_rec_end_date           date                        ,
    cc_closed_date_sk         integer                     ,
    cc_open_date_sk           integer                     ,
    cc_name                   varchar(50)                 ,
    cc_class                  varchar(50)                 ,
    cc_employees              integer                     ,
    cc_sq_ft                  integer                     ,
    cc_hours                  char(20)                    ,
    cc_manager                varchar(40)                 ,
    cc_mkt_id                 integer                     ,
    cc_mkt_class              char(50)                    ,
    cc_mkt_desc               varchar(100)                ,
    cc_market_manager         varchar(40)                 ,
    cc_division               integer                     ,
    cc_division_name          varchar(50)                 ,
    cc_company                integer                     ,
    cc_company_name           char(50)                    ,
    cc_street_number          char(10)                    ,
    cc_street_name            varchar(60)                 ,
    cc_street_type            char(15)                    ,
    cc_suite_number           char(10)                    ,
    cc_city                   varchar(60)                 ,
    cc_county                 varchar(30)                 ,
    cc_state                  char(2)                     ,
    cc_zip                    char(10)                    ,
    cc_country                varchar(20)                 ,
    cc_gmt_offset             decimal(5,2)                ,
    cc_tax_percentage         decimal(5,2)
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/call_center/call_center',
```



```
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_call_center_001;

CREATE FOREIGN TABLE obs_from_customer_001
(
    c_customer_sk             integer               not null,
    c_customer_id             char(16)              not null,
    c_current_cdemo_sk        integer                       ,
    c_current_hdemo_sk        integer                       ,
    c_current_addr_sk         integer                       ,
    c_first_shipto_date_sk    integer                       ,
    c_first_sales_date_sk     integer                       ,
    c_salutation              char(10)                      ,
    c_first_name              char(20)                      ,
    c_last_name               char(30)                      ,
    c_preferred_cust_flag     char(1)                       ,
    c_birth_day               integer                       ,
    c_birth_month             integer                       ,
    c_birth_year              integer                       ,
    c_birth_country           varchar(20)                   ,
    c_login                   char(13)                      ,
    c_email_address           char(50)                      ,
    c_last_review_date        char(10)
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/customer/customer' ,
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_customer_001;

CREATE FOREIGN TABLE obs_from_web_site_001
(
    web_site_sk               integer               not null,
    web_site_id               char(16)              not null,
    web_rec_start_date        date                          ,
    web_rec_end_date          date                          ,
    web_name                  varchar(50)                   ,
    web_open_date_sk          integer                       ,
    web_close_date_sk         integer                       ,
    web_class                 varchar(50)                   ,
    web_manager               varchar(40)                   ,
    web_mkt_id                integer                       ,
    web_mkt_class             varchar(50)                   ,
    web_mkt_desc              varchar(100)                  ,
    web_market_manager        varchar(40)                   ,
    web_company_id            integer                       ,
    web_company_name          char(50)                      ,
    web_street_number         char(10)                      ,
    web_street_name           varchar(60)                   ,
    web_street_type           char(15)                      ,
    web_suite_number          char(10)                      ,
    web_city                  varchar(60)                   ,
    web_county                varchar(30)                   ,
```

```
    web_state                 char(2)                       ,
    web_zip                   char(10)                      ,
    web_country               varchar(20)                   ,
    web_gmt_offset            decimal(5,2)                  ,
    web_tax_percentage        decimal(5,2)
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/web_site/web_site' ,
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_web_site_001;

CREATE FOREIGN TABLE obs_from_store_returns_001
(
    sr_returned_date_sk       integer                       ,
    sr_return_time_sk         integer                       ,
    sr_item_sk                integer               not null,
    sr_customer_sk            integer                       ,
    sr_cdemo_sk               integer                       ,
    sr_hdemo_sk               integer                       ,
    sr_addr_sk                integer                       ,
    sr_store_sk               integer                       ,
    sr_reason_sk              integer                       ,
    sr_ticket_number          bigint                not null,
    sr_return_quantity        integer                       ,
    sr_return_amt             decimal(7,2)                  ,
    sr_return_tax             decimal(7,2)                  ,
    sr_return_amt_inc_tax     decimal(7,2)                  ,
    sr_fee                    decimal(7,2)                  ,
    sr_return_ship_cost       decimal(7,2)                  ,
    sr_refunded_cash          decimal(7,2)                  ,
    sr_reversed_charge        decimal(7,2)                  ,
    sr_store_credit           decimal(7,2)                  ,
    sr_net_loss               decimal(7,2)
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/store_returns/store_returns' ,
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_store_returns_001;

CREATE FOREIGN TABLE obs_from_household_demographics_001
(
    hd_demo_sk                integer               not null,
    hd_income_band_sk         integer                       ,
    hd_buy_potential          char(15)                      ,
    hd_dep_count              integer                       ,
    hd_vehicle_count          integer
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/household_demographics/
household_demographics' ,
```

```
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_household_demographics_001;

CREATE FOREIGN TABLE obs_from_web_page_001
(
    wp_web_page_sk            integer               not null,
    wp_web_page_id            char(16)              not null,
    wp_rec_start_date         date                          ,
    wp_rec_end_date           date                          ,
    wp_creation_date_sk       integer                       ,
    wp_access_date_sk         integer                       ,
    wp_autogen_flag           char(1)                       ,
    wp_customer_sk            integer                       ,
    wp_url                    varchar(100)                  ,
    wp_type                   char(50)                      ,
    wp_char_count             integer                       ,
    wp_link_count             integer                       ,
    wp_image_count            integer                       ,
    wp_max_ad_count           integer
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/web_page/web_page' ,
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_web_page_001;

CREATE FOREIGN TABLE obs_from_promotion_001
(
    p_promo_sk                integer               not null,
    p_promo_id                char(16)              not null,
    p_start_date_sk           integer                       ,
    p_end_date_sk             integer                       ,
    p_item_sk                 integer                       ,
    p_cost                    decimal(15,2)                 ,
    p_response_target         integer                       ,
    p_promo_name              char(50)                      ,
    p_channel_dmail           char(1)                       ,
    p_channel_email           char(1)                       ,
    p_channel_catalog         char(1)                       ,
    p_channel_tv              char(1)                       ,
    p_channel_radio           char(1)                       ,
    p_channel_press           char(1)                       ,
    p_channel_event           char(1)                       ,
    p_channel_demo            char(1)                       ,
    p_channel_details         varchar(100)                  ,
    p_purpose                 char(15)                      ,
    p_discount_active         char(1)
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/promotion/promotion' ,
format 'text',
delimiter '|',
```

```
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_promotion_001;

CREATE FOREIGN TABLE obs_from_catalog_page_001
(
    cp_catalog_page_sk        integer               not null,
    cp_catalog_page_id        char(16)              not null,
    cp_start_date_sk          integer                       ,
    cp_end_date_sk            integer                       ,
    cp_department             varchar(50)                   ,
    cp_catalog_number         integer                       ,
    cp_catalog_page_number    integer                       ,
    cp_description            varchar(100)                  ,
    cp_type                   varchar(100)
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/catalog_page/catalog_page' ,
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_catalog_page_001;

