GreatSQL TPC-H 性能测试报告

GreatSQL TPC-H 性能测试报告

（2024年11月02日）

GreatSQL 社区

【文档声明】

GreatSQL 社区提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。您应当通过 GreatSQL 社区网站或 GreatSQL 社区提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为 GreatSQL 社区的保密信息，您应当严格遵守保密义务；未经 GreatSQL 社区事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。

未经 GreatSQL 社区事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行替换和宣传。

由于产品版本升级、调整或其他原因，本文档内容有可能变更。GreatSQL 社区保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在 GreatSQL 社区授权通道中不定期发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过 GreatSQL 社区授权渠道下载、获取最新版的用户文档。

本文档仅作为用户使用 GreatSQL 社区产品及服务的参考性指引。GreatSQL 社区在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但 GreatSQL 社区在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，GreatSQL 社区不承担任何法律责任。在任何情况下，GreatSQL 社区均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使 GreatSQL 社区已被告知该等损失的可能性）。

GreatSQL 社区文档中所有内容，包括但不限于图片、架构设计、页面布局、文字描述，均由 GreatSQL 社区和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经 GreatSQL 社区和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开替换、改变、散布、发行或公开发表 GreatSQL 社区网站、产品程序或内容。此外，未经 GreatSQL 社区事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制 GreatSQL 社区的名称（包括但不限于单独为或以组合形式包含“GreatSQL 社区”、“GreatSQL”等 GreatSQL 社区和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别 GreatSQL 社区和/或其关联公司）。

如若发现本文档存在任何错误，请与 GreatSQL 社区取得直接联系。

GreatSQL社区官网：https://greatsql.cn。

概述

本次测试针对GreatSQL数据库基于标准 TPC-H 场景的测试。

TPC-H（商业智能计算测试）是美国交易处理效能委员会（TPC，TransactionProcessing Performance Council）组织制定的用来模拟决策支持类应用的一个测试集。目前，学术界和工业界普遍采用 TPC-H 来评价决策支持技术方面应用的性能。这种商业测试可以全方位评测系统的整体商业计算综合能力，对厂商的要求更高，同时也具有普遍的商业实用意义，目前在银行信贷分析和信用卡分析、电信运营分析、税收分析、烟草行业决策分析中都有广泛的应用，TPC-H 查询包含八张数据表和 22 条复杂 SQL 查询，大多数查询包含多表联接（JOIN）、子查询和聚合查询等。

GreatSQL 数据库是一款 **开源免费** 数据库，可在普通硬件上满足金融级应用场景，具有 **高可用、高性能、高兼容、高安全** 等特性，可作为 MySQL 或 Percona 的理想可选替换。

