阿里云
分析型数据库MySQL版

性能白皮书

文档版本：20200525

法律声明

阿里云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。

1. 您应当通过阿里云网站或阿里云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。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，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。
2. 未经阿里云事先书面许可，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传播和宣传。
3. 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。阿里云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，并在阿里云授权通道中不时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阿里云授权渠道下载、获取最新版的用户文档。
4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云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，但阿里云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，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、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（即使阿里云已被告知该等损失的可能性）。
5. 阿里云文档中所有内容，包括但不限于图片、架构设计、页面布局、文字描述，均由阿里云和/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。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/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阿里云和/或其关联公司）。
6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	 **禁止：** 重置操作将丢失用户配置数据。
	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	 **警告：** 重启操作将导致业务中断，恢复业务时间约十分钟。
	用于警示信息、补充说明等，是用户必须了解的内容。	 **注意：** 权重设置为0，该服务器不会再接受新请求。
	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	 **说明：** 您也可以通过按Ctrl + A选中全部文件。
>	多级菜单递进。	单击**设置** > **网络** > **设置网络类型**。
粗体	表示按键、菜单、页面名称等UI元素。	在**结果确认**页面，单击**确定**。
Courier字体	命令。	执行cd /d C:/window命令，进入Windows系统文件夹。
斜体	表示参数、变量。	bae log list --instanceid Instance_ID
[]或者[a|b]	表示可选项，至多选择一个。	ipconfig [-all|-t]
{}或者{a|b}	表示必选项，至多选择一个。	switch {active|stand}

目录

1 产品概述

分析型数据库MySQL版（AnalyticDB for MySQL）是云端托管的PB级高并发实时数据仓库，是专注于服务OLAP领域的数据仓库。在数据存储模型上，采用关系模型进行数据存储，可以使用SQL进行自由灵活的计算分析，无需预先建模。利用云端的无缝伸缩能力，AnalyticDB for MySQL版在处理百亿条甚至更多量级的数据时真正实现毫秒级计算。

AnalyticDB for MySQL提供了按量付费和包年包月两种付费方式，您可以根据业务压力配置AnalyticDB for MySQL集群的规格。其中：

- 按量付费集群支持随时进行规格的扩缩容。
- 包年包月集群支持随时扩容和续费时扩缩容。

本次性能测试基于阿里云基础环境，分别在同等（或接近）硬件配置和同等数据规模下，对比AnalyticDB for MySQL与MySQL、Presto、Spark、Impala基于标准TPC-H的测试对比。

TPC-H是由TPC委员会制定发布，用于评测数据库的分析查询能力。TPC-H查询包含八张数据表和二十二条复杂SQL查询，大多数查询包含多表Join、子查询和Group By等。

2 测试方法

2.1 测试环境

下表列出了本次性能测试所使用的环境信息。

说明:

测试基础环境均在阿里云申请。

产品	规格	架构	CPU	MEM	存储	版本
AnalyticDB for MySQL	analyticdb.c8.xlarge.n1	2组C8（每组3个节点）	48 核（每节点8核）	384GB（每节点64G）	SSD云盘，400GB	AnalyticDB for MySQL 3.0
MySQL	mysql.x8.13large.2c	1个节点	52核	384GB	SSD云盘，400GB	MySQL 8.0
Presto	• master：ecs.g6.6xlarge • worker：ecs.se1ne.2xlarge	• master：1个节点 • worker：6个节点	• master：24核 • worker：48 核（每节点8核）	• master：96GB • worker：384GB（每节点64G）	• master：SSD云盘，120GB • worker：SSD云盘，80GB X 4块	• HDFS：2.8.5 • YARN：2.8.5 • Presto：0.228
Spark	• master：ecs.g5.6xlarge • worker：ecs.se1ne.2xlarge	• master：1个节点 • worker：6个节点	• master：24核 • worker：48 核（每节点8核）	• master：96GB • worker：384GB（每节点64G）	• master：SSD云盘，120GB • worker：SSD云盘，120GB X 1块	• HDFS：2.8.5 • YARN：2.8.5 • Presto：2.4.3

产品	规格	架构	CPU	MEM	存储	版本
Impala	• master：ecs.g6.6xlarge • worker：ecs.se1ne.2xlarge	• master：1个节点 • worker：6个节点	• master：24核 • worker：48核（每节点8核）	• master：96GB • worker：384GB（每节点64G）	• master：SSD云盘，120GB • worker：SSD云盘，80GB X 4块	• HDFS：2.8.5 • YARN：2.8.5 • Presto：2.12.2

2.2 测试场景

本文介绍AnalyticDB for MySQL性能测试的场景信息。

AnalyticDB for MySQL性能测试的场景信息包含：

1. 构建数据结构
   - AnalyticDB for MySQL
   - MySQL
   - Presto
   - Spark
   - Impala
2. 数据初始化
   a. 构建数据
   b. 导入数据
3. TPC-H测试集

2.3 测试方法

测试方法使用自定义脚本，依次执行TPC-H测试集。

测试优化

为保证顺利完成测试，在对个产品进行测试之前，进行了必要的配置优化。

- AnalyticDB for MySQL

  缺失配置，未进行配置优化。

- MySQL
  - 数据库参数取自阿里云默认模板。
  - 数据结构在原有基础上，增加了部分索引，请参见MySQL。
- Presto

  部分执行参数做了以下调整。

```
task.concurrency=128
exchange.max-reponse-size=16MB
query.max-memory-per-node=2GB
task.max-partial-aggregation-memory=32MB
exchange.max-buffer-size=128MB
query.max-total-memory-per-node=4GB
sink.max-buffer-size=64MB
```

- Spark

  部分执行参数做了以下调整。

```
spark-sql --num-executors 12
--executor-cores 3
--executor-memory 16G
```

- Impala

  配置EMR，关闭内存限制即disable_pool_mem_limits值为true。

测试方法

> **说明：**
>
> - 测试前，均执行两轮测试集作为预热环节。
> - 连续执行三轮测试，取三次测试结果的平均值。

您可以参照MySQL测试脚本示例，编写其他测试脚本。

```
while [ $n -lt 23 ]
do
 echo "query ${n} starting"
 time mysql -f tpch <tpch_${n}.sql
 echo "query $n ended!"
 n=`expr ${n} + 1`
done
```

2.4 测试指标

测试指标包括整个TPC-H测试集和单条SQL的执行时长。

3 测试结果

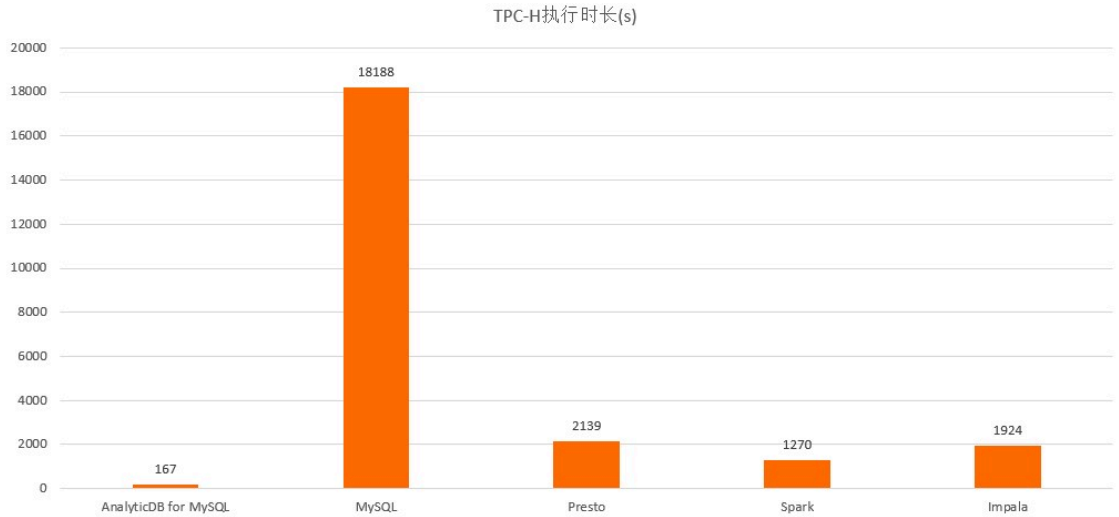
本文介绍AnalyticDB for MySQL、MySQL、Presto、Spark以及Impala的性能测试结果。

