第 5 章

openGauss 查询优化

随着数据和用户体量的快速增长，传统的数据库面临多方面难以解决的性能优化问题，比如如何按需配置数据库参数、如何准确估计查询基数、如何选择物理执行计划、如何自动诊断造成性能瓶颈的查询语句等。针对以上问题，本章首先介绍查询优化的整体流程（5.1 节）；然后分别从查询命令和人工智能算法两个方面提供性能优化能力：一方面，openGauss 提供 SQL 执行信息的查询接口（如解释命令（EXPLAIN）、分析命令（ANALYZE））和 SQL 物理优化接口（如 HINT），方便数据库用户手动分析和优化查询语句（5.2～5.4 节）；另一方面，openGauss 内置了一些智能调优算法，利用人工智能算法（如卷积网络、循环神经网络）高维拟合和自学习等能力，支撑特定场景下的性能优化，如自动参数优化、查询性能预测、索引推荐（5.5～5.7 节）等。

5.1 查询优化

由于一条 SQL 语句可能对应多种等价的执行计划，而且计划的执行效率受到数据库多方面组件（如优化器参数、并发负载）的影响，openGauss 主要从两个方面提供查询优化能力。一方面，openGauss 通过提供执行信息的接口（如解释命令、分析命令、提示命令），方便用户和数据库运维人员查看物理计划、估计的执行效率和真实的执行效率，并对物理连接顺序、连接类型进行指定；另一方面，openGauss 利用人工智能技术优化数据库的性能，从而获得更好的执行表现，包括自动参数优化、查询性能预测和索引推荐。如表 5-1 所示，首先，openGauss 根据数据库历史负载和当前状态推荐合适的参数（5.5 节）；其次，opengGauss 在数据库运行中，通过预测未来负载的执行时间，决定是否执行查询诊断和修复等功能（5.6 节）；最后，针对执行效率较低的查询或整个负载，openGauss 可以利用强化学习等算法自动推荐合适的索引（见 5.7 节）。

表 5-1 查询优化功能

查询优化方式	功　能	功 能 描 述
查询优化接口	解释命令	打印出执行计划和估计的执行开销
	分析命令	打印出执行计划和实际的执行时间
	提示命令	指定物理执行逻辑，包括算子类型、连接顺序等
智能查询优化	参数调优	利用多种统计和强化学习算法自动调整参数配置
	性能预测	利用深度图嵌入等算法估计查询的执行时间
	索引推荐	利用爬山法、强化学习等算法推荐索引

5.2 查询解释命令

查询解释命令(EXPLAIN)是提供查询优化信息的第一环。对于复杂、执行效率低的SQL 语句,数据库使用者通常会利用查询解释命令打印出的执行计划分析 SQL 语句。openGauss 在传统 PostgreSQL 查询解释命令的语法基础上做了进一步扩充。本节将从功能描述、语法格式、参数说明和具体案例四个方面详细介绍查询解释命令的使用。

5.2.1 功能描述

EXPLAIN 用于显示查询语句的执行计划。执行计划将显示查询语句所引用的表会采用什么样的扫描方式,如简单的顺序扫描、索引扫描等。如果引用了多个表,执行计划还会显示用到的连接算法。执行计划最关键的部分是语句的预计执行开销,这是计划生成器估算执行该语句将花费多长时间。若指定分析(ANALYZE)选项,则该语句会被执行,然后根据实际的运行结果显示统计数据,包括每个计划节点内的时间总开销(毫秒为单位)和实际返回的总行数。这对判断计划生成器的估计是否接近现实非常有用。

5.2.2 语法格式

EXPLAIN 字段的基本语法用于显示查询语句的执行计划,支持多种选项,对选项顺序无要求,格式如下:

```
EXPLAIN [ ( option [, ...] ) ] statement;
```

其中选项 option 字段可以通过指定参数查看物理执行计划、执行开销、调试状态等多种信息,语法如下:

```
ANALYZE [ boolean ] |
ANALYSE [ boolean ] |
VERBOSE [ boolean ] |
COSTS [ boolean ] |
CPU [ boolean ] |
DETAIL [ boolean ] |
NODES [ boolean ] |
NUM_NODES [ boolean ] |
BUFFERS [ boolean ] |
TIMING [ boolean ] |
PLAN [ boolean ] |
FORMAT { TEXT | XML | JSON | YAML }
```

其中,当需要显示 SQL 语句的执行计划时,要按顺序给出选项:

```
EXPLAIN { [ { ANALYZE | ANALYSE } ] [ VERBOSE ] | PERFORMANCE } statement;
```

在指定 ANALYZE 选项时，语句会被执行。如果用户想使用解释语句分析插入(INSERT)、更新(UPDATE)、删除(DELETE)、创建新表(CREATE TABLE AS)或执行(EXECUTE)语句，而不想改动数据(执行这些语句会影响数据)，请采用下面这种方法：

```
START TRANSACTION;
EXPLAIN ANALYZE ...;
ROLLBACK;
```

5.2.3　参数说明

5.2.2 节举例介绍了 ANALYZE、VERBOSE 等参数在 openGauss 中的应用。表 5-2 给出 openGauss 数据库中解释命令支持的所有参数。

表 5-2　openGauss 数据库中解释命令支持的所有参数

参　数　名	含　　义
STATEMENT	指定要分析的查询语句
ANALYZE boolean | ANALYSE boolean	显示实际运行时间和其他统计数据
VERBOSE boolean	显示有关计划的额外信息
COSTS boolean	包括每个规划节点的估计总成本，以及估计的行数和每行的宽度
CPU boolean	打印 CPU 的使用情况的信息
DETAIL boolean	打印数据库节点上的信息
NODES boolean	打印查询执行的节点信息
NUM_NODES boolean	打印执行中的节点的个数信息
BUFFERS boolean	包括缓冲区的使用情况的信息
TIMING boolean	包括实际的启动时间和花费在输出节点上的时间信息
PLAN	是否将执行计划存储在 plan_table 中。当该选项开启时，会将执行计划存储在 PLAN_TABLE 中，不打印到当前屏幕，因此该选项为 on 时，不能与其他选项同时使用
FORMAT	指定输出格式：TEXT(默认值)、XML、JSON 和 YAML

5.2.4　示例

本节给出一个 openGauss 中使用 EXPLAIN 解释查询语句的示例。

(1) 创建一个表 tpcds.customer_address_p1：

```
opengauss = # CREATE TABLE tpcds.customer_address_p1 AS TABLE tpcds.customer_address;
```

（2）修改 explain_perf_mode 为 normal：

```
opengauss = # SET explain_perf_mode = normal;
```

（3）显示简单查询的执行计划：

```
opengauss = # EXPLAIN SELECT * FROM tpcds.customer_address_p1;
QUERY PLAN
-------------------------------------------------------
Data Node Scan (cost = 0.00..0.00 rows = 0 width = 0)
Node/s: All dbnodes
(2 rows)
```

（4）以 JSON 格式输出的执行计划（explain_perf_mode 为 normal 时）：

```
opengauss = # EXPLAIN(FORMAT JSON) SELECT * FROM tpcds.customer_address_p1;
QUERY PLAN
------------------------------------------------------------------------------
 [                                  +
   {                                +
     "Plan": {                      +
       "Node Type": "Data Node Scan",  +
       "Startup Cost": 0.00,        +
       "Total Cost": 0.00,          +
       "Plan Rows": 0,              +
       "Plan Width": 0,             +
       "Node/s": "All dbnodes"      +
     }                              +
   }                                +
 ]
(1 row)
```