CREATE FOREIGN TABLE obs_from_inventory_001
(
    inv_date_sk             integer             not null,
    inv_item_sk             integer             not null,
    inv_warehouse_sk        integer             not null,
    inv_quantity_on_hand    integer
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/inventory/inventory' ,
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_inventory_001;

CREATE FOREIGN TABLE obs_from_catalog_returns_001
(
    cr_returned_date_sk       integer                     ,
    cr_returned_time_sk       integer                     ,
    cr_item_sk                integer             not null,
    cr_refunded_customer_sk   integer                     ,
    cr_refunded_cdemo_sk      integer                     ,
    cr_refunded_hdemo_sk      integer                     ,
    cr_refunded_addr_sk       integer                     ,
    cr_returning_customer_sk  integer                     ,
    cr_returning_cdemo_sk     integer                     ,
    cr_returning_hdemo_sk     integer                     ,
    cr_returning_addr_sk      integer                     ,
```

```
    cr_call_center_sk         integer                       ,
    cr_catalog_page_sk        integer                       ,
    cr_ship_mode_sk           integer                       ,
    cr_warehouse_sk           integer                       ,
    cr_reason_sk              integer                       ,
    cr_order_number           bigint               not null,
    cr_return_quantity        integer                       ,
    cr_return_amount          decimal(7,2)                  ,
    cr_return_tax             decimal(7,2)                  ,
    cr_return_amt_inc_tax     decimal(7,2)                  ,
    cr_fee                    decimal(7,2)                  ,
    cr_return_ship_cost       decimal(7,2)                  ,
    cr_refunded_cash          decimal(7,2)                  ,
    cr_reversed_charge        decimal(7,2)                  ,
    cr_store_credit           decimal(7,2)                  ,
    cr_net_loss               decimal(7,2)
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/catalog_returns/catalog_returns' ,
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_catalog_returns_001;

CREATE FOREIGN TABLE obs_from_web_returns_001
(
    wr_returned_date_sk       integer                       ,
    wr_returned_time_sk       integer                       ,
    wr_item_sk                integer               not null,
    wr_refunded_customer_sk   integer                       ,
    wr_refunded_cdemo_sk      integer                       ,
    wr_refunded_hdemo_sk      integer                       ,
    wr_refunded_addr_sk       integer                       ,
    wr_returning_customer_sk  integer                       ,
    wr_returning_cdemo_sk     integer                       ,
    wr_returning_hdemo_sk     integer                       ,
    wr_returning_addr_sk      integer                       ,
    wr_web_page_sk            integer                       ,
    wr_reason_sk              integer                       ,
    wr_order_number           bigint               not null,
    wr_return_quantity        integer                       ,
    wr_return_amt             decimal(7,2)                  ,
    wr_return_tax             decimal(7,2)                  ,
    wr_return_amt_inc_tax     decimal(7,2)                  ,
    wr_fee                    decimal(7,2)                  ,
    wr_return_ship_cost       decimal(7,2)                  ,
    wr_refunded_cash          decimal(7,2)                  ,
    wr_reversed_charge        decimal(7,2)                  ,
    wr_account_credit         decimal(7,2)                  ,
    wr_net_loss               decimal(7,2)
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/web_returns/web_returns' ,
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
```



```
)
with err_obs_from_web_returns_001;

CREATE FOREIGN TABLE obs_from_web_sales_001
(
    ws_sold_date_sk           integer                       ,
    ws_sold_time_sk           integer                       ,
    ws_ship_date_sk           integer                       ,
    ws_item_sk                integer               not null,
    ws_bill_customer_sk       integer                       ,
    ws_bill_cdemo_sk          integer                       ,
    ws_bill_hdemo_sk          integer                       ,
    ws_bill_addr_sk           integer                       ,
    ws_ship_customer_sk       integer                       ,
    ws_ship_cdemo_sk          integer                       ,
    ws_ship_hdemo_sk          integer                       ,
    ws_ship_addr_sk           integer                       ,
    ws_web_page_sk            integer                       ,
    ws_web_site_sk            integer                       ,
    ws_ship_mode_sk           integer                       ,
    ws_warehouse_sk           integer                       ,
    ws_promo_sk               integer                       ,
    ws_order_number           bigint                not null,
    ws_quantity               integer                       ,
    ws_wholesale_cost         decimal(7,2)                  ,
    ws_list_price             decimal(7,2)                  ,
    ws_sales_price            decimal(7,2)                  ,
    ws_ext_discount_amt       decimal(7,2)                  ,
    ws_ext_sales_price        decimal(7,2)                  ,
    ws_ext_wholesale_cost     decimal(7,2)                  ,
    ws_ext_list_price         decimal(7,2)                  ,
    ws_ext_tax                decimal(7,2)                  ,
    ws_coupon_amt             decimal(7,2)                  ,
    ws_ext_ship_cost          decimal(7,2)                  ,
    ws_net_paid               decimal(7,2)                  ,
    ws_net_paid_inc_tax       decimal(7,2)                  ,
    ws_net_paid_inc_ship      decimal(7,2)                  ,
    ws_net_paid_inc_ship_tax  decimal(7,2)                  ,
    ws_net_profit             decimal(7,2)
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/web_sales/web_sales' ,
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_web_sales_001;

CREATE FOREIGN TABLE obs_from_catalog_sales_001
(
    cs_sold_date_sk           integer                       ,
    cs_sold_time_sk           integer                       ,
    cs_ship_date_sk           integer                       ,
    cs_bill_customer_sk       integer                       ,
    cs_bill_cdemo_sk          integer                       ,
    cs_bill_hdemo_sk          integer                       ,
    cs_bill_addr_sk           integer                       ,
    cs_ship_customer_sk       integer                       ,
    cs_ship_cdemo_sk          integer                       ,
    cs_ship_hdemo_sk          integer                       ,
    cs_ship_addr_sk           integer                       ,
    cs_call_center_sk         integer                       ,
    cs_catalog_page_sk        integer                       ,
```

```
    cs_ship_mode_sk           integer                       ,
    cs_warehouse_sk           integer                       ,
    cs_item_sk                integer               not null,
    cs_promo_sk               integer                       ,
    cs_order_number           bigint                not null,
    cs_quantity               integer                       ,
    cs_wholesale_cost         decimal(7,2)                  ,
    cs_list_price             decimal(7,2)                  ,
    cs_sales_price            decimal(7,2)                  ,
    cs_ext_discount_amt       decimal(7,2)                  ,
    cs_ext_sales_price        decimal(7,2)                  ,
    cs_ext_wholesale_cost     decimal(7,2)                  ,
    cs_ext_list_price         decimal(7,2)                  ,
    cs_ext_tax                decimal(7,2)                  ,
    cs_coupon_amt             decimal(7,2)                  ,
    cs_ext_ship_cost          decimal(7,2)                  ,
    cs_net_paid               decimal(7,2)                  ,
    cs_net_paid_inc_tax       decimal(7,2)                  ,
    cs_net_paid_inc_ship      decimal(7,2)                  ,
    cs_net_paid_inc_ship_tax  decimal(7,2)                  ,
    cs_net_profit             decimal(7,2)
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/catalog_sales/catalog_sales' ,
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_catalog_sales_001;