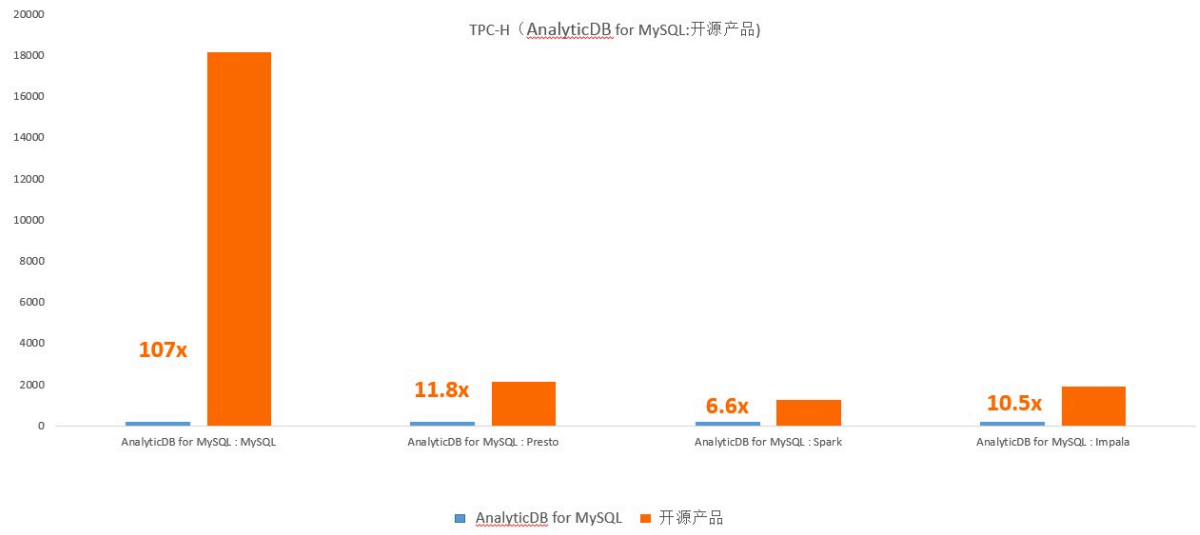
测试样本	执行时间（s）								
	AnalyticDB for MySQL	MySQL	执行时间提升倍数	Presto	执行时间提升倍数	Spark	执行时间提升倍数	Impala	执行时间提升/降低倍数
tpch_sql1	19.959	900.535	44.12	130.83	5.55	64.088	2.211	6.080	-0.695
tpch_sql2	0.889	4.135	3.65	44.94	49.55	41.219	45.366	18.660	19.990
tpch_sql3	4.368	156.144	34.75	121.21	26.75	41.913	8.595	93.240	20.346
tpch_sql4	7.506	34.811	3.64	115.68	14.41	24.764	2.299	138.240	17.417
tpch_sql5	4.563	55.333	11.13	-	-	67.922	13.885	-	-
tpch_sql6	0.186	166.924	896.44	91.62	491.58	10.462	55.247	1.440	6.742
tpch_sql7	3.444	81.284	22.60	143.34	40.62	162.78	46.265	136.350	38.591
tpch_sql8	6.701	92.881	12.86	-	-	77.574	10.576	277.470	40.407
tpch_sql9	37.154	417.531	10.24	-	-	100.33	1.700	300.480	7.087
tpch_sql10	4.503	322.436	70.60	127.17	27.24	46.066	9.230	44.670	8.920
tpch_sql11	1.056	21.790	19.63	39.91	36.79	49.354	45.737	2.380	1.254
tpch_sql12	2.270	14,656.916	11,539.88	120.28	93.71	21.242	15.726	5.260	3.142
tpch_sql13	7.303	517.782	69.90	34.43	3.71	34.51	3.725	88.200	11.077
tpch_sql14	4.634	43.885	68.22	94.69	148.35	20.322	31.054	7.980	11.587
tpch_sql15	5.922	149.901	76.99	179.16	92.22	23.899	11.434	3.650	0.899

测试样本	执行时间（s）								
	AnalyticDB for MySQL	MySQL	执行时间提升倍数	Presto	执行时间提升倍数	Spark	执行时间提升倍数	Impala	执行时间提升/降低倍数
tpch_sql1	16.940	45.487	22.45	25.92	12.36	79.557	40.009	7.180	2.701
tpch_sql15	7.128	20.985	3.09	210.68	40.08	116.033	21.627	46.650	8.097
tpch_sql18	26.591	236.179	7.88	-	-	119.788	3.505	158.340	4.955
tpch_sql12	2.566	15.867	5.18	97.32	36.93	21.68	7.449	11.080	3.318
tpch_sql20	3.537	71.311	19.16	116.28	31.88	43.971	11.432	20.300	4.739
tpch_sql21	18.946	162.672	7.59	201.77	9.65	80.793	3.264	509.870	25.912
tpch_sql22	2.159	13.323	0.86	32.9	3.60	21.832	2.050	46.220	5.456
总时长（s）	167.325	18,188.112	107.70	2139.423	11.79	1270.099	6.591	1923.74	10.497

说明：

- Impala中不允许HAVING中有子查询，tpch_sql5没有执行。
- Presto在部分示例中出现了OOM，无法完成查询。




TPC-H（AnalyticDB for MySQL:开源产品）
20000
18000
16000
14000
12000
10000
8000
6000
4000
2000
0
107x
11.8x
6.6x
10.5x
AnalyticDB for MySQL : MySQL
AnalyticDB for MySQL : Presto
AnalyticDB for MySQL : Spark
AnalyticDB for MySQL : Impala
AnalyticDB for MySQL
开源产品

4 性能测试指导

4.1 构建数据结构

4.1.1 AnalyticDB for MySQL

本文介绍AnalyticDB for MySQL性能测试的场景信息。

本次性能测试将在AnalyticDB for MySQL中创建以下八张数据表。

- CUSTOMER表

```
CREATE TABLE `CUSTOMER` (
 `C_CUSTKEY` int NOT NULL,
 `C_NAME` varchar NOT NULL,
 `C_ADDRESS` varchar NOT NULL,
 `C_NATIONKEY` int NOT NULL,
 `C_PHONE` varchar NOT NULL,
 `C_ACCTBAL` decimal(12, 2) NOT NULL,
 `C_MKTSEGMENT` varchar NOT NULL,
 `C_COMMENT` varchar NOT NULL,
 primary key (c_custkey)
)
DISTRIBUTE BY HASH(`c_custkey`)
INDEX_ALL='Y';
```

- LINEITEM表

```
CREATE TABLE `LINEITEM` (
 `L_ORDERKEY` bigint NOT NULL,
 `L_PARTKEY` int NOT NULL,
 `L_SUPPKEY` int NOT NULL,
 `L_LINENUMBER` bigint NOT NULL,
 `L_QUANTITY` decimal(12, 2) NOT NULL,
 `L_EXTENDEDPRICE` decimal(12, 2) NOT NULL,
 `L_DISCOUNT` decimal(12, 2) NOT NULL,
 `L_TAX` decimal(12, 2) NOT NULL,
 `L_RETURNFLAG` varchar NOT NULL,
 `L_LINESTATUS` varchar NOT NULL,
 `L_SHIPDATE` date NOT NULL,
 `L_COMMITDATE` date NOT NULL,
 `L_RECEIPTDATE` date NOT NULL,
 `L_SHIPINSTRUCT` varchar NOT NULL,
 `L_SHIPMODE` varchar NOT NULL,
 `L_COMMENT` varchar NOT NULL,
 primary key (l_orderkey,l_linenumber,l_shipdate)
)
DISTRIBUTE BY HASH(`l_orderkey`)
PARTITION BY VALUE(`date_format(l_shipdate, '%Y%m')`)
LIFECYCLE 90
INDEX_ALL='Y';
```