（5）如果有一个索引，使用一个带索引条件（WHERE）的查询时，可能会显示一个不同的计划：

```
opengauss = # EXPLAIN SELECT * FROM tpcds.customer_address_p1 WHERE ca_address_sk = 10000;
QUERY PLAN
------------------------------------------------------------------------------
Data Node Scan (cost = 0.00..0.00 rows = 0 width = 0)
Node/s: dn_6005_6006
(2 rows)
```

（6）以 YAML 格式输出的执行计划（explain_perf_mode 为 normal 时）：

```
opengauss = # EXPLAIN(FORMAT YAML) SELECT * FROM tpcds.customer_address_p1 WHERE ca_address_
sk = 10000;
```

```
QUERY PLAN
-----------------------------------
 - Plan:                          +
     Node Type: "Data Node Scan"  +
     Startup Cost: 0.00           +
     Total Cost: 0.00             +
     Plan Rows: 0                 +
     Plan Width: 0                +
     Node/s: "dn_6005_6006"
(1 row)
```

（7）禁止开销估计的执行计划：

```
opengauss = # EXPLAIN(COSTS FALSE)SELECT * FROM tpcds.customer_address_p1 WHERE ca_address_
sk = 10000;
QUERY PLAN
------------------------------------------------------------------------------
 Data Node Scan
   Node/s: dn_6005_6006
(2 rows)
```

（8）带有聚集函数查询的执行计划：

```
opengauss = # EXPLAIN SELECT SUM(ca_address_sk) FROM tpcds.customer_address_p1 WHERE ca_
address_sk < 10000;
QUERY PLAN
------------------------------------------------------------------------------
 Aggregate  (cost = 18.19..14.32 rows = 1 width = 4)
   -> Streaming (type: GATHER)  (cost = 18.19..14.32 rows = 3 width = 4)
        Node/s: All dbnodes
        -> Aggregate  (cost = 14.19..14.20 rows = 3 width = 4)
             -> Seq Scan on customer_address_p1  (cost = 0.00..14.18 rows = 10 width = 4)
                  Filter: (ca_address_sk < 10000)
(6 rows)
```

（9）删除表 tpcds.customer_address_p1：

```
opengauss = # DROP TABLE tpcds.customer_address_p1;
```

5.3　查询分析命令

查询分析命令(ANALYZE)本质上是一个查询性能分析工具，可以详细显示出查询语句执行过程中，比如计划在哪儿花费了多少时间。它会做出查询计划，并且会实际执行查

询计划，以测量查询计划中各个关键点的实际指标，例如耗时、条数，最后详细打印出来，方便用户分析执行表现和慢查询的原因。本节分别从功能描述、语法格式和具体案例 3 个方面详细介绍查询分析命令的使用。

5.3.1 功能描述

ANALYZE 收集与数据库中普通表内容相关的统计信息，统计结果存储在系统表 PG_STATISTIC 下。执行计划生成器会使用这些统计数据，以确定最有效的执行计划。如果没有指定参数，ANALYZE 会分析当前数据库中的每个表和分区表。同时也可以通过指定表名、列名和分区名参数把分析限定在特定的表、列或分区表中。ANALYZE | ANALYSE VERIFY 用于检测数据库中普通表(行存储、列存储)的数据文件是否损坏。

5.3.2 语法格式

ANALYZE 方便用户分步解析一条查询语句在执行过程中的表信息、分区表信息、列信息等统计数据。

(1) 如果用户需要收集表的统计信息，则执行如下命令：

```
{ ANALYZE | ANALYSE } [ VERBOSE ]
[ table_name [ ( column_name [, ...] ) ] ];
```

(2) 如果用户需要收集分区表的统计信息，则执行如下命令：

```
{ ANALYZE | ANALYSE } [ VERBOSE ]
    [ table_name [ ( column_name [, ...] ) ] ]
    PARTITION ( patrition_name ) ;
```

普通分区表目前支持针对某个分区的统计信息的语法，但功能上不支持针对某个分区的统计信息收集。

(3) 如果用户需要收集多列统计信息，则执行如下命令：

```
{ANALYZE | ANALYSE} [ VERBOSE ]
    table_name (( column_1_name, column_2_name [, ...] ));
```

收集多列统计信息时，请设置 GUC 参数 default_statistics_target 为负数，以使用百分比采样方式。此外，每组多列统计信息最多支持 32 列，而且目前 openGauss 不支持收集多列统计信息的表。

(4) 当用户需要检测当前库的数据文件时，执行如下命令：

```
{ANALYZE | ANALYSE} VERIFY {FAST|COMPLETE};
```

执行检测数据文件的命令时，需要注意以下几点：

① 支持对全库进行操作，由于涉及的表较多，建议以重定向保存结果，命令如下：

```
gsql -d database -p port -f "verify.sql"> verify_warning.txt 2>&1
```

② 对外提示(NOTICE)只核对外可见的表，内部表的检测会包含在它所依赖的外部表中，不对外显示和呈现。

③ 命令的可容错级别的处理。由于调试版本的断言(Assert)可能导致 core 无法继续执行命令，建议在发布模式(Release Mode)下操作。全库操作时，若关键系统表出现损坏，则直接报错，不再继续执行。

④ ANALYZE 参数非临时表不能在一个匿名块、事务块、函数或存储过程内被执行。支持存储过程中分析临时表，不支持统计信息回滚操作。ANALYZE VERIFY 操作处理的大多为异常场景检测，需要使用发布版本。ANALYZE VERIFY 场景不触发远程读，因此远程读参数不生效。关键系统表出现错误被系统检测出页面损坏时，将直接报错，不再继续检测。

(5) 当需要检测表和索引的数据文件时，执行以下命令：

```
{ANALYZE | ANALYSE} VERIFY {FAST|COMPLETE} table_name|index_name [CASCADE];
```

支持对普通表的操作和对索引表的操作，但不支持对索引表索引使用瀑布操作(CASCADE Operation)，原因是瀑布模式用于处理主表的所有索引表，当单独对索引表进行检测时，无须使用瀑布模式(CASCADE Mode)。

此外，对主表进行检测会同步检测主表的内部表，例如 toast 表、cudesc 表等。当提示索引表损坏时，建议使用重新索引(Reindex)命令进行重建索引操作。

(6) 如果需要检测分区表的数据文件，则执行以下命令：

```
{ANALYZE | ANALYSE} VERIFY {FAST|COMPLETE} table_name PARTITION {(patrition_name)}[CASCADE];
```

该语法支持对表的单独分区进行检测操作，但不支持对索引表 index 使用 CASCADE 操作。

5.3.3　示例

本节给出一个 openGauss 中使用 ANALYZE 解释查询语句的示例。

(1) 创建表：

```
opengauss=# CREATE TABLE customer_info
(
WR_RETURNED_DATE_SK       INTEGER                       ,
WR_RETURNED_TIME_SK       INTEGER                       ,
WR_ITEM_SK                 INTEGER               NOT NULL,
WR_REFUNDED_CUSTOMER_SK   INTEGER);
```

（2）创建分区表：

```
opengauss = # CREATE TABLE customer_par
(
WR_RETURNED_DATE_SK       INTEGER                       ,
WR_RETURNED_TIME_SK       INTEGER                       ,
WR_ITEM_SK                INTEGER               NOT NULL,
WR_REFUNDED_CUSTOMER_SK   INTEGER
)
PARTITION BY RANGE(WR_RETURNED_DATE_SK)
(
PARTITION P1 VALUES LESS THAN(2452275),
PARTITION P2 VALUES LESS THAN(2452640),
PARTITION P3 VALUES LESS THAN(2453000),
PARTITION P4 VALUES LESS THAN(MAXVALUE)
)
ENABLE ROW MOVEMENT;
```

