数据仓库服务
8.1.3

性能白皮书

文档版本	22
发布日期	2024-10-25

目 录

1 概述

本性能测试基于TPC-H和TPC-DS标准测试集进行测试。

TPC-H

TPC-H由国际事务处理性能委员会（Transaction Processing Performance Council）制定发布，用于评测数据库的分析查询能力。TPC-H查询包含8张数据表和22条复杂SQL查询，大多数查询包含多表Join、子查询和Group By等。

TPC-DS

TPC-DS由国际事务处理性能委员会（Transaction Processing Performance Council）制定发布，用于决策支持系统测试基准，主要用于衡量大数据产品的分析性能。TPC-DS查询共包含24张表，99个查询测试语句。

2 测试结果

2.1 TPC-H 单并发查询

测试环境

表 2-1 测试环境

产品	规格	CPU	内存	存储	单节点建议存储	查询性能（TPC-H 1000X）	版本	节点数
GaussDB(DWS)	8xlarge	32U	256G	SSD云盘	6400GB	596.90s	8.1.3	3

测试结果

如下为scale=1000的TPC-H的性能测试结果，查询执行时间以秒（s）为单位。

表 2-2 TPC-H 1000X 测试结果

编号	TPC-H查询	测试结果（s）
1	Q1	22.15
2	Q2	2.65
3	Q3	28.29
4	Q4	49.07
5	Q5	44.45

编号	TPC-H查询	测试结果（s）
6	Q6	1.24
7	Q7	33.26
8	Q8	32.10
9	Q9	112.67
10	Q10	20.63
11	Q11	6.49
12	Q12	13.73
13	Q13	20.61
14	Q14	3.10
15	Q15	3.45
16	Q16	7.77
17	Q17	16.97
18	Q18	81.23
19	Q19	16.74
20	Q20	16.99
21	Q21	53.11
22	Q22	10.20
23	总时长（s）	596.90

2.2 TPC-DS 单并发查询

测试环境

表 2-3 测试环境

产品	规格	CPU	内存	存储	单节点建议存储	查询性能（TPC-DS 1000X）	版本	节点数
GaussDB(DWS)	8xlarge	32U	256G	SSD云盘	6400GB	1545.93s	8.1.3	3

测试结果

如下为scale=1000的TPC-DS的性能测试结果，查询执行时间以秒（s）为单位。

表 2-4 TPC-DS 1000X 测试结果

编号	TPC-DS查询	测试结果（s）
1	Q1	1.46
2	Q2	9.33
3	Q3	2.40
4	Q4	142.24
5	Q5	6.36
6	Q6	0.82
7	Q7	3.20
8	Q8	1.02
9	Q9	10.92
10	Q10	1.68
11	Q11	78.92
12	Q12	0.53
13	Q13	3.95
14	Q14	113.09
15	Q15	2.99
16	Q16	8.08
17	Q17	6.94
18	Q18	3.58
19	Q19	1.20
20	Q20	0.51
21	Q21	0.51
22	Q22	11.37
23	Q23	188.13
24	Q24	14.51
25	Q25	6.90
26	Q26	1.30
27	Q27	3.42

编号	TPC-DS查询	测试结果（s）
28	Q28	18.44
29	Q29	5.27
30	Q30	1.09
31	Q31	7.13
32	Q32	2.21
33	Q33	2.35
34	Q34	4.00
35	Q35	3.60
36	Q36	10.27
37	Q37	0.56
38	Q38	52.97
39	Q39	7.00
40	Q40	1.29
41	Q41	0.08
42	Q42	0.66
43	Q43	2.99
44	Q44	3.05
45	Q45	1.38
46	Q46	5.86
47	Q47	10.98
48	Q48	2.32
49	Q49	3.54
50	Q50	10.30
51	Q51	13.90
52	Q52	0.63
53	Q53	1.68
54	Q54	6.27
55	Q55	0.57
56	Q56	1.60
57	Q57	4.76

编号	TPC-DS查询	测试结果（s）
58	Q58	1.89
59	Q59	20.85
60	Q60	3.21
61	Q61	1.89
62	Q62	1.83
63	Q63	2.10
64	Q64	35.18
65	Q65	14.18
66	Q66	2.71
67	Q67	231.01
68	Q68	5.41
69	Q69	1.48
70	Q70	10.46
71	Q71	5.34
72	Q72	7.81
73	Q73	2.19
74	Q74	49.95
75	Q75	19.64
76	Q76	6.32
77	Q77	2.55
78	Q78	136.59
79	Q79	8.08
80	Q80	4.90
81	Q81	1.77
82	Q82	1.32
83	Q83	0.46
84	Q84	0.52
85	Q85	3.43
86	Q86	5.01
87	Q87	53.19

编号	TPC-DS查询	测试结果（s）
88	Q88	8.94
89	Q89	2.23
90	Q90	0.77
91	Q91	0.42
92	Q92	1.06
93	Q93	13.28
94	Q94	4.90
95	Q95	41.46
96	Q96	2.16
97	Q97	12.01
98	Q98	2.04
99	Q99	3.35
100	总时长（s）	1545.93

3 测试方法

3.1 总体流程

图 3-1 整体流程
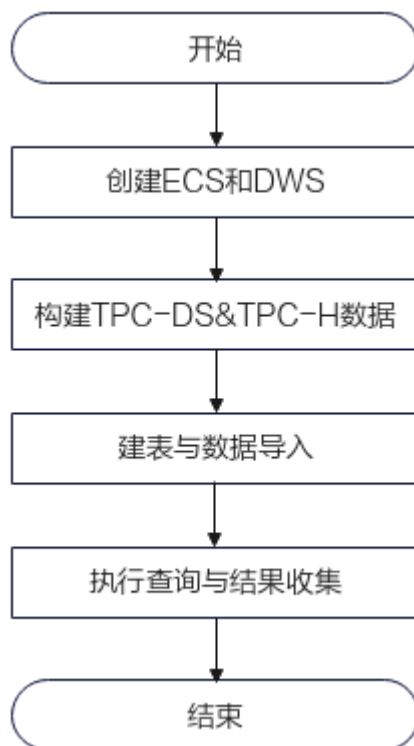
3.2 创建弹性云服务器 ECS 和数据仓库 GaussDB(DWS)

3.2.1 创建弹性云服务器 ECS

参考《弹性云服务器用户指南》创建弹性云服务器，创建的规格可参见下表。

📖 **说明**

由于TPC-DS、TPC-H数据集占用空间较大，以TPC-DS 1000X和TPC-H 1000X为例，分别占用930GB和1100GB。请创建弹性云服务器时，根据需求添加数据盘，举例如下：

- 单测TPC-DS或者TPC-H时：挂载2块超高IO 600GB数据盘。
- 同时测TPC-DS和TPC-H时：挂载2块超高IO 1200GB数据盘。

表 3-1 ECS 规格

参数项	参数取值
计费模式	按需计费
区域	华北-北京4
可用区	可用区1
CPU架构	鲲鹏计算
规格	鲲鹏通用计算增强型 | kc1.8xlarge.2 32vCPUs|64 GiB
镜像	EulerOS 2.8 64bit with ARM（40GB）
数据盘	系统盘通用型SSD 40GB，数据盘要求如下： ● 单测TPC-DS或者TPC-H时：挂载2块超高IO 600GB数据盘。 ● 同时测TPC-DS和TPC-H时：挂载2块超高IO 1200GB数据盘。

3.2.2 创建数据仓库 GaussDB(DWS)

参见"**创建集群**"章节创建GaussDB(DWS)数据仓库。**创建成功后，记录集群的内网IP。**

> **须知**
>
> 为确保ECS与GaussDB(DWS)网络互通，GaussDB(DWS)数据仓库需要与ECS在同一个区域，同一个虚拟私有云和子网下。

表 3-2 DWS 规格

参数项	参数取值
区域	华北-北京4
可用区	可用区1
产品类型	标准数仓
节点规格	8xlarge | 32 vCPUs | 256GB
每节点可用存储	500GB

参数项	参数取值
节点数	3

3.3 构建 TPC-H&TPC-DS 使用数据

3.3.1 准备数据构建工具编译环境

步骤1 远程连接ECS弹性云服务器。

步骤2 执行以下命令，安装git。

yum install git

步骤3 执行以下命令，安装gcc。

yum install gcc

----结束

3.3.2 TPC-H 数据构建

步骤1 从官网获取TPC-H工具。

步骤2 登录ECS云服务器，执行如下命令创建TPC-H存放目录。

mkdir -p /data1/script/tpch-kit/tpch1000X
mkdir -p /data2/script/tpch-kit/tpch1000X

步骤3 将获取的TPC-H工具通过SFTP工具上传到ECS的/data1/script/tpch-kit目录执行以下命令解压。

cd /data1/script/tpch-kit && unzip tpch_v3.0.0.zip

步骤4 执行如下命令编译生成数据构建工具dbgen。

须知

编译之前需要修改dbgen目录下的两个文件：makefile.suite和tpcd.h

1. 修改makefile.suite

```
#makefile.suite 的更改参数如下（103行-111行）

CC      = gcc
# Current values for DATABASE are: INFORMIX, DB2, TDAT (Teradata)
#                          SQLSERVER, SYBASE, ORACLE, VECTORWISE
# Current values for MACHINE are:  ATT, DOS, HP, IBM, ICL, MVS,
#                          SGI, SUN, U2200, VMS, LINUX, WIN32
# Current values for WORKLOAD are:  TPCH

DATABASE = POSTGRESQL     #程序给定参数没有postgresql，修改tpcd.h 添加POSTGRESQL脚本
MACHINE = LINUX
WORKLOAD = TPCH
```

2. 修改tpcd.h

```
//在tpcd.h文件增加如下语句
#ifdef POSTGRESQL
#define GEN_QUERY_PLAN   "EXPLAIN"
#define START_TRAN       "BEGIN TRANSACTION"
#define END_TRAN         "COMMIT;"
#define SET_OUTPUT       ""
#define SET_ROWCOUNT     "LIMIT %d\n"
#define SET_DBASE        ""
#endif /* POSTGRESQL */
```

```
cd TPC-H_Tools_v3.0.0/dbgen
cp makefile.suite makefile
make -f makefile
cp -R /data1/script/tpch-kit/TPC-H_Tools_v3.0.0/ /data2/script/tpch-kit/
```

步骤5 登录ECS，执行如下命令生成tpch 1000X数据，本示例分两个数据盘同步生成tpch 1000x数据。

须知

Tpch 1000X数据文件总大小约1100GB，请确认ECS的磁盘空间足够。

1. 进入**/data1/script/tpch-kit/TPC-H_Tools_v3.0.0/dbgen**目录后，执行如下命令：

   for c in {1..5};do (./dbgen -s *1000* **-C 10 -S** *${c}* **-f > /dev/null 2>&1 &);done**

2. 进入**/data2/script/tpch-kit/TPC-H_Tools_v3.0.0/dbgen**目录后，执行如下命令：

   for c in {6..10};do (./dbgen -s *1000* **-C 10 -S** *${c}* **-f > /dev/null 2>&1 &);done**

   其中：

   - -s 指定数据规模，本例为1000。
   - -C 指定分成几个chunk，本例为10。
   - -S 指定当前是第几个chunk，此处不需修改。

步骤6 执行以下命令，判断数据文件的生成进度。也可以通过**ps ux|grep dbgen**，查看生成数据文件的进程是否退出。

   du -sh /data1/script/tpch-kit/TPC-H_Tools_v3.0.0/dbgen/*.tbl*
   du -sh /data2/script/tpch-kit/TPC-H_Tools_v3.0.0/dbgen/*.tbl*

步骤7 将Tpch 1000X数据转移至指定目录。

```
mv /data1/script/tpch-kit/TPC-H_Tools_v3.0.0/dbgen/*.tbl* /data1/script/tpch-kit/tpch1000X
mv /data2/script/tpch-kit/TPC-H_Tools_v3.0.0/dbgen/*.tbl* /data2/script/tpch-kit/tpch1000X
```

----结束

3.3.3 TPC-DS 数据构建

步骤1 登录ECS云服务器，执行如下命令创建TPC-DS存放目录。

```
mkdir -p /data1/script/tpcds-kit/tpcds1000X
mkdir -p /data2/script/tpcds-kit/tpcds1000X
```

步骤2 从官网获取TPC-DS数据构建工具dsdgen最新版本，并通过SFTP工具上传到ECS的/
data1/script/tpcds-kit目录。

步骤3 执行如下命令解压tpcds的包并编译生成数据构建工具dsdgen。

- "*tpcds_3.2.0.zip*"替换为实际的软件包名。

- "*DSGen-software-code-3.2.0rc1*"替换为实际解压的文件夹名。

```
cd /data1/script/tpcds-kit && unzip tpcds_3.2.0.zip
cd DSGen-software-code-3.2.0rc1/tools && make
```

步骤4 进入**/data1/script/tpcds-kit/DSGen-software-code-3.2.0rc1/tools**目录后，执行以
下命令生成数据。

📖 **说明**

- 因为tpcds 1000X的数据，单个表的数据文件较大，我们采取分片生成的策略。

- Tpcds 1000X数据文件总大小约930GB，请确认ECS的磁盘空间足够。

- 由于生成的数据较大，只启动一个GDS导入数据会比较慢，建议均分两个数据盘来生成数
据，如下示例，1-5分片存放至*/data1/script/tpcds-kit/tpcds1000X*，6-10分片存放至*/
data2/script/tpcds-kit/tpcds1000X*。

```
for c in {1..5};do (./dsdgen -scale 1000 -dir /data1/script/tpcds-kit/tpcds1000X -TERMINATE N -parallel
10 -child ${c} -force Y > /dev/null 2>&1 &);done
for c in {6..10};do (./dsdgen -scale 1000 -dir /data2/script/tpcds-kit/tpcds1000X -TERMINATE N -parallel
10 -child ${c} -force Y > /dev/null 2>&1 &);done
```

其中：

- -scale 指定数据规模，本例为1000。

- -dir 指定生成数据文件存放的目录，本例为*/data1/script/tpcds-kit/tpcds1000X/
data2/script/tpcds-kit/tpcds1000X*。

- -TERMINATE 控制每行记录的末尾是否需要分隔符。

- -parallel 指定分片数，本例为10片。

- -child 指定当前是生成分片中的第几片，本例不需修改。

步骤5 执行以下命令，判断数据文件的生成进度。也可以通过**ps ux|grep dsdgen**，查看生成
数据文件的进程是否退出。

```
du -sh /data1/script/tpcds-kit/tpcds1000X/*.dat
du -sh /data2/script/tpcds-kit/tpcds1000X/*.dat
```
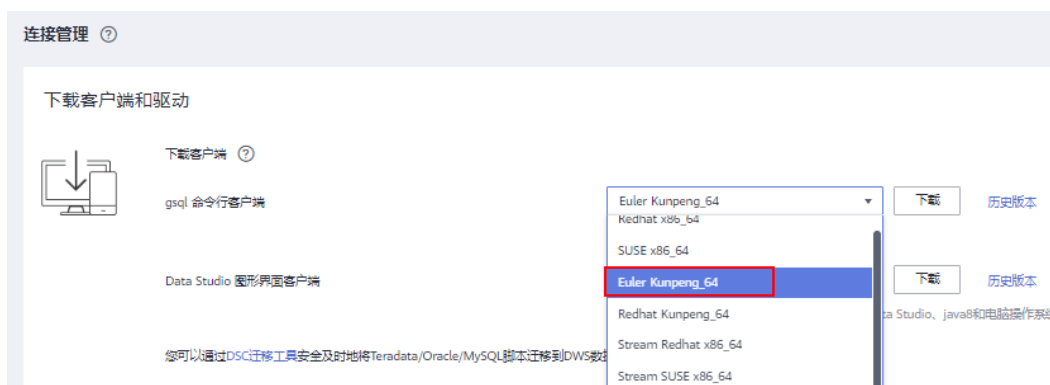
----结束

3.4 建表与数据导入

3.4.1 GDS 方式导入

3.4.1.1 安装和启动 GDS

安装和启动 GDS

步骤1 登录GaussDB(DWS)管理控制台。

步骤2 在左侧导航栏中，单击"连接客户端"。

步骤3 在"gsql 命令行客户端"的下拉列表中，选择对应版本的GaussDB(DWS)客户端。

请根据集群版本和安装客户端的操作系统，选择对应版本。

> 📖 **说明**
>
> 客户端CPU架构要和集群一致，如果集群是X86规格，则也应该选择X86客户端。

步骤4 单击"下载"。

步骤5 将GDS工具包上传至ECS的/opt目录中，本例以上传Euler Kunpeng版本的工具包为例。

步骤6 在工具包所在目录下，解压工具包。

cd */opt/*
unzip *dws_client_8.1.x_euler_kunpeng_x64.zip*

步骤7 创建用户gds_user及其所属的用户组gdsgrp。此用户用于启动GDS，且需要拥有读取数据源文件目录的权限。

groupadd *gdsgrp*
useradd -g *gdsgrp gds_user*

步骤8 修改工具包以及数据源文件目录属主为创建的用户gds_user及其所属的用户组gdsgrp。

chown -R *gds_user:gdsgrp /opt/*
chown -R *gds_user:gdsgrp /data1*
chown -R *gds_user:gdsgrp /data2*