- NATION表

```
CREATE TABLE `NATION` (
```

```
 `N_NATIONKEY` int NOT NULL,
 `N_NAME` varchar NOT NULL,
 `N_REGIONKEY` int NOT NULL,
 `N_COMMENT` varchar,
 primary key (n_nationkey)
) DISTRIBUTE BY BROADCAST INDEX_ALL='Y';
```

- ORDERS表

```
CREATE TABLE `ORDERS` (
 `O_ORDERKEY` bigint NOT NULL,
 `O_CUSTKEY` int NOT NULL,
 `O_ORDERSTATUS` varchar NOT NULL,
 `O_TOTALPRICE` decimal(12, 2) NOT NULL,
 `O_ORDERDATE` date NOT NULL,
 `O_ORDERPRIORITY` varchar NOT NULL,
 `O_CLERK` varchar NOT NULL,
 `O_SHIPPRIORITY` int NOT NULL,
 `O_COMMENT` varchar NOT NULL,
 primary key (o_orderkey,o_orderdate)
)
DISTRIBUTE BY HASH(`o_orderkey`)
PARTITION BY VALUE(`date_format(O_ORDERDATE, '%Y%m')`)
LIFECYCLE 90
INDEX_ALL='Y';
```

- PART表

```
CREATE TABLE `PART` (
 `P_PARTKEY` int NOT NULL,
 `P_NAME` varchar NOT NULL,
 `P_MFGR` varchar NOT NULL,
 `P_BRAND` varchar NOT NULL,
 `P_TYPE` varchar NOT NULL,
 `P_SIZE` int NOT NULL,
 `P_CONTAINER` varchar NOT NULL,
 `P_RETAILPRICE` decimal(12, 2) NOT NULL,
 `P_COMMENT` varchar NOT NULL,
 primary key (p_partkey)
)
DISTRIBUTE BY HASH(`p_partkey`)
INDEX_ALL='Y';
```

- PARTSUPP表

```
CREATE TABLE `PARTSUPP` (
 `PS_PARTKEY` int NOT NULL,
 `PS_SUPPKEY` int NOT NULL,
 `PS_AVAILQTY` int NOT NULL,
 `PS_SUPPLYCOST` decimal(12, 2) NOT NULL,
 `PS_COMMENT` varchar NOT NULL,
 primary key (ps_partkey,ps_suppkey)
)
DISTRIBUTE BY HASH(`ps_partkey`)
INDEX_ALL='Y';
```

- REGION表

```
CREATE TABLE `REGION` (
 `R_REGIONKEY` int NOT NULL,
 `R_NAME` varchar NOT NULL,
 `R_COMMENT` varchar,
```

```
 primary key (r_regionkey)
)
DISTRIBUTE BY BROADCAST
INDEX_ALL='Y';
```

- SUPPLIER表

```
CREATE TABLE `SUPPLIER` (
 `S_SUPPKEY` int NOT NULL,
 `S_NAME` varchar NOT NULL,
 `S_ADDRESS` varchar NOT NULL,
 `S_NATIONKEY` int NOT NULL,
 `S_PHONE` varchar NOT NULL,
 `S_ACCTBAL` decimal(12, 2) NOT NULL,
 `S_COMMENT` varchar NOT NULL,
 primary key (s_suppkey)
)
DISTRIBUTE BY HASH(`s_suppkey`)
INDEX_ALL='Y';
```

4.1.2 MySQL

本文介绍MySQL性能测试的场景信息。

本次性能测试将在MySQL中创建以下八张数据表。

- CUSTOMER表

```
CREATE TABLE `customer` (
  `C_CUSTKEY` int(11) NOT NULL,
  `C_NAME` varchar(25) NOT NULL,
  `C_ADDRESS` varchar(40) NOT NULL,
  `C_NATIONKEY` int(11) NOT NULL,
  `C_PHONE` varchar(15) NOT NULL,
  `C_ACCTBAL` decimal(12,2) NOT NULL,
  `C_MKTSEGMENT` varchar(10) NOT NULL,
  `C_COMMENT` varchar(117) NOT NULL,
  PRIMARY KEY (`C_CUSTKEY`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

- LINEITEM表

```
CREATE TABLE `lineitem` (
  `L_ORDERKEY` bigint(20) NOT NULL,
  `L_PARTKEY` int(11) NOT NULL,
  `L_SUPPKEY` int(11) NOT NULL,
  `L_LINENUMBER` bigint(20) NOT NULL,
  `L_QUANTITY` decimal(12,2) NOT NULL,
  `L_EXTENDEDPRICE` decimal(12,2) NOT NULL,
  `L_DISCOUNT` decimal(12,2) NOT NULL,
  `L_TAX` decimal(12,2) 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,
  PRIMARY KEY (`L_ORDERKEY`,`L_LINENUMBER`,`L_SHIPDATE`)
```

```
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

- NATION表

```
CREATE TABLE `nation` (
  `N_NATIONKEY` int(11) NOT NULL,
  `N_NAME` varchar(25) NOT NULL,
  `N_REGIONKEY` int(11) NOT NULL,
  `N_COMMENT` varchar(152) DEFAULT NULL,
  PRIMARY KEY (`N_NATIONKEY`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

- ORDERS表

```
CREATE TABLE `orders` (
  `O_ORDERKEY` bigint(20) NOT NULL,
  `O_CUSTKEY` int(11) NOT NULL,
  `O_ORDERSTATUS` varchar(1) NOT NULL,
  `O_TOTALPRICE` decimal(12,2) NOT NULL,
  `O_ORDERDATE` date NOT NULL,
  `O_ORDERPRIORITY` varchar(15) NOT NULL,
  `O_CLERK` varchar(15) NOT NULL,
  `O_SHIPPRIORITY` int(11) NOT NULL,
  `O_COMMENT` varchar(79) NOT NULL,
  PRIMARY KEY (`O_ORDERKEY`,`O_ORDERDATE`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

- PART表

```
CREATE TABLE `part` (
  `P_PARTKEY` int(11) 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` int(11) NOT NULL,
  `P_CONTAINER` varchar(10) NOT NULL,
  `P_RETAILPRICE` decimal(12,2) NOT NULL,
  `P_COMMENT` varchar(23) NOT NULL,
  PRIMARY KEY (`P_PARTKEY`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

- PARTSUPP表

```
CREATE TABLE `partsupp` (
  `PS_PARTKEY` int(11) NOT NULL,
  `PS_SUPPKEY` int(11) NOT NULL,
  `PS_AVAILQTY` int(11) NOT NULL,
  `PS_SUPPLYCOST` decimal(12,2) NOT NULL,
  `PS_COMMENT` varchar(199) NOT NULL,
  PRIMARY KEY (`PS_PARTKEY`,`PS_SUPPKEY`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

- REGION表

```
CREATE TABLE `region` (
  `R_REGIONKEY` int(11) NOT NULL,
  `R_NAME` varchar(25) NOT NULL,
  `R_COMMENT` varchar(152) DEFAULT NULL,
  PRIMARY KEY (`R_REGIONKEY`)
```

```
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

- SUPPLIER表

```
CREATE TABLE `supplier` (
  `S_SUPPKEY` int(11) NOT NULL,
  `S_NAME` varchar(25) NOT NULL,
  `S_ADDRESS` varchar(40) NOT NULL,
  `S_NATIONKEY` int(11) NOT NULL,
  `S_PHONE` varchar(15) NOT NULL,
  `S_ACCTBAL` decimal(12,2) NOT NULL,
  `S_COMMENT` varchar(101) NOT NULL,
  PRIMARY KEY (`S_SUPPKEY`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