（3）使用 ANALYZE 语句更新统计信息：

```
opengauss = # ANALYZE customer_info;
opengauss = # ANALYZE customer_par;
```

（4）使用 ANALYZE VERBOSE 语句更新统计信息，并输出表的相关信息：

```
opengauss = # ANALYZE VERBOSE customer_info;
INFO:  analyzing "cstore.pg_delta_3394584009"(cn_5002 pid = 53078)
INFO:  analyzing "public.customer_info"(cn_5002 pid = 53078)
INFO:  analyzing "public.customer_info" inheritance tree(cn_5002 pid = 53078)
ANALYZE
```

若环境有故障，需查看数据库主节点的 log。

（5）删除表，恢复原始实验环境：

```
opengauss = # DROP TABLE customer_info;
opengauss = # DROP TABLE customer_par;
```

5.4 优化提示命令

优化提示命令（HINT）可以帮助用户对查询语句的执行计划进行“提示”，比如改变部分表的执行顺序、连接方式等。

5.4.1　功能描述

HINT 为用户提供了直接影响执行计划生成的手段。SQL HINT 影响执行计划的生成、SQL 查询性能的提升。用户可以通过指定连接顺序连接流处理、表扫描等方法，或者指定结果行数等多个手段进行执行计划的调优，以提升查询的性能。

5.4.2　连接顺序提示

连接顺序提示用于指明连接的顺序，包括不指定内外表顺序和指定内外表顺序。

(1) 仅指定连接顺序，不指定内外表顺序：

```
leading(join_table_list)
```

(2) 同时指定 join 顺序和内外表顺序，内外表顺序仅在最外层生效：

```
leading((join_table_list))
```

其中，join_table_list 为表示表连接顺序的提示字符串，可以包含当前层的任意个表(别名)，或对于子查询提升的场景，也可以包含子查询的提示别名，同时任意表可以使用括号指定优先级，表之间使用空格分隔。注意：表只能用单个字符串表示，不能带 schema；表如果存在别名，需要优先使用别名表示该表。

此外，join table list 中指定的表需要满足以下要求，否则会报语义错误。

(1) 表必须在当前层或提升的子查询中存在。

(2) 表在当前层或提升的子查询中必须是唯一的。如果不唯一，需要使用不同的别名进行区分。

(3) 同一个表只能在 list 里出现一次。

(4) 如果表存在别名，则 list 中的表需要使用别名。例如，leading(t1 t2 t3 t4 t5)表示 t1，t2，t3，t4，t5 先 join，5 个表的 join 顺序及内外表不限。leading((t1 t2 t3 t4 t5))表示：t1 和 t2 先 join，t2 做内表；再和 t3 join，t3 做内表；再和 t4 join，t4 做内表；再和 t5 join，t5 做内表。leading(t1 (t2 t3 t4) t5)表示：t2，t3，t4 先 join，内外表不限；再和 t1，t5 join，内外表不限。leading((t1 (t2 t3 t4) t5))表示：t2，t3，t4 先 join，内外表不限；在最外层，t1 再和 t2，t3，t4 的 join 表 join，t1 为外表，再和 t5 join，t5 为内表。leading((t1 (t2 t3) t4 t5)) leading((t3 t2))表示：t2，t3 先 join，t2 做内表；然后再和 t1 join，t2，t3 的 join 表做内表；之后依次跟 t4，t5 做 join，t4，t5 做内表。对示例中的原语句使用如下的 HINT：

```
explain
select /* + leading((((((store_sales store) promotion) item) customer) ad2) store_returns)
leading((store store_sales)) */ i_product_name product_name ...
```

该提示命令表示表之间的 join 关系是：store_sales 和 store 先 join，store_sales 做内表，然后依次与 promotion，item，customer，ad2，store_returns 做 join。生成的计划如图 5-1 所示。

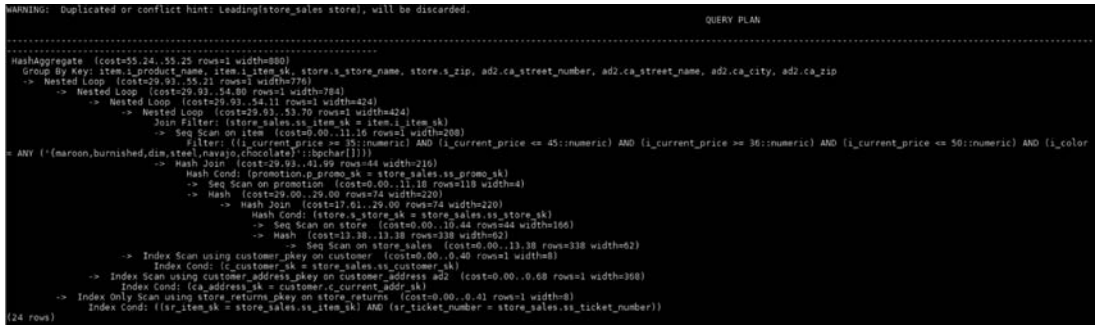

图 5-1　包括顺序提示的执行计划

5.4.3　连接方式提示

连接方式提示用于选择 Join 使用的方法，包括 Nested Loop、Hash Join 和 Merge Join，命令语法如下：

```
[no] nestloop|hashjoin|mergejoin(table_list)
```

（1）no 表示 HINT 的 join 方式不使用。例如，no nestloop(t1 t2 t3)表示生成 t1，t2，t3 三表连接计划时，不使用 nestloop。

（2）table_list 表示 HINT 表集合的字符串，该字符串中的表与 join_table_list 相同，只是中间不允许出现括号指定 join 的优先级。

（3）三表连接计划可能是 t2，t3 先 join，再与 t1 join，或 t1，t2 先 join，再与 t3 join。对于多表连接，HINT 只能指定最后一次 join 使用的连接方式，对于两表连接的方法不 HINT。如果需要，可以单独指定。例如，任意表均不允许 nestloop 连接，且希望 t2，t3 先 join，则增加 HINT：no nestloop(t2 t3)。示例如下：

```
explain
select /* + nestloop(store_sales store_returns item) */ i_product_name product_name ...
```

该 HINT 表示表之间的 join 关系是：store_sales 和 store 先 join，store_sales 做内表，然后依次与 promotion，item，customer，ad2，store_returns 做 join。

5.4.4　行数方式提示

行数方式提示用于指明中间结果集的大小，支持绝对值和相对值的 HINT：

```
rows(table_list #|+|-|* const)
```

(1) #,+,-,* 为进行行数估算 HINT 的四种操作符号。# 表示直接使用后面的行数进行 HINT。+,-,* 表示对原来估算的行数进行加、减、乘操作,运算后的行数最小值为 1 行。table_list 为 HINT 对应的单表或多表 join 结果集,与 Join 方式 HINT 中的 table_list 相同。

(2) const 可以是任意非负数,支持科学计数法。例如,rows(t1 #5)表示指定 t1 表的结果集为 5 行。rows(t1 t2 t3 * 1000)表示指定 t1,t2,t3 join 完的结果集的行数乘以 1000。推荐使用两个表 * 的 HINT。对于两个表的采用 * 操作符的 HINT,只要两个表出现在 join 的两端,都会触发 HINT。例如,设置 HINT 为 rows(t1 t2 * 3),对于(t1 t3 t4)和(t2 t5 t6)join 时,由于 t1 和 t2 出现在 join 的两端,所以其 join 的结果集也会应用该 HINT 规则乘以 3。rows HINT 支持在单表、多表、function table 及 subquery scan table 的结果集上指定 HINT。对示例中的原语句使用如下的 HINT:

```
explain
select /* + rows(store_sales store_returns * 50) */ i_product_name product_name ...
```