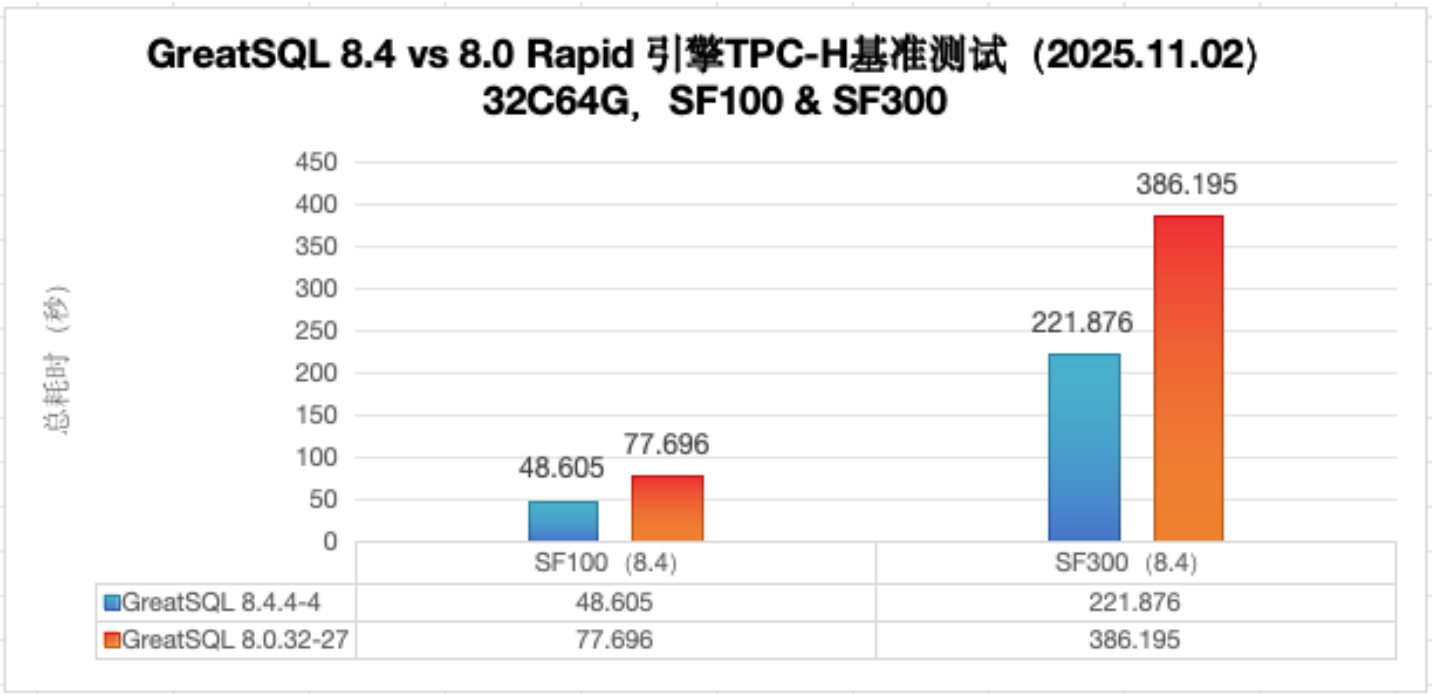
测试结果

从本次测试的结果来看，可以得到以下结论：

本次测试结果表明：GreatSQL 8.4.4-4 相比 GreatSQL 8.0.32-27 在 TPC-H 测试场景中性能有明显提升，在SF100 和 SF300 中分别提升 37.44% 和 42.55%，当数据量更大时性能表现更优异。

以上结论，仅基于本次测试的几个场景的总结。

GreatSQL 8.4.4-4 vs 8.0.32-27 Rapid 引擎 TPC-H 基准测试对比示意图如下：



测试环境：

配置	备注

操作系统	OS：CentOS Linux release 8.5.2111 内核：4.18.0-240.el8.x86_64
CPU	Intel(R) Xeon(R) Gold 6238 CPU @ 2.10GHz * 4
内存	256G
磁盘	INTEL SSDPE2KE032T8
数据库	GreatSQL 8.4.4-4 Revision d73de75905d
测试工具	tpch 3.0.1
测试数据量	SF100 & SF300

