数据仓库服务
9.1.1.x
性能白皮书

文档版本 01

发布日期 2025-08-22

华为云计算技术有限公司

商标声明


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

注意

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

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

目 录

1 9.1.1 版本性能测试概述

目的

本次性能测试基于华为云基础环境，分别在同等硬件配置和同等数据规模下，基于TPC-H、TPC-DS标准测试集，对DWS 9.1.1版本和9.1.0版本进行性能对比测试。基于SSB-Flat测试集，对DWS 9.1.0版本和开源OLAP产品ClickHouse进行对比测试。本次性能测试时间为2025年4月。

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。

SSB

SSB（Star Schema Benchmark）是一种在学术界和工业界广泛应用的数据库系统性能评估基准测试方法。它能够对比不同数据仓库在处理星型模型查询时的性能，帮助数据库管理员和决策者选择最符合需求的数据库系统。此外，参考OLAP行业的做法，将SSB中的星型模型展平转化为宽表，还可以改造成单一表测试Benchmark（SSB Flat）。

2 9.1.1 版本测试结论

2.1 性能总览

在9.1.1版本，我们实现了很多性能优化特性，提升整体开箱的SQL查询性能。以TPC-H、TPC-DS 1TB作为性能测试对比的基准，重点对比最新9.1.1版本与9.1.0版本的性能提升。集群规模为6节点，其中各节点的规格为16U 64G，累计96U 384G。从以下测试结果可以看到：

- 9.1.1版本存算一体架构TPC-H总查询耗时为162.62秒，相较9.1.0版本的185.47秒，性能提升14%；TPC-DS总查询耗时为417.83秒，相较9.1.0版本的658.06秒，性能提升57.5%。
- 9.1.1版本存算分离架构与存算一体架构性能劣化在15%以内。
- TPC-H 1000x测试基准22个SQL中，9.1.1.100版本相比9.1.0版本所有SQL性能提升，特别是Q10和Q20提升达1倍。
- TPC-DS 1000x测试基准99个SQL中，9.1.1.100版本相比9.1.0版本43个SQL有明显提升，6个SQL性能提升有2~7倍。
- 不论是简单的过滤、排序、聚集，还是复杂的多表关联、窗口计算、CTE查询，9.1.1版本都有明显性能优势。

表 2-1 TPC-H 和 TPC-DS 性能总览

1000x	DWS开箱性能			
版本	9.1.0		9.1.1	
	存算一体	存算分离	存算一体	存算分离
TPC-H	185.47	188.29	162.62	185.22
TPC-DS	658.06	663.64	417.83	445.34

在9.1.0版本，我们使用存算分离架构指定二级分区，基于SSB-Flat 100 GB测试基准，对比DWS和ClickHouse的性能表现。从以下测试结果可以看到：

- 开箱性能相比开源厂商ClickHouse有200%性能优势；

- 开箱性能相比9.1.0.100版本提升了133%；

表 2-2 SSB 性能总览

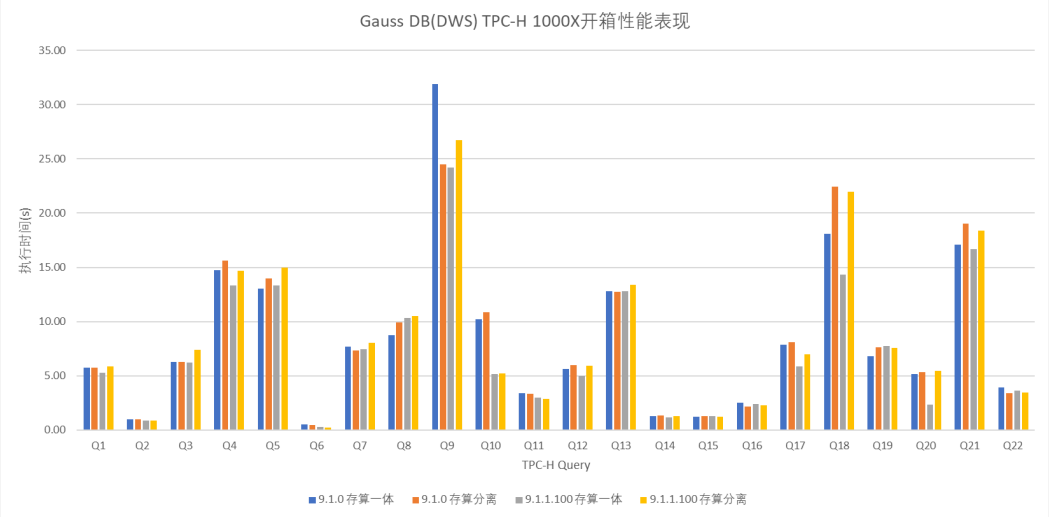
100x	DWS		ClickHouse
SSB-Flat	**9.1.0.200开箱（s）**	**9.1.0.100开箱（s）**	-
	0.91	2.12	2.73

3 TPC-H 性能测试

3.1 TPC-H 测试结果

DWS 测试了使用存算一体和存算分离两种部署架构下，TPC-H 1T规模数据集的开箱查询性能，共22个查询。存算一体查询总耗时为162.62s，存算分离查询总耗时为185.22s。

图 3-1 TPC-H 1000X 开箱性能



详细性能数据见下表。

表 3-1 TPC-H 测试结果

TPC-H查询	9.1.0		9.1.1.100	
-	存算一体	存算分离	存算一体	存算分离
Q1	5.78	5.73	5.29	5.87
Q2	0.97	0.98	0.87	0.87

TPC-H查询	9.1.0		9.1.1.100	
Q3	6.26	6.27	6.24	7.42
Q4	14.71	15.63	13.35	14.67
Q5	13.02	13.96	13.32	14.95
Q6	0.50	0.46	0.28	0.23
Q7	7.68	7.35	7.46	8.02
Q8	8.75	9.89	10.33	10.48
Q9	31.87	24.51	24.22	26.71
Q10	10.19	10.84	5.14	5.23
Q11	3.40	3.34	2.97	2.86
Q12	5.62	5.96	5.01	5.92
Q13	12.77	12.71	12.78	13.41
Q14	1.26	1.32	1.16	1.30
Q15	1.22	1.28	1.27	1.23
Q16	2.49	2.19	2.38	2.30
Q17	7.85	8.09	5.85	6.96
Q18	18.11	22.43	14.31	21.96
Q19	6.80	7.61	7.73	7.55
Q20	5.15	5.36	2.36	5.46
Q21	17.11	19.01	16.67	18.36
Q22	3.96	3.37	3.63	3.44
总时长（s）	**185.47**	**188.29**	**162.62**	**185.22**

3.2 TPC-H 测试环境

硬件环境

每个测试环境6个节点，配置如下：

- CPU 16核：Intel Ice Lake
- 内存：64GB
- 网络带宽：9Gbit/s
- 磁盘：SSD云盘，每块600GB，共2块

软件环境

- 内核版本：Linux 3.10.0-862.14.1.5.h757.eulerosv2r7.x86_64
- 操作系统：EulerOS release 2.0 (SP5)
- 数据库版本：DWS 3.0

3.3 TPC-H 测试过程

3.3.1 TPC-H 测试数据

表 3-2 TPC-H 测试数据

序号	表名	行数	表大小
1	region	5	294KB
2	nation	25	298KB
3	supplier	10,000,000	1020MB
4	customer	150,000,000	8226MB
5	part	200,000,000	5216MB
6	partsupp	800,000,000	29GB
7	orders	1,500,000,000	43GB
8	lineitem	5,999,989,709	171GB

3.3.2 TPC-H 数据生成

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工具上传到ECS的/data1/script/tpch-kit目录执行以下命令解压。

“tpch_3.0.1.zip”替换为实际的软件包名。

```
cd /data1/script/tpch-kit && unzip tpch_v3.0.1.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文件。

```
#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文件。

```
//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_v3.0.1/dbgen
$ cp makefile.suite makefile
$ make -f makefile
$ cp -R /data1/script/tpch-kit/TPC-H_Tools_v3.0.1/ /data2/script/tpch-kit/
```

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

须知

TPC-H 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_v3.0.1/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_v3.0.1/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_v3.0.1/dbgen/*.tbl*
du -sh /data2/script/tpch-kit/TPC-H_Tools_v3.0.1/dbgen/*.tbl*
```

步骤7 将TPC-H 1000X数据转移至指定目录。

```
mv /data1/script/tpch-kit/TPC-H_Tools_v3.0.1/dbgen/*.tbl* /data1/script/tpch-kit/tpch1000X
mv /data2/script/tpch-kit/TPC-H_Tools_v3.0.1/dbgen/*.tbl* /data2/script/tpch-kit/tpch1000X
```

----结束

3.3.3 建表与导入 TPC-H 数据

创建 TPC-H 目标表

连接DWS数据库后执行以下命令创建目标表。

```
CREATE TABLE REGION
(
   R_REGIONKEY  INT NOT NULL
 , R_NAME VARCHAR(25) NOT NULL
 , R_COMMENT VARCHAR(152)
) WITH (orientation=column, colversion=2.0, enable_hstore=true,
enable_hstore_opt=true,bucketnums=16384)
DISTRIBUTE BY replication;

CREATE TABLE NATION
(
   N_NATIONKEY  INT NOT NULL
  , N_NAME VARCHAR(25) NOT NULL
  , N_REGIONKEY  INT NOT NULL
  , N_COMMENT VARCHAR(152)
) WITH (orientation=column, colversion=2.0, enable_hstore=true,
enable_hstore_opt=true,bucketnums=16384)
DISTRIBUTE BY replication;