步骤9 切换到gds_user用户。

su - *gds_user*

步骤10 执行环境依赖脚本（仅8.1.x版本适用）。

cd /opt/gds/bin
source gds_env

步骤11 启动GDS。

```
/opt/gds/bin/gds -d /data1/script/tpch-kit/tpch1000X -p 192.168.0.90:5000 -H 192.168.0.0/24 -l /opt/gds/
gds01_log.txt -D      #TPC-H使用
/opt/gds/bin/gds -d /data2/script/tpch-kit/tpch1000X -p 192.168.0.90:5001 -H 192.168.0.0/24 -l /opt/gds/
gds02_log.txt -D      #TPC-H使用
/opt/gds/bin/gds -d /data1/script/tpcds-kit/tpcds1000X/ -p 192.168.0.90:5002 -H 192.168.0.0/24 -
l /opt/gds/gds03_log.txt -D      #TPC-DS使用
/opt/gds/bin/gds -d /data2/script/tpcds-kit/tpcds1000X/ -p 192.168.0.90:5003 -H 192.168.0.0/24 -
l /opt/gds/gds04_log.txt -D      #TPC-DS使用
```

> **须知**
>
> - 命令中的斜体部分请根据实际填写，如果数据分片存放至多个数据盘目录，需要启动对应目录数量的GDS。
> - 如果TPC-H和TPC-DS数据同时测试，需要启动以上4个GDS，如果只测试TPC-DS或TPC-H数据，请根据后面的"#xxx"备注启动对应的GDS服务即可。

- **-d** *dir*：保存有待导入数据的数据文件所在目录。
- **-p** *ip:port*：GDS监听IP和监听端口。IP替换为**ECS的内网IP**，确保GaussDB(DWS)能通过此IP与GDS的通讯；端口对于TPC-H取5000、5001，对于TPC-DS取5002、5003。
- **-H** *address_string*：允许哪些主机连接和使用GDS服务。参数需为CIDR格式。此地址配置成**GaussDB(DWS)的集群内网网段**（即GDS所在的ECS与GaussDB(DWS)在同一个VPC下，以内网通讯即可），例如*192.168.0.0/24*。
- **-l** *log_file*：存放GDS的日志文件路径及文件名。
- **-D**：后台运行GDS。仅支持Linux操作系统下使用。

----结束

3.4.1.2 使用 gsql 连接 DWS

使用 gsql 连接 DWS 集群

步骤1 进入到ECS的/opt目录下，并执行环境变量。

```
cd /opt
source gsql_env.sh
```

步骤2 执行以下命令连接GaussDB(DWS)数据库。其中，*<DWS的内网IP>*、*<用户dbadmin密码>*从**创建数据仓库GaussDB(DWS)**获取。

```
gsql -d gaussdb -h <DWS的内网IP> -U dbadmin -p 8000 -r -W <用户dbadmin密码>;
```

----结束

3.4.1.3 创建 GDS 外表并导入 TPC-H 数据

本文介绍如何通过GDS外表导入TPC-H 1000x数据，**表3-3**列出了TPC-H测试数据集中的表数据行数。

> **须知**
>
> TPC-DS请跳过本章节。

表数据行数

表 3-3 TPC-H

序号	表名	行数
1	region	5
2	nation	25
3	supplier	10,000,000
4	customer	150,000,000
5	part	200,000,000
6	partsupp	800,000,000
7	orders	1,500,000,000
8	lineitem	5,999,989,709

操作步骤

步骤1 使用gsql连接DWS成功后，执行以下命令创建目标表（8张表）。

```
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)
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)
distribute by replication;

DROP TABLE IF EXISTS supplier;
CREATE TABLE supplier
(
    S_SUPPKEY    INT 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)
distribute by hash(S_SUPPKEY);

DROP TABLE IF EXISTS customer;
CREATE TABLE customer
```

```
(
    C_CUSTKEY      INT 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)
distribute by hash(C_CUSTKEY);

DROP TABLE IF EXISTS part;
CREATE TABLE part
(
    P_PARTKEY     INT 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        INT NOT NULL,
    P_CONTAINER   CHAR(10) NOT NULL,
    P_RETAILPRICE DECIMAL(15,2) NOT NULL,
    P_COMMENT     VARCHAR(23) NOT NULL
)
with (orientation = column)
distribute by hash(P_PARTKEY);

DROP TABLE IF EXISTS partsupp;
CREATE TABLE partsupp
(
    PS_PARTKEY     INT NOT NULL,
    PS_SUPPKEY     INT NOT NULL,
    PS_AVAILQTY    INT NOT NULL,
    PS_SUPPLYCOST  DECIMAL(15,2)  NOT NULL,
    PS_COMMENT     VARCHAR(199) NOT NULL
)
with (orientation = column)
distribute by hash(PS_PARTKEY);

DROP TABLE IF EXISTS orders;
CREATE TABLE orders
(
    O_ORDERKEY       BIGINT NOT NULL,
    O_CUSTKEY        INT 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   INT NOT NULL,
    O_COMMENT        VARCHAR(79) NOT NULL
)
with (orientation = column)
distribute by hash(O_ORDERKEY)
PARTITION BY RANGE(O_ORDERDATE)
(
    PARTITION O_ORDERDATE_1 VALUES LESS THAN('1993-01-01 00:00:00'),
    PARTITION O_ORDERDATE_2 VALUES LESS THAN('1994-01-01 00:00:00'),
    PARTITION O_ORDERDATE_3 VALUES LESS THAN('1995-01-01 00:00:00'),
    PARTITION O_ORDERDATE_4 VALUES LESS THAN('1996-01-01 00:00:00'),
    PARTITION O_ORDERDATE_5 VALUES LESS THAN('1997-01-01 00:00:00'),
    PARTITION O_ORDERDATE_6 VALUES LESS THAN('1998-01-01 00:00:00'),
    PARTITION O_ORDERDATE_7 VALUES LESS THAN('1999-01-01 00:00:00')
);

DROP TABLE IF EXISTS lineitem;
CREATE TABLE lineitem
```

```
(
    L_ORDERKEY    BIGINT NOT NULL,
    L_PARTKEY     INT NOT NULL,
    L_SUPPKEY     INT NOT NULL,
    L_LINENUMBER  INT 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)
distribute by hash(L_ORDERKEY)
PARTITION BY RANGE(L_SHIPDATE)
(
    PARTITION L_SHIPDATE_1 VALUES LESS THAN('1993-01-01 00:00:00'),
    PARTITION L_SHIPDATE_2 VALUES LESS THAN('1994-01-01 00:00:00'),
    PARTITION L_SHIPDATE_3 VALUES LESS THAN('1995-01-01 00:00:00'),
    PARTITION L_SHIPDATE_4 VALUES LESS THAN('1996-01-01 00:00:00'),
    PARTITION L_SHIPDATE_5 VALUES LESS THAN('1997-01-01 00:00:00'),
    PARTITION L_SHIPDATE_6 VALUES LESS THAN('1998-01-01 00:00:00'),
    PARTITION L_SHIPDATE_7 VALUES LESS THAN('1999-01-01 00:00:00')
);
```

步骤2 执行以下命令创建GDS外表（8张表）。

> **须知**
>
> 以下每个外表的"gsfs://***192.168.0.90:500x***/xxx | gsfs://***192.168.0.90:500x***/xxx"中的IP地址和端口，请替换成**安装和启动GDS**中的对应的GDS的监听IP和端口。如启动两个GDS，则使用"|"区分。如果启动多个GDS，需要将所有GDS的监听IP和端口配置到外表中。

```
DROP FOREIGN TABLE IF EXISTS region_load;
CREATE FOREIGN TABLE region_load
(
    R_REGIONKEY  INT,
    R_NAME       CHAR(25),
    R_COMMENT    VARCHAR(152)
)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5000/region* | gsfs://192.168.0.90:5001/region*',
format 'text',
delimiter '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
);

DROP FOREIGN TABLE IF EXISTS nation_load;
CREATE FOREIGN TABLE nation_load
(
    N_NATIONKEY  INT,
    N_NAME       CHAR(25),
    N_REGIONKEY  INT,
    N_COMMENT    VARCHAR(152)
)
```

```
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5000/nation* | gsfs://192.168.0.90:5001/nation*',
format 'text',
delimiter '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
);

DROP FOREIGN TABLE IF EXISTS supplier_load;
CREATE FOREIGN TABLE supplier_load
(
    S_SUPPKEY     INT,
    S_NAME        CHAR(25),
    S_ADDRESS     VARCHAR(40),
    S_NATIONKEY   INT,
    S_PHONE       CHAR(15),
    S_ACCTBAL     DECIMAL(15,2),
    S_COMMENT     VARCHAR(101)
)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5000/supplier* | gsfs://192.168.0.90:5001/supplier*',
format 'text',
delimiter '|',
encoding 'utf8',
mode 'Normal'
);

DROP FOREIGN TABLE IF EXISTS customer_load;
CREATE FOREIGN TABLE customer_load
(
    C_CUSTKEY     INT,
    C_NAME        VARCHAR(25),
    C_ADDRESS     VARCHAR(40),
    C_NATIONKEY   INT,
    C_PHONE       CHAR(15),
    C_ACCTBAL     DECIMAL(15,2),
    C_MKTSEGMENT  CHAR(10),
    C_COMMENT     VARCHAR(117)
)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5000/customer* | gsfs://192.168.0.90:5001/customer*',
format 'text',
delimiter '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
);

DROP FOREIGN TABLE IF EXISTS part_load;
CREATE FOREIGN TABLE part_load
(
    P_PARTKEY     INT,
    P_NAME        VARCHAR(55),
    P_MFGR        CHAR(25),
    P_BRAND       CHAR(10),
    P_TYPE        VARCHAR(25),
    P_SIZE        INT,
    P_CONTAINER   CHAR(10),
    P_RETAILPRICE DECIMAL(15,2),
    P_COMMENT     VARCHAR(23)
)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5000/part.* | gsfs://192.168.0.90:5001/part.*',
format 'text',
delimiter '|',
encoding 'utf8',
```

```
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
);

DROP FOREIGN TABLE IF EXISTS partsupp_load;
CREATE FOREIGN TABLE partsupp_load
(
    PS_PARTKEY     INT,
    PS_SUPPKEY     INT,
    PS_AVAILQTY    INT,
    PS_SUPPLYCOST  DECIMAL(15,2),
    PS_COMMENT     VARCHAR(199)
)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5000/partsupp* | gsfs://192.168.0.90:5001/partsupp*',
format 'text',
delimiter '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
);


DROP FOREIGN TABLE IF EXISTS orders_load;
CREATE FOREIGN TABLE orders_load
(
    O_ORDERKEY      BIGINT,
    O_CUSTKEY       INT,
    O_ORDERSTATUS   CHAR(1),
    O_TOTALPRICE    DECIMAL(15,2),
    O_ORDERDATE     DATE,
    O_ORDERPRIORITY CHAR(15),
    O_CLERK         CHAR(15),
    O_SHIPPRIORITY  INT,
    O_COMMENT       VARCHAR(79)
)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5000/orders* | gsfs://192.168.0.90:5001/orders*',
format 'text',
delimiter '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
);

DROP FOREIGN TABLE IF EXISTS lineitem_load;
CREATE FOREIGN TABLE lineitem_load
(
    L_ORDERKEY    BIGINT,
    L_PARTKEY     INT,
    L_SUPPKEY     INT,
    L_LINENUMBER  INT,
    L_QUANTITY    DECIMAL(15,2),
    L_EXTENDEDPRICE  DECIMAL(15,2),
    L_DISCOUNT    DECIMAL(15,2),
    L_TAX         DECIMAL(15,2),
    L_RETURNFLAG  CHAR(1),
    L_LINESTATUS  CHAR(1),
    L_SHIPDATE    DATE,
    L_COMMITDATE  DATE,
    L_RECEIPTDATE DATE,
    L_SHIPINSTRUCT CHAR(25),
    L_SHIPMODE    CHAR(10),
    L_COMMENT     VARCHAR(44)
)
SERVER gsmpp_server
```

```
OPTIONS(location 'gsfs://192.168.0.90:5000/lineitem* | gsfs://192.168.0.90:5001/lineitem*',
format 'text',
delimiter '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
);
```

步骤3 执行以下命令导入数据。

```
INSERT INTO region SELECT * FROM region_load;
INSERT INTO nation SELECT * FROM nation_load;
INSERT INTO supplier SELECT * FROM supplier_load;
INSERT INTO customer SELECT * FROM customer_load;
INSERT INTO part SELECT * FROM part_load;
INSERT INTO partsupp SELECT * FROM partsupp_load;
INSERT INTO orders SELECT * FROM orders_load;
INSERT INTO lineitem SELECT * FROM lineitem_load;
```

----结束

3.4.1.4 创建 GDS 外表并导入 TPC-DS 数据

本文介绍如何通过GDS外表导入TPC-DS 1000x数据，**表3-4**列出了TPC-DS测试数据集中的表数据行数。

须知

TPC-H请跳过本章节。

表数据行数

表 3-4 TPC-DS

序号	表名	行数
1	customer_address	6,000,000
2	customer_demographics	1,920,800
3	date_dim	73,049
4	warehouse	20
5	ship_mode	20
6	time_dim	86,400
7	reason	65
8	income_band	20
9	item	300,000
10	store	1,002
11	call_center	42

序号	表名	行数
12	customer	12,000,000
13	web_site	54
14	household_demographics	7,200
15	web_page	3,000
16	promotion	1,500
17	catalog_page	30,000
18	inventory	783,000,000
19	catalog_returns	143,996,756
20	web_returns	71,997,522
21	store_returns	287,999,764
22	web_sales	720,000,376
23	catalog_sales	1,439,980,416
24	store_sales	2,879,987,999