CREATE FOREIGN TABLE obs_from_store_sales_001
(
    ss_sold_date_sk           integer                       ,
    ss_sold_time_sk           integer                       ,
    ss_item_sk                integer               not null,
    ss_customer_sk            integer                       ,
    ss_cdemo_sk               integer                       ,
    ss_hdemo_sk               integer                       ,
    ss_addr_sk                integer                       ,
    ss_store_sk               integer                       ,
    ss_promo_sk               integer                       ,
    ss_ticket_number          bigint                not null,
    ss_quantity               integer                       ,
    ss_wholesale_cost         decimal(7,2)                  ,
    ss_list_price             decimal(7,2)                  ,
    ss_sales_price            decimal(7,2)                  ,
    ss_ext_discount_amt       decimal(7,2)                  ,
    ss_ext_sales_price        decimal(7,2)                  ,
    ss_ext_wholesale_cost     decimal(7,2)                  ,
    ss_ext_list_price         decimal(7,2)                  ,
    ss_ext_tax                decimal(7,2)                  ,
    ss_coupon_amt             decimal(7,2)                  ,
    ss_net_paid               decimal(7,2)                  ,
    ss_net_paid_inc_tax       decimal(7,2)                  ,
    ss_net_profit             decimal(7,2)
)
SERVER gsmpp_server
OPTIONS (
location 'obs://dws/download/dws_sample_database_data_files/store_sales/store_sales' ,
format 'text',
delimiter '|',
encoding 'utf8',
noescaping 'true',
```



```
ACCESS_KEY 'access_key_value_to_be_replaced',
SECRET_ACCESS_KEY 'secret_access_key_value_to_be_replaced',
reject_limit 'unlimited',
chunksize '64'
)
with err_obs_from_store_sales_001;
```

3. 使用INSERT命令导入数据。

```
INSERT INTO customer_address SELECT * FROM obs_from_customer_address_001 ;
INSERT INTO customer_demographics SELECT * FROM obs_from_customer_demographics_001 ;
INSERT INTO date_dim SELECT * FROM obs_from_date_dim_001;
INSERT INTO warehouse SELECT * FROM obs_from_warehouse_001;
INSERT INTO ship_mode SELECT * FROM obs_from_ship_mode_001;
INSERT INTO time_dim SELECT * FROM obs_from_time_dim_001;
INSERT INTO reason SELECT * FROM obs_from_reason_001;
INSERT INTO income_band SELECT * FROM obs_from_income_band_001;
INSERT INTO item SELECT * FROM obs_from_item_001;
INSERT INTO store SELECT * FROM obs_from_store_001;
INSERT INTO call_center SELECT * FROM obs_from_call_center_001;
INSERT INTO customer SELECT * FROM obs_from_customer_001;
INSERT INTO web_site SELECT * FROM obs_from_web_site_001;
INSERT INTO store_returns SELECT * FROM obs_from_store_returns_001;
INSERT INTO household_demographics SELECT * FROM obs_from_household_demographics_001;
INSERT INTO web_page SELECT * FROM obs_from_web_page_001;
INSERT INTO promotion SELECT * FROM obs_from_promotion_001;
INSERT INTO catalog_page SELECT * FROM obs_from_catalog_page_001;
INSERT INTO inventory SELECT * FROM obs_from_inventory_001;
INSERT INTO catalog_returns SELECT * FROM obs_from_catalog_returns_001;
INSERT INTO web_returns SELECT * FROM obs_from_web_returns_001;
INSERT INTO web_sales SELECT * FROM obs_from_web_sales_001;
INSERT INTO catalog_sales SELECT * FROM obs_from_catalog_sales_001;
INSERT INTO store_sales SELECT * FROM obs_from_store_sales_001;
```

4. 优化表性能。

```
ANALYZE customer_address;
ANALYZE customer_demographics;
ANALYZE date_dim;
ANALYZE warehouse;
ANALYZE ship_mode;
ANALYZE time_dim;
ANALYZE reason;
ANALYZE income_band;
ANALYZE item;
ANALYZE store;
ANALYZE call_center;
ANALYZE customer;
ANALYZE web_site;
ANALYZE store_returns;
ANALYZE household_demographics;
ANALYZE web_page;
ANALYZE promotion;
ANALYZE catalog_page;
ANALYZE inventory;
ANALYZE catalog_returns;
ANALYZE web_returns;
ANALYZE web_sales;
ANALYZE catalog_sales;
ANALYZE store_sales;
```

5. 使用SELECT语句进行查询。更多语法信息，请参考SELECT。

```
--查询所有记录，且按字母升序排列。
SELECT r_reason_desc FROM tpcds.reason ORDER BY r_reason_desc;

--根据查询条件过滤，并对结果进行分组。
SELECT r_reason_id, AVG(r_reason_sk) FROM tpcds.reason GROUP BY r_reason_id HAVING
AVG(r_reason_sk) > 25;
```


2.6 查看系统表

除了创建的表以外，数据库还包含很多系统表。这些系统表包含集群安装信息以及GaussDB(DWS)上运行的各种查询和进程的信息。可以通过查询系统表来收集有关数据库的信息。

“**系统表和系统视图**”中每个表的说明指出了表是对所有用户可见还是只对初始化用户可见。必须以初始化用户身份登录才能查询只对初始化用户可见的表。

查看数据库中包含的表

例如，在PG_TABLES系统表中查看public schema中包含的所有表。

```
SELECT distinct(tablename) FROM pg_tables WHERE SCHEMANAME = 'public';
```

结果类似如下这样：

```
    tablename
------------------
 err_hr_staffs
 test
 err_hr_staffs_ft3
 web_returns_p1
 mig_seq_table
 films4
(6 rows)
```

查看数据库用户

通过PG_USER可以查看数据库中所有用户的列表，还可以查看用户ID（USESYSID）和用户权限。

```
SELECT * FROM pg_user;
             usename             | usesysid | usecreatedb | usesuper | usecatupd | userepl |   passwd   | valbegin |
valuntil |   respool
   | parent | spacelimit | useconfig
-----------------------------------+----------+-------------+----------+-----------+---------+----------+----------
+----------+----------
----+--------+------------+-----------
dfc22b86afbd9a745668c3ecd0f15ec18 |    17107 | f           | f        | f         | f       | ******** |          |          |
default_p
ool |      0 |            |
guest                             |    17103 | f           | f        | f         | f       | ******** |          |          | default_p
ool |      0 |            |
Ruby                              |       10 | t           | t        | t         | t       | ******** |          |          | default_p
ool |      0 |            |
dbadmin                           |    16404 | f           | f        | f         | f       | ******** |          |          | default_p
ool |      0 |            |
lily                              |    16482 | f           | f        | f         | f       | ******** |          |          | default_p
ool |      0 |            |
jack                              |    16478 | f           | f        | f         | f       | ******** |          |          | default_p
ool |      0 |            |