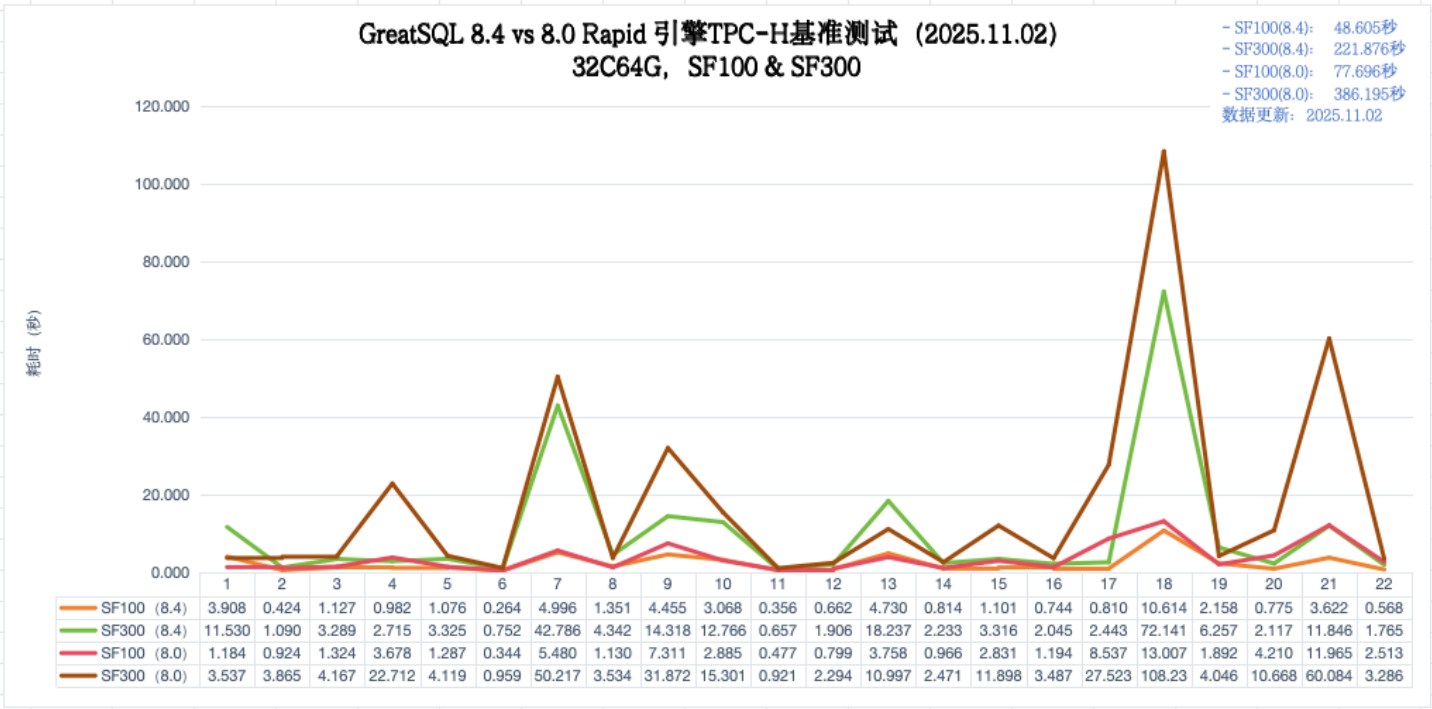
测试结果详细数据

每条SQL详细耗时如下表所示：

TPC-H Query	GreatSQL 8.4.4-4 SF100耗时（秒）	GreatSQL 8.4.4-4 SF300耗时（秒）	GreatSQL 8.0.32-27 SF100耗时（秒）	GreatSQL 8.0.32-27 SF300耗时（秒）
Q1	3.908	11.530	1.184	3.537
Q2	0.424	1.090	0.924	3.865
Q3	1.127	3.289	1.324	4.167
Q4	0.982	2.715	3.678	22.712
Q5	1.076	3.325	1.287	4.119
Q6	0.264	0.752	0.344	0.959
Q7	4.996	42.786	5.480	50.217
Q8	1.351	4.342	1.130	3.534
Q9	4.455	14.318	7.311	31.872
Q10	3.068	12.766	2.885	15.301
Q11	0.356	0.657	0.477	0.921
Q12	0.662	1.906	0.799	2.294
Q13	4.730	18.237	3.758	10.997
Q14	0.814	2.233	0.966	2.471

Q15	1.101	3.316	2.831	11.898
Q16	0.744	2.045	1.194	3.487
Q17	0.810	2.443	8.537	27.523
Q18	10.614	72.141	13.007	108.237
Q19	2.158	6.257	1.892	4.046
Q20	0.775	2.117	4.210	10.668
Q21	3.622	11.846	11.965	60.084
Q22	0.568	1.765	2.513	3.286
总耗时	**48.605**	**221.876**	**77.696**	**386.195**

GreatSQL 8.4.4-4 vs 8.0.32-27 Rapid 引擎 TPC-H 基准测试每条SQL耗时对比示意图如下：



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
SF100（8.4）	3.908	0.424	1.127	0.982	1.076	0.264	4.996	1.351	4.455	3.068	0.356	0.662	4.730	0.814	1.101	0.744	0.810	10.614	2.158	0.775	3.622	0.568
SF300（8.4）	11.530	1.090	3.289	2.715	3.325	0.752	42.786	4.342	14.318	12.766	0.657	1.906	18.237	2.233	3.316	2.045	2.443	72.141	6.257	2.117	11.846	1.765
SF100（8.0）	1.184	0.924	1.324	3.678	1.287	0.344	5.480	1.130	7.311	2.885	0.477	0.799	3.758	0.966	2.831	1.194	8.537	13.007	1.892	4.210	11.965	2.513
SF300（8.0）	3.537	3.865	4.167	22.712	4.119	0.959	50.217	3.534	31.872	15.301	0.921	2.294	10.997	2.471	11.898	3.487	27.523	108.23	4.046	10.668	60.084	3.286

附录

测试步骤

参考手册内容 TPC-H性能测试，执行 TPC-H 测试，详细过程不赘述。

测试工具

TPC-H 3.0.1。

适用于 Rapid 引擎的相应 SQL 查询文件及辅助的批量生成数据、导入数据工具代码仓库：https://gitee.com/GreatSQL/tpch。

测试模式

- 执行 tpch-create-table.sql 脚本，创建相应的数据库。
- 调用 pdbgen.sh 脚本构造测试数据集，分别为 SF100 和 SF300 规模。
- 调用 pload.sh 脚本将测试数据集并行导入到 GreatSQL 数据库中。
- 调整 Rapid 引擎两个参数：`rapid_memory_limit=64G` 和 `rapid_worker_threads=32`
- 分别对各个表执行 `ALTER TABLE x SECONDARY_LOAD;` 操作，将 InnoDB 引擎数据加载到 Rapid 引擎中。这个过程需要一定时间，请耐心等待。
- 修改脚本 run-tpch.sh 中的变量，执行测试。

GreatSQL 主要相关参数如下

代码块

```
innodb_buffer_pool_size=128G
rapid_memory_limit=64G
rapid_worker_threads=32
```