操作步骤

步骤1 使用gsql连接DWS成功后，执行以下SQL创建目标表（共24张表）。

```
CREATE TABLE customer_address
(
    ca_address_sk           bigint          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)
distribute by hash (ca_address_sk);

CREATE TABLE customer_demographics
(
    cd_demo_sk              bigint          not null ,
    cd_gender               char(1)                 ,
    cd_marital_status       char(1)                 ,
    cd_education_status     char(20)                ,
    cd_purchase_estimate    bigint                  ,
    cd_credit_rating        char(10)                ,
    cd_dep_count            bigint                  ,
    cd_dep_employed_count   bigint                  ,
    cd_dep_college_count    bigint
```

```
)
 with (orientation = column)
distribute by hash (cd_demo_sk);


CREATE TABLE date_dim
(
    d_date_sk               bigint              not null,
    d_date_id               char(16)            not null,
    d_date                  date                    ,
    d_month_seq             bigint                  ,
    d_week_seq              bigint                  ,
    d_quarter_seq           bigint                  ,
    d_year                  bigint                  ,
    d_dow                   bigint                  ,
    d_moy                   bigint                  ,
    d_dom                   bigint                  ,
    d_qoy                   bigint                  ,
    d_fy_year               bigint                  ,
    d_fy_quarter_seq        bigint                  ,
    d_fy_week_seq           bigint                  ,
    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             bigint                  ,
    d_last_dom              bigint                  ,
    d_same_day_ly           bigint                  ,
    d_same_day_lq           bigint                  ,
    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)
 DISTRIBUTE by hash(d_date_sk)
 PARTITION BY Range(d_year) (
     partition p1 values less than(1950),
     partition p2 values less than(2000),
     partition p3 values less than(2050),
     partition p4 values less than(2100),
     partition p5 values less than(3000),
     partition p6 values less than(maxvalue)
);


CREATE TABLE warehouse
(
    w_warehouse_sk          bigint              not null,
    w_warehouse_id          char(16)            not null,
    w_warehouse_name        varchar(20)             ,
    w_warehouse_sq_ft       bigint                  ,
    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)
distribute by replication;
```

```
CREATE TABLE ship_mode
(
    sm_ship_mode_sk         bigint              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)
distribute by replication;


CREATE TABLE time_dim
(
    t_time_sk             bigint              not null,
    t_time_id             char(16)              not null,
    t_time               bigint                 ,
    t_hour               bigint                 ,
    t_minute             bigint                 ,
    t_second             bigint                 ,
    t_am_pm              char(2)                  ,
    t_shift              char(20)                 ,
    t_sub_shift           char(20)                 ,
    t_meal_time           char(20)
)
 with (orientation = column)
distribute by hash (t_time_sk);


CREATE TABLE reason
(
    r_reason_sk            bigint              not null,
    r_reason_id            char(16)              not null,
    r_reason_desc          char(100)
)
 with (orientation = column)
distribute by replication;


CREATE TABLE income_band
(
    ib_income_band_sk       bigint              not null,
    ib_lower_bound          bigint                 ,
    ib_upper_bound          bigint
)
 with (orientation = column)
distribute by replication;

CREATE TABLE item
(
    i_item_sk             bigint              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            bigint                 ,
    i_brand              char(50)                 ,
    i_class_id            bigint                 ,
    i_class              char(50)                 ,
    i_category_id           bigint                 ,
    i_category            char(50)                 ,
    i_manufact_id           bigint                 ,
    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       bigint                ,
    i_product_name     char(50)
)
 with (orientation = column)
distribute by hash (i_item_sk);


CREATE TABLE store
(
    s_store_sk           bigint           not null,
    s_store_id           char(16)           not null,
    s_rec_start_date     date                  ,
    s_rec_end_date       date                  ,
    s_closed_date_sk     bigint                ,
    s_store_name         varchar(50)            ,
    s_number_employees   bigint                 ,
    s_floor_space        bigint                ,
    s_hours              char(20)              ,
    s_manager            varchar(40)            ,
    s_market_id          bigint                ,
    s_geography_class    varchar(100)            ,
    s_market_desc        varchar(100)            ,
    s_market_manager     varchar(40)             ,
    s_division_id        bigint                ,
    s_division_name      varchar(50)            ,
    s_company_id         bigint                ,
    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_precentage     decimal(5,2)
)
 with (orientation = column)
distribute by replication;

CREATE TABLE call_center
(
    cc_call_center_sk      bigint             not null,
    cc_call_center_id      char(16)             not null,
    cc_rec_start_date      date                  ,
    cc_rec_end_date        date                  ,
    cc_closed_date_sk      bigint                 ,
    cc_open_date_sk        bigint                 ,
    cc_name                varchar(50)            ,
    cc_class               varchar(50)            ,
    cc_employees           bigint                 ,
    cc_sq_ft               bigint                ,
    cc_hours               char(20)              ,
    cc_manager             varchar(40)             ,
    cc_mkt_id              bigint                ,
    cc_mkt_class           char(50)               ,
    cc_mkt_desc            varchar(100)            ,
    cc_market_manager      varchar(40)              ,
    cc_division            bigint                ,
    cc_division_name       varchar(50)             ,
    cc_company             bigint                 ,
    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)
distribute by replication;

drop table if exists customer;
CREATE TABLE customer
(
    c_customer_sk           bigint              not null,
    c_customer_id           char(16)            not null,
    c_current_cdemo_sk      bigint                ,
    c_current_hdemo_sk      bigint                ,
    c_current_addr_sk       bigint              ,
    c_first_shipto_date_sk  bigint              ,
    c_first_sales_date_sk   bigint              ,
    c_salutation            char(10)             ,
    c_first_name            char(20)             ,
    c_last_name             char(30)             ,
    c_preferred_cust_flag   char(1)               ,
    c_birth_day             bigint              ,
    c_birth_month           bigint                ,
    c_birth_year            bigint              ,
    c_birth_country         varchar(20)           ,
    c_login                 char(13)            ,
    c_email_address         char(50)             ,
    c_last_review_date_sk   char(10)
)
 with (orientation = column)
distribute by hash (c_customer_sk);


CREATE TABLE web_site
(
    web_site_sk             bigint              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        bigint               ,
    web_close_date_sk       bigint               ,
    web_class               varchar(50)           ,
    web_manager             varchar(40)             ,
    web_mkt_id              bigint              ,
    web_mkt_class           varchar(50)            ,
    web_mkt_desc            varchar(100)            ,
    web_market_manager      varchar(40)              ,
    web_company_id          bigint                ,
    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)
```

```
distribute by replication;

CREATE TABLE household_demographics
(
    hd_demo_sk              bigint              not null,
    hd_income_band_sk        bigint                    ,
    hd_buy_potential         char(15)                  ,
    hd_dep_count             bigint                    ,
    hd_vehicle_count         bigint
)
 with (orientation = column)
distribute by hash (hd_demo_sk);


CREATE TABLE web_page
(
    wp_web_page_sk           bigint              not null,
    wp_web_page_id           char(16)            not null,
    wp_rec_start_date        date                      ,
    wp_rec_end_date          date                      ,
    wp_creation_date_sk      bigint                    ,
    wp_access_date_sk        bigint                    ,
    wp_autogen_flag          char(1)                   ,
    wp_customer_sk           bigint                    ,
    wp_url                   varchar(100)              ,
    wp_type                  char(50)                  ,
    wp_char_count            bigint                    ,
    wp_link_count            bigint                    ,
    wp_image_count           bigint                    ,
    wp_max_ad_count          bigint
)
 with (orientation = column)
distribute by replication;


CREATE TABLE promotion
(
    p_promo_sk               bigint              not null,
    p_promo_id               char(16)            not null,
    p_start_date_sk          bigint                    ,
    p_end_date_sk            bigint                    ,
    p_item_sk                bigint                    ,
    p_cost                   decimal(15,2)             ,
    p_response_target        bigint                    ,
    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)
DISTRIBUTE BY HASH(p_promo_sk);


CREATE TABLE catalog_page
(
    cp_catalog_page_sk       bigint              not null,
    cp_catalog_page_id       char(16)            not null,
    cp_start_date_sk         bigint                    ,
    cp_end_date_sk           bigint                    ,
    cp_department            varchar(50)               ,
    cp_catalog_number        bigint                    ,
```

```
    cp_catalog_page_number    bigint                    ,
    cp_description            varchar(100)              ,
    cp_type                   varchar(100)
)
 with (orientation = column)
distribute by hash (cp_catalog_page_sk);

CREATE TABLE inventory
(
    inv_date_sk              bigint            not null,
    inv_item_sk              bigint            not null,
    inv_warehouse_sk         bigint            not null,
    inv_quantity_on_hand     integer
)
 with (orientation = column)
distribute by hash (inv_item_sk)
partition by range(inv_date_sk)
(
    partition p1 values less than(2451180),
    partition p2 values less than(2451545),
    partition p3 values less than(2451911),
    partition p4 values less than(2452276),
    partition p5 values less than(2452641),
    partition p6 values less than(2453006),
    partition p7 values less than(maxvalue)
)
;

CREATE TABLE catalog_returns
(
    cr_returned_date_sk      bigint                        ,
    cr_returned_time_sk      bigint                        ,
    cr_item_sk               bigint            not null,
    cr_refunded_customer_sk  bigint                        ,
    cr_refunded_cdemo_sk     bigint                        ,
    cr_refunded_hdemo_sk     bigint                        ,
    cr_refunded_addr_sk      bigint                        ,
    cr_returning_customer_sk bigint                        ,
    cr_returning_cdemo_sk    bigint                        ,
    cr_returning_hdemo_sk    bigint                        ,
    cr_returning_addr_sk     bigint                        ,
    cr_call_center_sk        bigint                        ,
    cr_catalog_page_sk       bigint                        ,
    cr_ship_mode_sk          bigint                        ,
    cr_warehouse_sk          bigint                        ,
    cr_reason_sk             bigint                        ,
    cr_order_number          bigint            not null,
    cr_return_quantity       bigint                        ,
    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)
distribute by hash (cr_item_sk)
partition by range(cr_returned_date_sk)
(
    partition p1 values less than(2450815),
    partition p2 values less than(2451180),
    partition p3 values less than(2451545),
    partition p4 values less than(2451911),
    partition p5 values less than(2452276),
    partition p6 values less than(2452641),
    partition p7 values less than(2453006),
```

```
   partition p8 values less than(maxvalue)
)
;


CREATE TABLE web_returns
(
   wr_returned_date_sk      bigint                    ,
   wr_returned_time_sk      bigint                    ,
   wr_item_sk               bigint           not null,
   wr_refunded_customer_sk  bigint                    ,
   wr_refunded_cdemo_sk     bigint                    ,
   wr_refunded_hdemo_sk     bigint                    ,
   wr_refunded_addr_sk      bigint                    ,
   wr_returning_customer_sk bigint                    ,
   wr_returning_cdemo_sk    bigint                    ,
   wr_returning_hdemo_sk    bigint                    ,
   wr_returning_addr_sk     bigint                    ,
   wr_web_page_sk           bigint                    ,
   wr_reason_sk             bigint                    ,
   wr_order_number          bigint           not null,
   wr_return_quantity       bigint                    ,
   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)
distribute by hash (wr_item_sk)
partition by range(wr_returned_date_sk)
(
   partition p1 values less than(2450815),
   partition p2 values less than(2451180),
   partition p3 values less than(2451545),
   partition p4 values less than(2451911),
   partition p5 values less than(2452276),
   partition p6 values less than(2452641),
   partition p7 values less than(2453006),
   partition p8 values less than(maxvalue)
)
;

CREATE TABLE store_returns
(
   sr_returned_date_sk      bigint                    ,
   sr_return_time_sk        bigint                    ,
   sr_item_sk               bigint           not null,
   sr_customer_sk           bigint                    ,
   sr_cdemo_sk              bigint                    ,
   sr_hdemo_sk              bigint                    ,
   sr_addr_sk               bigint                    ,
   sr_store_sk              bigint                    ,
   sr_reason_sk             bigint                    ,
   sr_ticket_number         bigint           not null,
   sr_return_quantity       bigint                    ,
   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)
distribute by hash (sr_item_sk)
partition by range(sr_returned_date_sk)
(
    partition p1 values less than (2451180) ,
    partition p2 values less than (2451545) ,
    partition p3 values less than (2451911) ,
    partition p4 values less than (2452276) ,
    partition p5 values less than (2452641) ,
    partition p6 values less than (2453006) ,
    partition p7 values less than (maxvalue)
)
;


CREATE TABLE web_sales
(
    ws_sold_date_sk          bigint                  ,
    ws_sold_time_sk          bigint                  ,
    ws_ship_date_sk          bigint                  ,
    ws_item_sk               bigint          not null,
    ws_bill_customer_sk      bigint                  ,
    ws_bill_cdemo_sk         bigint                  ,
    ws_bill_hdemo_sk         bigint                  ,
    ws_bill_addr_sk          bigint                  ,
    ws_ship_customer_sk      bigint                  ,
    ws_ship_cdemo_sk         bigint                  ,
    ws_ship_hdemo_sk         bigint                  ,
    ws_ship_addr_sk          bigint                  ,
    ws_web_page_sk           bigint                  ,
    ws_web_site_sk           bigint                  ,
    ws_ship_mode_sk          bigint                  ,
    ws_warehouse_sk          bigint                  ,
    ws_promo_sk              bigint                  ,
    ws_order_number          bigint          not null,
    ws_quantity              bigint                  ,
    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)
distribute by hash (ws_item_sk)
partition by range(ws_sold_date_sk)
(
    partition p1 values less than(2451180),
    partition p2 values less than(2451545),
    partition p3 values less than(2451911),
    partition p4 values less than(2452276),
    partition p5 values less than(2452641),
    partition p6 values less than(2453006),
    partition p7 values less than(maxvalue)
)
;

CREATE TABLE catalog_sales
(
```

```
    cs_sold_date_sk            bigint                    ,
    cs_sold_time_sk            bigint                    ,
    cs_ship_date_sk            bigint                    ,
    cs_bill_customer_sk        bigint                    ,
    cs_bill_cdemo_sk           bigint                    ,
    cs_bill_hdemo_sk           bigint                    ,
    cs_bill_addr_sk            bigint                   ,
    cs_ship_customer_sk        bigint                     ,
    cs_ship_cdemo_sk           bigint                    ,
    cs_ship_hdemo_sk           bigint                    ,
    cs_ship_addr_sk            bigint                   ,
    cs_call_center_sk          bigint                   ,
    cs_catalog_page_sk         bigint                    ,
    cs_ship_mode_sk            bigint                   ,
    cs_warehouse_sk            bigint                   ,
    cs_item_sk                 bigint           not null,
    cs_promo_sk                bigint                 ,
    cs_order_number            bigint           not null,
    cs_quantity                bigint                 ,
    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)
distribute by hash (cs_item_sk)
partition by range(cs_sold_date_sk)
(
    partition p1 values less than(2451180),
    partition p2 values less than(2451545),
    partition p3 values less than(2451911),
    partition p4 values less than(2452276),
    partition p5 values less than(2452641),
    partition p6 values less than(2453006),
    partition p7 values less than(maxvalue)
)
;

CREATE TABLE store_sales
(
    ss_sold_date_sk            bigint                   ,
    ss_sold_time_sk            bigint                   ,
    ss_item_sk                 bigint           not null,
    ss_customer_sk             bigint                    ,
    ss_cdemo_sk                bigint                   ,
    ss_hdemo_sk                bigint                   ,
    ss_addr_sk                 bigint                  ,
    ss_store_sk                bigint                  ,
    ss_promo_sk                bigint                  ,
    ss_ticket_number           bigint           not null,
    ss_quantity                bigint                  ,
    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)
distribute by hash (ss_item_sk)
partition by range(ss_sold_date_sk)
(
    partition p1 values less than(2451180),
    partition p2 values less than(2451545),
    partition p3 values less than(2451911),
    partition p4 values less than(2452276),
    partition p5 values less than(2452641),
    partition p6 values less than(2453006),
    partition p7 values less than(maxvalue)
)
;
```