(6 rows)
```

GaussDB(DWS)在内部使用Ruby执行日常管理和维护任务。可以向SELECT语句添加WHERE usesysid > 10来筛选查询，使其只显示用户定义的用户名称。

```
SELECT * FROM pg_user WHERE usesysid > 10;
             usename             | usesysid | usecreatedb | usesuper | usecatupd | userepl |   passwd   | valbegin |
valuntil |   respool
   | parent | spacelimit | useconfig
```

```
-----------------------------------+----------+-------------+----------+-----------+---------+----------+----------
+----------+----------
----+--------+------------+-----------
dfc22b86afbd9a745668c3ecd0f15ec18 |    17107 | f          | f       | f        | f       | ******** |         |         |
default_p
ool |      0 |           |
guest                             |    17103 | f          | f       | f        | f       | ******** |         |         | default_p
ool |      0 |           |
dbadmin                           |    16404 | f          | f       | f        | f       | ******** |         |         | default_p
ool |      0 |           |
lily                              |    16482 | f          | f       | f        | f       | ******** |         |         | default_p
ool |      0 |           |
jack                              |    16478 | f          | f       | f        | f       | ******** |         |         | default_p
ool |      0 |           |


(5 rows)
```

查看和停止正在运行的查询语句

通过视图**PG_STAT_ACTIVITY** 可以查看正在运行的查询语句。方法如下：

步骤1 设置参数track_activities为on。

```
SET track_activities = on;
```

当此参数为on时，数据库系统才会收集当前活动查询的运行信息。

步骤2 查看正在运行的查询语句。以查看正在运行的查询语句所连接的数据库名、执行查询的用户、查询状态及查询对应的PID为例：

```
SELECT datname, usename, state,pid FROM pg_stat_activity;
 datname  | usename | state  |       pid
----------+---------+--------+-----------------
 postgres | Ruby    | active | 140298793514752
 postgres | Ruby    | active | 140298718004992
 postgres | Ruby    | idle   | 140298650908416
 postgres | Ruby    | idle   | 140298625742592
 postgres | dbadmin | active | 140298575406848
(5 rows)
```

如果state字段显示为idle，则表明此连接处于空闲，等待用户输入命令。

如果仅需要查看非空闲的查询语句，则使用如下命令查看：

```
SELECT datname, usename, state FROM pg_stat_activity WHERE state != 'idle';
```

步骤3 若需要取消运行时间过长的查询，通过PG_TERMINATE_BACKEND函数，根据线程ID结束会话。

```
SELECT PG_TERMINATE_BACKEND(139834759993104);
```

显示类似如下信息，表示结束会话成功。

```
PG_TERMINATE_BACKEND
----------------------
 t
(1 row)
```

显示类似如下信息，表示用户执行了结束当前会话的操作。

```
FATAL:  terminating connection due to administrator command
FATAL:  terminating connection due to administrator command
```


说明

gsql客户端使用PG_TERMINATE_BACKEND函数结束当前会话后台线程时，客户端不会退出而是自动重连。即还会返回“The connection to the server was lost. Attempting reset: Succeeded.”

```
FATAL:  terminating connection due to administrator command
FATAL:  terminating connection due to administrator command
The connection to the server was lost. Attempting reset: Succeeded.
```

----结束

2.7 创建和管理 schema

背景信息

schema又称作模式。通过管理schema，允许多个用户使用同一数据库而不相互干扰，可以将数据库对象组织成易于管理的逻辑组，同时便于将第三方应用添加到相应的schema下而不引起冲突。 管理schema包括：创建schema、使用schema、删除schema、设置schema的搜索路径以及schema的权限控制。

注意事项

- 数据库集群包含一个或多个已命名数据库。用户和用户组在整个集群范围内是共享的，但是其数据并不共享。任何与服务器连接的用户都只能访问连接请求里声明的那个数据库。
- 一个数据库可以包含一个或多个已命名的schema，schema又包含表及其他数据库对象，包括数据类型、函数、操作符等。同一对象名可以在不同的schema中使用而不会引起冲突。例如，schema1和schema2都可以包含一个名为mytable的表。
- 和数据库不同，schema不是严格分离的。用户根据其对schema的权限，可以访问所连接数据库的schema中的对象。进行schema权限管理首先需要对数据库的权限控制进行了解。
- 不能创建以PG_为前缀的schema名，该类schema为数据库系统预留的。
- 在初始数据库gaussdb中创建用户时，系统会为新用户创建一个同名Schema。在其他数据库中，若需要同名Schema，则需要用户手动创建。
- 通过未修饰的表名（名字中只含有表名，没有“schema名”）引用表时，系统会通过search_path（搜索路径）来判断该表是哪个schema下的表。pg_temp和pg_catalog始终会作为搜索路径顺序中的前两位，无论二者是否出现在search_path中，或者出现在search_path中的任何位置。search_path（搜索路径）是一个schema名列表，在其中找到的第一个表就是目标表，如果没有找到则报错。（某个表即使存在，如果它的schema不在search_path中，依然会查找失败）在搜索路径中的第一个schema叫做"当前schema"。它是搜索时查询的第一个schema，同时在没有声明schema名时，新创建的数据库对象会默认存放在该schema下。
- 每个数据库都包含一个pg_catalog schema，它包含系统表和所有内置数据类型、函数、操作符。pg_catalog是搜索路径中的一部分，始终在临时表所属的模式后面，并在search_path中所有模式的前面，即具有第二搜索优先级。这样确保可以搜索到数据库内置对象。如果用户需要使用和系统内置对象重名的自定义对象时，可以在操作自定义对象时带上自己的模式。

操作步骤

- 创建schema
  - 执行如下命令来创建一个schema。
    ```
    CREATE SCHEMA myschema;
    ```
    当结果显示为如下信息，则表示成功创建一个名为myschema的schema。
    ```
    CREATE SCHEMA
    ```
    如果需要在模式中创建或者访问对象，其完整的对象名称由模式名称和具体的对象名称组成。中间由符号“.”隔开。例如：myschema.table。
  - 执行如下命令在创建schema时指定owner。
    ```
    CREATE SCHEMA myschema AUTHORIZATION dbadmin;
    ```
    当结果显示为如下信息，则表示成功创建一个属于dbadmin用户，名为myschema的schema。
    ```
    CREATE SCHEMA
    ```
- 使用schema

  在特定schema下创建对象或者访问特定schema下的对象，需要使用有schema修饰的对象名。该名字包含schema名以及对象名，他们之间用“.”号分开。
  - 执行如下命令在myschema下创建mytable表。
    ```
    CREATE TABLE myschema.mytable(id int, name varchar(20));
    CREATE TABLE
    ```
    如果在数据库中指定对象的位置，就需要使用有schema修饰的对象名称。
  - 执行如下命令查询myschema下mytable表的所有数据。
    ```
    SELECT * FROM myschema.mytable;
    id | name
    ----+------
    (0 rows)
    ```
- schema的搜索路径

  可以设置search_path配置参数指定寻找对象可用schema的顺序。在搜索路径列出的第一个schema会变成默认的schema。如果在创建对象时不指定schema，则会创建在默认的schema中。
  - 执行如下命令查看搜索路径。
    ```
    SHOW SEARCH_PATH;
     search_path
    ----------------
     "$user",public
    (1 row)
    ```
  - 执行如下命令将搜索路径设置为myschema、public，首先搜索myschema。
    ```
    SET SEARCH_PATH TO myschema, public;
    SET
    ```
- schema的权限控制

  默认情况下，用户只能访问属于自己的schema中的数据库对象。如果需要访问其他schema的对象，则该schema的所有者应该赋予他对该schema的usage权限。

  通过将模式的CREATE权限授予某用户，被授权用户就可以在此模式中创建对象。
  - 使用以下命令查看现有的schema：
    ```
    SELECT current_schema();
     current_schema
    ----------------
     myschema
    (1 row)
    ```
  - 执行如下命令创建用户jack，并将myschema的usage权限赋给用户jack。
    ```
    CREATE USER jack IDENTIFIED BY 'Bigdata@123';
    CREATE USER
    ```