测试表结构和数据量

各表数据量对比：

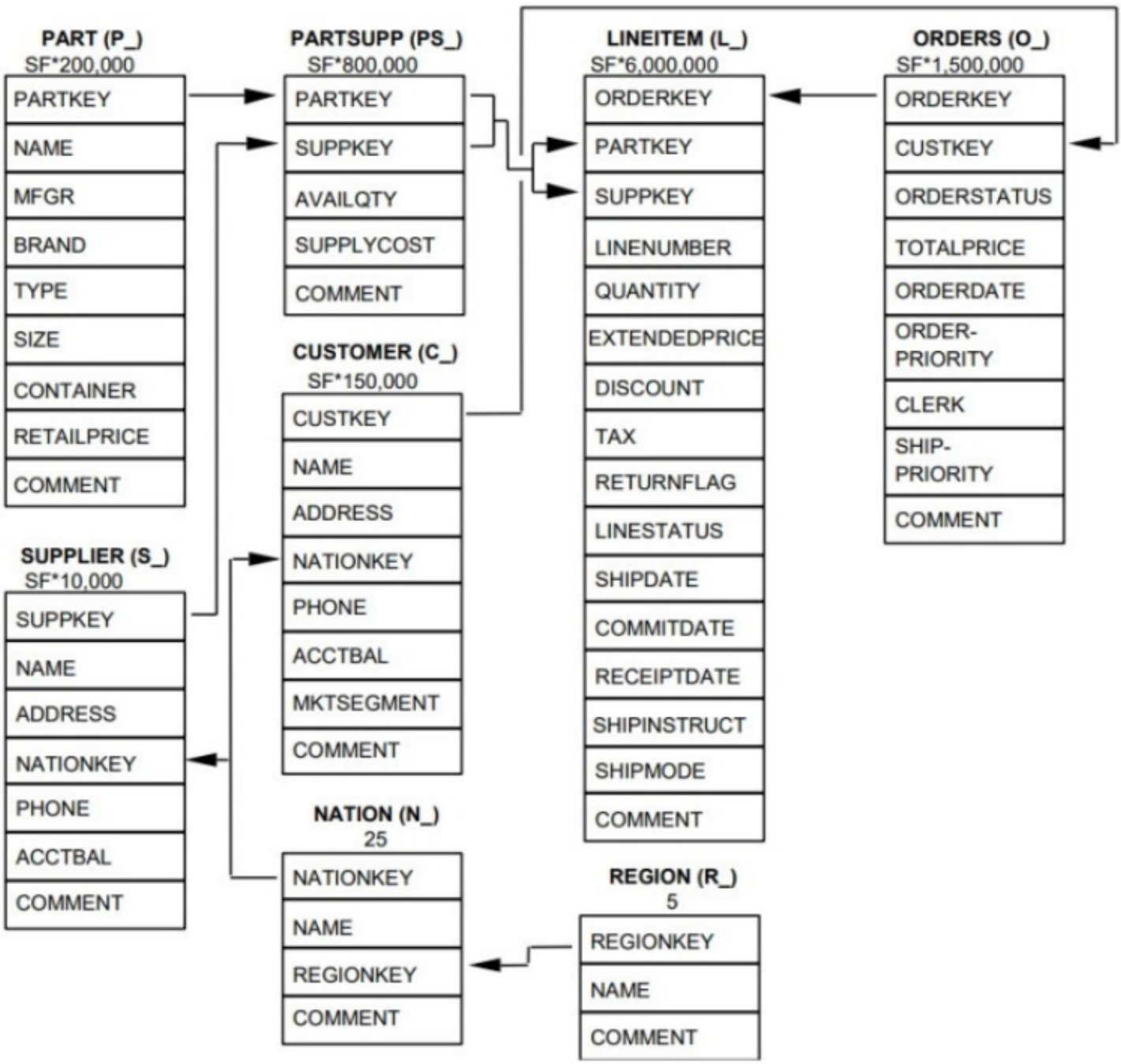
表名	TPC-H SF100数据量	TPC-H SF300数据量	备注
region	5	5	地区信息
nation	25	25	国家表
supplier	1000000	3000000	供应商信息
part	20000000	60000000	零件表
customer	15000000	45000000	消费者表
partsupp	80000000	240000000	配件供应表
orders	150000000	450000000	订单表

lineitem	600037902	1799989091	订单明细表

Rapid引擎表空间压缩比：

库名	InnoDB表空间文件总大小	Rapid引擎表空间总大小	压缩比
TPC-H SF100	184570593436	28728373248	6.42
TPC-H SF300	591644573888	74334864443	7.96

各表结构关系如下图所示：



测试环境

服务器详细信息

1. 操作系统

代码块

```
$ cat /etc/os-release

NAME="CentOS Linux"
VERSION="8"
ID="centos"
ID_LIKE="rhel fedora"
VERSION_ID="8"
PLATFORM_ID="platform:el8"
```

```
PRETTY_NAME="CentOS Linux 8"
ANSI_COLOR="0;31"
CPE_NAME="cpe:/o:centos:centos:8"
HOME_URL="https://centos.org/"
BUG_REPORT_URL="https://bugs.centos.org/"
CENTOS_MANTISBT_PROJECT="CentOS-8"
CENTOS_MANTISBT_PROJECT_VERSION="8"
```

