# 云数据库 GaussDB

# 工具参考（集中式_V2.0-8.x）

**文档版本** 01

**发布日期** 2024-11-05

**华为云计算技术有限公司**

### 商标声明


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

### 注意

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

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

## 华为云计算技术有限公司

地址： 贵州省贵安新区黔中大道交兴功路华为云数据中心 邮编：550029

网址： https://www.huaweicloud.com/

# 目 录



  

# 1 gsql

gsql是GaussDB提供在命令行下运行的数据库连接工具，可以通过此工具连接服务器并对其进行操作和维护，除了具备操作数据库的基本功能，gsql还提供了若干**高级特性**，便于用户使用。

## 1.1 gsql 概述

### 基本功能

- **连接数据库：**默认只支持从服务器本机连接，如果需要连接到远端的数据库，必须在服务端进行配置。详细操作请参见《开发指南》中“数据库使用入门 > 连接数据库 > 使用gsql连接 > 远程连接数据库”章节。

  **说明**

  gsql创建连接时，会有5分钟超时时间。如果在这个时间内，数据库未正确地接受连接并对身份进行认证，gsql将超时退出。

  针对此问题，可以参考**常见问题处理**。
- **执行SQL语句：**支持交互式地键入并执行SQL语句，也可以执行一个文件中指定的SQL语句。
- **执行元命令：**元命令可以帮助管理员查看数据库对象的信息、查询缓存区信息、格式化SQL输出结果，以及连接到新的数据库等。元命令的详细说明请参见**元命令参考**。

### 高级特性

gsql的高级特性如**表1-1**所示。

  

**表 1-1** gsql 高级特性

| 特性名称 | 描述 |
| --- | --- |
| 变量 | gsql提供类似于Linux的shell命令的变量特性，可以使用gsql的元命令\set设置一个变量，格式如下：<br>**\set** *varname value*<br>删除由\set命令设置的变量请使用如下方式：<br>**\unset** *varname*<br>**说明**<br>• 变量只是简单的名称/值对，这里的值可以是任意长度。<br>• 变量名称必须由字母（包括非拉丁字母）、数字和下划线组成，且对大小写敏感。<br>• 如果使用\set varname的格式（不带第二个参数），则只是设置这个变量而没有给变量赋值。<br>• 可以使用不带参数的\set来显示所有变量的值。<br>变量的示例和详细说明请参见**变量**。 |
| SQL代换 | 利用gsql的变量特性，可以将常用的SQL语句设置为变量，以简化操作。<br>SQL代换的示例和详细说明请参见**SQL代换**。 |
| 自定义提示符 | gsql使用的提示符支持用户自定义。可以通过修改gsql预留的三个变量PROMPT1、PROMPT2、PROMPT3来改变提示符。<br>这三个变量的值可以用户自定义，也可以使用gsql预定义的值。详细请参见**提示符**。 |
| 客户端操作历史记录 | gsql支持客户端操作历史记录，当客户端连接时指定“-r”参数，此功能被打开。可以通过\set设置记录历史的条数，例如，\set HISTSIZE 50，将记录历史的条数设置为50，\set HISTSIZE 0，不记录历史。<br>**说明**<br>• 客户端操作历史记录条数默认设置为32条，最多支持记录500条。当客户端交互式输入包含中文字符时，只支持UTF-8 的编码环境。<br>• 出于安全考虑，将包含PASSWORD、IDENTIFIED、GS_ENCRYPT_AES128、GS_DECRYPT_AES128、GS_ENCRYPT、GS_DECRYPT、GS_ENCRYPT_BYTEA、GS_DECRYPT_BYTEA、PG_CREATE_PHYSICAL_REPLICATION_SLOT_EXTERN、SECRET_ACCESS_KEY、SECRETKEY、CREATE_CREDENTIAL、ACCESSKEY、SECRET_KEY等字符串（不区分大小写）的SQL语句记录识别为包含敏感信息的语句，不会记录到历史信息中，即不能通过上下翻回显。 |