该 HINT 表示表之间的 join 关系是: store_sales 和 store 先 join,store_sales 做内表,然后依次与 promotion,item,customer,ad2,store_returns 做 join。

5.4.5　提示命令的错误、冲突及告警

提示命令的结果会体现在计划的变化上,可以通过解释语句(EXPLAIN)查看变化。提示中的错误不会影响语句的执行,只是不能生效,该错误会根据语句类型以不同方式提示用户。对于解释语句,提示命令的错误会以警告(warning)形式显示在界面上; 对于非解释语句,提示命令的错误会以调试级别日志显示在日志中,关键字为 PLANHINT。HINT 的错误分为以下 6 种类型。

(1) 语法错误: 语法规则树归约失败,会报错,指出出错的位置。例如,HINT 关键字错误,leading HINT 或 join HINT 指定 2 个表以下,其他 HINT 未指定表等。一旦发现语法错误,则立即终止 HINT 的解析,所以此时只有错误前面的解析完的 HINT 有效,例如下面的命令:

```
leading((t1 t2)) nestloop(t1) rows(t1 t2 #10)
```

若 nestloop(t1)存在语法错误,则终止解析,可用提示命令只有之前解析的 leading((t1 t2))。

(2) 语义错误: 表不存在,存在多个,或在 leading 或 join 中出现多次,均会报语义错误。scanHINT 中的 index 不存在,会报语义错误。另外,如果子查询提升后,同一层出现多个名称相同的表,且其中某个表需要被 HINT,HINT 会存在歧义,无法使用,需要为相同表增加别名规避。

(3) 提示命令重复或冲突：如果存在提示命令重复或冲突，则只有第一个提示命令生效，其他提示命令均会失效。提示重复是指提示命令的方法及表名均相同。例如，nestloop (t1 t2) nestloop(t1 t2)。提示冲突是指 table list 一样的提示命令，存在不一样的提示，提示的冲突仅对于每一类提示方法检测冲突。例如，nestloop (t1 t2) hashjoin (t1 t2)，则后面与前面冲突，此时 hashjoin 的提示失效。注意，nestloop(t1 t2)和 no mergejoin(t1 t2)不冲突。leading HINT 中的多个表会进行拆解。例如，leading ((t1 t2 t3))会拆解成 leading((t1 t2)) leading(((t1 t2) t3))，此时如果存在 leading((t2 t1))，则两者冲突，后面的会被丢弃。(例外：指定内外表的 HINT 若与不指定内外表的 HINT 重复，则始终丢弃不指定内外表的 HINT。)

(4) 子链接提升后 HINT 失效：子链接提升后的提示失效，会给出提示。这种情况通常出现在子链接中存在多个表链接的场景中。提升后，子链接中的多个表不再作为一个整体出现在链接中。

(5) 列类型不支持重分布：对于 skew HINT 来说，目的是进行重分布时的调优，所以当提示列的类型不支持重分布时，提示命令将无效。

(6) 提示未被使用：非等值 join 使用 hashjoin HINT 或 mergejoin HINT；不包含索引的表使用 indexscan HINT 或 indexonlyscan HINT。通常，只有在索引列上使用过滤条件才会生成相应的索引路径，全表扫描将不会使用索引，因此使用 indexscan HINT 或 indexonlyscan HINT 将不会使用索引。indexonlyscan 只有输出列仅包含索引列时才会使用，否则指定时仅尝试有等值连接条件的表连接，此时没有关联条件的表之间的路径将不会生成，所以指定相应的 leading，join，rows HINT 将不使用。例如，t1，t2，t3 表 join，t1 和 t2，t2 和 t3 有等值连接条件，则 t1 和 t3 不会优先连接，leading(t1 t3)不会被使用。

5.5 自动参数优化

5.5.1 工作原理

传统的数据库调优通常依靠雇佣专家(DBA)完成：专家针对指定的负载，在线下反复进行瓶颈检测、参数调整和性能对比，直至达到满意效果，这是一项非常耗时的工作，而且严重依赖专家自身的经验和知识。此外，在负载动态变化的场景下，要面临更加繁多的数据库状态和负载类型，极大地增加了这项工作的难度。虽然目前有一些基于学习的调优方法，它们由于没有充分利用系统查询负载的信息，而不具备动态适应负载变化的能力。此外，它们直接利用现有的机器学习模型，如高斯过程等，缺少针对调参问题的特征融合模型，提高调参的表现和泛化能力。

为了解决以上挑战，openGauss 提出了一种基于学习的数据库自动调优工具 X-Tuner。它是一款数据库自带的参数调优工具，通过结合深度强化学习和启发式算法，实现在无须人工干预的情况下，获取最佳数据库参数的途径。其核心思想是：基于深度强化学习算法

学习在不同的环境条件下推荐参数的策略。需要注意的是，调优程序是一个独立于数据库内核之外的工具，需要提供数据库及其所在实例的用户名和登录密码信息，以便控制数据库执行基准测试集(Benchmark)进行性能测试；在启动调优程序前，要求用户测试环境交互正常，能够正常跑通基准测试集测试脚本、能够正常连接数据库。

5.5.2　实验部署

启动调优程序之前，可以通过如下命令获取帮助信息，相关参数见表 5-3。

```
python main.py -- help
```

表 5-3　相关参数

参　数	参 数 说 明	取 值 范 围
-mode,-m	指定调优程序运行的模式	train,tune
-config-file,-f	调优程序的配置文件，可选	—
-db-name	指定调优的数据库	—
-db-user	指定调优的数据库用户名	—
-port	数据库的监听端口	—
-host	数据库实例的宿主机 IP	—
-host-user	数据库安装时的 DBA 用户名	—
-host-ssh-port	数据库实例所在宿主机的 SSH 端口号，可选	—
-scenario	指定调优的模式，对应 3 种不同的调优列表，用户可以对该调优列表进行修改	ap,htap,tp
-benchmark	由用户指定的 benchmark 脚本文件名	—
-model-path	调优强化学习模型存储或加载的文件路径	—
-version,-v	返回当前工具的版本号	—

部署 X-Tuner 工具的步骤如下：

(1) 用户进行数据库安全配置，并验证调优程序所在客户机能够正常访问到数据库实例所在的服务器。

(2) 用户向数据库实例导入数据(如 TPC-C、TPC-H)，根据调优程序给出的示例代码编写符合自己实际业务的基准测试集(脚本路径在基准测试集目录中)，手动验证基准测试集可以正常跑通并可获得稳定的测试结果，记录下此时的测试结果，以方便后续对比调优效果。

(3) 用户在确保数据库运行正常并在无其他人使用时备份现有参数，修改调优参数列表配置文件(文件路径在参数目录中，默认配置文件是 knobs_htap.py)，设定需要调整的参数及其范围。

(4) 用户输入数据库链接信息，选择当前调优模式为“训练”或“调优”，启动参数调优程序。例如，在 X-Tuner 根目录中输入如下命令：

```
python main.py -m train -db-name opengauss\
-db-user dba -port 1234 \
-host 192.168.1.2 -host-user opengauss\
-benchmark tpcc -model-path mymodel
```

(5) 若为“训练”模式,则输出训练后的模型,程序退出;若为“调优”模式,则输出调优后的最优参数列表,程序退出。用户通过对比调优结果,自行判断是否应该设置为该参数,并手动设置为推荐参数或重置为调优前的参数。