2. CPU

代码块

```
$ lscpu

Architecture:        x86_64
CPU op-mode(s):      32-bit, 64-bit
Byte Order:          Little Endian
CPU(s):              176
On-line CPU(s) list: 0-175
Thread(s) per core:  2
Core(s) per socket:  22
Socket(s):           4
NUMA node(s):        1
Vendor ID:           GenuineIntel
BIOS Vendor ID:      Intel
CPU family:          6
Model:               85
Model name:          Intel(R) Xeon(R) Gold 6238 CPU @ 2.10GHz
BIOS Model name:     Intel(R) Xeon(R) Gold 6238 CPU @ 2.10GHz
Stepping:            7
CPU MHz:             2799.999
CPU max MHz:         3700.0000
CPU min MHz:         1000.0000
BogoMIPS:            4200.00
Virtualization:      VT-x
L1d cache:           32K
L1i cache:           32K
L2 cache:            1024K
L3 cache:            30976K
NUMA node0 CPU(s):   0-175
Flags:               fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm co
nstant_tsc art arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc cpuid aperfmperf pni pclmulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3 sdbg fma cx16 xtpr pdcm
```

```
pcid dca sse4_1 sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm abm 3dnowprefetch cpuid_fault epb cat_l3 cdp_l3 invpcid_single intel_ppin ss
bd mba ibrs ibpb stibp ibrs_enhanced tpr_shadow vnmi flexpriority ept vpid ept_ad fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2 erms invpcid cqm mpx rdt_a avx512f avx512dq rd
seed adx smap clflushopt clwb intel_pt avx512cd avx512bw avx512vl xsaveopt xsavec xgetbv1 xsaves cqm_llc cqm_occup_llc cqm_mbm_total cqm_mbm_local dtherm ida arat pln pts p
ku ospke avx512_vnni md_clear flush_l1d arch_capabilities
```

3. 内存

代码块

```
$ free -ht
               total        used        free      shared  buff/cache   available
Mem:           251Gi       146Gi       1.7Gi        17Mi       102Gi       102Gi
Swap:          4.0Gi       318Mi       3.7Gi
Total:         255Gi       146Gi       5.4Gi
```

4. 磁盘

磁盘设备型号

代码块

```
$ nvme list

Node             SN                   Model                                    Namespace Usage                      Format           FW Rev
---------------- -------------------- ---------------------------------------- --------- -------------------------- ---------------- --------
/dev/nvme0n1          PHLN018200FD3P2BGN   INTEL SSDPE2KE032T8
     1           3.20  TB /   3.20  TB    512   B +  0 B   VDV10152
```

磁盘挂载参数、文件系统、ioscheduler

代码块

```
$ df -hT | grep /ssd1
/dev/nvme0n1        xfs       3.0T  682G  2.3T  23% /ssd1

$ mount | grep ssd1
```

```
/dev/nvme0n1 on /ssd1 type xfs
(rw,noatime,nodiratime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,noquota)

$ cat /sys/block/nvme0n1/queue/scheduler
[mq-deadline] kyber bfq none
```

NVMe SSD设备简单测速

代码块

```
$ dd oflag=direct if=/dev/zero of=./zero bs=1M count=20480

20480+0 records in
20480+0 records out
21474836480 bytes (21 GB, 20 GiB) copied, 11.389 s, 1.9 GB/s
```

5. 服务器关闭 NUMA 设置

代码块

```
$ cat /etc/default/grub

GRUB_TIMEOUT=5
GRUB_DISTRIBUTOR="$(sed 's, release .*$,,g' /etc/system-release)"
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL_OUTPUT="console"
GRUB_CMDLINE_LINUX="crashkernel=auto resume=/dev/mapper/cl-swap
rd.lvm.lv=cl/root rd.lvm.lv=cl/swap numa=off"
GRUB_DISABLE_RECOVERY="true"
GRUB_ENABLE_BLSCFG=true

$ dmesg | grep -i numa

[    0.000000] Command line: BOOT_IMAGE=(hd0,gpt2)/vmlinuz-4.18.0-
240.el8.x86_64 root=/dev/mapper/cl-root ro crashkernel=auto
resume=/dev/mapper/cl-swap rd.lvm.lv=cl/root r
d.lvm.lv=cl/swap numa=off
[    0.000000] NUMA turned off
[    0.000000] Kernel command line: BOOT_IMAGE=(hd0,gpt2)/vmlinuz-4.18.0-
240.el8.x86_64 root=/dev/mapper/cl-root ro crashkernel=auto
resume=/dev/mapper/cl-swap rd.lvm.lv=cl
/root rd.lvm.lv=cl/swap numa=off
```