CREATE TABLE SUPPLIER
(
   S_SUPPKEY  BIGINT NOT NULL
  , S_NAME  VARCHAR(25) NOT NULL
  , S_ADDRESS  VARCHAR(40) NOT NULL
  , S_NATIONKEY   INT NOT NULL
  , S_PHONE VARCHAR(15) NOT NULL
  , S_ACCTBAL  DECIMAL(15,2) NOT NULL
  , S_COMMENT  VARCHAR(101) NOT NULL
) WITH (orientation=column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384,secondary_part_column='s_su
ppkey',secondary_part_num=72)
DISTRIBUTE BY hash(S_SUPPKEY)
PARTITION BY RANGE(S_NATIONKEY)
(
PARTITION S_NATIONKEY_1 VALUES LESS THAN(1),
PARTITION S_NATIONKEY_2 VALUES LESS THAN(2),
PARTITION S_NATIONKEY_3 VALUES LESS THAN(3),
PARTITION S_NATIONKEY_4 VALUES LESS THAN(4),
PARTITION S_NATIONKEY_5 VALUES LESS THAN(5),
PARTITION S_NATIONKEY_6 VALUES LESS THAN(6),
PARTITION S_NATIONKEY_7 VALUES LESS THAN(7),
PARTITION S_NATIONKEY_8 VALUES LESS THAN(8),
PARTITION S_NATIONKEY_9 VALUES LESS THAN(9),
PARTITION S_NATIONKEY_10 VALUES LESS THAN(10),
PARTITION S_NATIONKEY_11 VALUES LESS THAN(11),
PARTITION S_NATIONKEY_12 VALUES LESS THAN(12),
PARTITION S_NATIONKEY_13 VALUES LESS THAN(13),
PARTITION S_NATIONKEY_14 VALUES LESS THAN(14),
PARTITION S_NATIONKEY_15 VALUES LESS THAN(15),
PARTITION S_NATIONKEY_16 VALUES LESS THAN(16),
PARTITION S_NATIONKEY_17 VALUES LESS THAN(17),
PARTITION S_NATIONKEY_18 VALUES LESS THAN(18),
PARTITION S_NATIONKEY_19 VALUES LESS THAN(19),
PARTITION S_NATIONKEY_20 VALUES LESS THAN(20),
PARTITION S_NATIONKEY_21 VALUES LESS THAN(21),
PARTITION S_NATIONKEY_22 VALUES LESS THAN(22),
PARTITION S_NATIONKEY_23 VALUES LESS THAN(23),
PARTITION S_NATIONKEY_24 VALUES LESS THAN(24),
PARTITION S_NATIONKEY_25 VALUES LESS THAN(25)
);

CREATE TABLE CUSTOMER
```

```
(
  C_CUSTKEY  BIGINT NOT NULL
 , C_NAME  VARCHAR(25) NOT NULL
 , C_ADDRESS  VARCHAR(40) NOT NULL
 , C_NATIONKEY   INT NOT NULL
 , C_PHONE VARCHAR(15) NOT NULL
 , C_ACCTBAL  DECIMAL(15,2)   NOT NULL
 , C_MKTSEGMENT  VARCHAR(10) NOT NULL
 , C_COMMENT  VARCHAR(117) NOT NULL
) WITH (orientation=column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384,secondary_part_column='c_cu
stkey',secondary_part_num=72)
DISTRIBUTE BY hash(C_CUSTKEY)
PARTITION BY RANGE(C_NATIONKEY)
(
PARTITION C_NATIONKEY_1 VALUES LESS THAN(1),
PARTITION C_NATIONKEY_2 VALUES LESS THAN(2),
PARTITION C_NATIONKEY_3 VALUES LESS THAN(3),
PARTITION C_NATIONKEY_4 VALUES LESS THAN(4),
PARTITION C_NATIONKEY_5 VALUES LESS THAN(5),
PARTITION C_NATIONKEY_6 VALUES LESS THAN(6),
PARTITION C_NATIONKEY_7 VALUES LESS THAN(7),
PARTITION C_NATIONKEY_8 VALUES LESS THAN(8),
PARTITION C_NATIONKEY_9 VALUES LESS THAN(9),
PARTITION C_NATIONKEY_10 VALUES LESS THAN(10),
PARTITION C_NATIONKEY_11 VALUES LESS THAN(11),
PARTITION C_NATIONKEY_12 VALUES LESS THAN(12),
PARTITION C_NATIONKEY_13 VALUES LESS THAN(13),
PARTITION C_NATIONKEY_14 VALUES LESS THAN(14),
PARTITION C_NATIONKEY_15 VALUES LESS THAN(15),
PARTITION C_NATIONKEY_16 VALUES LESS THAN(16),
PARTITION C_NATIONKEY_17 VALUES LESS THAN(17),
PARTITION C_NATIONKEY_18 VALUES LESS THAN(18),
PARTITION C_NATIONKEY_19 VALUES LESS THAN(19),
PARTITION C_NATIONKEY_20 VALUES LESS THAN(20),
PARTITION C_NATIONKEY_21 VALUES LESS THAN(21),
PARTITION C_NATIONKEY_22 VALUES LESS THAN(22),
PARTITION C_NATIONKEY_23 VALUES LESS THAN(23),
PARTITION C_NATIONKEY_24 VALUES LESS THAN(24),
PARTITION C_NATIONKEY_25 VALUES LESS THAN(25)
);

CREATE TABLE PART
(
  P_PARTKEY  BIGINT NOT NULL
 , P_NAME  VARCHAR(55) NOT NULL
 , P_MFGR  VARCHAR(25) NOT NULL
 , P_BRAND VARCHAR(10) NOT NULL
 , P_TYPE  VARCHAR(25) NOT NULL
 , P_SIZE  BIGINT NOT NULL
 , P_CONTAINER   VARCHAR(10) NOT NULL
 , P_RETAILPRICE DECIMAL(15,2) NOT NULL
 , P_COMMENT  VARCHAR(23) NOT NULL

) WITH (orientation=column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384,secondary_part_column='p_pa
rtkey',secondary_part_num=72)
DISTRIBUTE BY hash(P_PARTKEY)
PARTITION BY RANGE(P_SIZE)
(
PARTITION P_SIZE_1 VALUES LESS THAN(11),
PARTITION P_SIZE_2 VALUES LESS THAN(21),
PARTITION P_SIZE_3 VALUES LESS THAN(31),
PARTITION P_SIZE_4 VALUES LESS THAN(41),
PARTITION P_SIZE_5 VALUES LESS THAN(51)
);

CREATE TABLE PARTSUPP
(
```



```
  PS_PARTKEY  BIGINT NOT NULL
  , PS_SUPPKEY  BIGINT NOT NULL
  , PS_AVAILQTY BIGINT NOT NULL
  , PS_SUPPLYCOST  DECIMAL(15,2)  NOT NULL
  , PS_COMMENT  VARCHAR(199) NOT NULL
) WITH (orientation=column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384,secondary_part_column='ps_p
artkey',secondary_part_num=72)
DISTRIBUTE BY hash(PS_PARTKEY)
PARTITION BY RANGE(PS_AVAILQTY)
(
PARTITION PS_AVAILQTY_1 VALUES LESS THAN(1000),
PARTITION PS_AVAILQTY_2 VALUES LESS THAN(2000),
PARTITION PS_AVAILQTY_3 VALUES LESS THAN(3000),
PARTITION PS_AVAILQTY_4 VALUES LESS THAN(4000),
PARTITION PS_AVAILQTY_5 VALUES LESS THAN(5000),
PARTITION PS_AVAILQTY_6 VALUES LESS THAN(6000),
PARTITION PS_AVAILQTY_7 VALUES LESS THAN(7000),
PARTITION PS_AVAILQTY_8 VALUES LESS THAN(8000),
PARTITION PS_AVAILQTY_9 VALUES LESS THAN(9000),
PARTITION PS_AVAILQTY_10 VALUES LESS THAN(10000)
) ;

CREATE TABLE ORDERS
(
   O_ORDERKEY BIGINT NOT NULL
  , O_CUSTKEY  BIGINT NOT NULL
  , O_ORDERSTATUS VARCHAR(1) NOT NULL
  , O_TOTALPRICE  DECIMAL(15,2) NOT NULL
  , O_ORDERDATE   DATE NOT NULL
  , O_ORDERPRIORITY  VARCHAR(15) NOT NULL
  , O_CLERK VARCHAR(15) NOT NULL
  , O_SHIPPRIORITY   BIGINT NOT NULL
  , O_COMMENT  VARCHAR(79) NOT NULL
) WITH (orientation=column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384,secondary_part_column='o_or
derkey',secondary_part_num=72)
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')
) ;

CREATE TABLE LINEITEM
(
   L_ORDERKEY BIGINT NOT NULL
  , L_PARTKEY  BIGINT NOT NULL
  , L_SUPPKEY  BIGINT NOT NULL
  , L_LINENUMBER  BIGINT NOT NULL
  , L_QUANTITY DECIMAL(15,2) NOT NULL
  , L_EXTENDEDPRICE  DECIMAL(15,2) NOT NULL
  , L_DISCOUNT DECIMAL(15,2) NOT NULL
  , L_TAX   DECIMAL(15,2) NOT NULL
  , L_RETURNFLAG  VARCHAR(1) NOT NULL
  , L_LINESTATUS  VARCHAR(1) NOT NULL
  , L_SHIPDATE DATE NOT NULL
  , L_COMMITDATE  DATE NOT NULL
  , L_RECEIPTDATE DATE NOT NULL
  , L_SHIPINSTRUCT VARCHAR(25) NOT NULL
  , L_SHIPMODE  VARCHAR(10) NOT NULL
  , L_COMMENT   VARCHAR(44) NOT NULL
) WITH (orientation=column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384,secondary_part_column='l_ord
```



```
erkey',secondary_part_num=72)
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')
) ;
```

安装和启动 GDS

步骤1 参见**工具下载**下载GDS客户端（与gsql客户端在一个包）。

步骤2 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3 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4 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5 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6 切换到gds_user用户。

```
su - gds_user
```

步骤7 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8 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
/opt/gds/bin/gds -d /data1/script/ssb-kit/ssb100X/ -p 192.168.0.90:5004 -H 192.168.0.0/24 -l /opt/gds/
gds05_log.txt -D    #SSB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DWS的集群内网网段（即GDS所在的ECS与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

创建 TPC-H 数据集的 GDS 外表

连接DWS数据库后执行以下SQL语句创建。

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.tbl* | gsfs://192.168.0.90:5001/region.tbl*',
format 'text',
deLIMITer '|',
encoding 'utf8',
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.tbl* | gsfs://192.168.0.90:5001/nation.tbl*',
format 'text',
deLIMITer '|',
encoding 'utf8',
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.tbl* | gsfs://192.168.0.90:5001/supplier.tbl*',
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.tbl* | gsfs://192.168.0.90:5001/customer.tbl*',
format 'text',
deLIMITer '|',
encoding 'utf8',
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.tbl* | gsfs://192.168.0.90:5001/part.tbl*',
format 'text',
deLIMITer '|',
encoding 'utf8',
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.tbl* | gsfs://192.168.0.90:5001/partsupp.tbl*',
format 'text',
```



```
deLIMITer '|',
encoding 'utf8',
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.tbl* | gsfs://192.168.0.90:5001/orders.tbl*',
format 'text',
deLIMITer '|',
encoding 'utf8',
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.tbl* | gsfs://192.168.0.90:5001/lineitem.tbl*',
format 'text',
deLIMITer '|',
encoding 'utf8',
mode 'Normal'
);
```

导入 TPC-H 数据

执行以下SQL语句导入数据。

```
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;
```

设置 GUC 参数

在执行测试之前，可以设置如下几个参数，这将给性能带来5-10%的提升。

```
cpu_tuple_cost=0.0058401054702699184
cpu_operator_cost=0.002450424712151289
cpu_index_tuple_cost=0.004948411136865616
seq_page_cost=0.3447857201099396
random_page_cost=2.3901453018188477
allocate_mem_cost=0.5275554060935974
effective_cache_size=502
smp_thread_cost=278.525634765625
stream_multiple=0.5654585361480713
```