5.6 查询性能预测

查询性能预测(Predictor)主要预测查询的执行时间,包括工具原理和实验部署两部分。

5.6.1 工作原理

查询性能预测是基于机器学习且具有在线学习能力的查询时间预测工具,通过不断学习数据库内收集的历史执行信息,实现计划的执行时间预测功能。本特性需要拉起进程AiEngine,用于模型的训练和推理。首先,为了保证 openGauss 处于正常状态,用户通过身份验证成功登录 openGauss;用户执行的 SQL 语法正确无报错,且不会导致数据库异常等;历史性能数据窗口内 openGauss 并发量稳定,表结构、表数量不变,数据量无突变,涉及查询性能的 guc 参数不变;进行预测时,需要保证模型已训练并收敛;AiEngine 运行环境稳定。相关接口见表 5-4。

表 5-4 对外接口

Request-API	功 能	Request-API	功 能
/check	检查模型是否被正常拉起	/track_process	查看模型训练日志
/configure	设置模型参数	/setup	加载历史模型
/train	模型训练	/predict	模型预测

AiEngine 进程与内核进程发送请求进行通信,请求样例如下:

```
curl -X POST -d '{"modelName":"modelname"}' -H 'Content-Type: application/json' 'https://
IP-address:port/request-API'
```

使用此功能前,需使用 openssl 工具生成通信双方认证所需的证书,保证通信安全。

(1) 搭建证书生成环境,证书文件的保存路径为 $GAUSSHOME/CA。复制证书生成脚本及相关文件:

```
cp path_to_predictor/install/ssl.sh $GAUSSHOME/
cp path_to_predictor/install/ca_ext.txt $GAUSSHOME/
```

（2）复制配置文件 openssl.cnf 到 $GAUSSHOME 路径下：

```
cp $GAUSSHOME/share/om/openssl.cnf $GAUSSHOME/
```

（3）修改 openssl.conf 配置参数：

```
dir = $GAUSSHOME/CA/demoCA
default_md = sha256
```

至此，通信证书生成环境的准备工作完成。

（4）生成证书及密钥：

```
cd $GAUSSHOME ; sh ssl.sh
```

（5）根据提示设置密码。要求密码至少包含 3 种不同类型的字符，长度至少为 8 位。

```
Please enter your password:
```

（6）根据提示输入选项：

```
Certificate Details:
        Serial Number: 1 (0x1)
        Validity
            Not Before: May 15 08:32:44 2020 GMT
            Not After : May 15 08:32:44 2021 GMT
        Subject:
            countryName               = CN
            stateOrProvinceName       = SZ
            organizationName          = HW
            organizationalUnitName    = GS
            commonName                = CA
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:TRUE
Certificate is to be certified until May 15 08:32:44 2021 GMT (365 days)
Sign the certificate? [y/n]:y
1 out of 1 certificate requests certified, commit? [y/n]y
```

（7）输入拉起 AiEngine 的 IP 地址，如 IP 为 127.0.0.1。

```
Please enter your aiEngine IP: 127.0.0.1
```

（8）根据提示输入选项：

```
Certificate Details:
        Serial Number: 2 (0x2)
        Validity
```

```
            Not Before: May 15 08:38:07 2020 GMT
            Not After : May 13 08:38:07 2030 GMT
        Subject:
            countryName               = CN
            stateOrProvinceName       = SZ
            organizationName          = HW
            organizationalUnitName    = GS
            commonName                = 127.0.0.1
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
Certificate is to be certified until May 13 08:38:07 2030 GMT (3650 days)
Sign the certificate? [y/n]:y 1 out of 1 certificate requests certified, commit? [y/n]y
```

(9) 输入启动 openGauss 的 IP 地址,如 IP 为 127.0.0.1。

```
Please enter your gaussdb IP: 127.0.0.1
        Serial Number: 3 (0x3)
        Validity
            Not Before: May 15 08:41:46 2020 GMT
            Not After : May 13 08:41:46 2030 GMT
        Subject:
            countryName               = CN
            stateOrProvinceName       = SZ
            organizationName          = HW
            organizationalUnitName    = GS
            commonName                = 127.0.0.1
        X509v3 extensions:
            X509v3 Basic Constraints:
                CA:FALSE
Certificate is to be certified until May 13 08:41:46 2030 GMT (3650 days)
Sign the certificate? [y/n]:y
1 out of 1 certificate requests certified, commit? [y/n]y
```

5.6.2 实验部署

部署 Predictor 工具的步骤如下:

(1) 打开数据收集,设置 ActiveSQL operator 信息相关参数:

```
enable_resource_track = on
resource_track_level = operator
```

(2) 关闭数据收集,设置 ActiveSQL operator 信息相关参数:

```
enable_resource_track = off
resource_track_level = query
```

(3) 执行业务查询语句,等待 3min 后查看当前节点上的数据。

```
select * from gs_wlm_plan_operator_info;
```

(4) 数据持久化保存,设置 ActiveSQL operator 信息相关参数。

```
enable_resource_track = on
resource_track_level = operator
enable_resource_record = on
resource_track_duration = 0(默认值为 60s)
resource_track_cost = 10(默认值为 100000)
```

(5) 执行业务查询语句,等待 3min 后查看当前节点上的数据:

```
select * from gs_wlm_plan_operator_info;
```

1. 模型管理(系统管理员用户)

模型管理是指对训练好的机器学习或强化学习模型的存储、调度及使用策略。模型管理操作需要在数据库正常的状态下进行。

(1) 新增加一个机器学习模型:

```
INSERT INTO gs_opt_model values('rlstm', 'model_name', 'datname', '127.0.0.1', 5000, 2000, 1, -1,
64, 512, 0 , false, false, '{S, T}', '{0,0}', '{0,0}', 'Text');
```

(2) 修改一个机器学习模型的超参数或数据库统计参数:

```
UPDATE gs_opt_model SET <attribute> = <value> WHERE model_name = <target_model_name>;
```

(3) 删除一个机器学习模型:

```
DELETE FROM gs_opt_model WHERE model_name = <target_model_name>;
```

(4) 查询现有机器学习模型及其工作状态:

```
SELECT * FROM gs_opt_model;
```

2. 模型训练(系统管理员用户)

模型训练是指当要处理新的业务场景或者机器学习表现变差时,系统管理人员需要训练现有模型,以满足用户需求。

(1) 配置/添加模型训练参数:参考模型管理(系统管理员用户)进行模型添加、模型参数修改,来指定训练参数。

```
INSERT INTO gs_opt_model values('rlstm', 'default', 'opengauss', '127.0.0.1', 5000, 2000, 1, -
1, 64, 512, 0 , false, false, '{S, T}', '{0,0}', '{0,0}', 'Text');
```

(2) 训练参数更新。

```
UPDATE gs_opt_model SET < attribute > = < value > WHERE model_name = < target_model_name >;
```

(3) 前提条件为数据库状态正常且历史数据正常收集时,删除原有的 encoding 数据。

```
DELETE FROM gs_wlm_plan_encoding_table;
```

(4) 进行数据编码,指定数据库名。

```
SELECT gather_encoding_info('opengauss');
```

(5) 开始模型训练。

```
SELECT model_train_opt('rlstm', 'default');
```

(6) 查看模型训练状态,返回 TensorBoard 工具(见图 5-2)所用 URL。

```
SELECT * FROM track_model_train_opt('rlstm', 'default');
```

```
postgres=# select track_model_train_opt('rlstm', 'openai');
   track_model_train_opt
-------------------------
 http://10.90.56.229:6006/
(1 row)
```

图 5-2 TensorBoard 工具所用 URL

(7) 打开 URL 查看模型训练状态,返回 TensorBoard 可视化训练界面。

3. 模型预测(系统管理员用户|普通用户)