```
GRANT USAGE ON schema myschema TO jack;
GRANT
```

  - 将用户jack对于myschema的usage权限收回。

```
REVOKE USAGE ON schema myschema FROM jack;
REVOKE
```

- 删除schema
  - 当schema为空时，即该schema下没有数据库对象，使用DROP SCHEMA命令进行删除。例如删除名为nullschema的空schema。

```
DROP SCHEMA IF EXISTS nullschema;
DROP SCHEMA
```

  - 当schema非空时，如果要删除一个schema及其包含的所有对象，需要使用CASCADE关键字。例如删除myschema及该schema下的所有对象。

```
DROP SCHEMA myschema CASCADE;
DROP SCHEMA
```

  - 执行如下命令删除用户jack。

```
DROP USER jack;
DROP USER
```

2.8 创建和管理分区表

背景信息

GaussDB(DWS)数据库支持的分区表为范围分区表。

范围分区表：将数据基于范围映射到每一个分区，这个范围是由创建分区表时指定的分区键决定的。这种分区方式是最为常用的，并且分区键经常采用日期，例如将销售数据按照月份进行分区。

分区表和普通表相比具有以下优点：

- 改善查询性能：对分区对象的查询可以仅搜索自己关心的分区，提高检索效率。
- 增强可用性：如果分区表的某个分区出现故障，表在其他分区的数据仍然可用。
- 方便维护：如果分区表的某个分区出现故障，需要修复数据，只修复该分区即可。
- 均衡I/O：可以把不同的分区映射到不同的磁盘以平衡I/O，改善整个系统性能。

普通表若要转成分区表，需要新建分区表，然后把普通表中的数据导入到新建的分区表中。因此在初始设计表时，请根据业务提前规划是否使用分区表。

操作步骤

按照以下方式对范围分区表的进行操作。

- 创建分区表

```
CREATE TABLE tpcds.customer_address
(
    ca_address_sk       integer                 NOT NULL   ,
    ca_address_id       character(16)           NOT NULL   ,
    ca_street_number    character(10)                      ,
    ca_street_name      character varying(60)              ,
    ca_street_type      character(15)                      ,
    ca_suite_number     character(10)                      ,
    ca_city             character varying(60)              ,
    ca_county           character varying(30)              ,
    ca_state            character(2)                       ,
    ca_zip              character(10)                      ,
    ca_country          character varying(20)              ,
```



```
    ca_gmt_offset       numeric(5,2)                ,
    ca_location_type    character(20)
)
TABLESPACE example1
DISTRIBUTE BY HASH (ca_address_sk)
PARTITION BY RANGE (ca_address_sk)
(
        PARTITION P1 VALUES LESS THAN(5000),
        PARTITION P2 VALUES LESS THAN(10000),
        PARTITION P3 VALUES LESS THAN(15000),
        PARTITION P4 VALUES LESS THAN(20000),
        PARTITION P5 VALUES LESS THAN(25000),
        PARTITION P6 VALUES LESS THAN(30000),
        PARTITION P7 VALUES LESS THAN(40000),
        PARTITION P8 VALUES LESS THAN(MAXVALUE) TABLESPACE example2
)
ENABLE ROW MOVEMENT;
```

当结果显示为如下信息，则表示创建成功。

```
CREATE TABLE
```

说明

创建列存分区表的数量建议不超过1000个。

- 插入数据

  将表tpcds.customer_address的数据插入到表tpcds.web_returns_p2中。

  例如在数据库中创建了一个表tpcds.customer_address的备份表tpcds.web_returns_p2，现在需要将表tpcds.customer_address中的数据插入到表tpcds.web_returns_p2中，则可以执行如下命令。