3.3.4 TPC-H 查询测试

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。

包含Q1~Q22共22个查询，具体的查询SQL代码如下显示，仅供参考。

了解更多内容，请访问TPC-H官网。

Q1

```
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;
```

Q2

```
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;
```

Q3

```
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;
```

Q4

```
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;
```

Q5

```
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;
```

Q6

```
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;
```

Q7

```
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;
```

Q8

```
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;
```

Q9

```
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;
```

Q10

```
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;
```

Q11

```
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;
```

Q12

```
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;
```

Q13

```
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;
```

Q14

```
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;
```

Q15

```
WITH revenue (supplier_no, total_revenue) as
(
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;
```

Q16

```
SELECT
p_brand,
p_type,
p_size,
```

```
count(distinct ps_suppkey) as supplier_cnt
FROM
partsupp,
part
WHERE
p_partkey = ps_partkey
and p_brand <> 'Brand#45'
and p_type not like 'MEDIUM POLISHED%'
and p_size in (49, 14, 23, 45, 19, 3, 36, 9)
and ps_suppkey not in (
SELECT
s_suppkey
FROM
supplier
WHERE
s_comment like '%Customer%Complaints%'
)
GROUP BY
p_brand,
p_type,
p_size
ORDER BY
supplier_cnt desc,
p_brand,
p_type,
p_size
LIMIT 100;
```

Q17

```
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
);
```

Q18

```
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;
```

Q19

```
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'
);
```

Q20

```
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;
```

Q21

```
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;
```

Q22

```
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 TPC-DS 性能测试

4.1 TPC-DS 测试结果

DWS 测试了使用存算一体和 存算分离两种部署架构下，TPC-DS 1T规模数据集的开箱查询性能，共99个查询。存算一体查询总耗时为417.84s，存算分离查询总耗时为445.34s。详细结果见下表。

表 4-1 TPC-DS 测试结果

TPC-DS查询 / 时间(s)	**9.1.0**		**9.1.1.100**	
-	**存算一体**	**存算分离**	**存算一体**	**存算分离**
Q1	0.495	0.537	0.498	0.531
Q2	4.098	4.259	3.877	4.227
Q3	1.224	1.397	1.137	1.393
Q4	134.21	118.603	32.966	32.537
Q5	1.426	1.66	1.222	1.323
Q6	0.759	0.762	0.744	0.575
Q7	1.729	2.446	1.79	2.238
Q8	0.474	0.47	0.425	0.47
Q9	13.166	12.471	12.228	11.984
Q10	0.963	1.035	0.719	0.697
Q11	66.002	61.731	7.918	8.255
Q12	0.135	0.133	0.132	0.144
Q13	3.717	4.176	3.901	4.308

TPC-DS查询 / 时间(s)	9.1.0		9.1.1.100	
Q14	18.118	19.244	17.14	19.311
Q15	0.441	0.457	0.636	0.65
Q16	3.828	3.763	0.648	0.648
Q17	2.298	1.456	2.698	1.573
Q18	1.571	1.868	1.382	1.978
Q19	0.712	0.781	0.709	0.755
Q20	0.105	0.092	0.087	0.087
Q21	0.08	0.082	0.073	0.074
Q22	1.046	1.458	1.025	1.246
Q23	98.08	94.936	49.471	54.571
Q24	9.076	9.581	9.937	10.422
Q25	1.261	1.375	1.333	1.409
Q26	0.535	0.725	0.437	0.449
Q27	1.941	2.44	1.87	2.401
Q28	6.501	6.962	6.615	6.867
Q29	2.175	2.464	1.793	2.71
Q30	0.401	0.393	0.404	0.366
Q31	3.094	3.227	3.285	3.462
Q32	0.092	0.088	0.072	1.02
Q33	0.926	0.905	0.925	0.975
Q34	2.794	3.13	2.557	2.938
Q35	2.352	2.62	2.857	2.406
Q36	1.815	2.156	1.706	1.989
Q37	0.323	0.361	0.347	0.303
Q38	10.249	17.642	12.393	15.065
Q39	5.188	5.971	5.077	5.628
Q40	0.124	0.105	0.103	0.092
Q41	0.032	0.036	0.039	0.036
Q42	0.594	0.639	0.355	0.448

TPC-DS查询 / 时间(s)	9.1.0		9.1.1.100	
Q43	1.336	1.436	1.188	1.283
Q44	2.192	2.324	2.154	2.337
Q45	0.741	0.392	1.184	0.964
Q46	3.717	4.505	4.044	4.767
Q47	4.623	5.435	4.654	5.433
Q48	3.438	3.947	3.225	3.553
Q49	2.369	2.513	3.218	3.649
Q50	3.82	4.508	3.618	4.327
Q51	4.453	5.22	4.196	4.79
Q52	0.614	0.676	0.411	0.439
Q53	0.696	0.779	0.737	0.732
Q54	0.813	1.989	4.091	3.051
Q55	0.575	0.634	0.368	0.351
Q56	0.716	0.793	0.701	0.792
Q57	1.616	1.848	1.631	1.826
Q58	0.505	0.453	0.478	0.64
Q59	9.106	9.012	8.87	9.401
Q60	1.03	1.276	1.12	1.183
Q61	0.997	1.031	0.979	0.888
Q62	0.758	0.886	0.746	0.758
Q63	0.709	0.784	0.706	0.724
Q64	6.501	6.513	14.638	8.058
Q65	3.576	3.484	2.965	3.649
Q66	1.066	1.108	1.031	1.033
Q67	54.072	55.735	57.879	61.84
Q68	2.935	3.493	3.152	3.673
Q69	0.674	0.9	0.895	0.766
Q70	3.033	3.255	2.884	3.018
Q71	2.046	2.101	2.197	2.005

TPC-DS查询 / 时间(s)	9.1.0		9.1.1.100	
Q72	2.559	2.745	2.539	2.686
Q73	1.777	2.197	1.885	2.486
Q74	31.74	29.214	6.703	6.548
Q75	6.259	5.922	6.457	6.373
Q76	2.388	2.86	2.805	2.902
Q77	1.142	1.306	1.054	1.108
Q78	11.931	12.695	12.077	12.586
Q79	4.2	4.541	4.511	4.789
Q80	1.659	1.848	1.528	1.621
Q81	0.374	0.393	0.322	0.659
Q82	0.619	0.657	0.701	0.707
Q83	0.083	0.173	0.088	0.194
Q84	0.416	0.425	0.201	0.261
Q85	1.177	1.803	1.023	1.326
Q86	0.35	0.38	0.328	0.39
Q87	10.879	18.168	11.613	16.286
Q88	9.09	9.643	11.071	11.746
Q89	0.971	1.035	0.887	0.956
Q90	0.763	0.858	0.745	0.722
Q91	0.147	0.109	0.14	0.304
Q92	0.117	0.132	0.109	0.131
Q93	3.973	5.176	3.768	4.44
Q94	2.134	2.385	1.077	2.358
Q95	32.74	28.579	10.887	10.968
Q96	1.788	2.102	2.02	2.191
Q97	3.452	4.097	3.486	3.925
Q98	0.828	0.826	0.871	0.8
Q99	1.634	1.704	1.449	1.388
总时长(s)	**658.067**	**663.64**	**417.836**	**445.342**

4.2 TPC-DS 测试环境

硬件环境

每个测试环境6个节点，配置如下：

- CPU 16核：Intel Ice Lake
- 内存：64GB
- 网络带宽：9Gbit/s
- 磁盘：SSD云盘，每块600GB，共2块

软件环境

- 内核版本：Linux 3.10.0-862.14.1.5.h757.eulerosv2r7.x86_64
- 操作系统：EulerOS release 2.0 (SP5)
- 数据库版本：DWS 3.0

4.3 TPC-DS 测试过程

4.3.1 TPC-DS 测试数据

表 4-2 TPC-DS 测试数据

序号	表名	行数	表大小
1	customer_address	6,000,000	126MB
2	customer_demographics	1,920,800	11MB
3	date_dim	73,049	11MB
4	warehouse	20	1200KB
5	ship_mode	20	864KB
6	time_dim	86,400	1520KB
7	reason	65	720KB
8	income_band	20	720KB
9	item	300,000	23MB
10	store	1,002	2400KB
11	call_center	42	1968KB
12	customer	12,000,000	519MB

序号	表名	行数	表大小
13	web_site	54	1824KB
14	household_demographics	7,200	1208KB
15	web_page	3,000	2208KB
16	promotion	1,500	2112KB
17	catalog_page	30,000	3536KB
18	inventory	783,000,000	2499MB
19	catalog_returns	143,996,756	8454MB
20	web_returns	71,997,522	3990MB
21	store_returns	287,999,764	13GB
22	web_sales	720,000,376	54GB
23	catalog_sales	1,439,980,416	104GB
24	store_sales	2,879,987,999	142GB

4.3.2 TPC-DS 数据生成

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。

```
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

4.3.3 建表与导入 TPC-DS 数据

创建 TPC-DS 目标表

连接DWS数据库后执行以下SQL语句。

```
CREATE TABLE customer_address
(
    ca_address_sk             integer               not null,
    ca_address_id             varchar(16)           not null,
    ca_street_number          varchar(10)                   ,
    ca_street_name            varchar(60)                   ,
    ca_street_type            varchar(15)                   ,
    ca_suite_number           varchar(10)                   ,
    ca_city                   varchar(60)                   ,
    ca_county                 varchar(30)                   ,
    ca_state                  varchar(2)                    ,
    ca_zip                    varchar(10)                   ,
    ca_country                varchar(20)                   ,
    ca_gmt_offset             decimal(5,2)                  ,
    ca_location_type          varchar(20)
) WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384,secondary_part_column='ca_address_sk',secondary_part_num=72)
DISTRIBUTE BY hash (ca_address_sk);

CREATE TABLE customer_demographics
(
    cd_demo_sk                integer               not null,
    cd_gender                 varchar(1)                    ,
    cd_marital_status         varchar(1)                    ,
    cd_education_status       varchar(20)                   ,
    cd_purchase_estimate      integer                       ,
    cd_credit_rating          varchar(10)                   ,
    cd_dep_count              integer                       ,
    cd_dep_employed_count     integer                       ,
    cd_dep_college_count      integer
) WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384,secondary_part_column='cd_demo_sk',secondary_part_num=72)
DISTRIBUTE BY hash (cd_demo_sk);

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