模型预测功能需在数据库状态正常、指定模型已被训练且收敛的条件下进行。目前,模型训练参数的标签设置中需要包含 S 标签,解释命令中才可显示 p-time 预测值。例如:

```
INSERT INTO gs_opt_model values('rlstm','default','opengauss','127.0.0.1',5000,1000,1, - 1,50,
500,0 ,false,false,'{S,T}','{0,0}','{0,0}','Text');
```

(1) 调用解释命令接口:

```
explain (analyze on, predictor < model_name >)
```

(2) 预期结果:

```
Row Adapter (cost = 110481.35..110481.35 rows = 100 p - time = 99..182 width = 100) (actual time =
375.158..375.160 rows = 2 loops = 1)
```

（3）检查 AiEngine 是否可连接：

```
opengauss = # select check_engine_status('aiEngine - ip - address',running - port);
```

（4）查看模型对应日志在 AiEngine 侧的保存路径：

```
opengauss = # select track_model_train_opt('template_name', 'model_name');
```

5.7　索引推荐

本节介绍索引推荐(Index-Advisor)的功能，共包含 3 个子功能：单查询索引推荐、虚拟索引和负载级别索引推荐。

5.7.1　单查询索引推荐

单查询索引推荐功能支持用户在数据库中直接进行操作，本功能基于查询语句的语义信息和数据库的统计信息，对用户输入的单条查询语句生成推荐的索引。本功能涉及的函数接口见表 5-5。

表 5-5　函数接口

函　　数	参　　数	功　　能
gs_index_advise()	SQL 语句	针对单条 SQL 生成推荐索引

使用上述函数，获取针对该查询生成的推荐索引，推荐结果由索引的表名和列名组成。例如：

```
postgres = > select * from gs_index_advise('SELECT c_discount from bmsql_customer where c_w_id = 10');
     table        |  column
--------------+-----------
 bmsql_customer  | (c_w_id)
(1 row)
```

上述结果表明：应在 bmsql_customer 的 c_w_id 列上创建索引。例如，可以通过下述 SQL 语句创建索引：

```
CREATE INDEX idx on bmsql_customer(c_w_id);
```

某些查询语句也可能被推荐创建联合索引，例如：

```
postgres = # select * from gs_index_advise('select name, age, sex from t1 where age > = 18 and
age < 35 and sex = ''f'';');
```

```
table    | column
--------+-------------
 t1      | (age, sex)
(1 row)
```

上述语句表明,应该在表 t1 上创建一个联合索引 (age,sex)。可以通过下述命令创建:

```
CREATE INDEX idx1 on t1(age, sex);
```

5.7.2 虚拟索引

虚拟索引功能支持用户在数据库中直接进行操作,本功能将模拟真实索引的建立,避免创建真实索引所需的时间和空间开销。用户基于虚拟索引,可通过优化器评估该索引对指定查询语句的影响。虚拟索引功能的接口见表 5-6。

表 5-6 虚拟索引功能的接口

函 数 名	参 数	功 能
hypopg_create_index	创建索引语句的字符串	创建虚拟索引
hypopg_display_index	无	显示所有创建的虚拟索引信息
hypopg_drop_index	索引的 oid	删除指定的虚拟索引
hypopg_reset_index	无	清除所有虚拟索引
hypopg_estimate_size	索引的 oid	估计创建指定索引所需的空间大小

使用函数 hypopg_create_index()创建虚拟索引。例如:

```
postgres=> select * from hypopg_create_index('create index on bmsql_customer(c_w_id)');
 indexrelid     |                indexname
------------+---------------------------------------
     329726     | <329726>btree_bmsql_customer_c_w_id
(1 row)
```

开启 GUC 参数 enable_hypo_index,该参数控制数据库的优化器进行解释时是否考虑创建的虚拟索引。通过对特定的查询语句执行解释,用户可根据优化器给出的执行计划评估该索引是否能够提升该查询语句的执行效率。例如:

```
postgres=> set enable_hypo_index = on;
SET
```

开启 GUC 参数前,执行 EXPLAIN+查询语句:

```
postgres=> explain SELECT c_discount from bmsql_customer where c_w_id = 10;
                                QUERY PLAN
```

```
------------------------------------------------------------------------
 Seq Scan on bmsql_customer  (cost = 0.00..52963.06 rows = 31224 width = 4)
   Filter: (c_w_id = 10)
(2 rows)
```

开启 GUC 参数后，执行 EXPLAIN + 查询语句：

```
postgres = > explain SELECT c_discount from bmsql_customer where c_w_id = 10;
                                QUERY PLAN
------------------------------------------------------------------------
 [Bypass]
 Index Scan using < 329726 > btree_bmsql_customer_c_w_id on bmsql_customer  (cost = 0.00..
39678.69 rows = 31224 width = 4)
   Index Cond: (c_w_id = 10)
(3 rows)
```

通过对比两个执行计划可以观察到，该索引预计会降低指定查询语句的执行代价，用户可考虑创建对应的真实索引。使用函数 hypopg_display_index()可展示所有创建过的虚拟索引。例如：

```
postgres = > select * from hypopg_display_index();
                indexname                | indexrelid |     table      | column
-----------------------------------------+------------+----------------+-----------
 < 329726 > btree_bmsql_customer_c_w_id      | 329726     | bmsql_customer | (c_w_id)
 < 329729 > btree_bmsql_customer_c_d_id_c_w_id | 329729     | bmsql_customer  | (c_d_id, c_w_id)
(2 rows)
```

使用函数 hypopg_estimate_size()可估计创建虚拟索引所需的空间大小(单位：字节)。例如：

```
postgres = > select * from hypopg_estimate_size(329730);
hypopg_estimate_size
-----------------------
             15687680
(1 row)
```

删除虚拟索引。使用函数 hypopg_drop_index()删除指定 oid 的虚拟索引。例如：

```
postgres = > select * from hypopg_drop_index(329726);
 hypopg_drop_index
--------------------
 t
(1 row)
```

使用函数 hypopg_reset_index()一次性清除创建的所有虚拟索引。例如：

```
postgres=> select * from hypopg_reset_index();
 hypopg_reset_index
--------------------

(1 row)
```

5.7.3 负载级别索引推荐

对于负载级别索引推荐，用户可通过运行数据库外的脚本使用此功能。本功能将包含有多条 DML 语句的 workload 作为输入，最终生成一批可对整体 workload 的执行表现进行优化的索引。

首先，在构建负载级别索引之前，数据库需要满足 3 个前提：

(1) 数据库状态正常，客户端能够正常连接。

(2) 当前执行用户下安装有 gsql 工具，该工具路径已被加入 PATH 环境变量中。

(3) 具备 Python 3.6+的环境。

其次，准备好包含有多条 DML 语句的文件作为输入的 workload，文件中每条语句占据一行。用户可从数据库的离线日志中获得历史的业务语句。运行 Python 脚本 index_advisor_workload.py，命令如下：

```
python index_advisor_workload.py [p PORT] [d DATABASE] [f FILE] [--h HOST] [-U USERNAME]
[-W PASSWORD]
[--max_index_num MAX_INDEX_NUM] [--multi_iter_mode]
```

其中输入参数依次为：PORT，连接数据库的端口号；DATABASE，连接数据库的名字；FILE，包含 workload 语句的文件路径；HOST(可选)，连接数据库的主机号；USERNAME(可选)，连接数据库的用户名；PASSWORD(可选)，连接数据库用户的密码；MAX_INDEX_NUM(可选)，最大的推荐索引数目；multi_iter_mode(可选)，算法模式，可通过是否设置该参数来切换算法。例如：

```
python index_advisor_workload.py 6001 postgres tpcc_log.txt --max_index_num 10 --multi_
iter_mode
```