- 变量

  可以使用gsql元命令\set设置一个变量。例如把变量foo的值设置为bar：

  ```
  gaussdb=# \set foo bar
  ```

  要引用变量的值，在变量前面加冒号。例如查看变量的值：

  ```
  gaussdb=# \echo :foo
  bar
  ```

  这种变量的引用方法适用于规则的SQL语句和除\copy、\ef、\help、\sf、\!以外的元命令。

  gsql预定义了一些特殊变量，同时也规划了变量的取值。为了保证和后续版本最大限度地兼容，请避免以其他目的使用这些变量。所有特殊变量见**表1-2**。

  

**说明**

- 所有特殊变量都由大写字母、数字和下划线组成。
- 要查看特殊变量的默认值，请使用元命令**\echo :***varname*（例如\echo :DBNAME）。

**表 1-2** 特殊变量设置

| 变量 | 设置方法 | 变量说明 |
| --- | --- | --- |
| DBNAME | \set DBNAME *dbname* | 当前连接的数据库的名称。每次连接数据库时都会被重新设置。 |
| ECHO | \set ECHO all \| queries | • 如果设置为all，只显示查询信息。等效于使用gsql连接数据库时指定-a参数。<br>• 如果设置为queries，显示命令行和查询信息。等效于使用gsql连接数据库时指定-e参数。 |
| ECHO_HIDDEN | \set ECHO_HIDDEN on \| off \| noexec | 当使用元命令查询数据库信息（例如\dg）时，此变量的取值决定了查询的行为：<br>• 设置为on，先显示元命令实际调用的查询语句，然后显示查询结果。等效于使用gsql连接数据库时指定-E参数。<br>• 设置为off，则只显示查询结果。<br>• 设置为noexec，则只显示查询信息，不执行查询操作。 |
| ENCODING | \set ENCODING *encoding* | 当前客户端的字符集编码。 |
| FETCH_COUNT | \set FETCH_COUNT *variable* | • 如果该变量的值为大于0的整数，假设为n，则执行SELECT语句时每次从结果集中取n行到缓存并显示到屏幕。<br>• 如果不设置此变量，或设置的值小于等于0，则执行SELECT语句时一次性把结果都取到缓存。<br>**说明**<br>设置合理的变量值，将减少内存使用量。一般来说，设为100到1000之间的值比较合理。 |
| HISTCONTROL | \set HISTCONTROL ignorespace \| ignoredups \| ignoreboth \| none | • ignorespace：以空格开始的行将不会写入历史列表。<br>• ignoredups：与以前历史记录里匹配的行不会写入历史记录。<br>• ignoreboth、none或者其他值：所有以交互模式读入的行都被保存到历史列表。<br>**说明**<br>none表示不设置HISTCONTROL。 |
| HISTFILE | \set HISTFILE *filename* | 此文件用于存储历史名列表。缺省值是~/.bash_history。 |

  

| 变量 | 设置方法 | 变量说明 |
| --- | --- | --- |
| HISTSIZE | \set HISTSIZE *size* | 保存在历史命令里命令的个数。缺省值是500。 |
| HOST | \set HOST *hostname* | 已连接的数据库主机名称。 |
| IGNOREEOF | \set IGNOREEOF *variable* | • 若设置此变量为数值，假设为10，则在gsql中输入的前9次EOF字符（通常是Ctrl+C）都会被忽略，在第10次按Ctrl+C才能退出gsql程序。<br>• 若设置此变量为非数值，则缺省为10。<br>• 若删除此变量，则向交互的gsql会话发送一个EOF终止应用。 |
| LASTOID | \set LASTOID *oid* | 最后影响的oid值，即为从一条INSERT或lo_import命令返回的值。此变量只保证在下一条SQL语句的结果显示之前有效。 |
| ON_ERROR_ROLLBACK | \set ON_ERROR_ROLLBACK on \| interactive \| off | • 如果是on，当一个事务块里的语句产生错误的时候，这个错误将被忽略而事务继续。<br>• 如果是interactive，这样的错误只是在交互的会话里忽略。<br>• 如果是off（缺省），事务块里一个语句生成的错误将会回滚整个事务。on_error_rollback-on模式是通过在一个事务块的每个命令前隐含地发出一个SAVEPOINT的方式来工作的，在发生错误的时候回滚到该事务块。 |
| ON_ERROR_STOP | \set ON_ERROR_STOP on \| off | • on：命令执行错误时会立即停止，在交互模式下，gsql会立即返回已执行命令的结果。<br>• off（缺省）：命令执行错误时将会跳过错误继续执行。 |
| PORT | \set PORT *port* | 当前连接数据库的端口号。 |
| USER | \set USER *username* | 当前用于连接的数据库用户。 |
| VERBOSITY | \set VERBOSITY terse \| default \| verbose | 这个选项可以设置为值terse、default、verbose之一以控制错误报告的冗余行。<br>• terse：仅返回严重且主要的错误文本以及文本位置（一般适合于单行错误信息）。<br>• default：返回严重且主要的错误文本及其位置，还包括详细的错误细节、错误提示（可能会跨越多行）。<br>• verbose：返回所有的错误信息。 |