在100GB数据规模下，缺省MySQL表结构无法完成全部测试，可以通过增加索引提升数据查询性能。本手册性能白皮书中公布的数据，均为有索引情况下的测试结果。

```
create index idx_li_sd on lineitem(l_shipdate);
create index idx_li_rf_ls on lineitem(l_returnflag,l_linestatus);
create index idx_p_pk on part(p_partkey);
create index idx_ps_pk on partsupp(ps_partkey);
create index idx_s_sk on supplier(s_suppkey);
create index idx_ps_sk on partsupp(ps_suppkey);
create index idx_p_s on part(p_size);
create index idx_p_t on part(p_type);
create index idx_s_nk on supplier(s_nationkey);
create index idx_n_nk on nation(n_nationkey);
create index idx_n_rk on nation(n_regionkey);
create index idx_r_rk on region(r_regionkey);
create index idx_r_n on region(r_name);
create index idx_ps_sc on partsupp(ps_supplycost);
create index idx_c_mk on customer(c_mktsegment);
create index idx_c_ck on customer(c_custkey);
create index idx_o_ck on orders(o_custkey);
create index idx_li_ok on lineitem(l_orderkey);
create index idx_o_ok on orders(o_orderkey);
create index idx_o_od on orders(o_orderdate);
create index idx_o_op on orders(o_orderpriority);
create index idx_li_sk on lineitem(l_suppkey);
create index idx_c_nk on customer(c_nationkey);
create index idx_li_dc on lineitem(l_discount);
create index idx_li_q on lineitem(l_quantity);
create index idx_n_n on nation(n_name);
create index idx_li_rf on lineitem(l_returnflag);
create index idx_li_sm on lineitem(l_shipmode);
create index idx_li_cd on lineitem(l_commitdate);
create index idx_li_rd on lineitem(l_receiptdate);
create index idx_li_pk on lineitem(l_partkey);
create index idx_p_b on part(p_brand);
create index idx_p_c on part(p_container);
```

```
create index idx_o_os on orders(o_orderstatus);
```

4.1.3 Presto

本文介绍Presto性能测试的场景信息。

创建外表

本次性能测试将在Presto中创建以下八张文本格式的外表。

- CUSTOMER表

```
create external table customer(
  c_custkey integer,
  c_name varchar(25),
  c_address varchar(40),
  c_nationkey integer,
  c_phone char(15),
  c_acctbal decimal(15, 2),
  c_mktsegment char(10),
  c_comment varchar(117)
) row format delimited fields terminated by '|' location '/.../customer' TBLPROPERTIES(
  'serialization.null.format' = '',
  'serialization.encoding' = 'latin1'
);
```

- LINEITEM表

```
create external table lineitem(
  l_orderkey integer,
  l_partkey integer,
  l_suppkey integer,
  l_linenumber integer,
  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)
) row format delimited fields terminated by '|' location '/.../lineitem' TBLPROPERTIES(
  'serialization.null.format' = '',
  'serialization.encoding' = 'latin1'
);
```

- NATION表

```
create external table nation(
  n_nationkey integer,
  n_name char(25),
  n_regionkey integer,
  n_comment varchar(152)
) row format delimited fields terminated by '|' location '/.../nation' TBLPROPERTIES(
  'serialization.null.format' = '',
  'serialization.encoding' = 'latin1'
```

```
);
```

- ORDERS表

```
create external table orders(
  o_orderkey integer,
  o_custkey integer,
  o_orderstatus char(1),
  o_totalprice decimal(15, 2),
  o_orderdate date,
  o_orderpriority char(15),
  o_clerk char(15),
  o_shippriority integer,
  o_comment varchar(79)
) row format delimited fields terminated by '|' location '/.../orders' TBLPROPERTIES(
  'serialization.null.format' = '',
  'serialization.encoding' = 'latin1'
);
```

- PART表

```
create external table part(
  p_partkey integer,
  p_name varchar(55),
  p_mfgr char(25),
  p_brand char(10),
  p_type varchar(25),
  p_size integer,
  p_container char(10),
  p_retailprice decimal(15, 2),
  p_comment varchar(23)
) row format delimited fields terminated by '|' location '/.../part' TBLPROPERTIES(
  'serialization.null.format' = '',
  'serialization.encoding' = 'latin1'
);
```

- PARTSUPP表

```
create external table partsupp(
  ps_partkey integer,
  ps_suppkey integer,
  ps_availqty integer,
  ps_supplycost decimal(15, 2),
  ps_comment varchar(199)
) row format delimited fields terminated by '|' location '/.../partsupp' TBLPROPERTIES(
  'serialization.null.format' = '',
  'serialization.encoding' = 'latin1'
);
```

- REGION表

```
create external table region(
  r_regionkey integer,
  r_name char(25),
  r_comment varchar(152)
) row format delimited fields terminated by '|' location '/.../region/' TBLPROPERTIES(
  'serialization.null.format' = '',
  'serialization.encoding' = 'latin1'
```

```
);
```

- SUPPLIER表

```
create external table supplier(
  s_suppkey integer,
  s_name char(25),
  s_address varchar(40),
  s_nationkey integer,
  s_phone char(15),
  s_acctbal decimal(15, 2),
  s_comment varchar(101)
) row format delimited fields terminated by '|' location '/.../supplier/' TBLPROPERTIES(
  'serialization.null.format' = '',
  'serialization.encoding' = 'latin1'
);
```

创建表

本次性能测试将在Presto中创建以下八张内表。

- CUSTOMER表

```
create table customer(
        c_custkey     integer,
        c_name        varchar(25),
        c_address     varchar(40),
        c_nationkey   integer,
        c_phone       char(15),
        c_acctbal     decimal(15,2),
        c_mktsegment  char(10),
        c_comment     varchar(117)
)
stored as parquet
TBLPROPERTIES("parquet.compression"="SNAPPY");
```

- LINEITEM表

```
create table lineitem
(
    l_orderkey    integer,
    l_partkey     integer,
    l_suppkey     integer,
    l_linenumber  integer,
    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)
)
stored as parquet
```

```
TBLPROPERTIES("parquet.compression"="SNAPPY");
```

- NATION表

```
create table nation(
      n_nationkey  integer,
      n_name       char(25),
      n_regionkey  integer,
      n_comment    varchar(152)
)
stored as parquet
TBLPROPERTIES("parquet.compression"="SNAPPY");
```

- ORDERS表

```
create table orders(
   o_orderkey       integer,
      o_custkey       integer,
      o_orderstatus    char(1),
      o_totalprice     decimal(15,2),
      o_orderdate      date,
      o_orderpriority  char(15),
      o_clerk          char(15),
      o_shippriority   integer,
      o_comment        varchar(79)
)
stored as parquet
TBLPROPERTIES("parquet.compression"="SNAPPY");
```

- PART表

```
create table part(
      p_partkey     integer,
      p_name        varchar(55),
      p_mfgr        char(25),
      p_brand       char(10),
      p_type        varchar(25),
      p_size        integer,
      p_container   char(10),
      p_retailprice decimal(15,2),
      p_comment     varchar(23)
)
stored as parquet
TBLPROPERTIES("parquet.compression"="SNAPPY");
```

- PARTSUPP表

```
create table partsupp(
   ps_partkey     integer,
      ps_suppkey     integer,
      ps_availqty    integer,
      ps_supplycost  decimal(15,2),
      ps_comment     varchar(199)
)
stored as parquet
TBLPROPERTIES("parquet.compression"="SNAPPY");
```

- REGION表

```
create table region(
   r_regionkey  integer,
```

```
    r_name      char(25),
    r_comment    varchar(152)
)
stored as parquet
TBLPROPERTIES("parquet.compression"="SNAPPY");
```