步骤2 执行以下SQL语句创建GDS外表（共24张表）。

> **须知**
>
> 以下每个外表的"gsfs://**192.168.0.90:500x**/xxx | gsfs://**192.168.0.90:500x**/xxx"中的IP地址和端口，请替换成**安装和启动GDS**中的对应的GDS的监听IP和端口。如启动两个GDS，则使用"|"区分。如果启动多个GDS，需要将所有GDS的监听IP和端口配置到外表中。

```
DROP FOREIGN TABLE IF EXISTS customer_address_ext;
CREATE FOREIGN TABLE customer_address_ext
(
    ca_address_sk           bigint              ,
    ca_address_id           char(16)            ,
    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)
)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5002/customer_address* | gsfs://192.168.0.90:5003/
customer_address*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with  customer_address_err;

DROP FOREIGN TABLE IF EXISTS customer_demographics_ext;
CREATE FOREIGN TABLE customer_demographics_ext
(
    cd_demo_sk              bigint              ,
    cd_gender               char(1)             ,
    cd_marital_status       char(1)             ,
    cd_education_status     char(20)                ,
```

```
    cd_purchase_estimate    bigint                  ,
    cd_credit_rating        char(10)                ,
    cd_dep_count            bigint                  ,
    cd_dep_employed_count   bigint                  ,
    cd_dep_college_count    bigint
)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5002/customer_demographics* | gsfs://192.168.0.90:5003/
customer_demographics*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with  customer_demographics_err;

DROP FOREIGN TABLE IF EXISTS date_dim_ext;
CREATE FOREIGN TABLE date_dim_ext
(
    d_date_sk               bigint                  ,
    d_date_id               char(16)                ,
    d_date                  date                    ,
    d_month_seq             bigint                  ,
    d_week_seq              bigint                  ,
    d_quarter_seq           bigint                  ,
    d_year                  bigint                  ,
    d_dow                   bigint                  ,
    d_moy                   bigint                  ,
    d_dom                   bigint                  ,
    d_qoy                   bigint                  ,
    d_fy_year               bigint                  ,
    d_fy_quarter_seq        bigint                  ,
    d_fy_week_seq           bigint                  ,
    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             bigint                  ,
    d_last_dom              bigint                  ,
    d_same_day_ly           bigint                  ,
    d_same_day_lq           bigint                  ,
    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 'gsfs://192.168.0.90:5002/date_dim* | gsfs://192.168.0.90:5003/date_dim*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with  date_dim_err;

DROP FOREIGN TABLE IF EXISTS warehouse_ext;
CREATE FOREIGN TABLE warehouse_ext
(
    w_warehouse_sk          bigint                  ,
    w_warehouse_id          char(16)                ,
    w_warehouse_name        varchar(20)             ,
    w_warehouse_sq_ft       bigint                  ,
    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 'gsfs://192.168.0.90:5002/warehouse* | gsfs://192.168.0.90:5003/warehouse*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with  warehouse_err;

DROP FOREIGN TABLE IF EXISTS ship_mode_ext;
CREATE FOREIGN TABLE ship_mode_ext
(
sm_ship_mode_sk        bigint                   ,
sm_ship_mode_id        char(16)                 ,
sm_type                char(30)                 ,
sm_code                char(10)                 ,
sm_carrier             char(20)                 ,
sm_contract            char(20)
)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5002/ship_mode* | gsfs://192.168.0.90:5003/ship_mode*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with  ship_mode_err;

DROP FOREIGN TABLE IF EXISTS time_dim_ext;
CREATE FOREIGN TABLE time_dim_ext
(
t_time_sk              bigint                   ,
t_time_id              char(16)                 ,
t_time                 bigint                   ,
t_hour                 bigint                   ,
t_minute               bigint                   ,
t_second               bigint                   ,
t_am_pm                char(2)                  ,
t_shift                char(20)                 ,
t_sub_shift            char(20)                 ,
t_meal_time            char(20)
)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5002/time_dim* | gsfs://192.168.0.90:5003/time_dim*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with  time_dim_err;

DROP FOREIGN TABLE IF EXISTS reason_ext;
CREATE FOREIGN TABLE reason_ext
```

```
(
r_reason_sk          bigint          ,
r_reason_id          char(16)          ,
r_reason_desc          char(100)
)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5002/reason* | gsfs://192.168.0.90:5003/reason*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with reason_err;

DROP FOREIGN TABLE IF EXISTS income_band_ext;
CREATE FOREIGN TABLE income_band_ext
(
ib_income_band_sk          bigint          ,
ib_lower_bound          bigint          ,
ib_upper_bound          bigint
)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5002/income_band* | gsfs://192.168.0.90:5003/income_band*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with  income_band_err;

DROP FOREIGN TABLE IF EXISTS item_ext;
CREATE FOREIGN TABLE item_ext
(
   i_item_sk          bigint          ,
   i_item_id          char(16)          ,
   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          bigint          ,
   i_brand          char(50)          ,
   i_class_id          bigint          ,
   i_class          char(50)          ,
   i_category_id          bigint          ,
   i_category          char(50)          ,
   i_manufact_id          bigint          ,
   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          bigint          ,
   i_product_name          char(50)
)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5002/item* | gsfs://192.168.0.90:5003/item*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
```

```
with  item_err;

DROP FOREIGN TABLE IF EXISTS store_ext;
CREATE FOREIGN TABLE store_ext
(
    s_store_sk              bigint              ,
    s_store_id              char(16)              ,
    s_rec_start_date        date                 ,
    s_rec_end_date          date                 ,
    s_closed_date_sk        bigint               ,
    s_store_name            varchar(50)            ,
    s_number_employees      bigint                 ,
    s_floor_space           bigint              ,
    s_hours                 char(20)             ,
    s_manager               varchar(40)            ,
    s_market_id             bigint              ,
    s_geography_class       varchar(100)            ,
    s_market_desc           varchar(100)           ,
    s_market_manager        varchar(40)            ,
    s_division_id           bigint              ,
    s_division_name         varchar(50)            ,
    s_company_id            bigint              ,
    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_precentage        decimal(5,2)
)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5002/store_[^rs]* | gsfs://192.168.0.90:5003/storet_[^rs]*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with store_err;

DROP FOREIGN TABLE IF EXISTS call_center_ext;
CREATE FOREIGN TABLE call_center_ext
(
    cc_call_center_sk       bigint              ,
    cc_call_center_id       char(16)             ,
    cc_rec_start_date       date                 ,
    cc_rec_end_date         date                 ,
    cc_closed_date_sk       bigint               ,
    cc_open_date_sk         bigint               ,
    cc_name                 varchar(50)            ,
    cc_class                varchar(50)            ,
    cc_employees            bigint               ,
    cc_sq_ft                bigint              ,
    cc_hours                char(20)             ,
    cc_manager              varchar(40)            ,
    cc_mkt_id               bigint              ,
    cc_mkt_class            char(50)             ,
    cc_mkt_desc             varchar(100)           ,
    cc_market_manager       varchar(40)            ,
    cc_division             bigint               ,
    cc_division_name        varchar(50)            ,
    cc_company              bigint              ,
    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 'gsfs://192.168.0.90:5002/call_center* | gsfs://192.168.0.90:5003/call_center*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with call_center_err;

DROP FOREIGN TABLE IF EXISTS customer_ext;
CREATE FOREIGN TABLE customer_ext
(
    c_customer_sk           bigint              ,
    c_customer_id           char(16)            ,
    c_current_cdemo_sk      bigint                  ,
    c_current_hdemo_sk      bigint                  ,
    c_current_addr_sk       bigint                  ,
    c_first_shipto_date_sk  bigint                  ,
    c_first_sales_date_sk   bigint                  ,
    c_salutation            char(10)                ,
    c_first_name            char(20)                ,
    c_last_name             char(30)                ,
    c_preferred_cust_flag   char(1)                 ,
    c_birth_day             bigint                  ,
    c_birth_month           bigint                  ,
    c_birth_year            bigint                  ,
    c_birth_country         varchar(20)             ,
    c_login                 char(13)                ,
    c_email_address         char(50)                ,
    c_last_review_date_sk   char(10)
)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5002/customer_[^ad]* | gsfs://192.168.0.90:5003/customer_[^ad]*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'GBK',
mode 'Normal'
)
with  customer_err;

DROP FOREIGN TABLE IF EXISTS web_site_ext;
CREATE FOREIGN TABLE web_site_ext
(
    web_site_sk             bigint              ,
    web_site_id             char(16)                ,
    web_rec_start_date      date                    ,
    web_rec_end_date        date                    ,
    web_name                varchar(50)             ,
    web_open_date_sk        bigint                  ,
    web_close_date_sk       bigint                  ,
    web_class               varchar(50)             ,
    web_manager             varchar(40)             ,
    web_mkt_id              bigint                  ,
    web_mkt_class           varchar(50)             ,
    web_mkt_desc            varchar(100)            ,
```

```
    web_market_manager      varchar(40)           ,
    web_company_id          bigint                ,
    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 'gsfs://192.168.0.90:5002/web_site* | gsfs://192.168.0.90:5003/web_site*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with  web_site_err;

DROP FOREIGN TABLE IF EXISTS store_returns_ext;
CREATE FOREIGN TABLE store_returns_ext
(
    sr_returned_date_sk     bigint                ,
    sr_return_time_sk       bigint                ,
    sr_item_sk              bigint               ,
    sr_customer_sk          bigint                ,
    sr_cdemo_sk             bigint                ,
    sr_hdemo_sk             bigint                ,
    sr_addr_sk              bigint                ,
    sr_store_sk             bigint                ,
    sr_reason_sk            bigint                ,
    sr_ticket_number        bigint                ,
    sr_return_quantity      bigint                ,
    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 'gsfs://192.168.0.90:5002/store_returns* | gsfs://192.168.0.90:5003/store_returns*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with store_returns_err;

DROP FOREIGN TABLE IF EXISTS household_demographics_ext;
CREATE FOREIGN TABLE household_demographics_ext
(
    hd_demo_sk              bigint                ,
    hd_income_band_sk       bigint                 ,
    hd_buy_potential        char(15)                ,
    hd_dep_count            bigint                 ,
    hd_vehicle_count        bigint
```

```
)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5002/household_demographics* | gsfs://192.168.0.90:5003/
household_demographics*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with  household_demographics_err;

DROP FOREIGN TABLE IF EXISTS web_page_ext;
CREATE FOREIGN TABLE web_page_ext
(
    wp_web_page_sk          bigint              ,
    wp_web_page_id          char(16)            ,
    wp_rec_start_date       date                ,
    wp_rec_end_date         date                ,
    wp_creation_date_sk     bigint              ,
    wp_access_date_sk       bigint              ,
    wp_autogen_flag         char(1)             ,
    wp_customer_sk          bigint              ,
    wp_url                  varchar(100)        ,
    wp_type                 char(50)            ,
    wp_char_count           bigint              ,
    wp_link_count           bigint              ,
    wp_image_count          bigint              ,
    wp_max_ad_count         bigint
)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5002/web_page* | gsfs://192.168.0.90:5003/web_page*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with web_page_err;

DROP FOREIGN TABLE IF EXISTS promotion_ext;
CREATE FOREIGN TABLE promotion_ext
(
    p_promo_sk              bigint              ,
    p_promo_id              char(16)            ,
    p_start_date_sk         bigint              ,
    p_end_date_sk           bigint              ,
    p_item_sk               bigint              ,
    p_cost                  decimal(15,2)       ,
    p_response_target       bigint              ,
    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 'gsfs://192.168.0.90:5002/promotion* | gsfs://192.168.0.90:5003/promotion*',
FORMAT 'TEXT' ,
DELIMITER '|',
```

```
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with promotion_err;

DROP FOREIGN TABLE IF EXISTS catalog_page_ext;
CREATE FOREIGN TABLE catalog_page_ext
(
    cp_catalog_page_sk        bigint              ,
    cp_catalog_page_id        char(16)            ,
    cp_start_date_sk          bigint                  ,
    cp_end_date_sk            bigint                  ,
    cp_department             varchar(50)                 ,
    cp_catalog_number         bigint                  ,
    cp_catalog_page_number    bigint                      ,
    cp_description            varchar(100)                ,
    cp_type                   varchar(100)
)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5002/catalog_page* | gsfs://192.168.0.90:5003/catalog_page*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with  catalog_page_err;

DROP FOREIGN TABLE IF EXISTS inventory_ext;
CREATE FOREIGN TABLE inventory_ext
(
    inv_date_sk               bigint              ,
    inv_item_sk               bigint              ,
    inv_warehouse_sk          bigint                  ,
    inv_quantity_on_hand      integer
)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5002/inventory* | gsfs://192.168.0.90:5003/inventory*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with inventory_err;

DROP FOREIGN TABLE IF EXISTS catalog_returns_ext;
CREATE FOREIGN TABLE catalog_returns_ext
(
    cr_returned_date_sk       bigint                  ,
    cr_returned_time_sk       bigint                  ,
    cr_item_sk                bigint                  ,
    cr_refunded_customer_sk   bigint                      ,
    cr_refunded_cdemo_sk      bigint                  ,
    cr_refunded_hdemo_sk      bigint                  ,
    cr_refunded_addr_sk       bigint                  ,
    cr_returning_customer_sk  bigint                      ,
    cr_returning_cdemo_sk     bigint                  ,
    cr_returning_hdemo_sk     bigint                  ,
    cr_returning_addr_sk      bigint                  ,
    cr_call_center_sk         bigint                  ,
    cr_catalog_page_sk        bigint                  ,
    cr_ship_mode_sk           bigint                  ,
    cr_warehouse_sk           bigint                  ,
    cr_reason_sk              bigint                  ,
```

```
    cr_order_number         bigint                  ,
    cr_return_quantity      bigint                  ,
    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 'gsfs://192.168.0.90:5002/catalog_returns* | gsfs://192.168.0.90:5003/catalog_returns*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with catalog_returns_err;

DROP FOREIGN TABLE IF EXISTS web_returns_ext;
CREATE FOREIGN TABLE web_returns_ext
(
    wr_returned_date_sk     bigint                  ,
    wr_returned_time_sk     bigint                  ,
    wr_item_sk              bigint                  ,
    wr_refunded_customer_sk bigint                  ,
    wr_refunded_cdemo_sk    bigint                  ,
    wr_refunded_hdemo_sk    bigint                  ,
    wr_refunded_addr_sk     bigint                  ,
    wr_returning_customer_sk bigint                 ,
    wr_returning_cdemo_sk   bigint                  ,
    wr_returning_hdemo_sk   bigint                  ,
    wr_returning_addr_sk    bigint                  ,
    wr_web_page_sk          bigint                  ,
    wr_reason_sk            bigint                  ,
    wr_order_number         bigint                  ,
    wr_return_quantity      bigint                  ,
    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 'gsfs://192.168.0.90:5002/web_returns* | gsfs://192.168.0.90:5003/web_returns*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with web_returns_err;

DROP FOREIGN TABLE IF EXISTS web_sales_ext;
CREATE FOREIGN TABLE web_sales_ext
(
    ws_sold_date_sk         bigint                  ,
    ws_sold_time_sk         bigint                  ,
    ws_ship_date_sk         bigint                  ,
    ws_item_sk              bigint                  ,
```

```
    ws_bill_customer_sk       bigint                 ,
    ws_bill_cdemo_sk          bigint                ,
    ws_bill_hdemo_sk          bigint                ,
    ws_bill_addr_sk           bigint               ,
    ws_ship_customer_sk       bigint                 ,
    ws_ship_cdemo_sk          bigint                ,
    ws_ship_hdemo_sk          bigint                 ,
    ws_ship_addr_sk           bigint               ,
    ws_web_page_sk            bigint                ,
    ws_web_site_sk            bigint               ,
    ws_ship_mode_sk           bigint                ,
    ws_warehouse_sk           bigint                ,
    ws_promo_sk               bigint               ,
    ws_order_number           bigint                ,
    ws_quantity               bigint              ,
    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 'gsfs://192.168.0.90:5002/web_sales* | gsfs://192.168.0.90:5003/web_sales*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with web_sales_err;

DROP FOREIGN TABLE IF EXISTS catalog_sales_ext;
CREATE FOREIGN TABLE catalog_sales_ext
(
    cs_sold_date_sk           bigint                 ,
    cs_sold_time_sk           bigint                 ,
    cs_ship_date_sk           bigint                 ,
    cs_bill_customer_sk       bigint                  ,
    cs_bill_cdemo_sk          bigint                 ,
    cs_bill_hdemo_sk          bigint                  ,
    cs_bill_addr_sk           bigint                ,
    cs_ship_customer_sk       bigint                   ,
    cs_ship_cdemo_sk          bigint                  ,
    cs_ship_hdemo_sk          bigint                  ,
    cs_ship_addr_sk           bigint                 ,
    cs_call_center_sk         bigint                ,
    cs_catalog_page_sk        bigint                 ,
    cs_ship_mode_sk           bigint                 ,
    cs_warehouse_sk           bigint                  ,
    cs_item_sk                bigint               ,
    cs_promo_sk               bigint                ,
    cs_order_number           bigint                 ,
    cs_quantity               bigint               ,
    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 'gsfs://192.168.0.90:5002/catalog_sales* | gsfs://192.168.0.90:5003/catalog_sales*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with  catalog_sales_err;

DROP FOREIGN TABLE IF EXISTS store_sales_ext;
CREATE FOREIGN TABLE store_sales_ext
(
    ss_sold_date_sk        bigint              ,
    ss_sold_time_sk        bigint              ,
    ss_item_sk          bigint             ,
    ss_customer_sk        bigint              ,
    ss_cdemo_sk          bigint             ,
    ss_hdemo_sk          bigint             ,
    ss_addr_sk          bigint             ,
    ss_store_sk          bigint             ,
    ss_promo_sk          bigint             ,
    ss_ticket_number       bigint              ,
    ss_quantity          bigint             ,
    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 'gsfs://192.168.0.90:5002/store_sales* | gsfs://192.168.0.90:5003/store_sales*',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
FILL_MISSING_FIELDS 'true',
IGNORE_EXTRA_DATA 'true',
mode 'Normal'
)
with store_sales_err;
```

步骤3 执行以下SQL语句导入数据。

```
INSERT INTO customer_address SELECT * FROM customer_address_ext;
INSERT INTO customer_demographics SELECT * FROM customer_demographics_ext;
INSERT INTO date_dim SELECT * FROM date_dim_ext;
INSERT INTO warehouse SELECT * FROM warehouse_ext;
INSERT INTO ship_mode SELECT * FROM ship_mode_ext;
INSERT INTO time_dim SELECT * FROM time_dim_ext;
INSERT INTO reason SELECT * FROM reason_ext;
INSERT INTO income_band SELECT * FROM income_band_ext;
```

```
INSERT INTO item SELECT * FROM item_ext;
INSERT INTO store SELECT * FROM store_ext;
INSERT INTO call_center SELECT * FROM call_center_ext;
INSERT INTO customer SELECT * FROM customer_ext;
INSERT INTO web_site SELECT * FROM web_site_ext;
INSERT INTO household_demographics SELECT * FROM household_demographics_ext;
INSERT INTO web_page SELECT * FROM web_page_ext;
INSERT INTO promotion SELECT * FROM promotion_ext;
INSERT INTO catalog_page SELECT * FROM catalog_page_ext;
INSERT INTO inventory SELECT * FROM inventory_ext;
INSERT INTO catalog_returns SELECT * FROM catalog_returns_ext;
INSERT INTO web_returns SELECT * FROM web_returns_ext;
INSERT INTO store_returns SELECT * FROM store_returns_ext;
INSERT INTO web_sales SELECT * FROM web_sales_ext;
INSERT INTO catalog_sales SELECT * FROM catalog_sales_ext;
INSERT INTO store_sales SELECT * FROM store_sales_ext;
```

----结束

3.5 执行查询与结果收集

3.5.1 通过编写 shell 脚本自动化执行查询和结果收集

步骤1 登录ECS，进入到/opt目录下，使用vim命令生成query.conf和run_query.sh两个脚本文件。脚本内容如下，编辑后按:wq!保存脚本配置。

run_query.sh脚本如下：

```
#!/bin/bash
script_path=$(cd `dirname $0`;pwd)
query_mode=$1
query_path=$2
query_object=$3
query_log=${script_path}/query_log_`date +%y%m%d_%H%M%S`
source ${script_path}/query.conf

function usage()
{
   echo "[NOTICE]: This script is used to run queries and collect cost time, according to sepcified path and
query file name."
   echo "        You can run the script as below:"
   echo -e ""
   echo "        1. config the query.conf file."
   echo "        2. run the script in batch mode. "
   echo "           eg. sh run_query.sh batch [query file's absolute path]"
   echo -e ""
   echo "        3. run the script in single mode."
   echo "           eg. sh run_query.sh single [query file's absolute path] [specified query file name]"
}

function log_file_init()
{
   mkdir -p ${query_log}/explain_log
   mkdir -p ${query_log}/pre_warm_log
   mkdir -p ${query_log}/query_test_log
   touch ${query_log}/query_result.csv
   echo "query name,cost time1,cost time2,cost time3,average cost" > ${query_log}/query_result.csv
}

function single_query()
{
   echo "[INFO]: Single mode query is to start."
   echo "*****${query_object} begin*****"
   collect_plan
   pre_warm
```

```
    query_test
    echo "*****${query_object} end*****"
    echo "[INFO]: Single mode query is finished."
    echo "[NOTICE]: Get more details by query_result.csv and other logs in ${query_log}."
}

function batch_query()
{
    echo "[INFO]: Batch mode query is to start."
    for query_object in `ls ${query_path}`
    do
        echo "*****${query_object} begin*****"
        collect_plan
        pre_warm
        query_test
        echo "*****${query_object} end*****"
    done
    echo "[INFO]: Batch mode query is finished."
    echo "[NOTICE]: Get more details by query_result.csv and other logs in ${query_log}."
}

function collect_plan()
{
    echo "[STEP1]: Starting to collect plan."
    echo "explain performance" > ${query_log}/explain_log/${query_object}.tmp
    cat ${query_path}/${query_object} >> ${query_log}/explain_log/${query_object}.tmp
    gsql -h ${cluster_ip} -d ${db_name} -p ${db_port} -U ${db_user} -W "${user_pwd}" -f ${query_log}/
explain_log/${query_object}.tmp > ${query_log}/explain_log/${query_object}.explain 2>&1
    echo "[STEP1]: Finished."
}

function pre_warm()
{
    echo "[STEP2]: Starting to pre-warm."
    for i in {1..2}
    do
        gsql -h ${cluster_ip} -d ${db_name} -p ${db_port} -U ${db_user} -W "${user_pwd}" -f ${query_path}/$
{query_object} > ${query_log}/pre_warm_log/${query_object}.pre${i} 2>&1
    done
    echo "[STEP2]: Finished."
}

function query_test()
{
    time1=""
    time2=""
    time3=""
    echo "[STEP3]: Starting to do real test."
    for i in {1..3}
    do
        gsql -h ${cluster_ip} -d ${db_name} -p ${db_port} -U ${db_user} -W "${user_pwd}" -f ${query_path}/$
{query_object} > ${query_log}/query_test_log/${query_object}.real${i} 2>&1
        let "`eval echo "time"${i}`=`cat ${query_log}/query_test_log/${query_object}.real${i}|grep "total time:"|
awk {'print$3'}`"
    done
    time_ave=`echo "scale=2;(${time1}+${time2}+${time3})/3"|bc -l`
    echo "${query_object},${time1},${time2},${time3},${time_ave}" >> ${query_log}/query_result.csv
    echo "[step3]: Finished. The average time:${time_ave} ms."
}
case ${query_mode} in
    single)log_file_init;single_query;;
    batch)log_file_init;batch_query;;
    *)usage;;
esac
```

query.conf为集群信息配置文件，包含如下五个变量：

```
cluster_ip=127.0.0.1        集群主cn节点内网ip
db_name=tpcds_test          数据库名称
db_port=6000                数据库端口号
```

| db_user=tpcds_user | 数据库用户 |
| user_pwd=123456 | 数据库用户密码 |

步骤2 编辑query.conf为集群对应的信息后，先执行**source gsql_env**命令配置变量后，执行
sh run_query.sh即可开始查询执行和结果收集。