  

- SQL代换

  像元命令的参数一样，gsql变量的一个关键特性是可以把gsql变量替换成正规的SQL语句。此外，gsql还提供为变量更换新的别名或其他标识符等功能。使用SQL代换方式替换一个变量的值可在变量前加冒号。例如：

```
gaussdb=# \set foo 'HR.areaS'
gaussdb=# select * from :foo;
 area_id |       area_name
---------+------------------------
       4 | Middle East and Africa
       3 | Asia
       1 | Europe
       2 | Americas
(4 rows)
```

  执行以上命令，将会查询HR.areaS表。

**须知**

变量的值是逐字复制的，甚至可以包含不对称的引号或反斜杠命令。所以必须保证输入的内容有意义。

- 提示符

  通过表1-3的三个变量可以设置gsql的提示符，这些变量是由字符和特殊的转义字符所组成。

**表 1-3** 提示符变量

| 变量 | 描述 | 示例 |
| --- | --- | --- |
| PROMPT1 | gsql请求一个新命令时使用的正常提示符。<br>PROMPT1的默认值为：<br>`%/%R%#` | 使用变量PROMPT1切换提示符：<br>• 提示符变为[local]：<br>`gaussdb=> \set PROMPT1 %M`<br>`[local:/tmp/gaussdba_mppdb]`<br>• 提示符变为name：<br>`gaussdb=> \set PROMPT1 name`<br>`name`<br>• 提示符变为=：<br>`gaussdb=> \set PROMPT1 %R`<br>`=` |
| PROMPT2 | 在一个命令输入期待更多输入时（例如，查询没有用一个分号结束或者引号不完整）显示的提示符。 | 使用变量PROMPT2显示提示符：<br>`gaussdb=# \set PROMPT2 TEST`<br>`gaussdb=# select * from HR.areaS TEST;`<br>`area_id \|  area_name`<br>`---------+--------------------`<br>`1 \| Europe`<br>`2 \| Americas`<br>`4 \| Middle East and Africa`<br>`3 \| Asia`<br>`(4 rows))` |

  

| 变量 | 描述 | 示例 |
| --- | --- | --- |
| PROMPT3 | 当执行COPY命令，并期望在终端输入数据时（例如，COPY FROM STDIN），显示提示符。 | 使用变量PROMPT3显示COPY提示符：<br>gaussdb=# \set PROMPT3 '>>>>'<br>gaussdb=# copy HR.areaS from STDIN;<br>Enter data to be copied followed by a newline.<br>End with a backslash and a period on a line by itself.<br>>>>>1 aa<br>>>>>2 bb<br>>>>>\. |

提示符变量的值是按实际字符显示的，但是，当设置提示符的命令中出现“%”时，变量的值根据“%”后的字符，替换为已定义的内容，已定义的提示符请参见表1-4。

**表 1-4** 已定义的替换

| 符号 | 符号说明 |
| --- | --- |
| %M | 主机的全名（包含域名），若连接是通过Unix域套接字进行的，则全名为[local]，若Unix域套接字不是编译的缺省位置，就是[local:/dir/name]。 |
| %m | 主机名删去第一个点后面的部分。若通过Unix域套接字连接，则为[local]。 |
| %> | 主机正在侦听的端口号。 |
| %n | 数据库会话的用户名。 |
| %/ | 当前数据库名称。 |
| %~ | 类似 %/，如果数据库是缺省数据库时输出的是波浪线~。 |
| %# | 如果会话用户是数据库系统管理员，使用#，否则用>。 |
| %R | • 对于PROMPT1通常是“=”，如果是单行模式则是“^”，如果会话与数据库断开（如果\connect失败可能发生）则是“!”。<br>• 对于PROMPT2该序列被“ -”、“ *”、单引号、双引号或“$”(取决于gsql是否等待更多的输入：查询没有终止、正在一个 /* ... */ 注释里、正在引号或者美元符扩展里)代替。 |
| %x | 事务状态：<br>• 如果不在事务块里，则是一个空字符串。<br>• 如果在事务块里，则是“*”。<br>• 如果在一个失败的事务块里则是“!”。<br>• 如果无法判断事务状态时为“?”（比如没有连接）。 |
| %digits | 指定字节值的字符将被替换到该位置。 |
| %:name | gsql变量“name”的值。 |