```
    d_qoy                     integer                 ,
    d_fy_year                 integer                 ,
    d_fy_quarter_seq          integer                 ,
    d_fy_week_seq             integer                 ,
    d_day_name                varchar(9)              ,
    d_quarter_name            varchar(6)              ,
    d_holiday                 varchar(1)              ,
    d_weekend                 varchar(1)              ,
    d_following_holiday       varchar(1)              ,
    d_first_dom               integer                 ,
    d_last_dom                integer                 ,
    d_same_day_ly             integer                 ,
    d_same_day_lq             integer                 ,
    d_current_day             varchar(1)              ,
    d_current_week            varchar(1)              ,
    d_current_month           varchar(1)              ,
    d_current_quarter         varchar(1)              ,
    d_current_year            varchar(1)
) WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384,secondary_part_column='d_da
te_sk',secondary_part_num=72)
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            integer               not null,
    w_warehouse_id            varchar(16)           not null,
    w_warehouse_name          varchar(20)           ,
    w_warehouse_sq_ft         integer               ,
    w_street_number           varchar(10)           ,
    w_street_name             varchar(60)           ,
    w_street_type             varchar(15)           ,
    w_suite_number            varchar(10)           ,
    w_city                    varchar(60)           ,
    w_county                  varchar(30)           ,
    w_state                   varchar(2)            ,
    w_zip                     varchar(10)           ,
    w_country                 varchar(20)           ,
    w_gmt_offset              decimal(5,2)
)
WITH (orientation = column, colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384)
DISTRIBUTE BY replication;

CREATE TABLE ship_mode
(
    sm_ship_mode_sk           integer               not null,
    sm_ship_mode_id           varchar(16)           not null,
    sm_type                   varchar(30)           ,
    sm_code                   varchar(10)           ,
    sm_carrier                varchar(20)           ,
    sm_contract               varchar(20)
)
 WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384)
DISTRIBUTE BY replication;

CREATE TABLE time_dim
(
    t_time_sk                 integer               not null,
    t_time_id                 varchar(16)           not null,
    t_time                    integer               ,
```

```
    t_hour                    integer                     ,
    t_minute                  integer                     ,
    t_second                  integer                     ,
    t_am_pm                   varchar(2)                  ,
    t_shift                   varchar(20)                 ,
    t_sub_shift               varchar(20)                 ,
    t_meal_time               varchar(20)
)
 WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384,secondary_part_column='t_time_sk',secondary_part_num=72)
DISTRIBUTE BY hash (t_time_sk);

CREATE TABLE reason
(
    r_reason_sk               integer               not null,
    r_reason_id               varchar(16)           not null,
    r_reason_desc             varchar(100)
)
 WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384)
DISTRIBUTE BY replication;

CREATE TABLE income_band
(
    ib_income_band_sk         integer               not null,
    ib_lower_bound            integer                     ,
    ib_upper_bound            integer
)
 WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384)
DISTRIBUTE BY replication;

CREATE TABLE item
(
    i_item_sk                 integer               not null,
    i_item_id                 varchar(16)           not null,
    i_rec_start_date          date                        ,
    i_rec_end_date            date                        ,
    i_item_desc               varchar(200)                ,
    i_current_price           decimal(7,2)                ,
    i_wholesale_cost          decimal(7,2)                ,
    i_brand_id                integer                     ,
    i_brand                   varchar(50)                 ,
    i_class_id                integer                     ,
    i_class                   varchar(50)                 ,
    i_category_id             integer                     ,
    i_category                varchar(50)                 ,
    i_manufact_id             integer                     ,
    i_manufact                varchar(50)                 ,
    i_size                    varchar(20)                 ,
    i_formulation             varchar(20)                 ,
    i_color                   varchar(20)                 ,
    i_units                   varchar(10)                 ,
    i_container               varchar(10)                 ,
    i_manager_id              integer                     ,
    i_product_name            varchar(50)
)
 WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384,secondary_part_column='i_item_sk',secondary_part_num=72)
DISTRIBUTE BY hash (i_item_sk);

CREATE TABLE store
(
    s_store_sk                integer               not null,
    s_store_id                varchar(16)           not null,
    s_rec_start_date          date                        ,
    s_rec_end_date            date                        ,
```

```
    s_closed_date_sk          integer                   ,
    s_store_name              varchar(50)               ,
    s_number_employees        integer                   ,
    s_floor_space             integer                   ,
    s_hours                   varchar(20)               ,
    s_manager                 varchar(40)               ,
    s_market_id               integer                   ,
    s_geography_class         varchar(100)              ,
    s_market_desc             varchar(100)              ,
    s_market_manager          varchar(40)               ,
    s_division_id             integer                   ,
    s_division_name           varchar(50)               ,
    s_company_id              integer                   ,
    s_company_name            varchar(50)               ,
    s_street_number           varchar(10)               ,
    s_street_name             varchar(60)               ,
    s_street_type             varchar(15)               ,
    s_suite_number            varchar(10)               ,
    s_city                    varchar(60)               ,
    s_county                  varchar(30)               ,
    s_state                   varchar(2)                ,
    s_zip                     varchar(10)               ,
    s_country                 varchar(20)               ,
    s_gmt_offset              decimal(5,2)              ,
    s_tax_percentage          decimal(5,2)
)
 WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384)
DISTRIBUTE BY replication;

CREATE TABLE call_center
(
    cc_call_center_sk         integer               not null,
    cc_call_center_id         varchar(16)           not null,
    cc_rec_start_date         date                  ,
    cc_rec_end_date           date                  ,
    cc_closed_date_sk         integer               ,
    cc_open_date_sk           integer               ,
    cc_name                   varchar(50)           ,
    cc_class                  varchar(50)           ,
    cc_employees              integer               ,
    cc_sq_ft                  integer               ,
    cc_hours                  varchar(20)           ,
    cc_manager                varchar(40)           ,
    cc_mkt_id                 integer               ,
    cc_mkt_class              varchar(50)           ,
    cc_mkt_desc               varchar(100)          ,
    cc_market_manager         varchar(40)           ,
    cc_division               integer               ,
    cc_division_name          varchar(50)           ,
    cc_company                integer               ,
    cc_company_name           varchar(50)           ,
    cc_street_number          varchar(10)           ,
    cc_street_name            varchar(60)           ,
    cc_street_type            varchar(15)           ,
    cc_suite_number           varchar(10)           ,
    cc_city                   varchar(60)           ,
    cc_county                 varchar(30)           ,
    cc_state                  varchar(2)            ,
    cc_zip                    varchar(10)           ,
    cc_country                varchar(20)           ,
    cc_gmt_offset             decimal(5,2)          ,
    cc_tax_percentage         decimal(5,2)
)
 WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384)
DISTRIBUTE BY replication;

CREATE TABLE customer
```

```
(
    c_customer_sk             integer               not null,
    c_customer_id             varchar(16)           not null,
    c_current_cdemo_sk        integer                       ,
    c_current_hdemo_sk        integer                       ,
    c_current_addr_sk         integer                       ,
    c_first_shipto_date_sk    integer                       ,
    c_first_sales_date_sk     integer                       ,
    c_salutation              varchar(10)                   ,
    c_first_name              varchar(20)                   ,
    c_last_name               varchar(30)                   ,
    c_preferred_cust_flag     varchar(1)                    ,
    c_birth_day               integer                       ,
    c_birth_month             integer                       ,
    c_birth_year              integer                       ,
    c_birth_country           varchar(20)                   ,
    c_login                   varchar(13)                   ,
    c_email_address           varchar(50)                   ,
    c_last_review_date        varchar(10)
)
 WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384,secondary_part_column='c_customer_sk',secondary_part_num=72)
DISTRIBUTE BY hash (c_customer_sk);

CREATE TABLE web_site
(
    web_site_sk               integer               not null,
    web_site_id               varchar(16)           not null,
    web_rec_start_date        date                          ,
    web_rec_end_date          date                          ,
    web_name                  varchar(50)                   ,
    web_open_date_sk          integer                       ,
    web_close_date_sk         integer                       ,
    web_class                 varchar(50)                   ,
    web_manager               varchar(40)                   ,
    web_mkt_id                integer                       ,
    web_mkt_class             varchar(50)                   ,
    web_mkt_desc              varchar(100)                  ,
    web_market_manager        varchar(40)                   ,
    web_company_id            integer                       ,
    web_company_name          varchar(50)                   ,
    web_street_number         varchar(10)                   ,
    web_street_name           varchar(60)                   ,
    web_street_type           varchar(15)                   ,
    web_suite_number          varchar(10)                   ,
    web_city                  varchar(60)                   ,
    web_county                varchar(30)                   ,
    web_state                 varchar(2)                    ,
    web_zip                   varchar(10)                   ,
    web_country               varchar(20)                   ,
    web_gmt_offset            decimal(5,2)                  ,
    web_tax_percentage        decimal(5,2)
)
 WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384)
DISTRIBUTE BY replication;

CREATE TABLE household_demographics
(
    hd_demo_sk                integer               not null,
    hd_income_band_sk         integer                       ,
    hd_buy_potential          varchar(15)                   ,
    hd_dep_count              integer                       ,
    hd_vehicle_count          integer
)
 WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384,secondary_part_column='hd_demo_sk',secondary_part_num=72)
```

```
DISTRIBUTE BY hash (hd_demo_sk);

CREATE TABLE web_page
(
    wp_web_page_sk            integer               not null,
    wp_web_page_id            varchar(16)               not null,
    wp_rec_start_date         date                      ,
    wp_rec_end_date           date                      ,
    wp_creation_date_sk       integer                     ,
    wp_access_date_sk         integer                   ,
    wp_autogen_flag           varchar(1)                    ,
    wp_customer_sk            integer                   ,
    wp_url                    varchar(100)              ,
    wp_type                   varchar(50)                 ,
    wp_char_count             integer                   ,
    wp_link_count             integer                   ,
    wp_image_count            integer                     ,
    wp_max_ad_count           integer
)
 WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384)
DISTRIBUTE BY replication;

CREATE TABLE promotion
(
    p_promo_sk                integer               not null,
    p_promo_id                varchar(16)               not null,
    p_start_date_sk           integer                     ,
    p_end_date_sk             integer                   ,
    p_item_sk                 integer                   ,
    p_cost                    decimal(15,2)               ,
    p_response_target         integer                     ,
    p_promo_name              varchar(50)                     ,
    p_channel_dmail           varchar(1)                  ,
    p_channel_email           varchar(1)                  ,
    p_channel_catalog         varchar(1)                  ,
    p_channel_tv              varchar(1)                  ,
    p_channel_radio           varchar(1)                  ,
    p_channel_press           varchar(1)                  ,
    p_channel_event           varchar(1)                  ,
    p_channel_demo            varchar(1)                  ,
    p_channel_details         varchar(100)                ,
    p_purpose                 varchar(15)                 ,
    p_discount_active         varchar(1)
)
 WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384)
DISTRIBUTE BY replication;