- SUPPLIER表

```
create table supplier(
    s_suppkey     integer,
    s_name       char(25),
    s_address     varchar(40),
    s_nationkey   integer,
    s_phone      char(15),
    s_acctbal     decimal(15,2),
    s_comment     varchar(101)
)
stored as parquet
TBLPROPERTIES("parquet.compression"="SNAPPY");
```

4.1.4 Spark

本文介绍Spark性能测试的场景信息。

创建外表

本次性能测试将在Spark中创建以下八张文本格式的外表。

- CUSTOMER表

```
create external table customer(
  c_custkey integer,
  c_name varchar(25),
  c_address varchar(40),
  c_nationkey integer,
  c_phone char(15),
  c_acctbal decimal(15, 2),
  c_mktsegment char(10),
  c_comment varchar(117)
) row format delimited fields terminated by '|' location '/.../customer' TBLPROPERTIES(
  'serialization.null.format' = '',
  'serialization.encoding' = 'latin1'
);
```

- LINEITEM表

```
create external table lineitem(
  l_orderkey integer,
  l_partkey integer,
  l_suppkey integer,
  l_linenumber integer,
  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)
) row format delimited fields terminated by '|' location '/.../lineitem' TBLPROPERTIES(
  'serialization.null.format' = '',
  'serialization.encoding' = 'latin1'
);
```

- NATION表

```
create external table nation(
  n_nationkey integer,
  n_name char(25),
  n_regionkey integer,
  n_comment varchar(152)
) row format delimited fields terminated by '|' location '/.../nation' TBLPROPERTIES(
  'serialization.null.format' = '',
  'serialization.encoding' = 'latin1'
);
```

- ORDERS表

```
create external table orders(
  o_orderkey integer,
  o_custkey integer,
  o_orderstatus char(1),
  o_totalprice decimal(15, 2),
  o_orderdate date,
  o_orderpriority char(15),
  o_clerk char(15),
  o_shippriority integer,
  o_comment varchar(79)
) row format delimited fields terminated by '|' location '/.../orders' TBLPROPERTIES(
  'serialization.null.format' = '',
  'serialization.encoding' = 'latin1'
);
```

- PART表

```
create external table part(
  p_partkey integer,
  p_name varchar(55),
  p_mfgr char(25),
  p_brand char(10),
  p_type varchar(25),
  p_size integer,
  p_container char(10),
  p_retailprice decimal(15, 2),
  p_comment varchar(23)
) row format delimited fields terminated by '|' location '/.../part' TBLPROPERTIES(
  'serialization.null.format' = '',
  'serialization.encoding' = 'latin1'
);
```

- PARTSUPP表

```
create external table partsupp(
  ps_partkey integer,
  ps_suppkey integer,
  ps_availqty integer,
  ps_supplycost decimal(15, 2),
  ps_comment varchar(199)
```

```
) row format delimited fields terminated by '|' location '/.../partsupp' TBLPROPERTIES(
  'serialization.null.format' = '',
  'serialization.encoding' = 'latin1'
);
```

- REGION表

```
create external table region(
  r_regionkey integer,
  r_name char(25),
  r_comment varchar(152)
) row format delimited fields terminated by '|' location '/.../region/' TBLPROPERTIES(
  'serialization.null.format' = '',
  'serialization.encoding' = 'latin1'
);
```

- SUPPLIER表

```
create external table supplier(
  s_suppkey integer,
  s_name char(25),
  s_address varchar(40),
  s_nationkey integer,
  s_phone char(15),
  s_acctbal decimal(15, 2),
  s_comment varchar(101)
) row format delimited fields terminated by '|' location '/.../supplier/' TBLPROPERTIES(
  'serialization.null.format' = '',
  'serialization.encoding' = 'latin1'
);
```

创建表

本次性能测试将在Spark中创建以下八张内表。

- CUSTOMER表

```
create table customer(
        c_custkey     integer,
        c_name        varchar(25),
        c_address     varchar(40),
        c_nationkey   integer,
        c_phone       char(15),
        c_acctbal     decimal(15,2),
        c_mktsegment  char(10),
        c_comment     varchar(117)
)
stored as parquet
TBLPROPERTIES("parquet.compression"="SNAPPY");
```

- LINEITEM表

```
create table lineitem
(
    l_orderkey    integer,
    l_partkey     integer,
    l_suppkey     integer,
    l_linenumber  integer,
    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)
)
stored as parquet
TBLPROPERTIES("parquet.compression"="SNAPPY");
```

- NATION表

```
create table nation(
        n_nationkey  integer,
        n_name       char(25),
        n_regionkey  integer,
        n_comment    varchar(152)
)
stored as parquet
TBLPROPERTIES("parquet.compression"="SNAPPY");
```

- ORDERS表

```
create table orders(
    o_orderkey       integer,
        o_custkey        integer,
        o_orderstatus    char(1),
        o_totalprice     decimal(15,2),
        o_orderdate      date,
        o_orderpriority  char(15),
        o_clerk          char(15),
        o_shippriority   integer,
        o_comment        varchar(79)
)
stored as parquet
TBLPROPERTIES("parquet.compression"="SNAPPY");
```

- PART表

```
create table part(
        p_partkey     integer,
        p_name        varchar(55),
        p_mfgr        char(25),
        p_brand       char(10),
        p_type        varchar(25),
        p_size        integer,
        p_container   char(10),
        p_retailprice decimal(15,2),
        p_comment     varchar(23)
)
stored as parquet
TBLPROPERTIES("parquet.compression"="SNAPPY");
```

- PARTSUPP表

```
create table partsupp(
    ps_partkey     integer,
        ps_suppkey     integer,
```

```
    ps_availqty    integer,
    ps_supplycost  decimal(15,2),
    ps_comment     varchar(199)
)
stored as parquet
TBLPROPERTIES("parquet.compression"="SNAPPY");
```

- REGION表

```
create table region(
  r_regionkey  integer,
  r_name       char(25),
  r_comment    varchar(152)
)
stored as parquet
TBLPROPERTIES("parquet.compression"="SNAPPY");
```

- SUPPLIER表

```
create table supplier(
  s_suppkey     integer,
  s_name        char(25),
  s_address     varchar(40),
  s_nationkey   integer,
  s_phone       char(15),
  s_acctbal     decimal(15,2),
  s_comment     varchar(101)
)
stored as parquet
TBLPROPERTIES("parquet.compression"="SNAPPY");
```

4.1.5 Impala

本文介绍Impala性能测试的场景信息。

创建外表

本次性能测试将在Impala中创建以下八张外表。

- CUSTOMER表

```
CREATE EXTERNAL TABLE customer
  (
    C_CUSTKEY INT,
    C_NAME STRING,
    C_ADDRESS STRING,
    C_NATIONKEY INT,
    C_PHONE STRING,
    C_ACCTBAL DOUBLE,
    C_MKTSEGMENT STRING,
    C_COMMENT STRING
  ) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
  STORED AS TEXTFILE LOCATION '/.../customer';
```

- LINEITEM表

```
CREATE EXTERNAL TABLE lineitem
  (
    L_ORDERKEY INT,
    L_PARTKEY INT,
```

```
    L_SUPPKEY INT,
    L_LINENUMBER INT,
    L_QUANTITY DOUBLE,
    L_EXTENDEDPRICE DOUBLE,
    L_DISCOUNT DOUBLE,
    L_TAX DOUBLE,
    L_RETURNFLAG STRING,
    L_LINESTATUS STRING,
    L_SHIPDATE STRING,
    L_COMMITDATE STRING,
    L_RECEIPTDATE STRING,
    L_SHIPINSTRUCT STRING,
    L_SHIPMODE STRING,
    L_COMMENT STRING
  ) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
  STORED AS TEXTFILE LOCATION '/.../lineitem';
```

- NATION表

```
CREATE EXTERNAL TABLE nation
  (
   N_NATIONKEY INT,
  N_NAME STRING,
  N_REGIONKEY INT,
  N_COMMENT STRING
  )
  ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
  STORED AS TEXTFILE LOCATION '/.../nation'
```