  

| 符号 | 符号说明 |
| --- | --- |
| %command | command的输出，类似于使用“^”替换。 |
| %[ . . . %] | 提示可以包含终端控制字符，这些字符可以改变颜色、背景、提示文本的风格、终端窗口的标题。例如，<br>gaussdb=> \set PROMPT1 '%[%033[1;33;40m%]%n@%/%R%[%033[0m%]%#'<br>这个句式的结果是在VT100兼容的可显示彩色的终端上的一个宽体（1;）黑底黄字（33;40）。 |

## 环境变量

**表 1-5** 与 gsql 相关的环境变量

| 名称 | 描述 |
| --- | --- |
| COLUMNS | 如果\set columns为0，则由此参数控制wrapped格式的宽度。这个宽度用于决定在自动扩展的模式下，是否要把宽输出模式变成竖线的格式。 |
| PAGER | 如果查询结果无法在一页显示，它们就会被重定向到这个命令。可以用\pset命令关闭分页器。典型的是用命令more或less来实现逐页查看。缺省值是平台相关的。<br>**说明**<br>less的文本显示，受系统环境变量LC_CTYPE影响。 |
| PSQL_EDITOR | \e和\ef命令使用环境变量指定的编辑器。变量是按照列出的先后顺序检查的。在Unix系统上默认的编辑工具是vi。 |
| EDITOR | |
| VISUAL | |
| PSQL_EDITOR_LINENUMBER_ARG | 当\e和\ef带上一行数字参数使用时，这个变量指定的命令行参数用于向编辑器传递起始行数。像Emacs或vi这样的编辑器，这只是个加号。如果选项和行号之间需要空白，在变量的值后加一个空格。例如：<br>PSQL_EDITOR_LINENUMBER_ARG = '+'<br>PSQL_EDITOR_LINENUMBER_ARG='--line '<br>Unix系统默认的是+。 |
| PSQLRC | 用户的.gsqlrc文件的交互位置。 |
| SHELL | 使用\!命令跟shell执行的命令是一样的效果。 |
| TMPDIR | 存储临时文件的目录。缺省是/tmp。 |

  

# 1.2 使用指导

## 前提条件

- 连接数据库时使用的用户需要具备访问数据库的权限。
- gsql须与数据库版本配套。

## 背景信息

使用gsql命令可以连接本机的数据库服务，也可以连接远程数据库服务。连接远程数据库服务时，需要在服务器上设置允许远程连接，详细操作请参见《开发指南》中“数据库使用入门 > 连接数据库 > 使用gsql连接 > 远程连接数据库”章节。

## 操作步骤

**步骤1** 使用gsql连接到GaussDB服务器。

gsql工具使用-d参数指定目标数据库名、-U参数指定数据库用户名、-h参数指定主机名、-p参数指定端口号信息。

**说明**

若未指定数据库名称，则使用初始化时默认生成的数据库名称；若未指定数据库用户名，则默认使用当前操作系统用户作为数据库用户名；当某个值没有前面的参数（-d、-U等）时，若连接的命令中没有指定数据库名（-d）则该参数会被解释成数据库名；如果已经指定数据库名（-d）而没有指定数据库用户名（-U）时，该参数则会被解释成数据库用户名。

示例1，使用omm用户连接到postgres数据库的8000端口。

```
gsql -d postgres -p 8000
```

示例2，使用jack用户连接到远程主机postgres数据库的8000端口。

```
gsql -h 10.180.123.163 -d postgres -U jack -p 8000
```