测试表DDL

代码块

```
-- DROP DATABASE IF EXISTS tpch;
-- CREATE DATABASE IF NOT EXISTS tpch DEFAULT CHARACTER SET latin1;
-- USE tpch;

drop table if exists nation;
create table nation  ( n_nationkey  integer not null,
                                n_name       char(25) not null,
                                n_regionkey  integer not null,
                                n_comment    varchar(152),
                                primary key(n_nationkey),
                                key nation_fk1 (n_regionkey) )
secondary_engine = rapid;

drop table if exists region;
create table region  ( r_regionkey  integer not null,
                                r_name       char(25) not null,
                                r_comment    varchar(152),
                                primary key(r_regionkey) ) secondary_engine =
rapid;

drop table if exists part;
create table part  ( p_partkey     integer not null,
                                p_name        varchar(55) not null,
                                p_mfgr        char(25) not null,
                                p_brand       char(10) not null,
                                p_type        varchar(25) not null,
                                p_size        integer not null,
                                p_container   char(10) not null,
                                p_retailprice decimal(15,2) not null,
                                p_comment     varchar(23) not null,
                                primary key(p_partkey) ) secondary_engine =
rapid;

drop table if exists supplier;
create table supplier ( s_suppkey     integer not null,
                                s_name        char(25) not null,
                                s_address     varchar(40) not null,
                                s_nationkey   integer not null,
                                s_phone       char(15) not null,
                                s_acctbal     decimal(15,2) not null,
                                s_comment     varchar(101) not null,
                                primary key(s_suppkey),
```

```
                                key supplier_fk1 (s_nationkey) )
secondary_engine = rapid;

drop table if exists partsupp;
create table partsupp ( ps_partkey     integer not null,
                                ps_suppkey     integer not null,
                                ps_availqty    integer not null,
                                ps_supplycost  decimal(15,2)  not null,
                                ps_comment     varchar(199) not null,
                                primary key(ps_partkey,ps_suppkey),
                                key partsupp_fk1 (ps_suppkey),
                                key partsupp_fk2 (ps_partkey) )
secondary_engine = rapid;


drop table if exists customer;
create table customer ( c_custkey     integer not null,
                                c_name        varchar(25) not null,
                                c_address     varchar(40) not null,
                                c_nationkey   integer not null,
                                c_phone       char(15) not null,
                                c_acctbal     decimal(15,2)   not null,
                                c_mktsegment  char(10) not null,
                                c_comment     varchar(117) not null,
                                primary key(c_custkey),
                                key customer_fk1 (c_nationkey) )
secondary_engine = rapid;

drop table if exists orders;
create table orders  ( o_orderkey       integer not null,
                                o_custkey        integer not null,
                                o_orderstatus    char(1) not null,
                                o_totalprice     decimal(15,2) not null,
                                o_orderdate      date not null,
                                o_orderpriority  char(15) not null,
                                o_clerk          char(15) not null,
                                o_shippriority   integer not null,
                                o_comment        varchar(79) not null,
                                primary key(o_orderkey),
                                key orders_fk1 (o_custkey) ) secondary_engine =
 rapid;

drop table if exists lineitem;
create table lineitem ( l_orderkey    integer not null,
                                l_partkey     integer not null,
                                l_suppkey     integer not null,
                                l_linenumber  integer not null,
```

```
                                l_quantity    decimal(15,2) not null,
                                l_extendedprice  decimal(15,2) not null,
                                l_discount    decimal(15,2) not null,
                                l_tax         decimal(15,2) not null,
                                l_returnflag  char(1) not null,
                                l_linestatus  char(1) not null,
                                l_shipdate    date not null,
                                l_commitdate  date not null,
                                l_receiptdate date not null,
                                l_shipinstruct char(25) not null,
                                l_shipmode     char(10) not null,
                                l_comment      varchar(44) not null,
                                primary key(l_orderkey,l_linenumber),
                                key lineitem_fk1 (l_orderkey) ,
                                key lineitem_fk2 (l_partkey,l_suppkey) )
    secondary_engine = rapid;
```

22条TPC-H测试SQL

代码块

```sql
-- tpch_queries_1.sql
SELECT /*+ SET_VAR(use_secondary_engine=1)
SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q1 */
    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 <= CAST('1998-09-02' AS date)
GROUP BY
    l_returnflag,
    l_linestatus
ORDER BY
    l_returnflag,
    l_linestatus;