- ORDERS表

```
CREATE EXTERNAL TABLE orders
  (
    O_ORDERKEY INT,
    O_CUSTKEY INT,
    O_ORDERSTATUS STRING,
    O_TOTALPRICE DOUBLE,
    O_ORDERDATE STRING,
    O_ORDERPRIORITY STRING,
    O_CLERK STRING,
    O_SHIPPRIORITY INT,
    O_COMMENT STRING
  ) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
  STORED AS TEXTFILE LOCATION '/.../orders';
```

- PART表

```
CREATE EXTERNAL TABLE part
  (
    P_PARTKEY INT,
    P_NAME STRING,
    P_MFGR STRING,
    P_BRAND STRING,
    P_TYPE STRING,
    P_SIZE INT,
    P_CONTAINER STRING,
    P_RETAILPRICE DOUBLE,
    P_COMMENT STRING
  ) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
```

```
  STORED AS TEXTFILE LOCATION '/.../part';
```

- PARTSUPP表

```
CREATE EXTERNAL TABLE partsupp
  (
   PS_PARTKEY INT,
   PS_SUPPKEY INT,
   PS_AVAILQTY INT,
   PS_SUPPLYCOST DOUBLE,
   PS_COMMENT STRING
  ) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
  STORED AS TEXTFILE LOCATION '/.../partsupp';
```

- REGION表

```
CREATE EXTERNAL TABLE region
  (
   R_REGIONKEY INT,
   R_NAME STRING,
   R_COMMENT STRING
  )
  ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
  STORED AS TEXTFILE LOCATION '/.../region';
```

- SUPPLIER表

```
CREATE EXTERNAL TABLE supplier
  (
   S_SUPPKEY INT,
   S_NAME STRING,
   S_ADDRESS STRING,
   S_NATIONKEY INT,
   S_PHONE STRING,
   S_ACCTBAL DOUBLE,
   S_COMMENT STRING
  ) ROW FORMAT DELIMITED FIELDS TERMINATED BY '|'
  STORED AS TEXTFILE LOCATION '/.../supplier';
```

创建表

本次性能测试将在Impala中创建以下八张表。

- CUSTOMER表

```
DROP TABLE IF EXISTS customer;
CREATe  TABLE customer
  (
   C_CUSTKEY INT,
   C_NAME STRING,
   C_ADDRESS STRING,
   C_NATIONKEY INT,
   C_PHONE STRING,
   C_ACCTBAL DOUBLE,
   C_MKTSEGMENT STRING,
   C_COMMENT STRING
  )  STORED AS PARQUET
```

```
  TBLPROPERTIES("parquet.compression"="SNAPPY");
```

- LINEITEM表

```
DROP TABLE IF EXISTS lineitem;
CREATe  TABLE lineitem
   (
    L_ORDERKEY INT,
    L_PARTKEY INT,
    L_SUPPKEY INT,
    L_LINENUMBER INT,
    L_QUANTITY DOUBLE,
    L_EXTENDEDPRICE DOUBLE,
    L_DISCOUNT DOUBLE,
    L_TAX DOUBLE,
    L_RETURNFLAG STRING,
    L_LINESTATUS STRING,
    L_SHIPDATE STRING,
    L_COMMITDATE STRING,
    L_RECEIPTDATE STRING,
    L_SHIPINSTRUCT STRING,
    L_SHIPMODE STRING,
    L_COMMENT STRING
   )  STORED AS PARQUET
  TBLPROPERTIES("parquet.compression"="SNAPPY");
```

- NATION表

```
DROP TABLE IF EXISTS nation;
CREATe  TABLE nation
  (
    N_NATIONKEY INT,
   N_NAME STRING,
   N_REGIONKEY INT,
   N_COMMENT STRING
  ) STORED AS PARQUET
  TBLPROPERTIES("parquet.compression"="SNAPPY");
```

- ORDERS表

```
DROP TABLE IF EXISTS orders;
CREATe  TABLE orders
   (
    O_ORDERKEY INT,
    O_CUSTKEY INT,
    O_ORDERSTATUS STRING,
    O_TOTALPRICE DOUBLE,
    O_ORDERDATE STRING,
    O_ORDERPRIORITY STRING,
    O_CLERK STRING,
    O_SHIPPRIORITY INT,
    O_COMMENT STRING
   )  STORED AS PARQUET
  TBLPROPERTIES("parquet.compression"="SNAPPY");
```

- PART表

```
DROP TABLE IF EXISTS part;
CREATe  TABLE part
   (
    P_PARTKEY INT,
    P_NAME STRING,
```

```
    P_MFGR STRING,
    P_BRAND STRING,
    P_TYPE STRING,
    P_SIZE INT,
    P_CONTAINER STRING,
    P_RETAILPRICE DOUBLE,
    P_COMMENT STRING
  )  STORED AS PARQUET
 TBLPROPERTIES("parquet.compression"="SNAPPY");
```

- PARTSUPP表

```
DROP TABLE IF EXISTS partsupp;
CREATe  TABLE partsupp
  (
    PS_PARTKEY INT,
    PS_SUPPKEY INT,
    PS_AVAILQTY INT,
    PS_SUPPLYCOST DOUBLE,
    PS_COMMENT STRING
  )  STORED AS PARQUET
 TBLPROPERTIES("parquet.compression"="SNAPPY");
```

- REGION表

```
DROP TABLE IF EXISTS region;
CREATe  TABLE region
  (
    R_REGIONKEY INT,
    R_NAME STRING,
    R_COMMENT STRING
  ) STORED AS PARQUET
 TBLPROPERTIES("parquet.compression"="SNAPPY");
```

- SUPPLIER表

```
DROP TABLE IF EXISTS supplier;
CREATe  TABLE supplier
  (
    S_SUPPKEY INT,
    S_NAME STRING,
    S_ADDRESS STRING,
    S_NATIONKEY INT,
    S_PHONE STRING,
    S_ACCTBAL DOUBLE,
    S_COMMENT STRING
  )  STORED AS PARQUET
```

```
TBLPROPERTIES("parquet.compression"="SNAPPY");
```

4.2 数据初始化

4.2.1 构建数据

性能测试以TPC-H 100GB数据为测试数据，使用标准的DBGEN工具构造样本数据。

从TPC官网下载TPC-H标准的数据生成工具DBGEN，编译后生成二进制可执行文件dbgen。

```
./dbgen -s $scale -C $chunks -S $i -f
```

- -s：指定scale值，例如100GB时scale值为100，1TB时scale值为1000。
- -C：一共分成几个chunk。
- -S:：当前命令生成第几个 chunk。

> **说明：**
> 一条语句只能生成一个 chunk。

更多dbgen使用方法请参见tpch-dbgen。

4.2.2 导入数据

本文介绍如何将TPC-H 100GB测试数据分别导入AnalyticDB for MySQL、MySQL、Presto、Spark以及Impala中。

下表列出了TPC-H测试数据集中的表数据条数。

表名	数据条数
customer	15000000
lineitem	600037902
nation	25
orders	150000000
part	20000000
partsupp	80000000
region	5
supplier	1000000

AnalyticDB for MySQL/MySQL

- 在AnalyticDB for MySQL和MySQL中，使用LOAD DATA导入dbgen生成的文件。

```
LOAD DATA LOCAL INFILE 'customer.tbl' INTO TABLE CUSTOMER
FIELDS TERMINATED BY '|'  LINES TERMINATED BY '\r\n';
LOAD DATA LOCAL INFILE 'orders.tbl'   INTO TABLE ORDERS
FIELDS TERMINATED BY '|'  LINES TERMINATED BY '\r\n';
LOAD DATA LOCAL INFILE 'lineitem.tbl' INTO TABLE LINEITEM
FIELDS TERMINATED BY '|'  LINES TERMINATED BY '\r\n';
LOAD DATA LOCAL INFILE 'nation.tbl'   INTO TABLE NATION
FIELDS TERMINATED BY '|'  LINES TERMINATED BY '\r\n';
LOAD DATA LOCAL INFILE 'partsupp.tbl' INTO TABLE PARTSUPP
FIELDS TERMINATED BY '|'  LINES TERMINATED BY '\r\n';
LOAD DATA LOCAL NFILE 'part.tbl'     INTO TABLE PART
FIELDS TERMINATED BY '|'   LINES TERMINATED BY '\r\n';
LOAD DATA LOCAL INFILE 'region.tbl'   INTO TABLE REGION
FIELDS TERMINATED BY '|'  LINES TERMINATED BY '\r\n';
LOAD DATA LOCAL INFILE 'supplier.tbl' INTO TABLE SUPPLIER
FIELDS TERMINATED BY '|'  LINES TERMINATED BY '\r\n';
```