CREATE TABLE catalog_page
(
    cp_catalog_page_sk        integer               not null,
    cp_catalog_page_id        varchar(16)               not null,
    cp_start_date_sk          integer                   ,
    cp_end_date_sk            integer                   ,
    cp_department             varchar(50)                 ,
    cp_catalog_number         integer                   ,
    cp_catalog_page_number    integer                       ,
    cp_description            varchar(100)                ,
    cp_type                   varchar(100)
)
 WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384,secondary_part_column='cp_c
atalog_page_sk',secondary_part_num=72)
DISTRIBUTE BY hash (cp_catalog_page_sk);

CREATE TABLE inventory
(
    inv_date_sk               integer               not null,
```

```
    inv_item_sk               integer               not null,
    inv_warehouse_sk          integer               not null,
    inv_quantity_on_hand      integer
)
 WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384,secondary_part_column='inv_i
tem_sk',secondary_part_num=72)
DISTRIBUTE BY hash (inv_item_sk)
partition by range(inv_date_sk)
(
    partition p1 values less than(2451179),
    partition p2 values less than(2451544),
    partition p3 values less than(2451910),
    partition p4 values less than(2452275),
    partition p5 values less than(2452640),
    partition p6 values less than(2453005),
    partition p7 values less than(maxvalue)
)
;

CREATE TABLE catalog_returns
(
    cr_returned_date_sk       integer                     ,
    cr_returned_time_sk       integer                     ,
    cr_item_sk                integer               not null,
    cr_refunded_customer_sk   integer                       ,
    cr_refunded_cdemo_sk      integer                       ,
    cr_refunded_hdemo_sk      integer                       ,
    cr_refunded_addr_sk       integer                      ,
    cr_returning_customer_sk  integer                       ,
    cr_returning_cdemo_sk     integer                       ,
    cr_returning_hdemo_sk     integer                       ,
    cr_returning_addr_sk      integer                      ,
    cr_call_center_sk         integer                     ,
    cr_catalog_page_sk        integer                       ,
    cr_ship_mode_sk           integer                       ,
    cr_warehouse_sk           integer                       ,
    cr_reason_sk              integer                     ,
    cr_order_number           bigint                not null,
    cr_return_quantity        integer                       ,
    cr_return_amount          decimal(7,2)                  ,
    cr_return_tax             decimal(7,2)                  ,
    cr_return_amt_inc_tax     decimal(7,2)                    ,
    cr_fee                    decimal(7,2)                  ,
    cr_return_ship_cost       decimal(7,2)                    ,
    cr_refunded_cash          decimal(7,2)                    ,
    cr_reversed_charge        decimal(7,2)                    ,
    cr_store_credit           decimal(7,2)                  ,
    cr_net_loss               decimal(7,2)
)
 WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384,secondary_part_column='cr_it
em_sk',secondary_part_num=72)
DISTRIBUTE BY hash (cr_item_sk)
partition by range(cr_returned_date_sk)
(
    partition p1 values less than(2450815),
    partition p2 values less than(2451179),
    partition p3 values less than(2451544),
    partition p4 values less than(2451910),
    partition p5 values less than(2452275),
    partition p6 values less than(2452640),
    partition p7 values less than(2453005),
    partition p8 values less than(maxvalue)
)
;

CREATE TABLE web_returns
(
```

```
    wr_returned_date_sk       integer                       ,
    wr_returned_time_sk       integer                       ,
    wr_item_sk                integer               not null,
    wr_refunded_customer_sk   integer                       ,
    wr_refunded_cdemo_sk      integer                       ,
    wr_refunded_hdemo_sk      integer                       ,
    wr_refunded_addr_sk       integer                       ,
    wr_returning_customer_sk  integer                       ,
    wr_returning_cdemo_sk     integer                       ,
    wr_returning_hdemo_sk     integer                       ,
    wr_returning_addr_sk      integer                       ,
    wr_web_page_sk            integer                       ,
    wr_reason_sk              integer                       ,
    wr_order_number           bigint                not null,
    wr_return_quantity        integer                       ,
    wr_return_amt             decimal(7,2)                  ,
    wr_return_tax             decimal(7,2)                  ,
    wr_return_amt_inc_tax     decimal(7,2)                  ,
    wr_fee                    decimal(7,2)                  ,
    wr_return_ship_cost       decimal(7,2)                  ,
    wr_refunded_cash          decimal(7,2)                  ,
    wr_reversed_charge        decimal(7,2)                  ,
    wr_account_credit         decimal(7,2)                  ,
    wr_net_loss               decimal(7,2)
)
 WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384,secondary_part_column='wr_it
em_sk',secondary_part_num=72)
DISTRIBUTE BY hash (wr_item_sk)
partition by range(wr_returned_date_sk)
(
    partition p1 values less than(2450815),
    partition p2 values less than(2451179),
    partition p3 values less than(2451544),
    partition p4 values less than(2451910),
    partition p5 values less than(2452275),
    partition p6 values less than(2452640),
    partition p7 values less than(2453005),
    partition p8 values less than(maxvalue)
)
;

CREATE TABLE store_returns
(
    sr_returned_date_sk       integer                       ,
    sr_return_time_sk         integer                       ,
    sr_item_sk                integer               not null,
    sr_customer_sk            integer                       ,
    sr_cdemo_sk               integer                       ,
    sr_hdemo_sk               integer                       ,
    sr_addr_sk                integer                       ,
    sr_store_sk               integer                       ,
    sr_reason_sk              integer                       ,
    sr_ticket_number          bigint                not null,
    sr_return_quantity        integer                       ,
    sr_return_amt             decimal(7,2)                  ,
    sr_return_tax             decimal(7,2)                  ,
    sr_return_amt_inc_tax     decimal(7,2)                  ,
    sr_fee                    decimal(7,2)                  ,
    sr_return_ship_cost       decimal(7,2)                  ,
    sr_refunded_cash          decimal(7,2)                  ,
    sr_reversed_charge        decimal(7,2)                  ,
    sr_store_credit           decimal(7,2)                  ,
    sr_net_loss               decimal(7,2)
)
WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384,secondary_part_column='sr_it
em_sk',secondary_part_num=72)
DISTRIBUTE BY hash (sr_item_sk)
```

```
partition by range(sr_returned_date_sk)
(
    partition p1 values less than (2451179) ,
    partition p2 values less than (2451544) ,
    partition p3 values less than (2451910) ,
    partition p4 values less than (2452275) ,
    partition p5 values less than (2452640) ,
    partition p6 values less than (2453005) ,
    partition p7 values less than (maxvalue)
)
;

CREATE TABLE web_sales
(
    ws_sold_date_sk           integer                       ,
    ws_sold_time_sk           integer                       ,
    ws_ship_date_sk           integer                       ,
    ws_item_sk                integer               not null,
    ws_bill_customer_sk       integer                       ,
    ws_bill_cdemo_sk          integer                       ,
    ws_bill_hdemo_sk          integer                       ,
    ws_bill_addr_sk           integer                       ,
    ws_ship_customer_sk       integer                       ,
    ws_ship_cdemo_sk          integer                       ,
    ws_ship_hdemo_sk          integer                       ,
    ws_ship_addr_sk           integer                       ,
    ws_web_page_sk            integer                       ,
    ws_web_site_sk            integer                       ,
    ws_ship_mode_sk           integer                       ,
    ws_warehouse_sk           integer                       ,
    ws_promo_sk               integer                       ,
    ws_order_number           bigint                not null,
    ws_quantity               integer                       ,
    ws_wholesale_cost         decimal(7,2)                  ,
    ws_list_price             decimal(7,2)                  ,
    ws_sales_price            decimal(7,2)                  ,
    ws_ext_discount_amt       decimal(7,2)                  ,
    ws_ext_sales_price        decimal(7,2)                  ,
    ws_ext_wholesale_cost     decimal(7,2)                  ,
    ws_ext_list_price         decimal(7,2)                  ,
    ws_ext_tax                decimal(7,2)                  ,
    ws_coupon_amt             decimal(7,2)                  ,
    ws_ext_ship_cost          decimal(7,2)                  ,
    ws_net_paid               decimal(7,2)                  ,
    ws_net_paid_inc_tax       decimal(7,2)                  ,
    ws_net_paid_inc_ship      decimal(7,2)                  ,
    ws_net_paid_inc_ship_tax  decimal(7,2)                  ,
    ws_net_profit             decimal(7,2)
)
 WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384,secondary_part_column='ws_i
tem_sk',secondary_part_num=72)
DISTRIBUTE BY hash (ws_item_sk)
partition by range(ws_sold_date_sk)
(
    partition p1 values less than(2451179),
    partition p2 values less than(2451544),
    partition p3 values less than(2451910),
    partition p4 values less than(2452275),
    partition p5 values less than(2452640),
    partition p6 values less than(2453005),
    partition p7 values less than(maxvalue)
)
;

CREATE TABLE catalog_sales
(
    cs_sold_date_sk           integer                       ,
    cs_sold_time_sk           integer                       ,
```

```
    cs_ship_date_sk           integer                       ,
    cs_bill_customer_sk       integer                       ,
    cs_bill_cdemo_sk          integer                       ,
    cs_bill_hdemo_sk          integer                       ,
    cs_bill_addr_sk           integer                       ,
    cs_ship_customer_sk       integer                       ,
    cs_ship_cdemo_sk          integer                       ,
    cs_ship_hdemo_sk          integer                       ,
    cs_ship_addr_sk           integer                       ,
    cs_call_center_sk         integer                       ,
    cs_catalog_page_sk        integer                       ,
    cs_ship_mode_sk           integer                       ,
    cs_warehouse_sk           integer                       ,
    cs_item_sk                integer               not null,
    cs_promo_sk               integer                       ,
    cs_order_number           bigint                not null,
    cs_quantity               integer                       ,
    cs_wholesale_cost         decimal(7,2)                  ,
    cs_list_price             decimal(7,2)                  ,
    cs_sales_price            decimal(7,2)                  ,
    cs_ext_discount_amt       decimal(7,2)                  ,
    cs_ext_sales_price        decimal(7,2)                  ,
    cs_ext_wholesale_cost     decimal(7,2)                  ,
    cs_ext_list_price         decimal(7,2)                  ,
    cs_ext_tax                decimal(7,2)                  ,
    cs_coupon_amt             decimal(7,2)                  ,
    cs_ext_ship_cost          decimal(7,2)                  ,
    cs_net_paid               decimal(7,2)                  ,
    cs_net_paid_inc_tax       decimal(7,2)                  ,
    cs_net_paid_inc_ship      decimal(7,2)                  ,
    cs_net_paid_inc_ship_tax  decimal(7,2)                  ,
    cs_net_profit             decimal(7,2)
)
 WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384,secondary_part_column='cs_it
em_sk',secondary_part_num=72)
DISTRIBUTE BY hash (cs_item_sk)
partition by range(cs_sold_date_sk)
(
    partition p1 values less than(2451179),
    partition p2 values less than(2451544),
    partition p3 values less than(2451910),
    partition p4 values less than(2452275),
    partition p5 values less than(2452640),
    partition p6 values less than(2453005),
    partition p7 values less than(maxvalue)
)
;