示例：

sh run_query.sh *batch query1000x/*

参数1：批量执行选择batch，单个query执行选择single。

参数2：tpcds1000x或者tpch1000x query存放的绝对路径。

参数3：如果参数1选择batch，此参数忽略；如果参数1选择single，此参数为具体执
行的query名称，例如Q1。

须知

- gsql客户端的使用需要每次连接后，都要执行**source gsql_env**导入变量命令，执行
  查询脚本前请确认gsql可执行。
- query1000x文件夹中必须包含tpch 22个标准查询文件Q1~Q22或者tpcds 99个标准
  查询文件Q1~Q99。
- run_query.sh脚本依赖bc命令，执行前确认bc命令存在。
- 每个查询默认会跑6次，第1次收集执行计划，第2，3次预热，第4到第6次正式查
  询，最终结果取后3次查询的平均值。
- 查询脚本执行后会立即生成query_log_yymmdd_hhmmss名称的目录，其中：
  - explain_log子目录存放查询计划。
  - pre_warm子目录存放预热执行结果。
  - query_test子目录存放正式查询执行结果。
  - query_result.csv文件，csv格式汇总所有查询的执行结果。

----结束

4 附录

4.1 TPC-H 测试集

您可以通过**命令生成方法**生成TPC-H测试集，也可以直接通过**脚本生成方法**生成，另我们已经给出完整的TPC-H**测试集**供您参考。

> 📖 **说明**
>
> 由于版本差异，通过脚本生成的SQL测试集，可能会存在部分SQL执行不成功的情况，请参考**测试集**进行修正后执行。

命令生成方法

TPC-H 22个标准查询SQL可以用如下方法生成。

步骤1 登录测试过程申请的ECS，执行如下命令：

```
cd /data1/script/tpch-kit/TPC-H_Tools_v3.0.0/dbgen
cp qgen queries
cp dists.dss queries
cd queries
for i in {1..22}
do
   ./qgen -s 1000 -d $i > Q$i
done
```

步骤2 生成的query无法直接使用，需要修改，修改项如下：

- 语句中含有LILIMIT -1的需要删除LIMIT -1。
- Q15：将create view语句改为with语句，去除drop view语句。

----结束

脚本生成方法

推荐使用如下脚本直接生成GaussDB(DWS)可用的TPC-H查询：

```
# -*- coding: utf-8 -*-
import os
import sys
import re
```

```python
def genseq(qgenfile, scale, query_dir):
    flags = os.O_WRONLY | os.O_CREAT | os.O_EXCL
    modes = stat.S_IWUSR | stat.S_IRUSR
    if not os.path.exists(query_dir):
        os.mkdir(query_dir)

    for i in range(1, 23):
        cmd = qgenfile + ' -s ' + str(scale) + ' -d ' + str(i) + ' > ' + query_dir + '/Q' + str(i)
        os.system(cmd)

    os.chdir(query_dir)
    for i in range(1, 23):
        with open("Q" + str(i), 'r') as fr:
            qlist = list(fr)
            for j in range(len(qlist)):
                if i == 15:
                    if 'create view' in qlist[j]:
                        qlist[j] = qlist[j].replace('create view', 'with')
                    elif 'l_suppkey;' in qlist[j]:
                        qlist[j] = qlist[j].replace(';\n', ')')
                    elif 'drop view' in qlist[j] or 'LIMIT -1' in qlist[j]:
                        qlist[j] = ''
                    continue
                if 'LIMIT -1' in qlist[j]:
                    qlist[j] = ''
                elif 'LIMIT -1' not in qlist[j] and 'LIMIT' in qlist[j]:
                    qlist[j - 1] = qlist[j - 1].replace(';', '')
                    qlist[j] = qlist[j].replace('LIMIT', 'limit').replace('\n', ';\n')
        with os.fdopen(os.open("Q" + str(i), flags, modes), 'w') as fw:
            for z in range(len(qlist)):
                fw.write(qlist[z])
    print("TPCH Q1~Q22 query store at " + query_dir)


if __name__ == '__main__':
    if len(sys.argv) != 4:
        print('Wrong number of parameters！ ')
        print('Usage：python3 gen_tpch_thpseq.py qgen_file_path scale query_dir')
        print("""Parameter:
            qgen_file_path: tpch qgen文件路径
            scale: 生成查询对应的数据规模，单位GB
            query_dir: 生成文件的存放路径""")
        print("Notice: Make sure the qgen and dists.dss files are in the queries directory of the tpch tool ")
        print("""Example:
            python3 gen_tpch_thpseq.py ./qgen 1000 tpch_query1000x""")
        sys.exit(1)

    qgen_file_path = sys.argv[1]
    scale = sys.argv[2]
    query_dir = sys.argv[3]
    try:
        if not re.match(r'^\.?\/(\w+\/?)+$', qgen_file_path):
            print("error param qgenfilepath:", qgen_file_path)
        if not re.match(r'\d+', scale):
            print('error param scale:', scale)
        if not re.match(r'^\/?(\w+\/?)+$', query_dir):
            print('error param query_dir:', query_dir)
    except Exception as ex:
        print('exception: invalid param!')


    if not os.path.isfile(qgen_file_path):
        print('The file %s is not exist!' % qgen_file_path)
        sys.exit(1)

    genseq(qgen_file_path, int(scale), query_dir)
```

将脚本保存为gen_tpch_thpseq.py，并放在queries目录下，执行如下命令，其中1000为数据规模，代表1000x，tpch1000x_query为22个sql保存的位置：

python3 gen_tpch_thpseq.py ./qgen 1000 tpch1000x_query

测试集

性能测试集中将执行以下22个查询SQL。

SQL1

```
select
    l_returnflag,
    l_linestatus,
    sum(l_quantity) as sum_qty,
    sum(l_extendedprice) as sum_base_price,
    sum(l_extendedprice * (1 - l_discount)) as sum_disc_price,
    sum(l_extendedprice * (1 - l_discount) * (1 + l_tax)) as sum_charge,
    avg(l_quantity) as avg_qty,
    avg(l_extendedprice) as avg_price,
    avg(l_discount) as avg_disc,
    count(*) as count_order
from
    lineitem
where
    l_shipdate <= date '1998-12-01' - interval '90' day (3)
group by
    l_returnflag,
    l_linestatus
order by
    l_returnflag,
    l_linestatus;
```

SQL2

```
select
    s_acctbal,
    s_name,
    n_name,
    p_partkey,
    p_mfgr,
    s_address,
    s_phone,
    s_comment
from
    part,
    supplier,
    partsupp,
    nation,
    region
where
    p_partkey = ps_partkey
    and s_suppkey = ps_suppkey
    and p_size = 15
    and p_type like '%BRASS'
    and s_nationkey = n_nationkey
    and n_regionkey = r_regionkey
    and r_name = 'EUROPE'
    and ps_supplycost = (
        select
            min(ps_supplycost)
        from
            partsupp,
            supplier,
            nation,
            region
        where
            p_partkey = ps_partkey
            and s_suppkey = ps_suppkey
            and s_nationkey = n_nationkey
```

```
        and n_regionkey = r_regionkey
        and r_name = 'EUROPE'
    )
order by
    s_acctbal desc,
    n_name,
    s_name,
    p_partkey
    limit 100;
```

SQL3

```
select
    l_orderkey,
    sum(l_extendedprice * (1 - l_discount)) as revenue,
    o_orderdate,
    o_shippriority
from
    customer,
    orders,
    lineitem
where
    c_mktsegment = 'BUILDING'
    and c_custkey = o_custkey
    and l_orderkey = o_orderkey
    and o_orderdate < date '1995-03-15'
    and l_shipdate > date '1995-03-15'
group by
    l_orderkey,
    o_orderdate,
    o_shippriority
order by
    revenue desc,
    o_orderdate
    limit 10;
```

SQL4

```
select
    o_orderpriority,
    count(*) as order_count
from
    orders
where
    o_orderdate >= date '1993-07-01'
    and o_orderdate < date '1993-07-01' + interval '3' month
    and exists (
        select
            *
        from
            lineitem
        where
            l_orderkey = o_orderkey
            and l_commitdate < l_receiptdate
    )
group by
    o_orderpriority
order by
    o_orderpriority;
```

SQL5

```
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 >= date '1994-01-01'
    and o_orderdate < date '1994-01-01' + interval '1' year
group by
    n_name
order by
    revenue desc;
```

SQL6

```
select
    sum(l_extendedprice * l_discount) as revenue
from
    lineitem
where
    l_shipdate >= date '1994-01-01'
    and l_shipdate < date '1994-01-01' + interval '1' year
    and l_discount between .06 - 0.01 and .06 + 0.01
    and l_quantity < 24;
```

SQL7

```
select
    supp_nation,
    cust_nation,
    l_year,
    sum(volume) as revenue
from
    (
        select
            n1.n_name as supp_nation,
            n2.n_name as cust_nation,
            extract(year from l_shipdate) as l_year,
            l_extendedprice * (1 - l_discount) as volume
        from
            supplier,
            lineitem,
            orders,
            customer,
            nation n1,
            nation n2
        where
            s_suppkey = l_suppkey
            and o_orderkey = l_orderkey
            and c_custkey = o_custkey
            and s_nationkey = n1.n_nationkey
            and c_nationkey = n2.n_nationkey
            and (
                (n1.n_name = 'FRANCE' and n2.n_name = 'GERMANY')
                or (n1.n_name = 'GERMANY' and n2.n_name = 'FRANCE')
            )
            and l_shipdate between date '1995-01-01' and date '1996-12-31'
    ) as shipping
group by
    supp_nation,
    cust_nation,
    l_year
```

```
order by
    supp_nation,
    cust_nation,
    l_year;
```

SQL8

```
select
    o_year,
    sum(case
        when nation = 'BRAZIL' then volume
        else 0
    end) / sum(volume) as mkt_share
from
    (
        select
            extract(year from o_orderdate) as o_year,
            l_extendedprice * (1 - l_discount) as volume,
            n2.n_name as nation
        from
            part,
            supplier,
            lineitem,
            orders,
            customer,
            nation n1,
            nation n2,
            region
        where
            p_partkey = l_partkey
            and s_suppkey = l_suppkey
            and l_orderkey = o_orderkey
            and o_custkey = c_custkey
            and c_nationkey = n1.n_nationkey
            and n1.n_regionkey = r_regionkey
            and r_name = 'AMERICA'
            and s_nationkey = n2.n_nationkey
            and o_orderdate between date '1995-01-01' and date '1996-12-31'
            and p_type = 'ECONOMY ANODIZED STEEL'
    ) as all_nations
group by
    o_year
order by
    o_year;
```

SQL9

```
select
    nation,
    o_year,
    sum(amount) as sum_profit
from
    (
        select
            n_name as nation,
            extract(year from o_orderdate) as o_year,
            l_extendedprice * (1 - l_discount) - ps_supplycost * l_quantity as amount
        from
            part,
            supplier,
            lineitem,
            partsupp,
            orders,
            nation
        where
            s_suppkey = l_suppkey
            and ps_suppkey = l_suppkey
            and ps_partkey = l_partkey
            and p_partkey = l_partkey
```

```
            and o_orderkey = l_orderkey
            and s_nationkey = n_nationkey
            and p_name like '%green%'
        ) as profit
group by
    nation,
    o_year
order by
    nation,
    o_year desc;
```

SQL10

```
select
    c_custkey,
    c_name,
    sum(l_extendedprice * (1 - l_discount)) as revenue,
    c_acctbal,
    n_name,
    c_address,
    c_phone,
    c_comment
from
    customer,
    orders,
    lineitem,
    nation
where
    c_custkey = o_custkey
    and l_orderkey = o_orderkey
    and o_orderdate >= date '1993-10-01'
    and o_orderdate < date '1993-10-01' + interval '3' month
    and l_returnflag = 'R'
    and c_nationkey = n_nationkey
group by
    c_custkey,
    c_name,
    c_acctbal,
    c_phone,
    n_name,
    c_address,
    c_comment
order by
    revenue desc
    limit 20;
```

SQL11

```
select
    ps_partkey,
    sum(ps_supplycost * ps_availqty) as value
from
    partsupp,
    supplier,
    nation
where
    ps_suppkey = s_suppkey
    and s_nationkey = n_nationkey
    and n_name = 'GERMANY'
group by
    ps_partkey having
        sum(ps_supplycost * ps_availqty) > (
            select
                sum(ps_supplycost * ps_availqty) * 0.0000001000
            from
                partsupp,
                supplier,
                nation
            where
```

```
            ps_suppkey = s_suppkey
            and s_nationkey = n_nationkey
            and n_name = 'GERMANY'
    )
order by
    value desc;
```

SQL12

```
select
    l_shipmode,
    sum(case
        when o_orderpriority = '1-URGENT'
            or o_orderpriority = '2-HIGH'
            then 1
        else 0
    end) as high_line_count,
    sum(case
        when o_orderpriority <> '1-URGENT'
            and o_orderpriority <> '2-HIGH'
            then 1
        else 0
    end) as low_line_count
from
    orders,
    lineitem
where
    o_orderkey = l_orderkey
    and l_shipmode in ('MAIL', 'SHIP')
    and l_commitdate < l_receiptdate
    and l_shipdate < l_commitdate
    and l_receiptdate >= date '1994-01-01'
    and l_receiptdate < date '1994-01-01' + interval '1' year
group by
    l_shipmode
order by
    l_shipmode;
```

SQL13

```
select
    c_count,
    count(*) as custdist
from
    (
        select
            c_custkey,
            count(o_orderkey)
        from
            customer left outer join orders on
                c_custkey = o_custkey
                and o_comment not like '%special%requests%'
        group by
            c_custkey
    ) as c_orders (c_custkey, c_count)
group by
    c_count
order by
    custdist desc,
    c_count desc;
```

SQL14

```
select
    100.00 * sum(case
        when p_type like 'PROMO%'
            then l_extendedprice * (1 - l_discount)
        else 0
```

```
end) / sum(l_extendedprice * (1 - l_discount)) as promo_revenue
from
    lineitem,
    part
where
    l_partkey = p_partkey
    and l_shipdate >= date '1995-09-01'
    and l_shipdate < date '1995-09-01' + interval '1' month;
```

SQL15

```
with revenue (supplier_no, total_revenue) as --change view to cte
(
    select
        l_suppkey,
        sum(l_extendedprice * (1 - l_discount))
    from
        lineitem
    where
        l_shipdate >= date '1996-01-01'
        and l_shipdate < date '1996-01-01' + interval '3 month'
    group by
        l_suppkey
)
select
    s_suppkey,
    s_name,
    s_address,
    s_phone,
    total_revenue
from
    supplier,
    revenue
where
    s_suppkey = supplier_no
    and total_revenue = (
        select
            max(total_revenue)
        from
            revenue
    )
order by
    s_suppkey;
```

SQL16

```
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;
```

SQL17

```
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
   );
```

SQL18

```
select
   c_name,
   c_custkey,
   o_orderkey,
   o_orderdate,
   o_totalprice,
   sum(l_quantity)
from
   customer,
   orders,
   lineitem
where
   o_orderkey in (
      select
         l_orderkey
      from
         lineitem
      group by
         l_orderkey having
            sum(l_quantity) > 300
   )
   and c_custkey = o_custkey
   and o_orderkey = l_orderkey
group by
   c_name,
   c_custkey,
   o_orderkey,
   o_orderdate,
   o_totalprice
order by
   o_totalprice desc,
   o_orderdate
   limit 100;
```

SQL19

```
select
   sum(l_extendedprice* (1 - l_discount)) as revenue
```

```
from
   lineitem,
   part
where
   (
      p_partkey = l_partkey
      and p_brand = 'Brand#12'
      and p_container in ('SM CASE', 'SM BOX', 'SM PACK', 'SM PKG')
      and l_quantity >= 1 and l_quantity <= 1 + 10
      and p_size between 1 and 5
      and l_shipmode in ('AIR', 'AIR REG')
      and l_shipinstruct = 'DELIVER IN PERSON'
   )
   or
   (
      p_partkey = l_partkey
      and p_brand = 'Brand#23'
      and p_container in ('MED BAG', 'MED BOX', 'MED PKG', 'MED PACK')
      and l_quantity >= 10 and l_quantity <= 10 + 10
      and p_size between 1 and 10
      and l_shipmode in ('AIR', 'AIR REG')
      and l_shipinstruct = 'DELIVER IN PERSON'
   )
   or
   (
      p_partkey = l_partkey
      and p_brand = 'Brand#34'
      and p_container in ('LG CASE', 'LG BOX', 'LG PACK', 'LG PKG')
      and l_quantity >= 20 and l_quantity <= 20 + 10
      and p_size between 1 and 15
      and l_shipmode in ('AIR', 'AIR REG')
      and l_shipinstruct = 'DELIVER IN PERSON'
   );
```