- 在AnalyticDB for MySQL中，还可以通过OSS外表方式导入测试数据，请参见#unique_21。

Presto/Spark/Impala

在Presto、Spark以及Impala中，可以通过外表方式导入测试数据。

- CUSTOMER表

```
insert overwrite table customer
select
   c_custkey,
      c_name,
      c_address,
      c_nationkey,
      c_phone,
      c_acctbal,
      c_mktsegment,
      c_comment
from ${source_db}.customer;
```

- LINEITEM表

```
insert overwrite table lineitem
select
   l_orderkey,
   l_partkey,
   l_suppkey,
   l_linenumber,
   l_quantity,
   l_extendedprice,
   l_discount,
   l_tax,
   l_returnflag,
   l_linestatus,
   l_shipdate,
   l_commitdate,
   l_receiptdate,
   l_shipinstruct,
   l_shipmode,
```

```
    l_comment
from ${source_db}.lineitem;
```

- NATION表

```
insert overwrite table nation
select
        n_nationkey,
        n_name,
        n_regionkey,
        n_comment
from ${source_db}.nation;
```

- ORDERS表

```
insert overwrite table orders
select
        o_orderkey,
        o_custkey,
        o_orderstatus,
        o_totalprice,
        o_orderdate,
        o_orderpriority,
        o_clerk,
        o_shippriority,
        o_comment
from ${source_db}.orders;
```

- PART表

```
insert overwrite table part
select
        p_partkey,
        p_name,
        p_mfgr,
        p_brand,
        p_type,
        p_size,
        p_container,
        p_retailprice,
        p_comment
from ${source_db}.part;
```

- PARTSUPP表

```
insert overwrite table partsupp
select
        ps_partkey,
        ps_suppkey,
        ps_availqty,
        ps_supplycost,
        ps_comment
from ${source_db}.partsupp;
```

- REGION表

```
insert overwrite table region
select
        r_regionkey,
        r_name,
        r_comment
```

```
from ${source_db}.region;
```

- SUPPLIER表

```
insert overwrite table supplier
select
   s_suppkey,
   s_name,
   s_address,
   s_nationkey,
   s_phone,
   s_acctbal,
   s_comment
from ${source_db}.supplier;
```

4.3 TPC-H测试集

本次性能测试将分别在AnalyticDB for MySQL、MySQL、Presto、Spark以及Impala中执行以下二十二个查询SQL。

- SQL1

```
select
      l_returnflag,
      l_linestatus,
      sum(l_quantity) as sum_qty,
      sum(l_extendedprice) as sum_base_price,
      sum(l_extendedprice * (1 - l_discount)) as sum_disc_price,
      sum(l_extendedprice * (1 - l_discount) * (1 + l_tax)) as sum_charge,
      avg(l_quantity) as avg_qty,
      avg(l_extendedprice) as avg_price,
      avg(l_discount) as avg_disc,
      count(*) as count_order
from
      lineitem
where
      l_shipdate <= date '1998-12-01' - interval '120' day
group by
      l_returnflag,
      l_linestatus
order by
      l_returnflag,
      l_linestatus;
```

- SQL2

```
select
      s_acctbal,
      s_name,
      n_name,
      p_partkey,
      p_mfgr,
      s_address,
      s_phone,
      s_comment
from
      part,
      supplier,
      partsupp,
```

```
        nation,
        region
where
        p_partkey = ps_partkey
        and s_suppkey = ps_suppkey
        and p_size = 48
        and p_type like '%STEEL'
        and s_nationkey = n_nationkey
        and n_regionkey = r_regionkey
        and r_name = 'EUROPE'
        and ps_supplycost = (
                select
                        min(ps_supplycost)
                from
                        partsupp,
                        supplier,
                        nation,
                        region
                where
                        p_partkey = ps_partkey
                        and s_suppkey = ps_suppkey
                        and s_nationkey = n_nationkey
                        and n_regionkey = r_regionkey
                        and r_name = 'EUROPE'
        )
order by
        s_acctbal desc,
        n_name,
        s_name,
        p_partkey
limit 100;
```

- SQL3

```
select
        l_orderkey,
        sum(l_extendedprice * (1 - l_discount)) as revenue,
        o_orderdate,
        o_shippriority
from
        customer,
        orders,
        lineitem
where
        c_mktsegment = 'MACHINERY'
        and c_custkey = o_custkey
        and l_orderkey = o_orderkey
        and o_orderdate < date '1995-03-23'
        and l_shipdate > date '1995-03-23'
group by
        l_orderkey,
        o_orderdate,
        o_shippriority
order by
        revenue desc,
        o_orderdate
limit 10;
```

- SQL4

```
select
        o_orderpriority,
        count(*) as order_count
```

```
from
    orders
where
    o_orderdate >= date '1996-07-01'
    and o_orderdate < date '1996-07-01' + interval '3' month
    and exists (
        select
            *
        from
            lineitem
        where
            l_orderkey = o_orderkey
            and l_commitdate < l_receiptdate
    )
group by
    o_orderpriority
order by
    o_orderpriority;
```

- SQL5

```
select
    n_name,
    sum(l_extendedprice * (1 - l_discount)) as revenue
from
    customer,
    orders,
    lineitem,
    supplier,
    nation,
    region
where
    c_custkey = o_custkey
    and l_orderkey = o_orderkey
    and l_suppkey = s_suppkey
    and c_nationkey = s_nationkey
    and s_nationkey = n_nationkey
    and n_regionkey = r_regionkey
    and r_name = 'EUROPE'
    and o_orderdate >= date '1996-01-01'
    and o_orderdate < date '1996-01-01' + interval '1' year
group by
    n_name
order by
    revenue desc;
```

- SQL6

```
select
    sum(l_extendedprice * l_discount) as revenue
from
    lineitem
where
    l_shipdate >= date '1996-01-01'
    and l_shipdate < date '1996-01-01' + interval '1' year
    and l_discount between 0.02 - 0.01 and 0.02 + 0.01
    and l_quantity < 24;
```

- SQL7

```
select
    supp_nation,
```

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

- SQL8

```
select
    o_year,
    sum(case
        when nation = 'BRAZIL' then volume
        else 0
    end) / sum(volume) as mkt_share
from
    (
        select
            extract(year from o_orderdate) as o_year,
            l_extendedprice * (1 - l_discount) as volume,
            n2.n_name as nation
        from
            part,
            supplier,
            lineitem,
            orders,
            customer,
            nation n1,
            nation n2,
            region
        where
            p_partkey = l_partkey
```

```
                and s_suppkey = l_suppkey
                and l_orderkey = o_orderkey
                and o_custkey = c_custkey
                and c_nationkey = n1.n_nationkey
                and n1.n_regionkey = r_regionkey
                and r_name = 'AMERICA'
                and s_nationkey = n2.n_nationkey
                and o_orderdate between date '1995-01-01' and date '1996-12-31'
                and p_type = 'LARGE ANODIZED COPPER'
        ) as all_nations
group by
        o_year
order by
        o_year;
```