CREATE TABLE store_sales
(
    ss_sold_date_sk           integer                       ,
    ss_sold_time_sk           integer                       ,
    ss_item_sk                integer               not null,
    ss_customer_sk            integer                       ,
    ss_cdemo_sk               integer                       ,
    ss_hdemo_sk               integer                       ,
    ss_addr_sk                integer                       ,
    ss_store_sk               integer                       ,
    ss_promo_sk               integer                       ,
    ss_ticket_number          bigint                not null,
    ss_quantity               integer                       ,
    ss_wholesale_cost         decimal(7,2)                  ,
    ss_list_price             decimal(7,2)                  ,
    ss_sales_price            decimal(7,2)                  ,
    ss_ext_discount_amt       decimal(7,2)                  ,
    ss_ext_sales_price        decimal(7,2)                  ,
    ss_ext_wholesale_cost     decimal(7,2)                  ,
    ss_ext_list_price         decimal(7,2)                  ,
```

```
    ss_ext_tax              decimal(7,2)              ,
    ss_coupon_amt           decimal(7,2)              ,
    ss_net_paid             decimal(7,2)              ,
    ss_net_paid_inc_tax     decimal(7,2)              ,
    ss_net_profit           decimal(7,2)
)
 WITH (orientation = column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,bucketnums=16384,secondary_part_column='ss_item_sk',secondary_part_num=72)
DISTRIBUTE BY hash (ss_item_sk)
partition by range(ss_sold_date_sk)
(
    partition p1 values less than(2451179),
    partition p2 values less than(2451544),
    partition p3 values less than(2451910),
    partition p4 values less than(2452275),
    partition p5 values less than(2452640),
    partition p6 values less than(2453005),
    partition p7 values less than(maxvalue)
)
;
```

安装和启动 GDS

步骤1 参见**工具下载**下载GDS客户端（与gsql客户端在一个包）。

步骤2 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3 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4 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5 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6 切换到gds_user用户。

```
su - gds_user
```

步骤7 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8 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
/opt/gds/bin/gds -d /data1/script/ssb-kit/ssb100X/ -p 192.168.0.90:5004 -H 192.168.0.0/24 -l /opt/gds/gds05_log.txt -D    #SSB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DWS的集群内网网段（即GDS所在的ECS与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

创建 TPC-DS 数据集的 GDS 外表

连接DWS数据库后执行以下SQL语句。

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*.dat | gsfs://192.168.0.90:5003/
customer_address*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
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*.dat | gsfs://192.168.0.90:5003/
customer_demographics*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
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*.dat | gsfs://192.168.0.90:5003/date_dim*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
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*.dat | gsfs://192.168.0.90:5003/warehouse*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
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*.dat | gsfs://192.168.0.90:5003/ship_mode*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
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*.dat | gsfs://192.168.0.90:5003/time_dim*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
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*.dat | gsfs://192.168.0.90:5003/reason*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
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*.dat | gsfs://192.168.0.90:5003/income_band*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
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*.dat | gsfs://192.168.0.90:5003/item*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
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_percentage          decimal(5,2)
)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5002/store_[0-9]*.dat | gsfs://192.168.0.90:5003/store_[0-9]*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
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*.dat | gsfs://192.168.0.90:5003/call_center*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
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_[0-9]*.dat | gsfs://192.168.0.90:5003/
customer_[0-9]*.dat',
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*.dat | gsfs://192.168.0.90:5003/web_site*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
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*.dat | gsfs://192.168.0.90:5003/store_returns*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
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*.dat | gsfs://192.168.0.90:5003/
household_demographics*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
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*.dat | gsfs://192.168.0.90:5003/web_page*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
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*.dat | gsfs://192.168.0.90:5003/promotion*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
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*.dat | gsfs://192.168.0.90:5003/catalog_page*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
```

```
encoding 'utf8',
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*.dat | gsfs://192.168.0.90:5003/inventory*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
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*.dat | gsfs://192.168.0.90:5003/
catalog_returns*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
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*.dat | gsfs://192.168.0.90:5003/web_returns*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
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*.dat | gsfs://192.168.0.90:5003/web_sales*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
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*.dat | gsfs://192.168.0.90:5003/catalog_sales*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
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*.dat | gsfs://192.168.0.90:5003/store_sales*.dat',
FORMAT 'TEXT' ,
DELIMITER '|',
encoding 'utf8',
mode 'Normal'
) WITH store_sales_err;
```

导入 TPC-DS 数据

执行以下命令导入数据。

```
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;
```

4.3.4 TPC-DS 查询测试

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DWS可用的SQL语句：

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

5 SSB 性能测试

5.1 SSB 测试结果

经过针对SSB 宽表场景对DWS和某开源OLAP产品ClickHouse的对比测试发现：

- 使用hstore_opt表，配合turbo存储、turbo引擎， DWS查询性能整体优于开源产品ClickHouse 2倍。
- DWS开箱性能相比9.1.0.100版本提升了1.3倍。

表 5-1 SSB 测试结果

SSB	DWS	ClickHouse
Q1.1	0.074	0.059
Q1.2	0.039	0.021
Q1.3	0.102	0.022
Q2.1	0.052	0.254
Q2.2	0.236	0.281
Q2.3	0.064	0.214
Q3.1	0.101	0.434
Q3.2	0.049	0.348
Q3.3	0.032	0.299
Q3.4	0.010	0.025
Q4.1	0.085	0.456
Q4.2	0.047	0.171
Q4.3	0.023	0.146
总时长（s）	**0.913**	**2.73**

5.2 SSB 测试环境

硬件环境

每个测试环境6个节点，配置如下：

- CPU 16核：Intel Ice Lake
- 内存：64GB
- 网络带宽：9Gbit/s
- 磁盘：SSD云盘，每块600GB，共2块

软件环境

内核版本：Linux 3.10.0-862.14.1.5.h757.eulerosv2r7.x86_64

操作系统：EulerOS release 2.0 (SP5)

数据库版本：DWS 3.0

5.3 SSB 测试过程

5.3.1 SSB 测试数据

表 5-2 SSB 测试数据

序号	表名	行数	表大小
1	supplier	200000	-
2	customer	3000000	-
3	part	1400000	-
4	lineorder	60037902	-
5	lineorder_flat	60037902	-

5.3.2 SSB 数据生成

步骤1 下载ssb工具包并编译。

```
git clone http://github.com/vadimtk/ssb-dbgen.git
cd ssb-dbgen && make
```

步骤2 生成数据。

说明

文件生成路径最好符合**安装和启动GDS**中SSB所使用的路径，否则需要修改**安装和启动GDS**中GDS的启动路径。

```
./dbgen -s 100 -T c
./dbgen -s 100 -T l
./dbgen -s 100 -T p
./dbgen -s 100 -T s
./dbgen -s 100 -T d
```