```
CREATE TABLE tpcds.web_returns_p2
(
    ca_address_sk       integer               NOT NULL   ,
    ca_address_id       character(16)           NOT NULL   ,
    ca_street_number    character(10)                    ,
    ca_street_name      character varying(60)             ,
    ca_street_type      character(15)                   ,
    ca_suite_number     character(10)                    ,
    ca_city             character varying(60)           ,
    ca_county           character varying(30)            ,
    ca_state            character(2)                   ,
    ca_zip              character(10)                  ,
    ca_country          character varying(20)            ,
    ca_gmt_offset       numeric(5,2)                   ,
    ca_location_type    character(20)
)
TABLESPACE example1
DISTRIBUTE BY HASH (ca_address_sk)
PARTITION BY RANGE (ca_address_sk)
(
        PARTITION P1 VALUES LESS THAN(5000),
        PARTITION P2 VALUES LESS THAN(10000),
        PARTITION P3 VALUES LESS THAN(15000),
        PARTITION P4 VALUES LESS THAN(20000),
        PARTITION P5 VALUES LESS THAN(25000),
        PARTITION P6 VALUES LESS THAN(30000),
        PARTITION P7 VALUES LESS THAN(40000),
        PARTITION P8 VALUES LESS THAN(MAXVALUE) TABLESPACE example2
)
ENABLE ROW MOVEMENT;
CREATE TABLE
INSERT INTO tpcds.web_returns_p2 SELECT * FROM tpcds.customer_address;
INSERT 0 0
```


说明

分区表不显示指定则默认不开启行迁移开关**ROW MOVEMENT**，此时不允许跨分区更新。ENABLE ROW MOVEMENT开启则允许跨分区更新，但此时如果有SELECT FOR UPDATE查询该分区表并发执行，存在查询结果瞬时不一致的可能性，需要谨慎使用。

- 修改分区表行迁移属性
  ```
  ALTER TABLE tpcds.web_returns_p2 DISABLE ROW MOVEMENT;
  ALTER TABLE
  ```
- 删除分区

  删除分区P8。
  ```
  ALTER TABLE tpcds.web_returns_p2 DROP PARTITION P8;
  ALTER TABLE
  ```
- 增加分区

  增加分区P8，范围为 40000<= P8<=MAXVALUE。
  ```
  ALTER TABLE tpcds.web_returns_p2 ADD PARTITION P8 VALUES LESS THAN (MAXVALUE);
  ALTER TABLE
  ```
- 重命名分区
  - 重命名分区P8为P_9。
    ```
    ALTER TABLE tpcds.web_returns_p2 RENAME PARTITION P8 TO P_9;
    ALTER TABLE
    ```
  - 重命名分区P_9为P8。
    ```
    ALTER TABLE tpcds.web_returns_p2 RENAME PARTITION FOR (40000) TO P8;
    ALTER TABLE
    ```
- 查询分区

  查询分区P7。
  ```
  SELECT * FROM tpcds.web_returns_p2 PARTITION (P7);
  SELECT * FROM tpcds.web_returns_p2 PARTITION FOR (35888);
  ```
- 查看分区表信息，可使用系统表dba_tab_partitions。
  ```
  select * from dba_tab_partitions where table_name='tpcds.customer_address';
  ```
- 删除分区表
  ```
  DROP TABLE tpcds.web_returns_p2;
  DROP TABLE
  ```

2.9 创建和管理索引

背景信息

索引可以提高数据的访问速度，但同时也增加了插入、更新和删除操作的处理时间。所以是否要为表增加索引，索引建立在哪些字段上，是创建索引前必须要考虑的问题。需要分析应用程序的业务处理、数据使用、经常被用作查询的条件或者被要求排序的字段来确定是否建立索引。

索引建立在数据库表中的某些列上。因此，在创建索引时，应该仔细考虑在哪些列上创建索引。

- 在经常需要搜索查询的列上创建索引，可以加快搜索的速度。
- 在作为主键的列上创建索引，强制该列的唯一性和组织表中数据的排列结构。
- 在经常使用连接的列上创建索引，这些列主要是一些外键，可以加快连接的速度。
- 在经常需要根据范围进行搜索的列上创建索引，因为索引已经排序，其指定的范围是连续的。

- 在经常需要排序的列上创建索引，因为索引已经排序，这样查询可以利用索引的排序，加快排序查询时间。
- 在经常使用WHERE子句的列上创建索引，加快条件的判断速度。
- 为经常出现在关键字ORDER BY、GROUP BY、DISTINCT后面的字段建立索引。

  说明

  - 索引创建成功后，系统会自动判断何时引用索引。当系统认为使用索引比顺序扫描更快时，就会使用索引。
  - 索引创建成功后，必须和表保持同步以保证能够准确地找到新数据，这样就增加了数据操作的负荷。因此请定期删除无用的索引。
  - 索引创建成功后，会对表里面的现有数据生效。

操作步骤

创建分区表的步骤请参考**创建和管理分区表**。

- 创建索引
  - 创建分区表索引tpcds_web_returns_p2_index1，不指定索引分区的名字。

    ```
    CREATE INDEX tpcds_web_returns_p2_index1 ON tpcds.web_returns_p2 (ca_address_id) LOCAL;
    ```

    当结果显示为如下信息，则表示创建成功。

    ```
    CREATE INDEX
    ```

  - 创建分区索引tpcds_web_returns_p2_index2，并指定索引分区的名字。

    ```
    CREATE INDEX tpcds_web_returns_p2_index2 ON tpcds.web_returns_p2 (ca_address_sk) LOCAL
    (
        PARTITION web_returns_p2_P1_index,
        PARTITION web_returns_p2_P2_index TABLESPACE example3,
        PARTITION web_returns_p2_P3_index TABLESPACE example4,
        PARTITION web_returns_p2_P4_index,
        PARTITION web_returns_p2_P5_index,
        PARTITION web_returns_p2_P6_index,
        PARTITION web_returns_p2_P7_index,
        PARTITION web_returns_p2_P8_index
    ) TABLESPACE example2;
    ```

    当结果显示为如下信息，则表示创建成功。

    ```
    CREATE INDEX
    ```

- 重命名索引分区

  执行如下命令对索引分区*web_returns_p2_P8_index*重命名*web_returns_p2_P8_index*_new。

  ```
  ALTER INDEX tpcds.tpcds_web_returns_p2_index2 RENAME PARTITION web_returns_p2_P8_index TO web_returns_p2_P8_index_new;
  ```

  当结果显示为如下信息，则表示重命名成功。

  ```
  ALTER INDEX
  ```

- 查询索引
  - 执行如下命令查询系统和用户定义的所有索引。

    ```
    SELECT RELNAME FROM PG_CLASS WHERE RELKIND='i';
    ```

  - 执行如下命令查询指定索引的信息。

    ```
    \di+ tpcds.tpcds_web_returns_p2_index2
    ```

- 删除索引

  ```
  DROP INDEX tpcds.tpcds_web_returns_p2_index1;
  DROP INDEX tpcds.tpcds_web_returns_p2_index2;
  ```

  当结果显示为如下信息，则表示删除成功。

  ```
  DROP INDEX
  ```


GaussDB(DWS)支持4种创建索引的方式请参见表2-3。

说明

- 索引创建成功后，系统会自动判断何时引用索引。当系统认为使用索引比顺序扫描更快时，就会使用索引。
- 索引创建成功后，必须和表保持同步以保证能够准确地找到新数据，这样就增加了数据操作的负荷。因此请定期删除无用的索引。

表 2-3 索引方式

索引方式	描述
唯一索引	可用于约束索引属性值的唯一性，或者属性组合值的唯一性。如果一个表声明了唯一约束或者主键，则GaussDB(DWS)自动在组成主键或唯一约束的字段上创建唯一索引（可能是多字段索引），以实现这些约束。目前，GaussDB(DWS)只有B-Tree可以创建唯一索引。
多字段索引	一个索引可以定义在表中的多个属性上。目前，GaussDB(DWS)中的B-Tree支持多字段索引，且最多可在32个字段上创建索引。
部分索引	建立在一个表的子集上的索引，这种索引方式只包含满足条件表达式的元组。
表达式索引	索引建立在一个函数或者从表中一个或多个属性计算出来的表达式上。表达式索引只有在查询时使用与创建时相同的表达式才会起作用。

- 创建一个普通表。

```
CREATE TABLE tpcds.customer_address_bak AS TABLE tpcds.customer_address;
INSERT 0 0
```

- 创建普通索引

  如果对于tpcds.customer_address_bak表，需要经常进行以下查询。