- SQL9

```
select
        nation,
        o_year,
        sum(amount) as sum_profit
from
        (
                select
                        n_name as nation,
                        extract(year from o_orderdate) as o_year,
                        l_extendedprice * (1 - l_discount) - ps_supplycost * l_quantity as amount
                from
                        part,
                        supplier,
                        lineitem,
                        partsupp,
                        orders,
                        nation
                where
                        s_suppkey = l_suppkey
                        and ps_suppkey = l_suppkey
                        and ps_partkey = l_partkey
                        and p_partkey = l_partkey
                        and o_orderkey = l_orderkey
                        and s_nationkey = n_nationkey
                        and p_name like '%maroon%'
        ) as profit
group by
        nation,
        o_year
order by
        nation,
        o_year desc;
```

- SQL10

```
select
        c_custkey,
        c_name,
        sum(l_extendedprice * (1 - l_discount)) as revenue,
        c_acctbal,
        n_name,
        c_address,
        c_phone,
        c_comment
from
        customer,
```

```
        orders,
        lineitem,
        nation
where
        c_custkey = o_custkey
        and l_orderkey = o_orderkey
        and o_orderdate >= date '1993-02-01'
        and o_orderdate < date '1993-02-01' + interval '3' month
        and l_returnflag = 'R'
        and c_nationkey = n_nationkey
group by
        c_custkey,
        c_name,
        c_acctbal,
        c_phone,
        n_name,
        c_address,
        c_comment
order by
        revenue desc
limit 20;
```

- SQL11

```
select
        ps_partkey,
        sum(ps_supplycost * ps_availqty) as value
from
        partsupp,
        supplier,
        nation
where
        ps_suppkey = s_suppkey
        and s_nationkey = n_nationkey
        and n_name = 'EGYPT'
group by
        ps_partkey having
                sum(ps_supplycost * ps_availqty) > (
                        select
                                sum(ps_supplycost * ps_availqty) * 0.0001000000
                        from
                                partsupp,
                                supplier,
                                nation
                        where
                                ps_suppkey = s_suppkey
                                and s_nationkey = n_nationkey
                                and n_name = 'EGYPT'
                )
order by
        value desc;
```

- SQL12

```
select
        l_shipmode,
        sum(case
                when o_orderpriority = '1-URGENT'
                        or o_orderpriority = '2-HIGH'
                        then 1
                else 0
        end) as high_line_count,
        sum(case
```

```
            when o_orderpriority <> '1-URGENT'
                and o_orderpriority <> '2-HIGH'
                then 1
            else 0
        end) as low_line_count
from
        orders,
        lineitem
where
        o_orderkey = l_orderkey
        and l_shipmode in ('FOB', 'AIR')
        and l_commitdate < l_receiptdate
        and l_shipdate < l_commitdate
        and l_receiptdate >= date '1997-01-01'
        and l_receiptdate < date '1997-01-01' + interval '1' year
group by
        l_shipmode
order by
        l_shipmode;
```

- SQL13

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

- SQL14

```
select
        100.00 * sum(case
            when p_type like 'PROMO%'
                then l_extendedprice * (1 - l_discount)
            else 0
        end) / sum(l_extendedprice * (1 - l_discount)) as promo_revenue
from
        lineitem,
        part
where
        l_partkey = p_partkey
        and l_shipdate >= date '1997-06-01'
        and l_shipdate < date '1997-06-01' + interval '1' month;
```

- SQL15

```
create view revenue0 (supplier_no, total_revenue) as
```

```
        select
                l_suppkey,
                sum(l_extendedprice * (1 - l_discount))
        from
                lineitem
        where
                l_shipdate >= date '1995-02-01'
                and l_shipdate < date '1995-02-01' + interval '3' month
        group by
                l_suppkey;


select
        s_suppkey,
        s_name,
        s_address,
        s_phone,
        total_revenue
from
        supplier,
        revenue0
where
        s_suppkey = supplier_no
        and total_revenue = (
                select
                        max(total_revenue)
                from
                        revenue0
        )
order by
        s_suppkey;

drop view revenue0;
```

- SQL16

```
select
        p_brand,
        p_type,
        p_size,
        count(distinct ps_suppkey) as supplier_cnt
from
        partsupp,
        part
where
        p_partkey = ps_partkey
        and p_brand <> 'Brand#45'
        and p_type not like 'SMALL ANODIZED%'
        and p_size in (47, 15, 37, 30, 46, 16, 18, 6)
        and ps_suppkey not in (
                select
                        s_suppkey
                from
                        supplier
                where
                        s_comment like '%Customer%Complaints%'
        )
group by
        p_brand,
        p_type,
        p_size
order by
        supplier_cnt desc,
```

```
    p_brand,
    p_type,
    p_size;
```

- SQL17

```
select
    sum(l_extendedprice) / 7.0 as avg_yearly
from
    lineitem,
    part
where
    p_partkey = l_partkey
    and p_brand = 'Brand#51'
    and p_container = 'WRAP PACK'
    and l_quantity < (
        select
            0.2 * avg(l_quantity)
        from
            lineitem
        where
            l_partkey = p_partkey
    );
```

- SQL18

```
select
    c_name,
    c_custkey,
    o_orderkey,
    o_orderdate,
    o_totalprice,
    sum(l_quantity)
from
    customer,
    orders,
    lineitem
where
    o_orderkey in (
        select
            l_orderkey
        from
            lineitem
        group by
            l_orderkey having
                sum(l_quantity) > 312
    )
    and c_custkey = o_custkey
    and o_orderkey = l_orderkey
group by
    c_name,
    c_custkey,
    o_orderkey,
    o_orderdate,
    o_totalprice
order by
    o_totalprice desc,
    o_orderdate
```

```
limit 100;
```

- SQL19

```
select
    sum(l_extendedprice* (1 - l_discount)) as revenue
from
    lineitem,
    part
where
    (
        p_partkey = l_partkey
        and p_brand = 'Brand#52'
        and p_container in ('SM CASE', 'SM BOX', 'SM PACK', 'SM PKG')
        and l_quantity >= 3 and l_quantity <= 3 + 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#43'
        and p_container in ('MED BAG', 'MED BOX', 'MED PKG', 'MED PACK')
        and l_quantity >= 12 and l_quantity <= 12 + 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#52'
        and p_container in ('LG CASE', 'LG BOX', 'LG PACK', 'LG PKG')
        and l_quantity >= 21 and l_quantity <= 21 + 10
        and p_size between 1 and 15
        and l_shipmode in ('AIR', 'AIR REG')
        and l_shipinstruct = 'DELIVER IN PERSON'
    );
```

- SQL20

```
select
    s_name,
    s_address
from
    supplier,
    nation
where
    s_suppkey in (
        select
            ps_suppkey
        from
            partsupp
        where
            ps_partkey in (
                select
                    p_partkey
                from
                    part
                where
                    p_name like 'drab%'
            )
```

```
                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 '1996-01-01'
                        and l_shipdate < date '1996-01-01' + interval '1' year
                )
    )
    and s_nationkey = n_nationkey
    and n_name = 'KENYA'
order by
    s_name;
```

- SQL21

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

- SQL22

```
select
```

```
        cntrycode,
        count(*) as numcust,
        sum(c_acctbal) as totacctbal
from
        (
                select
                        substring(c_phone from 1 for 2) as cntrycode,
                        c_acctbal
                from
                        customer
                where
                        substring(c_phone from 1 for 2) in
                                ('24', '32', '17', '18', '12', '14', '22')
                        and c_acctbal > (
                                select
                                        avg(c_acctbal)
                                from
                                        customer
                                where
                                        c_acctbal > 0.00
                                        and substring(c_phone from 1 for 2) in
                                                ('24', '32', '17', '18', '12', '14', '22')
                        )
                        and not exists (
                                select
                                        *
                                from
                                        orders
                                where
                                        o_custkey = c_custkey
                        )
        ) as custsale
group by
        cntrycode
order by
        cntrycode;
```

说明:

在Spark中，需要将substring(c_phone from 1 for 2)该写为substring(c_phone, 1, 2)。