推荐结果为一批索引，以多个创建索引语句的格式显示在屏幕上，示例结果如下。

```
create index ind0 on bmsql_stock(s_i_id,s_w_id);
create index ind1 on bmsql_customer(c_w_id,c_id,c_d_id);
create index ind2 on bmsql_order_line(ol_w_id,ol_o_id,ol_d_id);
create index ind3 on bmsql_item(i_id);
```

```
create index ind4 on bmsql_oorder(o_w_id,o_id,o_d_id);
create index ind5 on bmsql_new_order(no_w_id,no_d_id,no_o_id);
create index ind6 on bmsql_customer(c_w_id,c_d_id,c_last,c_first);
create index ind7 on bmsql_new_order(no_w_id);
create index ind8 on bmsql_oorder(o_w_id,o_c_id,o_d_id);
create index ind9 on bmsql_district(d_w_id);
```

5.8　小结

本章概要介绍了 openGauss 性能优化的相关功能，包括查询解释、查询分析、参数配置、查询性能预测等。面对大规模的数据体量和多样的用户负载，首先，5.2 节和 5.3 节分别介绍了查询解释命令和查询分析命令的功能和用法；然后，为了进一步解决数据库配置导致的查询效率低下问题，5.5 节介绍了自动参数优化工具的功能和使用方法；最后，为了预判查询的执行效率，5.6 节介绍了 openGauss 内置的查询性能预测工具的基本原理和使用方法。

5.9　习题

1. 以下哪种方式不属于解释命令的功能？（　　）
   A. 扫描方式　　B. 连接方式
   C. 数据存储的分区号　　D. 预测的执行开销
2. （多选题）以下属于分析命令的功能有（　　）。
   A. 收集表的统计信息　　B. 检测数据文件是否损坏
   C. 打印缓存区的使用情况　　D. 更改连接方式
3. 如果需要提高代价估计的准确度，则使用（　　）优化提示功能。
   A. 连接顺序提示　　B. 连接方式提示
   C. 中间行数提示　　D. 最终行数提示
4. 以下不属于优化提示命令的功能有（　　）。
   A. 连接顺序提示　　B. 连接方式提示
   C. 中间行数提示　　D. 最终行数提示
5. （多选题）以下有关查询优化命令的说法，不正确的是（　　）。
   A. 提示命令的结果不可以通过 explain 查看
   B. openGauss 可以执行两条冲突的优化提示命令
   C. explain 可以获得计划的实际执行信息
   D. 如果表的分布列与连接列相同，就不会生成重分布计划（redistribute）

6.（多选题）以下有关参数调优的说法，不正确的是（　　）。

A. openGauss 支持基于搜索的参数调优算法

B. 传统的数据库调优通常依靠自动化算法独立完成

C. openGauss 内置的深度调优模型不需要训练

D. openGauss 的调优模块需要和用户定期交互

7.（多选题）以下有关查询时间预测的说法，不正确的是（　　）。

A. 并发和串行场景下，同一个查询的执行时间总是一样的

B. 查询的执行时间和数据库参数无关

C. 预测的查询时间可以用于负载调度、性能监控等模块

D. 预测的查询时间可以作为实际执行时间

第 6 章

openGauss 维护

6.1 openGauss 运行健康状态检查

openGauss 提供了相关的工具用于查询数据库和实例的状态，以确认其处于正常的运行状态，并可以对外提供数据服务。若发现异常，可利用此工具执行诊断操作，尽快恢复数据服务。本章用的是 openGauss 提供的 gs_check 工具。

6.1.1 注意事项

使用 gs_check 工具有以下几个注意事项。

(1) **执行用户**：扩容新节点检查必须由 root 用户执行，其他场景则必须由 omm 用户执行。

(2) **查询选项**：必须使用-i 或者-e 选项，其中

-i：检查指定的单项；

-e：检查指定场景配置中的多项。

(3) **root 权限免除**：如果-i 选项的参数中不包含 root 类检查项，或-e 指定的场景配置列表中没有 root 类检查项，则不需要交互输入 root 权限的用户和密码。在操作中可使用--skip-root-items 选项跳过检查项中包含的 root 类检查，以免除请求 root 权限。

6.1.2 操作步骤

下面讲解 3 种运行健康状态检查的基本操作。

1. 单项检查

单项检查的操作步骤如下：

(1) 以操作系统用户 omm 登录数据库主节点。

(2) 使用 gs_check 工具的-i 选项对数据库状态进行单项检查，命令如下：

```
gs_check -i CheckClusterState
```

其中，-i 指定检查项(注意区分大小写)，例如-i CheckClusterState 或-i CheckCPU。取值范围为支持的所有检查项的名称，具体可参见表 6-2。

结果示例如下：

```
[omm@ecs-c32a ~]$ gs_check -i CheckClusterState
Parsing the check items config file successfully
Distribute the context file to remote hosts successfully
Start to health check for the cluster. Total Items:1 Nodes:1

Checking...                [ ========================== ] 1/1
Start to analysis the check result
CheckClusterState.....................OK
The item run on 1 nodes.   success: 1

Analysis the check result successfully
Success. All check items run completed. Total:1   Success:1
For more information please refer to /opt/huawei/wisequery/script/gspylib/inspection/output/
CheckReport_20201118772852075250.tar.gz
```

可同时指定多个单项,注意区分大小写,例如-i CheckClusterState -i CheckCPU 或-i CheckClusterState,CheckCPU。结果示例如下:

```
[omm@ecs-c32a ~]$ gs_check -i CheckClusterState -i CheckCPU
# 或 gs_check -i CheckClusterState,CheckCPU
Parsing the check items config file successfully
Distribute the context file to remote hosts successfully
Start to health check for the cluster. Total Items:2 Nodes:1

Checking...                [ ========================== ] 2/2
Start to analysis the check result
CheckClusterState.....................OK
The item run on 1 nodes.   success: 1

CheckCPU..............................OK
The item run on 1 nodes.   success: 1

Analysis the check result successfully
Success.   All check items run completed. Total:2   Success:2
For more information please refer to /opt/huawei/wisequery/script/gspylib/inspection/output/
CheckReport_20201118793872230590.tar.gz
```

2. 场景检查(多项)

场景检查的操作步骤如下:

(1) 以操作系统用户 omm 登录数据库主节点。

(2) 使用 gs_check 工具的-e 选项对数据库状态进行场景检查,命令如下:

```
gs_check -e inspect
```

其中，-e 指定检查项，注意区分大小写，例如-e inspect 或-e upgrade。取值范围为支持的场景检查的所有名称，默认场景包括 inspect（例行检查）、upgrade（升级前检查）、binary_upgrade（就地升级前检查）、health（健康检查）、install（安装检查）等，用户可以根据需求自己编写场景。

结果示例如下：

```
[omm@ecs-c32a ~]$ gs_check -e inspect
Parsing the check items config file successfully
The below items require root privileges to execute: [CheckBlockdev CheckIOrequestqueue
CheckIOConfigure CheckMTU CheckRXTX CheckMultiQueue CheckFirewall CheckSshdService
CheckSshdConfig CheckCrondService CheckNoCheckSum CheckSctpService CheckMaxProcMemory
CheckBootItems CheckFilehandle CheckNICModel CheckDropCache]
Please enter root privileges user[root]:
Please enter password for user[root]:
Check root password connection successfully
Distribute the context file to remote hosts successfully
Start to health check for the cluster. Total Items:59 Nodes:1

Checking...                [ ========================== ] 59/59
Start to analysis the check result
CheckClusterState.....OK
The item run on 1 nodes.  success: 1

CheckDBParams.....OK
The item run on 1 nodes.  success: 1

CheckDebugSwitch.....OK
The item run on 1 nodes.  success: 1

…… (部分结果省略)

CheckDropCache.....WARNING
The item run on 1 nodes.  warning: 1
The warning[ecs-c32a] value:
No DropCache process is running

CheckMpprcFile.....NG
The item run on 1 nodes.  ng: 1
The ng[ecs-c32a] value:
There is no mpprc file

Analysis the check result successfully
Failed.  All check items run completed. Total:59  Success:49  Warning:2  NG:8
For more information please refer to /opt/huawei/wisequery/script/gspylib/inspection/output/
CheckReport_inspect_20201118794892240592.tar.gz
```