----结束

5.3.3 建表与导入 SSB 数据

创建 SSB 目标表

连接DWS数据库后执行以下SQL语句。

```
CREATE TABLE CUSTOMER
(
C_CUSTKEY     BIGINT NOT NULL,
C_NAME        VARCHAR(25) NOT NULL,
C_ADDRESS     VARCHAR(40) NOT NULL,
C_CITY        VARCHAR(25) NOT NULL,
C_NATION      VARCHAR(25) NOT NULL,
C_REGION      VARCHAR(25) NOT NULL,
C_PHONE       VARCHAR(15) NOT NULL,
C_MKTSEGMENT  VARCHAR(10) NOT NULL
)
WITH (orientation=column, colversion=2.0,enable_hstore=true,enable_hstore_opt=true)
DISTRIBUTE BY hash(C_CUSTKEY) ;
CREATE TABLE SUPPLIER
(
S_SUPPKEY  BIGINT NOT NULL
, S_NAME  VARCHAR(25) NOT NULL
, S_ADDRESS  VARCHAR(40) NOT NULL
, S_CITY        VARCHAR(25) NOT NULL
, S_NATION      VARCHAR(25) NOT NULL
, S_REGION      VARCHAR(25) NOT NULL
, S_PHONE VARCHAR(15) NOT NULL
)
WITH (orientation=column, colversion=2.0,enable_hstore=true,enable_hstore_opt=true)
DISTRIBUTE BY hash(S_SUPPKEY);
CREATE TABLE PART
(
P_PARTKEY  BIGINT NOT NULL
, P_NAME  VARCHAR(55) NOT NULL
, P_MFGR  VARCHAR(25) NOT NULL
, P_CATEGORY   VARCHAR(25) NOT NULL
, P_BRAND VARCHAR(10) NOT NULL
, P_COLOR VARCHAR(20) NOT NULL
, P_TYPE  VARCHAR(25) NOT NULL
, P_SIZE  BIGINT NOT NULL
, P_CONTAINER   VARCHAR(10) NOT NULL
)
WITH (orientation=column, colversion=2.0,enable_hstore=true,enable_hstore_opt=true)
DISTRIBUTE BY hash(P_PARTKEY);
CREATE TABLE lineorder
(
LO_ORDERKEY             BIGINT NOT NULL,
LO_LINENUMBER           BIGINT NOT NULL,
LO_CUSTKEY              BIGINT NOT NULL,
LO_PARTKEY              BIGINT NOT NULL,
LO_SUPPKEY              BIGINT NOT NULL,
LO_ORDERDATE            DATE NOT NULL,
LO_ORDERPRIORITY        VARCHAR(15) NOT NULL,
```

```
LO_SHIPPRIORITY         BIGINT NOT NULL,
LO_QUANTITY             BIGINT NOT NULL,
LO_EXTENDEDPRICE        BIGINT NOT NULL,
LO_ORDTOTALPRICE        BIGINT NOT NULL,
LO_DISCOUNT             BIGINT NOT NULL,
LO_REVENUE              BIGINT NOT NULL,
LO_SUPPLYCOST           BIGINT NOT NULL,
LO_TAX                  BIGINT NOT NULL,
LO_COMMITDATE           DATE NOT NULL,
LO_SHIPMODE             VARCHAR(10) NOT NULL
)
WITH (orientation=column, colversion=2.0,enable_hstore=true,enable_hstore_opt=true)
DISTRIBUTE BY hash(LO_ORDERKEY)
PARTITION BY RANGE(LO_ORDERDATE)
(
PARTITION LO_ORDERDATE_1 VALUES LESS THAN('1992-04-01 00:00:00'),
PARTITION LO_ORDERDATE_2 VALUES LESS THAN('1992-07-01 00:00:00'),
PARTITION LO_ORDERDATE_3 VALUES LESS THAN('1992-10-01 00:00:00'),
PARTITION LO_ORDERDATE_4 VALUES LESS THAN('1993-01-01 00:00:00'),
PARTITION LO_ORDERDATE_5 VALUES LESS THAN('1993-04-01 00:00:00'),
PARTITION LO_ORDERDATE_6 VALUES LESS THAN('1993-07-01 00:00:00'),
PARTITION LO_ORDERDATE_7 VALUES LESS THAN('1993-10-01 00:00:00'),
PARTITION LO_ORDERDATE_8 VALUES LESS THAN('1994-01-01 00:00:00'),
PARTITION LO_ORDERDATE_9 VALUES LESS THAN('1994-04-01 00:00:00'),
PARTITION LO_ORDERDATE_10 VALUES LESS THAN('1994-07-01 00:00:00'),
PARTITION LO_ORDERDATE_11 VALUES LESS THAN('1994-10-01 00:00:00'),
PARTITION LO_ORDERDATE_12 VALUES LESS THAN('1995-01-01 00:00:00'),
PARTITION LO_ORDERDATE_13 VALUES LESS THAN('1995-04-01 00:00:00'),
PARTITION LO_ORDERDATE_14 VALUES LESS THAN('1995-07-01 00:00:00'),
PARTITION LO_ORDERDATE_15 VALUES LESS THAN('1995-10-01 00:00:00'),
PARTITION LO_ORDERDATE_16 VALUES LESS THAN('1996-01-01 00:00:00'),
PARTITION LO_ORDERDATE_17 VALUES LESS THAN('1996-04-01 00:00:00'),
PARTITION LO_ORDERDATE_18 VALUES LESS THAN('1996-07-01 00:00:00'),
PARTITION LO_ORDERDATE_19 VALUES LESS THAN('1996-10-01 00:00:00'),
PARTITION LO_ORDERDATE_20 VALUES LESS THAN('1997-01-01 00:00:00'),
PARTITION LO_ORDERDATE_21 VALUES LESS THAN('1997-04-01 00:00:00'),
PARTITION LO_ORDERDATE_22 VALUES LESS THAN('1997-07-01 00:00:00'),
PARTITION LO_ORDERDATE_23 VALUES LESS THAN('1997-10-01 00:00:00'),
PARTITION LO_ORDERDATE_24 VALUES LESS THAN('1998-01-01 00:00:00'),
PARTITION LO_ORDERDATE_25 VALUES LESS THAN('1998-04-01 00:00:00'),
PARTITION LO_ORDERDATE_26 VALUES LESS THAN('1998-07-01 00:00:00'),
PARTITION LO_ORDERDATE_27 VALUES LESS THAN('1998-10-01 00:00:00'),
PARTITION LO_ORDERDATE_28 VALUES LESS THAN('1999-01-01 00:00:00')
);
SET enable_hstoreopt_auto_bitmap=true;
CREATE TABLE lineorder_flat
(
LO_ORDERKEY             BIGINT NOT NULL,
LO_LINENUMBER           BIGINT NOT NULL,
LO_CUSTKEY              BIGINT NOT NULL,
LO_PARTKEY              BIGINT NOT NULL,
LO_SUPPKEY              BIGINT NOT NULL,
LO_ORDERDATE            DATE NOT NULL,
LO_ORDERPRIORITY        VARCHAR(15) NOT NULL,
LO_SHIPPRIORITY         BIGINT NOT NULL,
LO_QUANTITY             BIGINT NOT NULL,
LO_EXTENDEDPRICE        BIGINT NOT NULL,
LO_ORDTOTALPRICE        BIGINT NOT NULL,
LO_DISCOUNT             BIGINT NOT NULL,
LO_REVENUE              BIGINT NOT NULL,
LO_SUPPLYCOST           BIGINT NOT NULL,
LO_TAX                  BIGINT NOT NULL,
LO_COMMITDATE           DATE NOT NULL,
LO_SHIPMODE             VARCHAR(10) NOT NULL,
C_NAME        VARCHAR(25) NOT NULL
, C_ADDRESS     VARCHAR(40) NOT NULL
, C_CITY        VARCHAR(25) NOT NULL
, C_NATION      VARCHAR(25) NOT NULL
, C_REGION      VARCHAR(25) NOT NULL
```

```
, C_PHONE       VARCHAR(15) NOT NULL
, C_MKTSEGMENT  VARCHAR(10) NOT NULL
, S_NAME        VARCHAR(25) NOT NULL
, S_ADDRESS     VARCHAR(40) NOT NULL
, S_CITY        VARCHAR(25) NOT NULL
, S_NATION      VARCHAR(25) NOT NULL
, S_REGION      VARCHAR(25) NOT NULL
, S_PHONE       VARCHAR(15) NOT NULL
, P_NAME  VARCHAR(55) NOT NULL
, P_MFGR  VARCHAR(25) NOT NULL
, P_CATEGORY    VARCHAR(25) NOT NULL
, P_BRAND VARCHAR(10) NOT NULL
, P_COLOR VARCHAR(20) NOT NULL
, P_TYPE  VARCHAR(25) NOT NULL
, P_SIZE  BIGINT NOT NULL
, P_CONTAINER   VARCHAR(10) NOT NULL
, Partial Cluster Key(s_region,s_nation,s_city)
) WITH (orientation=column,
colversion=2.0,enable_hstore=true,enable_hstore_opt=true,secondary_part_column='p_mfgr',
secondary_part_num=8)
DISTRIBUTE BY hash(LO_ORDERKEY)
PARTITION BY RANGE(LO_ORDERDATE)
(
PARTITION LO_ORDERDATE_1 VALUES LESS THAN('1992-04-01 00:00:00'),
PARTITION LO_ORDERDATE_2 VALUES LESS THAN('1992-07-01 00:00:00'),
PARTITION LO_ORDERDATE_3 VALUES LESS THAN('1992-10-01 00:00:00'),
PARTITION LO_ORDERDATE_4 VALUES LESS THAN('1993-01-01 00:00:00'),
PARTITION LO_ORDERDATE_5 VALUES LESS THAN('1993-04-01 00:00:00'),
PARTITION LO_ORDERDATE_6 VALUES LESS THAN('1993-07-01 00:00:00'),
PARTITION LO_ORDERDATE_7 VALUES LESS THAN('1993-10-01 00:00:00'),
PARTITION LO_ORDERDATE_8 VALUES LESS THAN('1994-01-01 00:00:00'),
PARTITION LO_ORDERDATE_9 VALUES LESS THAN('1994-04-01 00:00:00'),
PARTITION LO_ORDERDATE_10 VALUES LESS THAN('1994-07-01 00:00:00'),
PARTITION LO_ORDERDATE_11 VALUES LESS THAN('1994-10-01 00:00:00'),
PARTITION LO_ORDERDATE_12 VALUES LESS THAN('1995-01-01 00:00:00'),
PARTITION LO_ORDERDATE_13 VALUES LESS THAN('1995-04-01 00:00:00'),
PARTITION LO_ORDERDATE_14 VALUES LESS THAN('1995-07-01 00:00:00'),
PARTITION LO_ORDERDATE_15 VALUES LESS THAN('1995-10-01 00:00:00'),
PARTITION LO_ORDERDATE_16 VALUES LESS THAN('1996-01-01 00:00:00'),
PARTITION LO_ORDERDATE_17 VALUES LESS THAN('1996-04-01 00:00:00'),
PARTITION LO_ORDERDATE_18 VALUES LESS THAN('1996-07-01 00:00:00'),
PARTITION LO_ORDERDATE_19 VALUES LESS THAN('1996-10-01 00:00:00'),
PARTITION LO_ORDERDATE_20 VALUES LESS THAN('1997-01-01 00:00:00'),
PARTITION LO_ORDERDATE_21 VALUES LESS THAN('1997-04-01 00:00:00'),
PARTITION LO_ORDERDATE_22 VALUES LESS THAN('1997-07-01 00:00:00'),
PARTITION LO_ORDERDATE_23 VALUES LESS THAN('1997-10-01 00:00:00'),
PARTITION LO_ORDERDATE_24 VALUES LESS THAN('1998-01-01 00:00:00'),
PARTITION LO_ORDERDATE_25 VALUES LESS THAN('1998-04-01 00:00:00'),
PARTITION LO_ORDERDATE_26 VALUES LESS THAN('1998-07-01 00:00:00'),
PARTITION LO_ORDERDATE_27 VALUES LESS THAN('1998-10-01 00:00:00'),
PARTITION LO_ORDERDATE_28 VALUES LESS THAN('1999-01-01 00:00:00')
) ;
SET enable_hstoreopt_auto_bitmap=false;
```

创建 SSB 数据集的 GDS 外表

连接DWS数据库后执行以下SQL语句。

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_load;
CREATE FOREIGN TABLE customer_load
(
   C_CUSTKEY     BIGINT NOT NULL
  , C_NAME        VARCHAR(25) NOT NULL
  , C_ADDRESS     VARCHAR(40) NOT NULL
  , C_CITY        VARCHAR(25) NOT NULL
  , C_NATION      VARCHAR(25) NOT NULL
  , C_REGION      VARCHAR(25) NOT NULL
  , C_PHONE       VARCHAR(15) NOT NULL
  , C_MKTSEGMENT  VARCHAR(10) NOT NULL)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5004/customer.tbl*',
format 'text',
delimiter '|',
encoding 'utf8',
mode 'Normal'
);

DROP FOREIGN TABLE IF EXISTS supplier_load;
CREATE FOREIGN TABLE supplier_load
(
 S_SUPPKEY  BIGINT NOT NULL
  , S_NAME  VARCHAR(25) NOT NULL
  , S_ADDRESS  VARCHAR(40) NOT NULL
  , S_CITY        VARCHAR(25) NOT NULL
  , S_NATION      VARCHAR(25) NOT NULL
  , S_REGION      VARCHAR(25) NOT NULL
  , S_PHONE VARCHAR(15) NOT NULL)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5004/supplier.tbl*',
format 'text',
delimiter '|',
encoding 'utf8',
mode 'Normal'
);

DROP FOREIGN TABLE IF EXISTS part_load;
CREATE FOREIGN TABLE part_load
(
 P_PARTKEY  BIGINT NOT NULL
  , P_NAME  VARCHAR(55) NOT NULL
  , P_MFGR  VARCHAR(25) NOT NULL
  , P_CATEGORY   VARCHAR(25) NOT NULL
  , P_BRAND VARCHAR(10) NOT NULL
  , P_COLOR VARCHAR(20) NOT NULL
  , P_TYPE  VARCHAR(25) NOT NULL
  , P_SIZE  BIGINT NOT NULL
  , P_CONTAINER   VARCHAR(10) NOT NULL)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5004/part.tbl*',
format 'text',
delimiter '|',
encoding 'utf8',
mode 'Normal'
);