SQL20

```
select
   s_name,
   s_address
from
   supplier,
   nation
where
   s_suppkey in (
      select
         ps_suppkey
      from
         partsupp
      where
         ps_partkey in (
            select
               p_partkey
            from
               part
            where
               p_name like 'forest%'
         )
         and ps_availqty > (
            select
               0.5 * sum(l_quantity)
            from
               lineitem
            where
               l_partkey = ps_partkey
               and l_suppkey = ps_suppkey
               and l_shipdate >= date '1994-01-01'
               and l_shipdate < date '1994-01-01' + interval '1' year
         )
   )
```

```
        and s_nationkey = n_nationkey
        and n_name = 'CANADA'
order by
    s_name;
```

SQL21

```
select
    s_name,
    count(*) as numwait
from
    supplier,
    lineitem l1,
    orders,
    nation
where
    s_suppkey = l1.l_suppkey
    and o_orderkey = l1.l_orderkey
    and o_orderstatus = 'F'
    and l1.l_receiptdate > l1.l_commitdate
    and exists (
        select
            *
        from
            lineitem l2
        where
            l2.l_orderkey = l1.l_orderkey
            and l2.l_suppkey <> l1.l_suppkey
    )
    and not exists (
        select
            *
        from
            lineitem l3
        where
            l3.l_orderkey = l1.l_orderkey
            and l3.l_suppkey <> l1.l_suppkey
            and l3.l_receiptdate > l3.l_commitdate
    )
    and s_nationkey = n_nationkey
    and n_name = 'SAUDI ARABIA'
group by
    s_name
order by
    numwait desc,
    s_name
    limit 100;
```

SQL22

```
select
    cntrycode,
    count(*) as numcust,
    sum(c_acctbal) as totacctbal
from
    (
        select
            substring(c_phone from 1 for 2) as cntrycode,
            c_acctbal
        from
            customer
        where
            substring(c_phone from 1 for 2) in
                ('13', '31', '23', '29', '30', '18', '17')
            and c_acctbal > (
                select
                    avg(c_acctbal)
                from
                    customer
```

```
            where
                c_acctbal > 0.00
                and substring(c_phone from 1 for 2) in
                    ('13', '31', '23', '29', '30', '18', '17')
        )
        and not exists (
            select
                *
            from
                orders
            where
                o_custkey = c_custkey
        )
    ) as custsale
group by
    cntrycode
order by
    cntrycode;
```

4.2 TPC-DS 测试集

您可以通过**命令生成方法**生成TPC-DS测试集，也可以直接通过**脚本生成方法**生成，另我们已经给出前面20个的TPC-DS**测试集**供您参考。

命令生成方法

TPC-DS标准99个SQL查询语句可用如下方法生成：

步骤1 准备工作。生成TPC-DS查询语句前需要修改query_templates目录下的文件：

1. 登录测试过程申请的ECS，进入/data1/script/tpcds-kit/DSGen-software-code-3.2.0rc1/query_templates目录：
   cd /data1/script/tpcds-kit/*DSGen-software-code-3.2.0rc1/***query_templates**

2. 新建文件hwdws.tpl，内容为：
   ```
   define __LIMITA = "";
   define __LIMITB = "";
   define __LIMITC = "limit %d";
   define _BEGIN = "-- begin query " + [_QUERY] + " in stream " + [_STREAM] + " using template " + [_TEMPLATE];
   define _END = "-- end query " + [_QUERY] + " in stream " + [_STREAM] + " using template " + [_TEMPLATE];
   ```

3. 因TPC-DS工具中SQL语句生成模板有语法错误，需修改query77.tpl，将135行的'，coalesce(returns, 0) returns'改为'**, coalesce(returns, 0) as returns**'。

步骤2 执行以下命令生成查询语句：

cd /data1/script/tpcds-kit/DSGen-software-code-3.2.0rc1/tools
./dsqgen -input ../query_templates/templates.lst -directory ../query_templates/ -scale 1000 -dialect hwdws

执行后会生成query_0.sql文件，里面放着99个标准SQL语句，需要手动去切分成99个文件。

步骤3 生成的标准查询中如下日期函数语法在GaussDB(DWS)暂不支持，需要手动进行修改：

```
Q5:  and (cast('2001-08-19' as date) +  14 days)
修改为
and (cast('2001-08-19' as date) +  14)

Q12:and (cast('1999-02-28' as date) + 30 days)
修改为
and (cast('1999-02-28' as date) + 30)
```

Q16:(cast('1999-4-01' as date) + 60 days)
修改为
(cast('1999-4-01' as date) + 60)

Q20:and (cast('1998-05-05' as date) + 30 days)
修改为
and (cast('1998-05-05' as date) + 30)

Q21:and d_date between (cast ('2000-05-19' as date) - 30 days)
修改为
and d_date between (cast ('2000-05-19' as date) - 30)

and (cast ('2000-05-19' as date) + 30 days)
修改为
and (cast ('2000-05-19' as date) + 30)

Q32:(cast('1999-02-22' as date) + 90 days)
修改为
(cast('1999-02-22' as date) + 90)

Q37:and d_date between cast('1998-04-29' as date) and (cast('1998-04-29' as date) + 60 days)
修改为
and d_date between cast('1998-04-29' as date) and (cast('1998-04-29' as date) + 60)

Q40:and d_date between (cast ('2002-05-10' as date) - 30 days)
修改为
and d_date between (cast ('2002-05-10' as date) - 30)

and (cast ('2002-05-10' as date) + 30 days)
修改为
and (cast ('2002-05-10' as date) + 30)

Q77:and (cast('1999-08-29' as date) + 30 days)
修改为
and (cast('1999-08-29' as date) + 30)

Q80:and (cast('2002-08-04' as date) + 30 days)
修改为
and (cast('2002-08-04' as date) + 30)

Q82:and d_date between cast('1998-01-18' as date) and (cast('1998-01-18' as date) + 60 days)
修改为
and d_date between cast('1998-01-18' as date) and (cast('1998-01-18' as date) + 60)

Q92:(cast('2001-01-26' as date) + 90 days)
修改为
(cast('2001-01-26' as date) + 90)

Q94:(cast('1999-5-01' as date) + 60 days)
修改为
(cast('1999-5-01' as date) + 60)

Q95:(cast('1999-4-01' as date) + 60 days)
修改为
(cast('1999-4-01' as date) + 60)

Q98:and (cast('2002-04-01' as date) + 30 days)
修改为
and (cast('2002-04-01' as date) + 30)

----结束

脚本生成方法

建议使用如下脚本直接生成GaussDB(DWS)可用的SQL语句：

```
# -*- coding: utf-8 -*-
import os
import sys
```

```python
import re
import stat


def gen_thp_seq(dsqgen_file, scale, query_dir):
    flags = os.O_WRONLY | os.O_CREAT | os.O_EXCL
    modes = stat.S_IWUSR | stat.S_IRUSR
    if not os.path.exists(query_dir):
        os.mkdir(query_dir)

    cmd = dsqgen_file + ' -input ../query_templates/templates.lst -directory ../query_templates/ -scale ' + str(scale) + ' -dialect hwdws'
    os.system(cmd)
    with open('query_0.sql', 'r') as f1:
        line = f1.readline()
        queryname = ''
        while line:
            if '-- begin' in line.strip():
                #line:'-- begin query 1 in stream 0 using template query96.tpl\n'
                queryname = line.split(' ')[-1][5:-5]
                fquery = os.fdopen(os.open(query_dir + '/Q' + queryname, flags, modes), 'w+')
                line = f1.readline()
                continue

            if not queryname or line == '\n':
                line = f1.readline()
                continue

            if '-- end' in line.strip():
                fquery.close()
                line = f1.readline()
                continue

            if 'days)' in line:
                line = line.replace('days', '')
            fquery.write(line)
            line = f1.readline()

    print("TPCDS Q1~Q99 query store at " + query_dir)
    os.system('rm -rf query_0.sql')


if __name__ == '__main__':
    if len(sys.argv) != 4:
        print('Wrong number of parameters！ ')
        print('Usage：python3 gen_tpcds_thpseq.py dsqgen_file_path scale query_dir')
        print("""Parameter:
            qgen_file_path: tpcds dsqgen文件路径
            scale: 生成查询对应的数据规模
            query_dir: 生成文件的存放路径""")
        print("""Example:
            python3 gen_tpcds_thpseq.py ./dsqgen 1000 tpcds_query1000x""")
        sys.exit(1)

    dsqgen_file_path = sys.argv[1]
    scale = sys.argv[2]
    query_dir = sys.argv[3]
    try:
        if not re.match(r'^\.?\/(\w+\/?)+$', dsqgen_file_path):
            print("error param qgenfilepath:", dsqgen_file_path)
        if not re.match(r'\d+', scale):
            print('error param scale:', scale)
        if not re.match(r'^\/?(\w+\/?)+$', query_dir):
            print('error param query_dir:', query_dir)
    except Exception as ex:
        print('exception: invalid param!')

    if not os.path.isfile(dsqgen_file_path):
```

```
    print('The file %s is not exist!' % dsqgen_file_path)
    sys.exit(1)

  gen_thp_seq(dsqgen_file_path, int(scale), query_dir)
```

将以上脚本保存在**/data1/script/tpcds-kit/DSGen-software-code-3.2.0rc1/tools/**gen_tpcds_thpseq.py，执行如下命令可获得99个SQL语句，其中1000为数据规模，代表TPCDS 1000x，tpcds_query1000x为生成的SQL语句存放的位置：

cd **/data1/script/tpcds-kit/DSGen-software-code-3.2.0rc1/tools/**
python3 gen_tpcds_thpseq.py ./dsqgen 1000 tpcds_query1000x

测试集

TPC-DS测试集共包括99个SQL查询，本章节仅体现前20个SQL。其他请根据以上方法生成。

SQL1

```
with customer_total_return as
(select sr_customer_sk as ctr_customer_sk
,sr_store_sk as ctr_store_sk
,sum(SR_RETURN_AMT_INC_TAX) as ctr_total_return
from store_returns
,date_dim
where sr_returned_date_sk = d_date_sk
and d_year =2001
group by sr_customer_sk
,sr_store_sk)
 select  c_customer_id
from customer_total_return ctr1
,store
,customer
where ctr1.ctr_total_return > (select avg(ctr_total_return)*1.2
from customer_total_return ctr2
where ctr1.ctr_store_sk = ctr2.ctr_store_sk)
and s_store_sk = ctr1.ctr_store_sk
and s_state = 'PA'
and ctr1.ctr_customer_sk = c_customer_sk
order by c_customer_id
limit 100;
```

SQL2

```
with wscs as
 (select sold_date_sk
     ,sales_price
 from (select ws_sold_date_sk sold_date_sk
         ,ws_ext_sales_price sales_price
     from web_sales
     union all
     select cs_sold_date_sk sold_date_sk
         ,cs_ext_sales_price sales_price
     from catalog_sales)),
 wswscs as
 (select d_week_seq,
     sum(case when (d_day_name='Sunday') then sales_price else null end) sun_sales,
     sum(case when (d_day_name='Monday') then sales_price else null end) mon_sales,
     sum(case when (d_day_name='Tuesday') then sales_price else  null end) tue_sales,
     sum(case when (d_day_name='Wednesday') then sales_price else null end) wed_sales,
     sum(case when (d_day_name='Thursday') then sales_price else null end) thu_sales,
     sum(case when (d_day_name='Friday') then sales_price else null end) fri_sales,
     sum(case when (d_day_name='Saturday') then sales_price else null end) sat_sales
 from wscs
    ,date_dim
 where d_date_sk = sold_date_sk
 group by d_week_seq)
```

```
select d_week_seq1
    ,round(sun_sales1/sun_sales2,2)
    ,round(mon_sales1/mon_sales2,2)
    ,round(tue_sales1/tue_sales2,2)
    ,round(wed_sales1/wed_sales2,2)
    ,round(thu_sales1/thu_sales2,2)
    ,round(fri_sales1/fri_sales2,2)
    ,round(sat_sales1/sat_sales2,2)
from
(select wswscs.d_week_seq d_week_seq1
    ,sun_sales sun_sales1
    ,mon_sales mon_sales1
    ,tue_sales tue_sales1
    ,wed_sales wed_sales1
    ,thu_sales thu_sales1
    ,fri_sales fri_sales1
    ,sat_sales sat_sales1
 from wswscs,date_dim
 where date_dim.d_week_seq = wswscs.d_week_seq and
     d_year = 1999) y,
(select wswscs.d_week_seq d_week_seq2
    ,sun_sales sun_sales2
    ,mon_sales mon_sales2
    ,tue_sales tue_sales2
    ,wed_sales wed_sales2
    ,thu_sales thu_sales2
    ,fri_sales fri_sales2
    ,sat_sales sat_sales2
 from wswscs
    ,date_dim
 where date_dim.d_week_seq = wswscs.d_week_seq and
     d_year = 1999+1) z
where d_week_seq1=d_week_seq2-53
order by d_week_seq1;
```

SQL3

```
select  dt.d_year
    ,item.i_brand_id brand_id
    ,item.i_brand brand
    ,sum(ss_ext_sales_price) sum_agg
 from  date_dim dt
    ,store_sales
    ,item
 where dt.d_date_sk = store_sales.ss_sold_date_sk
   and store_sales.ss_item_sk = item.i_item_sk
   and item.i_manufact_id = 125
   and dt.d_moy=11
 group by dt.d_year
    ,item.i_brand
    ,item.i_brand_id
 order by dt.d_year
     ,sum_agg desc
     ,brand_id
 limit 100;
```

SQL4

```
with year_total as (
 select c_customer_id customer_id
    ,c_first_name customer_first_name
    ,c_last_name customer_last_name
    ,c_preferred_cust_flag customer_preferred_cust_flag
    ,c_birth_country customer_birth_country
    ,c_login customer_login
    ,c_email_address customer_email_address
    ,d_year dyear
    ,sum(((ss_ext_list_price-ss_ext_wholesale_cost-ss_ext_discount_amt)+ss_ext_sales_price)/2) year_total
    ,'s' sale_type
```

```
      from customer
         ,store_sales
         ,date_dim
      where c_customer_sk = ss_customer_sk
        and ss_sold_date_sk = d_date_sk
      group by c_customer_id
            ,c_first_name
            ,c_last_name
            ,c_preferred_cust_flag
            ,c_birth_country
            ,c_login
            ,c_email_address
            ,d_year
      union all
      select c_customer_id customer_id
         ,c_first_name customer_first_name
         ,c_last_name customer_last_name
         ,c_preferred_cust_flag customer_preferred_cust_flag
         ,c_birth_country customer_birth_country
         ,c_login customer_login
         ,c_email_address customer_email_address
         ,d_year dyear
         ,sum((((cs_ext_list_price-cs_ext_wholesale_cost-cs_ext_discount_amt)+cs_ext_sales_price)/2) ) year_total
         ,'c' sale_type
      from customer
         ,catalog_sales
         ,date_dim
      where c_customer_sk = cs_bill_customer_sk
        and cs_sold_date_sk = d_date_sk
      group by c_customer_id
            ,c_first_name
            ,c_last_name
            ,c_preferred_cust_flag
            ,c_birth_country
            ,c_login
            ,c_email_address
            ,d_year
      union all
      select c_customer_id customer_id
         ,c_first_name customer_first_name
         ,c_last_name customer_last_name
         ,c_preferred_cust_flag customer_preferred_cust_flag
         ,c_birth_country customer_birth_country
         ,c_login customer_login
         ,c_email_address customer_email_address
         ,d_year dyear
         ,sum((((ws_ext_list_price-ws_ext_wholesale_cost-ws_ext_discount_amt)+ws_ext_sales_price)/2) )
year_total
         ,'w' sale_type
      from customer
         ,web_sales
         ,date_dim
      where c_customer_sk = ws_bill_customer_sk
        and ws_sold_date_sk = d_date_sk
      group by c_customer_id
            ,c_first_name
            ,c_last_name
            ,c_preferred_cust_flag
            ,c_birth_country
            ,c_login
            ,c_email_address
            ,d_year
            )
      select
               t_s_secyear.customer_id
            ,t_s_secyear.customer_first_name
            ,t_s_secyear.customer_last_name
            ,t_s_secyear.customer_preferred_cust_flag
      from year_total t_s_firstyear
```

```
      ,year_total t_s_secyear
      ,year_total t_c_firstyear
      ,year_total t_c_secyear
      ,year_total t_w_firstyear
      ,year_total t_w_secyear
 where t_s_secyear.customer_id = t_s_firstyear.customer_id
   and t_s_firstyear.customer_id = t_c_secyear.customer_id
   and t_s_firstyear.customer_id = t_c_firstyear.customer_id
   and t_s_firstyear.customer_id = t_w_firstyear.customer_id
   and t_s_firstyear.customer_id = t_w_secyear.customer_id
   and t_s_firstyear.sale_type = 's'
   and t_c_firstyear.sale_type = 'c'
   and t_w_firstyear.sale_type = 'w'
   and t_s_secyear.sale_type = 's'
   and t_c_secyear.sale_type = 'c'
   and t_w_secyear.sale_type = 'w'
   and t_s_firstyear.dyear =  2000
   and t_s_secyear.dyear = 2000+1
   and t_c_firstyear.dyear =  2000
   and t_c_secyear.dyear =  2000+1
   and t_w_firstyear.dyear = 2000
   and t_w_secyear.dyear = 2000+1
   and t_s_firstyear.year_total > 0
   and t_c_firstyear.year_total > 0
   and t_w_firstyear.year_total > 0
   and case when t_c_firstyear.year_total > 0 then t_c_secyear.year_total / t_c_firstyear.year_total else null
end
        > case when t_s_firstyear.year_total > 0 then t_s_secyear.year_total / t_s_firstyear.year_total else null
end
   and case when t_c_firstyear.year_total > 0 then t_c_secyear.year_total / t_c_firstyear.year_total else null
end
        > case when t_w_firstyear.year_total > 0 then t_w_secyear.year_total / t_w_firstyear.year_total else
null end
 order by t_s_secyear.customer_id
        ,t_s_secyear.customer_first_name
        ,t_s_secyear.customer_last_name
        ,t_s_secyear.customer_preferred_cust_flag
limit 100;
```