6.1.3 常见错误与异常处理

下面介绍常见错误与异常处理。

(1) 单项检查 CheckCPU 报错：sar 工具未找到，结果如下：

```
[omm@ecs-c32a ~]$ gs_check -i CheckCPU
Parsing the check items config file successfully
Distribute the context file to remote hosts successfully
Start to health check for the cluster. Total Items:1 Nodes:1

Checking...                [ ========================== ] 1/1
Start to analysis the check result
CheckCPU.............................ERROR
The item run on 1 nodes.  error: 1
The error[ecs-c32a] value:
[GAUSS-53025]: ERROR: Execute Shell command faild: export LC_ALL=C; sar 1 5 2>&1 , the
exception is: /bin/sh: sar: command not found

Analysis the check result successfully
Failed.  All check items run completed. Total:1       Error:1
For more information please refer to /opt/huawei/wisequery/script/gspylib/inspection/output/
CheckReport_20201118781612133728.tar.gz
```

可能的问题：在某些 Linux 发行版(例如 openEuler)中 sar 工具未安装或 PATH 环境变量设置有误。

解决办法如下。

① 安装、启动 sar 工具，检查 PATH 环境变量设置是否正确，命令如下：

```
yum install sysstat       # openEuler, CentOS
service sysstat start     # 启动 sysstat 服务
```

② 在 omm 用户下检查是否能正常使用 sar 工具，命令如下：

```
sar -V                          # 检查 sar 是否正常运行
```

③ 再次进行单项检查，观察结果是否正常，命令如下：

```
[omm@ecs-c32a ~]$ gs_check -i CheckCPU
```

(2) 若检查结果出现异常，可按表 6-1 进行修复。

表 6-2 列出了 gs_check 工具支持的检查项。

表 6-1　检查结果异常与处理

检　查　项	异 常 状 态	处 理 方 法
CheckClusterState （检查 openGauss 状态）	openGauss 未启动或 openGauss 实例未启动	使用以下命令启动 openGauss 及实例： gs_om -t start
	openGauss 状态异常或 openGauss 实例异常	检查各主机、实例状态，根据状态信息进行排查，命令如下： gs_check -i CheckClusterState
CheckDBParams （检查 openGauss 参数）	数据库参数错误	通过 gs_guc 工具修改数据库参数为指定值
CheckDebugSwitch （检查日志级别）	日志级别不正确	通过 gs_guc 工具将 log_min_messages 参数改为指定内容
CheckDirPermissions （检查目录权限）	路径权限错误	修改对应目录权限为指定数值（750/700），命令如下： chmod 750 ＜DIR＞
CheckReadonlyMode （检查只读模式）	只读模式被打开	确认数据库节点所在磁盘使用率未超阈值（默认 60%）且未执行其他维护操作，命令如下： gs_check -i CheckDataDiskUsage ps ux 使用 gs_guc 工具关闭 openGauss 只读模式，命令如下： gs_guc reload -N all -l all -c 'default_transaction_read_only = off'
CheckEnvProfile （检查环境变量）	环境变量不一致	重新执行前更新环境变量信息

表 6-2　gs_check 工具支持的检查项

状态	检　查　项	检 查 内 容
os	CheckCPU （检查 CPU 使用率）	检查主机 CPU 占用率，如果 idle 值大于 30%并且 iowait 值小于 30%，则检查项通过，否则检查项不通过
	CheckFirewall （检查防火墙状态）	检查主机防火墙状态，如果防火墙关闭，则检查项通过，否则检查项不通过
	CheckTimeZone （检查时区一致性）	检查 openGauss 内各节点的时区，如果时区一致，则检查项通过，否则检查项不通过
	CheckSysParams （检查系统参数）	检查各节点操作系统参数，判断是否等于预期值，并根据检查项具体标准给出结果，详见操作系统参数
	CheckOSVer （检查操作系统版本）	检查 openGauss 内各个节点的操作系统版本信息，如果满足版本兼容列表且 openGauss 在同一混搭列表中，则检查项通过，否则检查项不通过
	CheckNTPD （检查 NTPD 服务）	检查系统 NTPD 服务，如果服务开启且各节点时间误差在 1min 内，则检查项通过，否则检查项不通过

续表

状态	检 查 项	检查内容
os	CheckTHP (检查 THP 服务)	检查系统 THP 服务,如果服务开启,则检查项通过,否则检查项不通过
	CheckSshdService (检查 sshd 服务是否已启动)	检查系统是否存在 sshd 服务,若存在,则检查项通过,否则检查项不通过
	CheckCrondService (检查 crontab 服务是否已启动)	检查系统是否存在 crontab 服务,若存在,则检查项通过,否则检查项不通过
	CheckCrontabLeft (检查 crontab 是否残留 Gauss 相关信息)	检查 crontab 是否残留 Gauss 相关信息,若无该信息,则检查项通过,否则检查项不通过
	CheckDirLeft (检查文件目录是否残留)	检查文件目录(/opt/huawei/Bigdata/、/var/log/Bigdata/、/home/omm)是否存在,若不存在(若 mount 目录包含此目录,则忽略),则检查项通过,否则检查项不通过
	CheckProcessLeft (检查进程是否有残留)	检查是否残留 gaussdb 和 omm 进程,若未残留,则检查项通过,否则检查项不通过
	CheckStack (栈深度检查)	检查栈深度,若各个节点不一致,则报出警告,若大于或等于 3072,则检查项通过,否则检查项不通过
	CheckNoCheckSum (检查 nochecksum 值是否为预期值且一致)	检查 nochecksum 值,分下列情况: Redhat 6.4/6.5 且使用 bond 网卡时,若各个节点都为 Y,则检查项通过,否则检查项不通过; 其他系统:若各个节点都为 N,则检查项通过,否则检查项不通过
	CheckOmmUserExist (检查 omm 用户是否存在)	检查是否存在 omm 用户,若不存在 omm 用户,则检查项通过,否则检查项不通过
	CheckPortConflict (检查数据库节点端口是否占用)	检查数据库节点端口是否已被占用,若未占用,则检查项通过,否则检查项不通过
	CheckSysPortRange (检查 ip_local_port_range 设置范围)	检查 ip_local_port_range 系统参数范围,若范围为 26000～65535,则检查项通过,否则检查项不通过
	CheckEtcHosts (检查/etc/hosts 中是否有重复地址及 localhost 配置)	检查"/etc/hosts"文件,若没有配置 localhost,则检查项不通过,若存在带有"#openGauss"注释的映射,则检查项不通过,若 IP 地址相同主机名(hostname)不同,则检查项不通过,否则检查项通过,若主机名相同但 IP 地址不同,则检查项不通过
	CheckCpuCount (检查 CPU 核数)	检查 CPU 核心,若与可用 CPU 不符,则检查项不通过;若与可用 CPU 相符但存在不可用信息,则报出警告。若所有节点 CPU 信息不相同,则检查项不通过
	CheckHyperThread (检查超线程是否打开)	检查超线程,若超线程打开,则检查项通过,否则检查项不通过
	CheckMemInfo (检查内存总大小)	检查各节点总内存大小是否一致,若检查结果一致,则检查项通过,否则报出警告