DROP FOREIGN TABLE IF EXISTS lineorder_load;
CREATE FOREIGN TABLE lineorder_load
(
    LO_ORDERKEY             BIGINT NOT NULL,
    LO_LINENUMBER           BIGINT NOT NULL,
    LO_CUSTKEY              BIGINT NOT NULL,
    LO_PARTKEY              BIGINT NOT NULL,
    LO_SUPPKEY              BIGINT NOT NULL,
    LO_ORDERDATE            DATE NOT NULL,
    LO_ORDERPRIORITY        VARCHAR(15) NOT NULL,
    LO_SHIPPRIORITY         BIGINT NOT NULL,
    LO_QUANTITY             BIGINT NOT NULL,
```

```
    LO_EXTENDEDPRICE        BIGINT NOT NULL,
    LO_ORDTOTALPRICE        BIGINT NOT NULL,
    LO_DISCOUNT             BIGINT NOT NULL,
    LO_REVENUE              BIGINT NOT NULL,
    LO_SUPPLYCOST           BIGINT NOT NULL,
    LO_TAX                  BIGINT NOT NULL,
    LO_COMMITDATE           DATE NOT NULL,
    LO_SHIPMODE             VARCHAR(10) NOT NULL)
SERVER gsmpp_server
OPTIONS(location 'gsfs://192.168.0.90:5004/lineorder.tbl*',
format 'text',
delimiter '|',
encoding 'utf8',
mode 'Normal'
);
```

导入 SSB 数据

执行以下命令导入数据。

```
INSERT INTO customer SELECT * FROM customer_load;
INSERT INTO supplier SELECT * FROM supplier_load;
INSERT INTO part SELECT * FROM part_load;
INSERT INTO lineorder SELECT * FROM lineorder_load;

INSERT INTO lineorder_flat
SELECT
 l.LO_ORDERKEY AS LO_ORDERKEY,
 l.LO_LINENUMBER AS LO_LINENUMBER,
 l.LO_CUSTKEY AS LO_CUSTKEY,
 l.LO_PARTKEY AS LO_PARTKEY,
 l.LO_SUPPKEY AS LO_SUPPKEY,
 l.LO_ORDERDATE AS LO_ORDERDATE,
 l.LO_ORDERPRIORITY AS LO_ORDERPRIORITY,
 l.LO_SHIPPRIORITY AS LO_SHIPPRIORITY,
 l.LO_QUANTITY AS LO_QUANTITY,
 l.LO_EXTENDEDPRICE AS LO_EXTENDEDPRICE,
 l.LO_ORDTOTALPRICE AS LO_ORDTOTALPRICE,
 l.LO_DISCOUNT AS LO_DISCOUNT,
 l.LO_REVENUE AS LO_REVENUE,
 l.LO_SUPPLYCOST AS LO_SUPPLYCOST,
 l.LO_TAX AS LO_TAX,
 l.LO_COMMITDATE AS LO_COMMITDATE,
 l.LO_SHIPMODE AS LO_SHIPMODE,
 c.C_NAME AS C_NAME,
 c.C_ADDRESS AS C_ADDRESS,
 c.C_CITY AS C_CITY,
 c.C_NATION AS C_NATION,
 c.C_REGION AS C_REGION,
 c.C_PHONE AS C_PHONE,
 c.C_MKTSEGMENT AS C_MKTSEGMENT,
 s.S_NAME AS S_NAME,
 s.S_ADDRESS AS S_ADDRESS,
 s.S_CITY AS S_CITY,
 s.S_NATION AS S_NATION,
 s.S_REGION AS S_REGION,
 s.S_PHONE AS S_PHONE,
 p.P_NAME AS P_NAME,
 p.P_MFGR AS P_MFGR,
 p.P_CATEGORY AS P_CATEGORY,
 p.P_BRAND AS P_BRAND,
 p.P_COLOR AS P_COLOR,
 p.P_TYPE AS P_TYPE,
 p.P_SIZE AS P_SIZE,
 p.P_CONTAINER AS P_CONTAINER
FROM lineorder AS l
INNER JOIN customer AS c ON c.C_CUSTKEY = l.LO_CUSTKEY
INNER JOIN supplier AS s ON s.S_SUPPKEY = l.LO_SUPPKEY
INNER JOIN part AS p ON p.P_PARTKEY = l.LO_PARTKEY;
```

5.3.4 SSB 查询测试

SSB（Star Schema Benchmark）是一种在学术界和工业界广泛应用的数据库系统性能评估基准测试方法。它能够对比不同数据仓库在处理星型模型查询时的性能，帮助数据库管理员和决策者选择最符合需求的数据库系统。此外，参考OLAP行业的做法，将SSB中的星型模型展平转化为宽表，还可以改造成单一表测试Benchmark（SSB Flat）。

共包含Q1.1~Q1.3、Q2.1~Q2.3、Q3.1~Q3.4、Q4.1~4.3的查询样例，以下为SQL查询语句，仅供参考。

了解更多SSB数据测试，请访问SSB官网。

Q1.1

```
SELECT SUM(LO_EXTENDEDPRICE * LO_DISCOUNT) AS revenue
FROM lineorder_flat
WHERE
LO_ORDERDATE >= date '19930101'
AND LO_ORDERDATE <= date '19931231'
AND LO_DISCOUNT BETWEEN 1 AND 3
AND LO_QUANTITY < 25;
```

Q1.2

```
SELECT SUM(LO_EXTENDEDPRICE * LO_DISCOUNT) AS revenue
FROM lineorder_flat
WHERE
LO_ORDERDATE >= 19940101
AND LO_ORDERDATE <= 19940131
AND LO_DISCOUNT BETWEEN 4 AND 6
AND LO_QUANTITY BETWEEN 26 AND 35;
```

Q1.3

```
SELECT SUM(LO_EXTENDEDPRICE * LO_DISCOUNT) AS revenue
FROM lineorder_flat
WHERE
weekofyear(to_date(LO_ORDERDATE)) = 6
AND LO_ORDERDATE >= 19940101
AND LO_ORDERDATE <= 19941231
AND LO_DISCOUNT BETWEEN 5 AND 7
AND LO_QUANTITY BETWEEN 26 AND 35;
```

Q2.1

```
SELECT
SUM(LO_REVENUE), div(LO_ORDERDATE,10000) AS YEAR,
P_BRAND
FROM lineorder_flat
WHERE P_CATEGORY = 'MFGR#12' AND S_REGION = 'AMERICA'
GROUP BY YEAR, P_BRAND
ORDER BY YEAR, P_BRAND;
```

Q2.2

```
SELECT
SUM(LO_REVENUE), div(LO_ORDERDATE,10000) AS YEAR,
P_BRAND
FROM lineorder_flat
WHERE
```

```
P_BRAND >= 'MFGR#2221'
AND P_BRAND <= 'MFGR#2228'
AND S_REGION = 'ASIA'
GROUP BY YEAR, P_BRAND
ORDER BY YEAR, P_BRAND;
```

Q2.3

```
SELECT
SUM(LO_REVENUE), div(LO_ORDERDATE,10000) AS YEAR,
P_BRAND
FROM lineorder_flat
WHERE
P_BRAND = 'MFGR#2239'
AND S_REGION = 'EUROPE'
GROUP BY YEAR, P_BRAND
ORDER BY YEAR, P_BRAND;
```

Q3.1

```
SELECT
C_NATION,
S_NATION, div(LO_ORDERDATE,10000) AS YEAR,
SUM(LO_REVENUE) AS revenue
FROM lineorder_flat
WHERE
C_REGION = 'ASIA'
AND S_REGION = 'ASIA'
AND LO_ORDERDATE >= 19920101
AND LO_ORDERDATE <= 19971231
GROUP BY C_NATION, S_NATION, YEAR
ORDER BY YEAR ASC, revenue DESC;
```

Q3.2

```
SELECT
C_CITY,
S_CITY, div(LO_ORDERDATE,10000) AS YEAR,
SUM(LO_REVENUE) AS revenue
FROM lineorder_flat
WHERE
C_NATION = 'UNITED STATES'
AND S_NATION = 'UNITED STATES'
AND LO_ORDERDATE >= 19920101
AND LO_ORDERDATE <= 19971231
GROUP BY C_CITY, S_CITY, YEAR
ORDER BY YEAR ASC, revenue DESC;
```

Q3.3

```
SELECT
C_CITY,
S_CITY, div(LO_ORDERDATE,10000) AS YEAR,
SUM(LO_REVENUE) AS revenue
FROM lineorder_flat
WHERE
C_CITY IN ('UNITED KI1', 'UNITED KI5')
AND S_CITY IN ('UNITED KI1', 'UNITED KI5')
AND LO_ORDERDATE >= 19920101
AND LO_ORDERDATE <= 19971231
GROUP BY C_CITY, S_CITY, YEAR
ORDER BY YEAR ASC, revenue DESC;
```

Q3.4

```
SELECT
C_CITY,
```



```
S_CITY, div(LO_ORDERDATE,10000) AS YEAR,
SUM(LO_REVENUE) AS revenue
FROM lineorder_flat
WHERE
C_CITY IN ('UNITED KI1', 'UNITED KI5')
AND S_CITY IN ('UNITED KI1', 'UNITED KI5')
AND LO_ORDERDATE >= 19971201
AND LO_ORDERDATE <= 19971231
GROUP BY C_CITY, S_CITY, YEAR
ORDER BY YEAR ASC, revenue DESC;
```

Q4.1

```
SELECT div(LO_ORDERDATE,10000) AS YEAR,
C_NATION,
SUM(LO_REVENUE - LO_SUPPLYCOST) AS profit
FROM lineorder_flat
WHERE
C_REGION = 'AMERICA'
AND S_REGION = 'AMERICA'
AND P_MFGR IN ('MFGR#1', 'MFGR#2')
GROUP BY YEAR, C_NATION
ORDER BY YEAR ASC, C_NATION ASC;
```

Q4.2

```
SELECT div(LO_ORDERDATE,10000) AS YEAR,
S_NATION,
P_CATEGORY,
SUM(LO_REVENUE - LO_SUPPLYCOST) AS profit
FROM lineorder_flat
WHERE
C_REGION = 'AMERICA'
AND S_REGION = 'AMERICA'
AND LO_ORDERDATE >= 19970101
AND LO_ORDERDATE <= 19981231
AND P_MFGR IN ('MFGR#1', 'MFGR#2')
GROUP BY YEAR, S_NATION, P_CATEGORY
ORDER BY
YEAR ASC,
S_NATION ASC,
P_CATEGORY ASC;
```

Q4.3

```
SELECT div(LO_ORDERDATE,10000) AS YEAR,
S_CITY,
P_BRAND,
SUM(LO_REVENUE - LO_SUPPLYCOST) AS profit
FROM lineorder_flat
WHERE
S_NATION = 'UNITED STATES'
AND LO_ORDERDATE >= 19970101
AND LO_ORDERDATE <= 19981231
AND P_CATEGORY = 'MFGR#14'
GROUP BY YEAR, S_CITY, P_BRAND
ORDER BY YEAR ASC, S_CITY ASC, P_BRAND ASC;
```