```
SELECT ca_address_sk FROM tpcds.customer_address_bak WHERE ca_address_sk=14888;
```

  通常，数据库系统需要逐行扫描整个tpcds.customer_address_bak表以寻找所有匹配的元组。如果表tpcds.customer_address_bak的规模很大，但满足WHERE条件的只有少数几个（可能是零个或一个），则这种顺序扫描的性能就比较差。如果让数据库系统在ca_address_sk属性上维护一个索引，用于快速定位匹配的元组，则数据库系统只需要在搜索树上查找少数的几层就可以找到匹配的元组，这将会大大提高数据查询的性能。同样，在数据库中进行更新和删除操作时，索引也可以提升这些操作的性能。

  使用以下命令创建索引。

```
CREATE INDEX index_wr_returned_date_sk ON tpcds.customer_address_bak (ca_address_sk);
CREATE INDEX
```

- 创建多字段索引

  假如用户需要经常查询表tpcds.customer_address_bak中ca_address_sk是5050，且ca_street_number小于1000的记录，使用以下命令进行查询。

```
SELECT ca_address_sk,ca_address_id FROM tpcds.customer_address_bak WHERE ca_address_sk = 5050 AND ca_street_number < 1000;
```

  使用以下命令在字段ca_address_sk和ca_street_number上定义一个多字段索引。

```
CREATE INDEX more_column_index ON
tpcds.customer_address_bak(ca_address_sk ,ca_street_number );
CREATE INDEX
```

- 创建部分索引

  如果只需要查询ca_address_sk为5050的记录，可以创建部分索引来提升查询效率。

  ```
  CREATE INDEX part_index ON tpcds.customer_address_bak(ca_address_sk) WHERE ca_address_sk = 5050;
  CREATE INDEX
  ```

- 创建表达式索引

  假如经常需要查询ca_street_number小于1000的信息，执行如下命令进行查询。

  ```
  SELECT * FROM tpcds.customer_address_bak WHERE trunc(ca_street_number) < 1000;
  ```

  可以为上面的查询创建表达式索引：

  ```
  CREATE INDEX para_index ON tpcds.customer_address_bak (trunc(ca_street_number));
  CREATE INDEX
  ```

- 删除tpcds.customer_address_bak表。

  ```
  DROP TABLE tpcds.customer_address_bak;
  DROP TABLE
  ```

2.10 创建和管理视图

背景信息

当用户对数据库中的一张或者多张表的某些字段的组合感兴趣，而又不想每次键入这些查询时，用户就可以定义一个视图，以便解决这个问题。

视图与基本表不同，不是物理上实际存在的，是一个虚表。数据库中仅存放视图的定义，而不存放视图对应的数据，这些数据仍存放在原来的基本表中。若基本表中的数据发生变化，从视图中查询出的数据也随之改变。从这个意义上讲，视图就像一个窗口，透过它可以看到数据库中用户感兴趣的数据及变化。视图每次被引用的时候都会运行一次。

管理视图

- 创建视图

  执行如下命令创建新视图MyView。

  ```
  CREATE OR REPLACE VIEW MyView AS SELECT * FROM tpcds.web_returns WHERE
  trunc(wr_refunded_cash) > 10000;
  CREATE VIEW
  ```

  说明

  CREATE VIEW中的OR REPLACE可有可无，当存在OR REPLACE时，表示若以前存在该视图就进行替换。

- 查询视图

  执行如下命令查询MyView视图。

  ```
  SELECT * FROM MyView;
  ```

- 查看当前用户下的视图

  ```
  SELECT * FROM user_views;
  ```

- 查看所有视图

  ```
  SELECT * FROM dba_views;
  ```

- 查看某视图的具体信息

  执行如下命令查询dba_users视图的详细信息。


```
\d+ dba_users
                    View "PG_CATALOG.DBA_USERS"
 Column  |         Type          | Modifiers | Storage  | Description
----------+-----------------------+-----------+----------+-------------
 USERNAME | CHARACTER VARYING(64) |           | extended |
View definition:
 SELECT PG_AUTHID.ROLNAME::CHARACTER VARYING(64) AS USERNAME
   FROM PG_AUTHID;
```

- 重建视图

  执行如下命令可以在不键入查询语句的情况下重建视图。

  ```
  ALTER VIEW MyView REBUILD;
  ALTER VIEW
  ```

- 删除视图

  执行如下命令删除MyView视图。

  ```
  DROP VIEW MyView;
  DROP VIEW
  ```

2.11 创建和管理序列

背景信息

序列Sequence是用来产生唯一整数的数据库对象。序列的值是按照一定规则自增的整数。因为自增所以不重复，因此说Sequence具有唯一标识性。这也是Sequence常被用作主键的原因。

通过序列使某字段成为唯一标识符的方法有两种：

- 一种是声明字段的类型为序列整型，由数据库在后台自动创建一个对应的Sequence。
- 另一种是使用CREATE SEQUENCE自定义一个新的Sequence，然后将nextval('sequence_name')函数读取的序列值，指定为某一字段的默认值，这样该字段就可以作为唯一标识符。

操作步骤

方法一： 声明字段类型为序列整型来定义标识符字段。例如：

```
CREATE TABLE T1
(
   id    serial,
   name  text
);
```

当结果显示为如下信息，则表示创建成功。

```
CREATE TABLE
```

方法二： 创建序列，并通过nextval('sequence_name')函数指定为某一字段的默认值。这种方式更灵活，可以为序列定义cache，一次预申请多个序列值，减少与GTM的交互次数，来提高性能。

1. 创建序列

   ```
   CREATE SEQUENCE seq1 cache 100;
   ```

   当结果显示为如下信息，则表示创建成功。

   ```
   CREATE SEQUENCE
   ```

2. 指定为某一字段的默认值，使该字段具有唯一标识属性。


```
CREATE TABLE T2
(
   id   int not null default nextval('seq1'),
   name text
);
```

当结果显示为如下信息，则表示默认值指定成功。

```
CREATE TABLE
```

3. 指定序列与列的归属关系。

   将序列和一个表的指定字段进行关联。这样，在删除那个字段或其所在表的时候会自动删除已关联的序列。

```
ALTER SEQUENCE seq1 OWNED BY T2.id;
```

当结果显示为如下信息，则表示指定成功。

```
ALTER SEQUENCE
```

说明

除了为序列指定了cache，方法二所实现的功能基本与方法一类似。但是一旦定义cache，序列将会产生空洞(序列值为不连贯的数值，如：1.4.5)，并且不能保序。另外为某序列指定从属列后，该列删除，对应的sequence也会被删除。 虽然数据库并不限制序列只能为一列产生默认值，但最好不要多列共用同一个序列。

当前版本只支持在定义表的时候指定自增列，或者指定某列的默认值为nextval('seqname')， 不支持在已有表中增加自增列或者增加默认值为nextval('seqname')的列。

注意事项

新序列值的产生是靠GTM维护的，默认情况下，每申请一个序列值都要向GTM发送一次申请，GTM在当前值的基础上加上步长值作为产生的新值返回给调用者。GTM作为全局唯一的节点，势必成为性能的瓶颈，所以对于需要大量频繁产生序列号的操作，如使用Bulkload工具进行数据导入场景，是非常不推荐产生默认序列值的。比如，在下面所示的场景中， INSERT FROM SELECT语句的性能会非常慢。