SQL5

```
with ssr as
 (select s_store_id,
      sum(sales_price) as sales,
      sum(profit) as profit,
      sum(return_amt) as returns,
      sum(net_loss) as profit_loss
 from
  ( select  ss_store_sk as store_sk,
        ss_sold_date_sk  as date_sk,
        ss_ext_sales_price as sales_price,
        ss_net_profit as profit,
        cast(0 as decimal(7,2)) as return_amt,
        cast(0 as decimal(7,2)) as net_loss
   from store_sales
   union all
   select sr_store_sk as store_sk,
        sr_returned_date_sk as date_sk,
        cast(0 as decimal(7,2)) as sales_price,
        cast(0 as decimal(7,2)) as profit,
        sr_return_amt as return_amt,
        sr_net_loss as net_loss
   from store_returns
  ) salesreturns,
   date_dim,
   store
 where date_sk = d_date_sk
     and d_date between cast('2002-08-05' as date)
            and (cast('2002-08-05' as date) +  14 )
```

```
                    and store_sk = s_store_sk
group by s_store_id)
,
csr as
(select cp_catalog_page_id,
      sum(sales_price) as sales,
      sum(profit) as profit,
      sum(return_amt) as returns,
      sum(net_loss) as profit_loss
from
 ( select  cs_catalog_page_sk as page_sk,
         cs_sold_date_sk  as date_sk,
         cs_ext_sales_price as sales_price,
         cs_net_profit as profit,
         cast(0 as decimal(7,2)) as return_amt,
         cast(0 as decimal(7,2)) as net_loss
   from catalog_sales
   union all
   select cr_catalog_page_sk as page_sk,
         cr_returned_date_sk as date_sk,
         cast(0 as decimal(7,2)) as sales_price,
         cast(0 as decimal(7,2)) as profit,
         cr_return_amount as return_amt,
         cr_net_loss as net_loss
   from catalog_returns
 ) salesreturns,
   date_dim,
   catalog_page
where date_sk = d_date_sk
     and d_date between cast('2002-08-05' as date)
            and (cast('2002-08-05' as date) + 14 )
     and page_sk = cp_catalog_page_sk
group by cp_catalog_page_id)
,
wsr as
(select web_site_id,
      sum(sales_price) as sales,
      sum(profit) as profit,
      sum(return_amt) as returns,
      sum(net_loss) as profit_loss
from
 ( select  ws_web_site_sk as wsr_web_site_sk,
         ws_sold_date_sk  as date_sk,
         ws_ext_sales_price as sales_price,
         ws_net_profit as profit,
         cast(0 as decimal(7,2)) as return_amt,
         cast(0 as decimal(7,2)) as net_loss
   from web_sales
   union all
   select ws_web_site_sk as wsr_web_site_sk,
         wr_returned_date_sk as date_sk,
         cast(0 as decimal(7,2)) as sales_price,
         cast(0 as decimal(7,2)) as profit,
         wr_return_amt as return_amt,
         wr_net_loss as net_loss
   from web_returns left outer join web_sales on
      ( wr_item_sk = ws_item_sk
        and wr_order_number = ws_order_number)
 ) salesreturns,
   date_dim,
   web_site
where date_sk = d_date_sk
     and d_date between cast('2002-08-05' as date)
            and (cast('2002-08-05' as date) + 14 )
     and wsr_web_site_sk = web_site_sk
group by web_site_id)
 select  channel
     , id
     , sum(sales) as sales
```

```
      , sum(returns) as returns
      , sum(profit) as profit
from
(select 'store channel' as channel
    , 'store' || s_store_id as id
    , sales
    , returns
    , (profit - profit_loss) as profit
from   ssr
union all
select 'catalog channel' as channel
    , 'catalog_page' || cp_catalog_page_id as id
    , sales
    , returns
    , (profit - profit_loss) as profit
from   csr
union all
select 'web channel' as channel
    , 'web_site' || web_site_id as id
    , sales
    , returns
    , (profit - profit_loss) as profit
from   wsr
) x
group by rollup (channel, id)
order by channel
        ,id
limit 100;
```

SQL6

```
select  a.ca_state state, count(*) cnt
 from customer_address a
    ,customer c
    ,store_sales s
    ,date_dim d
    ,item i
 where      a.ca_address_sk = c.c_current_addr_sk
   and c.c_customer_sk = s.ss_customer_sk
   and s.ss_sold_date_sk = d.d_date_sk
   and s.ss_item_sk = i.i_item_sk
   and d.d_month_seq =
       (select distinct (d_month_seq)
        from date_dim
           where d_year = 1998
         and d_moy = 7 )
   and i.i_current_price > 1.2 *
         (select avg(j.i_current_price)
       from item j
       where j.i_category = i.i_category)
 group by a.ca_state
 having count(*) >= 10
 order by cnt, a.ca_state
 limit 100;
```

SQL7

```
select  i_item_id,
      avg(ss_quantity) agg1,
      avg(ss_list_price) agg2,
      avg(ss_coupon_amt) agg3,
      avg(ss_sales_price) agg4
 from store_sales, customer_demographics, date_dim, item, promotion
 where ss_sold_date_sk = d_date_sk and
     ss_item_sk = i_item_sk and
     ss_cdemo_sk = cd_demo_sk and
     ss_promo_sk = p_promo_sk and
     cd_gender = 'M' and
     cd_marital_status = 'U' and
```

```
      cd_education_status = 'College' and
      (p_channel_email = 'N' or p_channel_event = 'N') and
      d_year = 1999
 group by i_item_id
 order by i_item_id
 limit 100;
```

SQL8

```
select  s_store_name
     ,sum(ss_net_profit)
 from store_sales
    ,date_dim
    ,store,
    (select ca_zip
    from (
     SELECT substr(ca_zip,1,5) ca_zip
     FROM customer_address
     WHERE substr(ca_zip,1,5) IN (
                   '74804','87276','13428','49436','56281','79805',
                   '46826','68570','20368','28846','41886',
                   '68164','68097','16113','18727','96789',
                   '63317','57937','19554','69911','83554',
                   '84246','61336','46999','25229','15960',
                   '61657','28058','64558','39712','74928',
                   '34018','87826','69733','26479','73630',
                   '88683','61704','81441','42706','54175',
                   '45152','49049','30850','63980','40484',
                   '71665','63755','23769','79855','24308',
                   '28241','16343','25663','85999','46359',
                   '93691','34706','99973','74947','60316',
                   '58637','48063','81363','19268','66228',
                   '78136','16368','99907','58139','17043',
                   '89764','14834','25152','70158','76080',
                   '81251','83972','48635','54671','35602',
                   '10788','57325','46354','92707','41103',
                   '89761','31840','69225','76139','18826',
                   '12556','51692','20579','50965','32136',
                   '71357','16309','82922','59273','40999',
                   '73273','93217','65679','12653','16978',
                   '27319','41973','65580','56237','17799',
                   '53192','63632','37089','65994','58048',
                   '14388','58085','80614','40042','79194',
                   '42268','61913','97332','37349','72146',
                   '52681','18176','39332','89283','69023',
                   '84175','11520','33483','60169','93562',
                   '10097','14536','70276','64042','22822',
                   '87229','51528','70269','44519','48044',
                   '78170','81440','60315','14543','30719',
                   '13240','62325','35517','51529','98085',
                   '79007','16582','95187','15625','88780',
                   '38656','74607','75117','62819','31929',
                   '27665','88890','98611','53527','48652',
                   '10324','62273','17726','36232','38526',
                   '50705','61179','30363','54408','58631',
                   '23622','50319','33299','78829','91267',
                   '25571','60347','44750','62797','10713',
                   '46494','91163','20973','42007','54724',
                   '89203','12561','71116','60404','70589',
                   '66744','46074','69138','34737','25092',
                   '59246','74778','40140','89476','71030',
                   '89861','93207','44996','34850','48752',
                   '79574','29570','76507','79728','43195',
                   '47596','46415','42514','68144','14169',
                   '17041','75747','33630','19378','32618',
                   '78704','75807','76800','80916','87272',
                   '37109','21714','14867','83806','33895',
                   '80637','20658','75224','92772','55791',
                   '58603','31681','38788','91922','42465',
```

```
                              '74371','72854','75746','80383','75909',
                              '37151','82077','80604','66771','46075',
                              '58723','15380','83174','53615','50347',
                              '98340','68957','63361','18705','47629',
                              '76013','68572','50588','31168','41563',
                              '38936','88746','19052','75648','46403',
                              '24332','54711','28218','80432','53870',
                              '25049','32562','94211','99803','49133',
                              '21202','50005','17953','14324','85525',
                              '51984','37304','69870','64321','66962',
                              '66453','40619','91199','54400','28804',
                              '65544','27059','31143','20303','52429',
                              '24476','91458','52514','55145','99015',
                              '51657','10001','96434','38325','39628',
                              '83338','62381','67697','61542','86076',
                              '89833','32657','56881','93983','85031',
                              '57530','56318','46934','34740','79458',
                              '88443','15861','56034','24808','32336',
                              '34312','96450','12923','91876','53509',
                              '30241','35816','52377','23946','23644',
                              '16413','35796','59100','21689','49199',
                              '40062','82510','14072','78823','49158',
                              '99933','75399','11365','44799','77549',
                              '19569','39186','78909','68143','70468',
                              '14944','33047','98329','42262','68647',
                              '65754','27357','56372','18073','12363',
                              '64467','26221','32914','70431','42436',
                              '55316','33335','27701','44687','22360',
                              '76124','44007','59525','51574','71555',
                              '43130','64199','19616','94285')
         intersect
          select ca_zip
          from (SELECT substr(ca_zip,1,5) ca_zip,count(*) cnt
                FROM customer_address, customer
                WHERE ca_address_sk = c_current_addr_sk and
                      c_preferred_cust_flag='Y'
                group by ca_zip
                having count(*) > 10)A1)A2) V1
 where ss_store_sk = s_store_sk
  and ss_sold_date_sk = d_date_sk
  and d_qoy = 1 and d_year = 1999
  and (substr(s_zip,1,2) = substr(V1.ca_zip,1,2))
 group by s_store_name
 order by s_store_name
 limit 100;
```

SQL9

```
select case when (select count(*)
                from store_sales
                where ss_quantity between 1 and 20) > 40845849
            then (select avg(ss_ext_tax)
                from store_sales
                where ss_quantity between 1 and 20)
            else (select avg(ss_net_paid)
                from store_sales
                where ss_quantity between 1 and 20) end bucket1 ,
       case when (select count(*)
                from store_sales
                where ss_quantity between 21 and 40) > 5712087
            then (select avg(ss_ext_tax)
                from store_sales
                where ss_quantity between 21 and 40)
            else (select avg(ss_net_paid)
                from store_sales
                where ss_quantity between 21 and 40) end bucket2,
       case when (select count(*)
                from store_sales
                where ss_quantity between 41 and 60) > 30393328
```

```
              then (select avg(ss_ext_tax)
                  from store_sales
                  where ss_quantity between 41 and 60)
              else (select avg(ss_net_paid)
                  from store_sales
                  where ss_quantity between 41 and 60) end bucket3,
          case when (select count(*)
                  from store_sales
                  where ss_quantity between 61 and 80) > 46385791
              then (select avg(ss_ext_tax)
                  from store_sales
                  where ss_quantity between 61 and 80)
              else (select avg(ss_net_paid)
                  from store_sales
                  where ss_quantity between 61 and 80) end bucket4,
          case when (select count(*)
                  from store_sales
                  where ss_quantity between 81 and 100) > 29981928
              then (select avg(ss_ext_tax)
                  from store_sales
                  where ss_quantity between 81 and 100)
              else (select avg(ss_net_paid)
                  from store_sales
                  where ss_quantity between 81 and 100) end bucket5
from reason
where r_reason_sk = 1
;
```

SQL10

```
select
 cd_gender,
 cd_marital_status,
 cd_education_status,
 count(*) cnt1,
 cd_purchase_estimate,
 count(*) cnt2,
 cd_credit_rating,
 count(*) cnt3,
 cd_dep_count,
 count(*) cnt4,
 cd_dep_employed_count,
 count(*) cnt5,
 cd_dep_college_count,
 count(*) cnt6
 from
 customer c,customer_address ca,customer_demographics
 where
 c.c_current_addr_sk = ca.ca_address_sk and
 ca_county in ('Clark County','Richardson County','Tom Green County','Sullivan County','Cass County') and
 cd_demo_sk = c.c_current_cdemo_sk and
 exists (select *
        from store_sales,date_dim
        where c.c_customer_sk = ss_customer_sk and
            ss_sold_date_sk = d_date_sk and
            d_year = 2000 and
            d_moy between 1 and 1+3) and
  (exists (select *
        from web_sales,date_dim
        where c.c_customer_sk = ws_bill_customer_sk and
            ws_sold_date_sk = d_date_sk and
            d_year = 2000 and
            d_moy between 1 ANd 1+3) or
   exists (select *
        from catalog_sales,date_dim
        where c.c_customer_sk = cs_ship_customer_sk and
            cs_sold_date_sk = d_date_sk and
            d_year = 2000 and
            d_moy between 1 and 1+3))
```

```
        group by cd_gender,
                cd_marital_status,
                cd_education_status,
                cd_purchase_estimate,
                cd_credit_rating,
                cd_dep_count,
                cd_dep_employed_count,
                cd_dep_college_count
        order by cd_gender,
                cd_marital_status,
                cd_education_status,
                cd_purchase_estimate,
                cd_credit_rating,
                cd_dep_count,
                cd_dep_employed_count,
                cd_dep_college_count
limit 100;
```

SQL11

```
with year_total as (
 select c_customer_id customer_id
       ,c_first_name customer_first_name
       ,c_last_name customer_last_name
       ,c_preferred_cust_flag customer_preferred_cust_flag
       ,c_birth_country customer_birth_country
       ,c_login customer_login
       ,c_email_address customer_email_address
       ,d_year dyear
       ,sum(ss_ext_list_price-ss_ext_discount_amt) year_total
       ,'s' sale_type
 from customer
     ,store_sales
     ,date_dim
 where c_customer_sk = ss_customer_sk
   and ss_sold_date_sk = d_date_sk
 group by c_customer_id
       ,c_first_name
       ,c_last_name
       ,c_preferred_cust_flag
       ,c_birth_country
       ,c_login
       ,c_email_address
       ,d_year
 union all
 select c_customer_id customer_id
       ,c_first_name customer_first_name
       ,c_last_name customer_last_name
       ,c_preferred_cust_flag customer_preferred_cust_flag
       ,c_birth_country customer_birth_country
       ,c_login customer_login
       ,c_email_address customer_email_address
       ,d_year dyear
       ,sum(ws_ext_list_price-ws_ext_discount_amt) year_total
       ,'w' sale_type
 from customer
     ,web_sales
     ,date_dim
 where c_customer_sk = ws_bill_customer_sk
   and ws_sold_date_sk = d_date_sk
 group by c_customer_id
       ,c_first_name
       ,c_last_name
       ,c_preferred_cust_flag
       ,c_birth_country
       ,c_login
       ,c_email_address
       ,d_year
       )
```

```
 select
             t_s_secyear.customer_id
            ,t_s_secyear.customer_first_name
            ,t_s_secyear.customer_last_name
            ,t_s_secyear.customer_birth_country
 from year_total t_s_firstyear
    ,year_total t_s_secyear
    ,year_total t_w_firstyear
    ,year_total t_w_secyear
 where t_s_secyear.customer_id = t_s_firstyear.customer_id
        and t_s_firstyear.customer_id = t_w_secyear.customer_id
        and t_s_firstyear.customer_id = t_w_firstyear.customer_id
        and t_s_firstyear.sale_type = 's'
        and t_w_firstyear.sale_type = 'w'
        and t_s_secyear.sale_type = 's'
        and t_w_secyear.sale_type = 'w'
        and t_s_firstyear.dyear = 2001
        and t_s_secyear.dyear = 2001+1
        and t_w_firstyear.dyear = 2001
        and t_w_secyear.dyear = 2001+1
        and t_s_firstyear.year_total > 0
        and t_w_firstyear.year_total > 0
        and case when t_w_firstyear.year_total > 0 then t_w_secyear.year_total / t_w_firstyear.year_total else
0.0 end
            > case when t_s_firstyear.year_total > 0 then t_s_secyear.year_total / t_s_firstyear.year_total else 0.0
end
 order by t_s_secyear.customer_id
        ,t_s_secyear.customer_first_name
        ,t_s_secyear.customer_last_name
        ,t_s_secyear.customer_birth_country
limit 100;
```