集中式数据库实例中，连接主DataNode时可以把DataNode的IP地址使用逗号分割全部添加到-h后，gsql将依次从前往后连接每个IP地址，查询当前DataNode是否为主DataNode，如果不是则断开连接尝试下一个IP地址，直到找到主DataNode为止。

```
gsql -h 10.180.123.163,10.180.123.164,10.180.123.165 -d postgres -U jack -p 8000
```

示例3，参数postgres和omm不属于任何选项时，分别被解释成了数据库名和用户名。

```
gsql postgres omm -p 8000
```

**等效于**

```
gsql -d postgres -U omm -p 8000
```

详细的gsql参数请参见**命令参考**。

**步骤2** 执行SQL语句。

以创建数据库human_staff为例。

```
gaussdb=# CREATE DATABASE human_staff;
CREATE DATABASE
```

  

通常，输入的命令行在遇到分号的时候结束。如果输入的命令行没有错误，结果就会输出到屏幕上。

**步骤3** 执行gsql元命令。

以列出GaussDB中所有的数据库和描述信息为例。

```
gaussdb=# \l
                                 List of databases
     Name      |  Owner   | Encoding  | Collate | Ctype |   Access privileges
---------------+----------+-----------+---------+-------+-----------------------
 human_resource | omm | SQL_ASCII | C       | C     |
 postgres       | omm | SQL_ASCII | C       | C     |
 template0      | omm | SQL_ASCII | C       | C     | =c/omm         +
                |     |           |         |       | omm=CTc/omm
 template1      | omm | SQL_ASCII | C       | C     | =c/omm         +
                |     |           |         |       | omm=CTc/omm
 human_staff    | omm | SQL_ASCII | C       | C     |
(5 rows)
```

更多gsql元命令请参见**元命令参考**。

**----结束**

## 示例

以把一个查询分成多行输入为例。注意提示符的变化：

```
gaussdb=# CREATE TABLE HR.areaS(
gaussdb(# area_ID    NUMBER,
gaussdb(# area_NAME VARCHAR2(25)
gaussdb-# )tablespace EXAMPLE;
CREATE TABLE
```

查看表的定义：

```
gaussdb=# \d HR.areaS
              Table "hr.areas"
  Column   |         Type          | Modifiers
-----------+-----------------------+-----------
 area_id   | numeric               | not null
 area_name | character varying(25) |
```

向HR.areaS表插入四行数据：

```
gaussdb=# INSERT INTO HR.areaS (area_ID, area_NAME) VALUES (1, 'Europe');
INSERT 0 1
gaussdb=# INSERT INTO HR.areaS (area_ID, area_NAME) VALUES (2, 'Americas');
INSERT 0 1
gaussdb=# INSERT INTO HR.areaS (area_ID, area_NAME) VALUES (3, 'Asia');
INSERT 0 1
gaussdb=# INSERT INTO HR.areaS (area_ID, area_NAME) VALUES (4, 'Middle East and Africa');
INSERT 0 1
```

切换提示符：

```
gaussdb=# \set PROMPT1 '%n@%m %~%R%#'
omm@[local] gaussdb=#
```

查看表：

```
omm@[local] gaussdb=# SELECT * FROM HR.areaS;
 area_id |       area_name
---------+------------------------
       1 | Europe
       4 | Middle East and Africa
       2 | Americas
```

  

```
      3 | Asia
(4 rows)
```

可以用\pset命令以不同的方法显示表：

```
omm@[local] gaussdb=# \pset border 2
Border style is 2.
omm@[local] gaussdb=# SELECT * FROM HR.areaS;
+---------+------------------------+
| area_id |       area_name        |
+---------+------------------------+
|       1 | Europe                 |
|       2 | Americas               |
|       3 | Asia                   |
|       4 | Middle East and Africa |
+---------+------------------------+
(4 rows)
omm@[local] gaussdb=# \pset border 0
Border style is 0.
omm@[local] gaussdb=# SELECT * FROM HR.areaS;
area_id       area_name
------- ----------------------
      1 Europe
      2 Americas
      3 Asia
      4 Middle East and Africa
(4 rows)
```

使用元命令：

```
omm@[local] gaussdb=# \a \t \x
Output format is unaligned.
Showing only tuples.
Expanded display is on.
omm@[local] gaussdb=# SELECT * FROM HR.areaS;
area_id|2
area_name|Americas

area_id|1
area_name|Europe

area_id|4
area_name|Middle East and Africa

area_id|3
area_name|Asia
omm@[local] gaussdb=#
```