```
CREATE SEQUENCE newSeq1;
CREATE TABLE newT1
        (
          id   int not null default nextval('newSeq1'),
          name text
        );
INSERT INTO newT1(name) SELECT name from T1;
```

可以提高性能的写法是（假设T1表导入newT1表中的数据为10000行）：

```
INSERT INTO newT1(id, name) SELECT id,name from T1;
SELECT SETVAL('newSeq1',10000);
```

说明

序列操作函数nextval()，setval() 等均不支持回滚。另外setval设置的新值，会对当前会话的nextval立即生效，但对其他会话，如果定义了cache，不会立即生效，在用尽所有缓存的值后，其变动才被其他会话感知。所以为了避免产生重复值，要谨慎使用setval，设置的新值不能是已经产生的值或者在缓存中的值。

如果必须要在bulkload场景下产生默认序列值，则一定要为newSeq1定义足够大的cache，并且不要定义Maxvalue或者Minvalue。数据库会试图将nextval('sequence_name')的调用下推到Data Node，以提高性能。 目前GTM对并发的连接请求是有限制的，当Data Node很多时，将产生大量并发连接， 这时一定要控制bulkload的并发数目，避免耗尽GTM的连接资源。如果目标表为复制表(DISTRIBUTE BY REPLICATION)时下推将不能进行。当数据量较大时，这对数据库将是个灾难。除了性能问题之外，空间也可能会剧烈膨胀，在导入结束后，需要用

vacuum full来恢复。最好的方式还是如上建议的，不要在bulkload的场景中产生默认序列值。

另外，序列创建后，在每个节点上都维护了一张单行表，存储序列的定义及当前值，但此当前值并非GTM上的当前值，只是保存本节点与GTM交互后的状态。如果其他节点也向GTM申请了新值，或者调用了Setval修改了序列的状态，不会刷新本节点的单行表，但因每次申请序列值是向GTM申请，所以对序列正确性没有影响。

2.12 创建和管理定时任务

背景信息

当客户在使用数据库过程中，如果白天执行一些耗时比较长的任务（例如：统计数据汇总之类或从其他数据库同步数据的任务），会对正常的业务有性能影响，所以客户经常选择在晚上执行，无形中增加了客户的工作量。因此数据库兼容Orcale数据库中定时任务的功能，可以由客户创建定时任务，当任务时间点到达后可以自动触发任务的执行，从而可以减少客户运维的工作量。

数据库兼容Oracle定时任务功能主要通过DBMS.JOB高级包提供的接口，可以实现定时任务的创建、任务到期自动执行、任务删除、修改任务属性（包括：任务id、任务的关闭开启、任务的触发时间、触发时间间隔、任务内容等）。

定时任务管理

步骤1 创建测试表：

```
CREATE TABLE test(id int, time date);
```

当结果显示为如下信息，则表示创建成功。

```
CREATE TABLE
```

步骤2 创建自定义存储过程：

```
CREATE OR REPLACE PROCEDURE PRC_JOB_1()
AS
N_NUM integer :=1;
BEGIN
FOR I IN 1..1000 LOOP
INSERT INTO test VALUES(I,SYSDATE);
END LOOP;
END;
/
```

当结果显示为如下信息，则表示创建成功。

```
CREATE PROCEDURE
```

步骤3 创建任务：

- 新创建的任务（未指定job_id）表示每隔1分钟执行一次存储过程PRC_JOB_1。

```
call dbms_job.submit('call public.prc_job_1(); ', sysdate, 'interval ''1 minute''', :a);
job
-----
1
(1 row)
```

- 指定job_id创建任务

```
call dbms_job.isubmit(2,'call public.prc_job_1(); ', sysdate, 'interval ''1 minute''');
isubmit
---------

(1 row)
```


步骤4 通过视图查看当前用户已创建的任务信息

```
select job,dbname,start_date,last_date,this_date,next_date,broken,status,interval,failures,what from
user_jobs;
job | dbname |     start_date      |         last_date          |         this_date          |      next_date      | broken |
status |      interval       | failures |           what
-----+--------+---------------------+----------------------------+----------------------------+---------------------
+--------+--------+---------------------+----------+---------------------------
1 | gaussdb   | 2017-07-18 11:38:03 | 2017-07-18 13:53:03.607838 | 2017-07-18 13:53:03.607838 |
2017-07-18 13:54:03 | n      | s      | interval '1 minute' |        0 | call public.prc_job_1();
(1 row)
```

步骤5 停止任务

```
call dbms_job.broken(1,true);
broken
--------

(1 row)
```

步骤6 启动任务

```
call dbms_job.broken(1,false);
broken
--------

(1 row)
```

步骤7 修改任务属性

- 修改JOB的Next_date参数信息

  --修改Job1的Next_date为1小时以后开始执行。

  ```
  call dbms_job.next_date(1, sysdate+1.0/24);
  next_date
  -----------

  (1 row)
  ```

- 修改JOB的Interval参数信息

  --修改Job1的Interval为每隔1小时执行一次。

  ```
  call dbms_job.interval(1,'sysdate + 1.0/24');
  interval
  ----------

  (1 row)
  ```

- 修改JOB的What参数信息

  --修改Job1的What为执行SQL语句“insert into public.test values(333, sysdate +5);”。

  ```
  call dbms_job.what(1,'insert into public.test values(333, sysdate+5);');
  what
  ------

  (1 row)
  ```

- 同时修改JOB的Next_date、Interval、What等多个参数信息

  ```
  call dbms_job.change(1, 'call public.prc_job_1();', sysdate, 'interval ''1 minute''');
  change
  --------

  (1 row)
  ```

步骤8 删除JOB

```
call dbms_job.remove(1);
remove
--------

(1 row)
```


步骤9 JOB的权限控制

- 当创建一个JOB时，该JOB会和创建该JOB的数据库和用户绑定（即：pg_job系统视图新增的JOB记录中的dbname和log_user）。
- 如果当前用户是DBA用户、系统管理员、该JOB的创建用户（即：pg_job中的log_user），那么该用户有权限通过高级包接口remove、change、next_data、what、interval删除或修改JOB的参数信息。否则，会提示当前用户没有权限操作该JOB。
- 如果当前数据库是该JOB创建所属的数据库（即：为pg_job系统视图中的dbname），那么连接到当前数据库上可以通过高级包接口remove、change、next_data、what、interval删除或修改JOB的参数信息。
- 当删除JOB所属的数据库（即：为pg_job系统视图中的dbname）时，系统会关联删除该数据库从属的JOB记录。
- 当删除JOB所属的用户（即：为pg_job系统视图中的log_user）时，系统会关联删除该用户从属的JOB记录。

----结束

3 导入数据入门示例

本文档为您提供了以下3个相互独立的导入数据入门样例，您可以根据需要选择其中一个或几个样例进行体验。

- **交通卡口通行车辆分析**
- **某公司供应链需求分析**
- **零售业百货公司经营状况分析**