SQL12

```
select  i_item_id
    ,i_item_desc
    ,i_category
    ,i_class
    ,i_current_price
    ,sum(ws_ext_sales_price) as itemrevenue
    ,sum(ws_ext_sales_price)*100/sum(sum(ws_ext_sales_price)) over
        (partition by i_class) as revenueratio
from
    web_sales
    ,item
    ,date_dim
where
    ws_item_sk = i_item_sk
    and i_category in ('Music', 'Shoes', 'Children')
    and ws_sold_date_sk = d_date_sk
    and d_date between cast('2000-05-14' as date)
            and (cast('2000-05-14' as date) + 30 )
group by
    i_item_id
        ,i_item_desc
        ,i_category
        ,i_class
        ,i_current_price
order by
    i_category
        ,i_class
        ,i_item_id
        ,i_item_desc
        ,revenueratio
limit 100;
```

SQL13

```
select avg(ss_quantity)
     ,avg(ss_ext_sales_price)
     ,avg(ss_ext_wholesale_cost)
     ,sum(ss_ext_wholesale_cost)
 from store_sales
    ,store
    ,customer_demographics
    ,household_demographics
    ,customer_address
    ,date_dim
 where s_store_sk = ss_store_sk
 and  ss_sold_date_sk = d_date_sk and d_year = 2001
 and((ss_hdemo_sk=hd_demo_sk
 and cd_demo_sk = ss_cdemo_sk
 and cd_marital_status = 'U'
 and cd_education_status = '4 yr Degree'
 and ss_sales_price between 100.00 and 150.00
 and hd_dep_count = 3
    )or
    (ss_hdemo_sk=hd_demo_sk
 and cd_demo_sk = ss_cdemo_sk
 and cd_marital_status = 'D'
 and cd_education_status = '2 yr Degree'
 and ss_sales_price between 50.00 and 100.00
 and hd_dep_count = 1
    ) or
    (ss_hdemo_sk=hd_demo_sk
 and cd_demo_sk = ss_cdemo_sk
 and cd_marital_status = 'S'
 and cd_education_status = 'Advanced Degree'
 and ss_sales_price between 150.00 and 200.00
 and hd_dep_count = 1
    ))
 and((ss_addr_sk = ca_address_sk
 and ca_country = 'United States'
 and ca_state in ('IL', 'WI', 'TN')
 and ss_net_profit between 100 and 200
    ) or
    (ss_addr_sk = ca_address_sk
 and ca_country = 'United States'
 and ca_state in ('MO', 'OK', 'WA')
 and ss_net_profit between 150 and 300
    ) or
    (ss_addr_sk = ca_address_sk
 and ca_country = 'United States'
 and ca_state in ('NE', 'VA', 'GA')
 and ss_net_profit between 50 and 250
    ))
;
```

SQL14

```
with  cross_items as
 (select i_item_sk ss_item_sk
 from item,
 (select iss.i_brand_id brand_id
    ,iss.i_class_id class_id
    ,iss.i_category_id category_id
 from store_sales
    ,item iss
    ,date_dim d1
 where ss_item_sk = iss.i_item_sk
   and ss_sold_date_sk = d1.d_date_sk
   and d1.d_year between 2000 AND 2000 + 2
 intersect
 select ics.i_brand_id
    ,ics.i_class_id
```

```
        ,ics.i_category_id
from catalog_sales
    ,item ics
    ,date_dim d2
where cs_item_sk = ics.i_item_sk
  and cs_sold_date_sk = d2.d_date_sk
  and d2.d_year between 2000 AND 2000 + 2
intersect
select iws.i_brand_id
    ,iws.i_class_id
    ,iws.i_category_id
from web_sales
    ,item iws
    ,date_dim d3
where ws_item_sk = iws.i_item_sk
  and ws_sold_date_sk = d3.d_date_sk
  and d3.d_year between 2000 AND 2000 + 2)
where i_brand_id = brand_id
    and i_class_id = class_id
    and i_category_id = category_id
),
avg_sales as
(select avg(quantity*list_price) average_sales
 from (select ss_quantity quantity
            ,ss_list_price list_price
        from store_sales
            ,date_dim
        where ss_sold_date_sk = d_date_sk
          and d_year between 2000 and 2000 + 2
        union all
        select cs_quantity quantity
            ,cs_list_price list_price
        from catalog_sales
            ,date_dim
        where cs_sold_date_sk = d_date_sk
          and d_year between 2000 and 2000 + 2
        union all
        select ws_quantity quantity
            ,ws_list_price list_price
        from web_sales
            ,date_dim
        where ws_sold_date_sk = d_date_sk
          and d_year between 2000 and 2000 + 2) x)
 select  channel, i_brand_id,i_class_id,i_category_id,sum(sales), sum(number_sales)
 from(
     select 'store' channel, i_brand_id,i_class_id
            ,i_category_id,sum(ss_quantity*ss_list_price) sales
            , count(*) number_sales
        from store_sales
            ,item
            ,date_dim
        where ss_item_sk in (select ss_item_sk from cross_items)
          and ss_item_sk = i_item_sk
          and ss_sold_date_sk = d_date_sk
          and d_year = 2000+2
          and d_moy = 11
        group by i_brand_id,i_class_id,i_category_id
        having sum(ss_quantity*ss_list_price) > (select average_sales from avg_sales)
        union all
        select 'catalog' channel, i_brand_id,i_class_id,i_category_id, sum(cs_quantity*cs_list_price) sales,
count(*) number_sales
        from catalog_sales
            ,item
            ,date_dim
        where cs_item_sk in (select ss_item_sk from cross_items)
          and cs_item_sk = i_item_sk
          and cs_sold_date_sk = d_date_sk
          and d_year = 2000+2
          and d_moy = 11
```

```
                    group by i_brand_id,i_class_id,i_category_id
                    having sum(cs_quantity*cs_list_price) > (select average_sales from avg_sales)
                    union all
                    select 'web' channel, i_brand_id,i_class_id,i_category_id, sum(ws_quantity*ws_list_price) sales , count(*)
number_sales
                    from web_sales
                       ,item
                       ,date_dim
                    where ws_item_sk in (select ss_item_sk from cross_items)
                      and ws_item_sk = i_item_sk
                      and ws_sold_date_sk = d_date_sk
                      and d_year = 2000+2
                      and d_moy = 11
                    group by i_brand_id,i_class_id,i_category_id
                    having sum(ws_quantity*ws_list_price) > (select average_sales from avg_sales)
 ) y
 group by rollup (channel, i_brand_id,i_class_id,i_category_id)
 order by channel,i_brand_id,i_class_id,i_category_id
 limit 100;
with  cross_items as
 (select i_item_sk ss_item_sk
 from item,
 (select iss.i_brand_id brand_id
     ,iss.i_class_id class_id
     ,iss.i_category_id category_id
 from store_sales
     ,item iss
     ,date_dim d1
 where ss_item_sk = iss.i_item_sk
   and ss_sold_date_sk = d1.d_date_sk
   and d1.d_year between 2000 AND 2000 + 2
 intersect
 select ics.i_brand_id
     ,ics.i_class_id
     ,ics.i_category_id
 from catalog_sales
     ,item ics
     ,date_dim d2
 where cs_item_sk = ics.i_item_sk
   and cs_sold_date_sk = d2.d_date_sk
   and d2.d_year between 2000 AND 2000 + 2
 intersect
 select iws.i_brand_id
     ,iws.i_class_id
     ,iws.i_category_id
 from web_sales
     ,item iws
     ,date_dim d3
 where ws_item_sk = iws.i_item_sk
   and ws_sold_date_sk = d3.d_date_sk
   and d3.d_year between 2000 AND 2000 + 2) x
 where i_brand_id = brand_id
     and i_class_id = class_id
     and i_category_id = category_id
),
 avg_sales as
(select avg(quantity*list_price) average_sales
  from (select ss_quantity quantity
           ,ss_list_price list_price
        from store_sales
           ,date_dim
        where ss_sold_date_sk = d_date_sk
          and d_year between 2000 and 2000 + 2
        union all
        select cs_quantity quantity
           ,cs_list_price list_price
        from catalog_sales
           ,date_dim
        where cs_sold_date_sk = d_date_sk
```

```
      and d_year between 2000 and 2000 + 2
    union all
    select ws_quantity quantity
         ,ws_list_price list_price
    from web_sales
       ,date_dim
    where ws_sold_date_sk = d_date_sk
      and d_year between 2000 and 2000 + 2) x)
 select  this_year.channel ty_channel
                    ,this_year.i_brand_id ty_brand
                    ,this_year.i_class_id ty_class
                    ,this_year.i_category_id ty_category
                    ,this_year.sales ty_sales
                    ,this_year.number_sales ty_number_sales
                    ,last_year.channel ly_channel
                    ,last_year.i_brand_id ly_brand
                    ,last_year.i_class_id ly_class
                    ,last_year.i_category_id ly_category
                    ,last_year.sales ly_sales
                    ,last_year.number_sales ly_number_sales
from
(select 'store' channel, i_brand_id,i_class_id,i_category_id
     ,sum(ss_quantity*ss_list_price) sales, count(*) number_sales
from store_sales
   ,item
   ,date_dim
where ss_item_sk in (select ss_item_sk from cross_items)
  and ss_item_sk = i_item_sk
  and ss_sold_date_sk = d_date_sk
  and d_week_seq = (select d_week_seq
              from date_dim
              where d_year = 2000 + 1
                and d_moy = 12
                and d_dom = 17)
group by i_brand_id,i_class_id,i_category_id
having sum(ss_quantity*ss_list_price) > (select average_sales from avg_sales)) this_year,
(select 'store' channel, i_brand_id,i_class_id
     ,i_category_id, sum(ss_quantity*ss_list_price) sales, count(*) number_sales
from store_sales
   ,item
   ,date_dim
where ss_item_sk in (select ss_item_sk from cross_items)
  and ss_item_sk = i_item_sk
  and ss_sold_date_sk = d_date_sk
  and d_week_seq = (select d_week_seq
              from date_dim
              where d_year = 2000
                and d_moy = 12
                and d_dom = 17)
group by i_brand_id,i_class_id,i_category_id
having sum(ss_quantity*ss_list_price) > (select average_sales from avg_sales)) last_year
where this_year.i_brand_id= last_year.i_brand_id
  and this_year.i_class_id = last_year.i_class_id
  and this_year.i_category_id = last_year.i_category_id
order by this_year.channel, this_year.i_brand_id, this_year.i_class_id, this_year.i_category_id
limit 100;
```

SQL15

```
select  ca_zip
     ,sum(cs_sales_price)
 from catalog_sales
    ,customer
    ,customer_address
    ,date_dim
where cs_bill_customer_sk = c_customer_sk
   and c_current_addr_sk = ca_address_sk
   and ( substr(ca_zip,1,5) in ('85669', '86197','88274','83405','86475',
                              '85392', '85460', '80348', '81792')
```

```
      or ca_state in ('CA','WA','GA')
      or cs_sales_price > 500)
  and cs_sold_date_sk = d_date_sk
  and d_qoy = 2 and d_year = 1999
group by ca_zip
order by ca_zip
limit 100;
```

SQL16

```
select
  count(distinct cs_order_number) as "order count"
 ,sum(cs_ext_ship_cost) as "total shipping cost"
 ,sum(cs_net_profit) as "total net profit"
from
  catalog_sales cs1
 ,date_dim
 ,customer_address
 ,call_center
where
  d_date between '1999-2-01' and
        (cast('1999-2-01' as date) + 60 )
and cs1.cs_ship_date_sk = d_date_sk
and cs1.cs_ship_addr_sk = ca_address_sk
and ca_state = 'TX'
and cs1.cs_call_center_sk = cc_call_center_sk
and cc_county in ('Barrow County','Luce County','Mobile County','Richland County',
          'Wadena County'
)
and exists (select *
        from catalog_sales cs2
        where cs1.cs_order_number = cs2.cs_order_number
          and cs1.cs_warehouse_sk <> cs2.cs_warehouse_sk)
and not exists(select *
          from catalog_returns cr1
          where cs1.cs_order_number = cr1.cr_order_number)
order by count(distinct cs_order_number)
limit 100;
```

SQL17

```
select  i_item_id
      ,i_item_desc
      ,s_state
      ,count(ss_quantity) as store_sales_quantitycount
      ,avg(ss_quantity) as store_sales_quantityave
      ,stddev_samp(ss_quantity) as store_sales_quantitystdev
      ,stddev_samp(ss_quantity)/avg(ss_quantity) as store_sales_quantitycov
      ,count(sr_return_quantity) as store_returns_quantitycount
      ,avg(sr_return_quantity) as store_returns_quantityave
      ,stddev_samp(sr_return_quantity) as store_returns_quantitystdev
      ,stddev_samp(sr_return_quantity)/avg(sr_return_quantity) as store_returns_quantitycov
      ,count(cs_quantity) as catalog_sales_quantitycount ,avg(cs_quantity) as catalog_sales_quantityave
      ,stddev_samp(cs_quantity) as catalog_sales_quantitystdev
      ,stddev_samp(cs_quantity)/avg(cs_quantity) as catalog_sales_quantitycov
 from store_sales
     ,store_returns
     ,catalog_sales
     ,date_dim d1
     ,date_dim d2
     ,date_dim d3
     ,store
     ,item
 where d1.d_quarter_name = '2000Q1'
   and d1.d_date_sk = ss_sold_date_sk
   and i_item_sk = ss_item_sk
   and s_store_sk = ss_store_sk
   and ss_customer_sk = sr_customer_sk
   and ss_item_sk = sr_item_sk
```

```
        and ss_ticket_number = sr_ticket_number
        and sr_returned_date_sk = d2.d_date_sk
        and d2.d_quarter_name in ('2000Q1','2000Q2','2000Q3')
        and sr_customer_sk = cs_bill_customer_sk
        and sr_item_sk = cs_item_sk
        and cs_sold_date_sk = d3.d_date_sk
        and d3.d_quarter_name in ('2000Q1','2000Q2','2000Q3')
 group by i_item_id
        ,i_item_desc
        ,s_state
 order by i_item_id
        ,i_item_desc
        ,s_state
limit 100;
```

SQL18

```
select  i_item_id,
        ca_country,
        ca_state,
        ca_county,
        avg( cast(cs_quantity as decimal(12,2))) agg1,
        avg( cast(cs_list_price as decimal(12,2))) agg2,
        avg( cast(cs_coupon_amt as decimal(12,2))) agg3,
        avg( cast(cs_sales_price as decimal(12,2))) agg4,
        avg( cast(cs_net_profit as decimal(12,2))) agg5,
        avg( cast(c_birth_year as decimal(12,2))) agg6,
        avg( cast(cd1.cd_dep_count as decimal(12,2))) agg7
 from catalog_sales, customer_demographics cd1,
      customer_demographics cd2, customer, customer_address, date_dim, item
 where cs_sold_date_sk = d_date_sk and
        cs_item_sk = i_item_sk and
        cs_bill_cdemo_sk = cd1.cd_demo_sk and
        cs_bill_customer_sk = c_customer_sk and
        cd1.cd_gender = 'M' and
        cd1.cd_education_status = 'Primary' and
        c_current_cdemo_sk = cd2.cd_demo_sk and
        c_current_addr_sk = ca_address_sk and
        c_birth_month in (10,1,8,7,3,5) and
        d_year = 1998 and
        ca_state in ('NE','OK','NC'
                ,'CO','ID','AR','MO')
 group by rollup (i_item_id, ca_country, ca_state, ca_county)
 order by ca_country,
        ca_state,
        ca_county,
    i_item_id
limit 100;
```

SQL19

```
select  i_brand_id brand_id, i_brand brand, i_manufact_id, i_manufact,
    sum(ss_ext_sales_price) ext_price
 from date_dim, store_sales, item,customer,customer_address,store
 where d_date_sk = ss_sold_date_sk
    and ss_item_sk = i_item_sk
    and i_manager_id=62
    and d_moy=11
    and d_year=2000
    and ss_customer_sk = c_customer_sk
    and c_current_addr_sk = ca_address_sk
    and substr(ca_zip,1,5) <> substr(s_zip,1,5)
    and ss_store_sk = s_store_sk
 group by i_brand
        ,i_brand_id
        ,i_manufact_id
        ,i_manufact
 order by ext_price desc
        ,i_brand
```

```
        ,i_brand_id
        ,i_manufact_id
        ,i_manufact
limit 100 ;
```

SQL20

```
select  i_item_id
       ,i_item_desc
       ,i_category
       ,i_class
       ,i_current_price
       ,sum(cs_ext_sales_price) as itemrevenue
       ,sum(cs_ext_sales_price)*100/sum(sum(cs_ext_sales_price)) over
           (partition by i_class) as revenueratio
 from    catalog_sales
    ,item
    ,date_dim
 where cs_item_sk = i_item_sk
   and i_category in ('Sports', 'Shoes', 'Women')
   and cs_sold_date_sk = d_date_sk
 and d_date between cast('2001-03-21' as date)
             and (cast('2001-03-21' as date) + 30)
 group by i_item_id
       ,i_item_desc
       ,i_category
       ,i_class
       ,i_current_price
 order by i_category
       ,i_class
       ,i_item_id
       ,i_item_desc
       ,revenueratio
limit 100;
```