```

```
24
25
26
27  -- tpch_queries_2.sql
28  SELECT /*+ SET_VAR(use_secondary_engine=1)
    SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q2 */
29      s_acctbal,
30      s_name,
31      n_name,
32      p_partkey,
33      p_mfgr,
34      s_address,
35      s_phone,
36      s_comment
37  FROM
38      part,
39      supplier,
40      partsupp,
41      nation,
42      region
43  WHERE
44      p_partkey = ps_partkey
45      AND s_suppkey = ps_suppkey
46      AND p_size = 15
47      AND p_type LIKE '%BRASS'
48      AND s_nationkey = n_nationkey
49      AND n_regionkey = r_regionkey
50      AND r_name = 'EUROPE'
51      AND ps_supplycost = (
52          SELECT
53              min(ps_supplycost)
54          FROM
55              partsupp,
56              supplier,
57              nation,
58              region
59          WHERE
60              p_partkey = ps_partkey
61              AND s_suppkey = ps_suppkey
62              AND s_nationkey = n_nationkey
63              AND n_regionkey = r_regionkey
64              AND r_name = 'EUROPE')
65  ORDER BY
66      s_acctbal DESC,
67      n_name,
68      s_name,
69      p_partkey
```

```
LIMIT 100;




-- tpch_queries_3.sql
SELECT /*+ SET_VAR(use_secondary_engine=1)
SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q3 */
    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 < CAST('1995-03-15' AS date)
    AND l_shipdate > CAST('1995-03-15' AS date)
GROUP BY
    l_orderkey,
    o_orderdate,
    o_shippriority
ORDER BY
    revenue DESC,
    o_orderdate
LIMIT 10;




-- tpch_queries_4.sql
SELECT /*+ SET_VAR(use_secondary_engine=1)
SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q4 */
    o_orderpriority,
    count(*) AS order_count
FROM
    orders
WHERE
    o_orderdate >= CAST('1993-07-01' AS date)
    AND o_orderdate < CAST('1993-10-01' AS date)
    AND EXISTS (
        SELECT
            *
```

```
        FROM
            lineitem
        WHERE
            l_orderkey = o_orderkey
            AND l_commitdate < l_receiptdate)
GROUP BY
    o_orderpriority
ORDER BY
    o_orderpriority;



-- tpch_queries_5.sql
SELECT /*+ SET_VAR(use_secondary_engine=1)
SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q5 */
    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 >= CAST('1994-01-01' AS date)
    AND o_orderdate < CAST('1995-01-01' AS date)
GROUP BY
    n_name
ORDER BY
    revenue DESC;



-- tpch_queries_6.sql
SELECT /*+ SET_VAR(use_secondary_engine=1)
SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q6 */
    sum(l_extendedprice * l_discount) AS revenue
```

```
FROM
    lineitem
WHERE
    l_shipdate >= CAST('1994-01-01' AS date)
    AND l_shipdate < CAST('1995-01-01' AS date)
    AND l_discount BETWEEN 0.05
    AND 0.07
    AND l_quantity < 24;




-- tpch_queries_7.sql
SELECT /*+ SET_VAR(use_secondary_engine=1)
SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q7 */
    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 CAST('1995-01-01' AS date)
        AND CAST('1996-12-31' AS date)) AS shipping
GROUP BY
    supp_nation,
    cust_nation,
```

```
    l_year
ORDER BY
    supp_nation,
    cust_nation,
    l_year;



-- tpch_queries_8.sql
SELECT /*+ SET_VAR(use_secondary_engine=1)
SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q8 */
    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 CAST('1995-01-01' AS date)
        AND CAST('1996-12-31' AS date)
        AND p_type = 'ECONOMY ANODIZED STEEL') AS all_nations
GROUP BY
    o_year
```

```
252 ORDER BY
253     o_year;
254
255
256
257
258 -- tpch_queries_9.sql
259 SELECT /*+ SET_VAR(use_secondary_engine=1)
    SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q9 */
260     nation,
261     o_year,
262     sum(amount) AS sum_profit
263 FROM (
264     SELECT
265         n_name AS nation,
266         extract(year FROM o_orderdate) AS o_year,
267         l_extendedprice * (1 - l_discount) - ps_supplycost * l_quantity AS
    amount
268     FROM
269         part,
270         supplier,
271         lineitem,
272         partsupp,
273         orders,
274         nation
275     WHERE
276         s_suppkey = l_suppkey
277         AND ps_suppkey = l_suppkey
278         AND ps_partkey = l_partkey
279         AND p_partkey = l_partkey
280         AND o_orderkey = l_orderkey
281         AND s_nationkey = n_nationkey
282         AND p_name LIKE '%green%') AS profit
283 GROUP BY
284     nation,
285     o_year
286 ORDER BY
287     nation,
288     o_year DESC;
289
290
291
292
293 -- tpch_queries_10.sql
294 SELECT /*+ SET_VAR(use_secondary_engine=1)
    SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q10 */
295     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 >= CAST('1993-10-01' AS date)
    AND o_orderdate < CAST('1994-01-01' AS date)
    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;




-- tpch_queries_11.sql
SELECT /*+ SET_VAR(use_secondary_engine=1)
SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q11 */
    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) * 0.0001000000 /* SF1 */
            /* sum(ps_supplycost * ps_availqty) * 0.0000100000 /* SF10 */
            /* sum(ps_supplycost * ps_availqty) * 0.0000010000 /* SF100 */
            /* sum(ps_supplycost * ps_availqty) * 0.0000003333 /* SF300 */
            /* sum(ps_supplycost * ps_availqty) * 0.0000001000 /* SF1000 */
        FROM
            partsupp,
            supplier,
            nation
        WHERE
            ps_suppkey = s_suppkey
            AND s_nationkey = n_nationkey
            AND n_name = 'GERMANY')
ORDER BY
    value DESC;




-- tpch_queries_12.sql
SELECT /*+ SET_VAR(use_secondary_engine=1)
SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q12 */
    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
```

```
388      AND l_shipmode IN ('MAIL', 'SHIP')
389      AND l_commitdate < l_receiptdate
390      AND l_shipdate < l_commitdate
391      AND l_receiptdate >= CAST('1994-01-01' AS date)
392      AND l_receiptdate < CAST('1995-01-01' AS date)
393  GROUP BY
394      l_shipmode
395  ORDER BY
396      l_shipmode;
397
398
399
400
401  -- tpch_queries_13.sql
402  SELECT /*+ SET_VAR(use_secondary_engine=1)
     SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q13 */
403      c_count,
404      count(*) AS custdist
405  FROM (
406      SELECT
407          c_custkey,
408          count(o_orderkey)
409      FROM
410          customer
411      LEFT OUTER JOIN orders ON c_custkey = o_custkey
412      AND o_comment NOT LIKE '%special%requests%'
413  GROUP BY
414      c_custkey) AS c_orders (c_custkey,
415          c_count)
416  GROUP BY
417      c_count
418  ORDER BY
419      custdist DESC,
420      c_count DESC;
421
422
423
424
425  -- tpch_queries_14.sql
426  SELECT /*+ SET_VAR(use_secondary_engine=1)
     SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q14 */
427      100.00 * sum(
428          CASE WHEN p_type LIKE 'PROMO%' THEN
429              l_extendedprice * (1 - l_discount)
430          ELSE
431              0
432          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 < CAST('1995-10-01' AS date);



-- tpch_queries_15.sql
SELECT /*+ SET_VAR(use_secondary_engine=1)
SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q15 */
    s_suppkey,
    s_name,
    s_address,
    s_phone,
    total_revenue
FROM
    supplier,
    (
        SELECT
            l_suppkey AS supplier_no,
            sum(l_extendedprice * (1 - l_discount)) AS total_revenue
        FROM
            lineitem
        WHERE
            l_shipdate >= CAST('1996-01-01' AS date)
            AND l_shipdate < CAST('1996-04-01' AS date)
        GROUP BY
            supplier_no) revenue0
WHERE
    s_suppkey = supplier_no
    AND total_revenue = (
        SELECT
            max(total_revenue)
        FROM (
            SELECT
                l_suppkey AS supplier_no,
                sum(l_extendedprice * (1 - l_discount)) AS total_revenue
            FROM
                lineitem
            WHERE
                l_shipdate >= CAST('1996-01-01' AS date)
                AND l_shipdate < CAST('1996-04-01' AS date)
            GROUP BY
```

```
                supplier_no) revenue1)
ORDER BY
    s_suppkey;




-- tpch_queries_16.sql
SELECT /*+ SET_VAR(use_secondary_engine=1)
SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q16 */
    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;




-- tpch_queries_17.sql
SELECT /*+ SET_VAR(use_secondary_engine=1)
SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q17 */
    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);




-- tpch_queries_18.sql
SELECT /*+ SET_VAR(use_secondary_engine=1)
SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q18 */
    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;



-- tpch_queries_19.sql
SELECT /*+ SET_VAR(use_secondary_engine=1)
SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q19 */
    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');



-- tpch_queries_20.sql
SELECT /*+ SET_VAR(use_secondary_engine=1)
SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q20 */
```

```
    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 >= CAST('1994-01-01' AS date)
                        AND l_shipdate < CAST('1995-01-01' AS date)))
            AND s_nationkey = n_nationkey
            AND n_name = 'CANADA'
        ORDER BY
            s_name;



-- tpch_queries_21.sql
SELECT /*+ SET_VAR(use_secondary_engine=1)
SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q21 */
    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;




-- tpch_queries_22.sql
SELECT /*+ SET_VAR(use_secondary_engine=1)
SET_VAR(secondary_engine_cost_threshold=0) */ /*+ Q22 */
    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
```

```
707             substring(c_phone FROM 1 FOR 2) IN ('13', '31', '23', '29', '30',
    '18', '17')
708             AND c_acctbal > (
709                 SELECT
710                     avg(c_acctbal)
711                 FROM
712                     customer
713                 WHERE
714                     c_acctbal > 0.00
715                     AND substring(c_phone FROM 1 FOR 2) IN ('13', '31', '23',
    '29', '30', '18', '17'))
716                 AND NOT EXISTS (
717                     SELECT
718                         *
719                     FROM
720                         orders
721                     WHERE
722                         o_custkey = c_custkey)) AS custsale
723     GROUP BY
724         cntrycode
725     ORDER BY
726         cntrycode;
```

参考资料

- TPC-H官网：http://www.tpc.org/tpch
- GreatSQL安装指南：https://greatsql.cn/docs/4-install-guide/0-install-guide.html
- TPC-H性能测试指南：https://greatsql.cn/docs/10-optimize/3-2-benchmark-tpch.html
- TPC-H测试建表DDL及查询SQL：https://gitee.com/GreatSQL/tpch/
- pdbgen.py脚本：https://gitee.com/GreatSQL/tpch/blob/master/pdbgen.sh