# 1.3 获取帮助

## 操作步骤

- 连接数据库时，可以使用如下命令获取帮助信息。

  ```
  gsql --help
  ```

  显示如下帮助信息：

  ```
  ......
  Usage:
    gsql [OPTION]... [DBNAME [USERNAME]]

  General options:
    -c, --command=COMMAND    run only single command (SQL or internal) and exit
    -d, --dbname=DBNAME      database name to connect to (default: "omm")
    -f, --file=<FILE_NAME>      execute commands from file, then exit
  ......
  ```

  

- 连接到数据库后，可以使用如下命令获取帮助信息。
  ```
  help
  ```
  显示如下帮助信息：
  ```
  You are using gsql, the command-line interface to gaussdb.
  Type:  \copyright for distribution terms
         \h for help with SQL commands
         \? for help with gsql commands
         \g or terminate with semicolon to execute query
         \q to quit
  ```

## 任务示例

**步骤1** 连接数据库，具体操作请参考《开发指南》中“使用数据库入门 > 连接数据库 > 使用gsql连接”章节。

**步骤2** 查看gsql的帮助信息。具体执行命令请参见**表1-6**。

**表 1-6** 使用 gsql 联机帮助

| 描述 | 示例 |
| --- | --- |
| 查看版权信息 | \copyright |
| 查看GaussDB支持的SQL语句的帮助 | 查看GaussDB支持的SQL语句的帮助<br>例如，查看GaussDB支持的所有SQL语句：<br>`gaussdb=# \h`<br>`Available help:`<br>`  ABORT`<br>`... ...`<br>例如，查看CREATE DATABASE命令的参数可使用下面的命令：<br>`gaussdb=# \help CREATE DATABASE`<br>`Command:     CREATE DATABASE`<br>`Description: create a new database`<br>`Syntax:`<br>`CREATE DATABASE database_name`<br>`    [ [ WITH ] {[ OWNER [=] user_name ]|`<br>`          [ TEMPLATE [=] template ]|`<br>`          [ ENCODING [=] encoding ]|`<br>`          [ LC_COLLATE [=] lc_collate ]|`<br>`          [ LC_CTYPE [=] lc_ctype ]|`<br>`          [ DBCOMPATIBILITY [=] compatibility_type ]|`<br>`          [ TABLESPACE [=] tablespace_name ]|`<br>`          [ CONNECTION LIMIT [=] connlimit ]}[...] ];` |
| 查看gsql命令的帮助 | 例如，查看gsql支持的命令：<br>`gaussdb=# \?`<br>`General`<br>`  \copyright            show GaussDB Kernel usage and distribution terms`<br>`  \g [FILE] or ;        execute query (and send results to file or \|pipe)`<br>`  \h(\help) [NAME]              help on syntax of SQL commands, * for all commands`<br>`  \q                    quit gsql`<br>`... ...` |

**----结束**

  

# 1.4 命令参考

详细的gsql参数请参见**表1-7**、**表1-8**、**表1-9**和**表1-10**。

**表 1-7** 常用参数

| 参数 | 参数说明 | 取值范围 |
| --- | --- | --- |
| -c, --command=COMMAND | 声明gsql要执行一条字符串命令然后退出。 | - |
| -d, --dbname=DBNAME | 指定想要连接的数据库名称。<br>另外，gsql允许使用扩展的DBNAME，即'postgres[ql]://[user[:password]@][netloc][:port][,...][/dbname][?param1=value1&...]'或'[key=value] [...]'形式的连接串作为DBNAME，gsql将从连接串中解析连接信息，并优先使用这些信息。<br>**注意**<br>gsql使用扩展的DBNAME创建连接时，不支持指定replication参数。 | 字符串。 |
| -f, --file=FILENAME | 使用文件作为命令源而不是交互式输入。gsql将在处理完文件后结束。如果FILENAME是-（连字符），则从标准输入读取。 | 绝对路径或相对路径，且满足操作系统路径命名规则。 |
| -l, --list | 列出所有可用的数据库，然后退出。 | - |
| -v, --set, --variable=NAME=VALUE | 设置gsql变量NAME为VALUE。<br>变量的示例和详细说明请参见**变量**。 | - |
| -X, --no-gsqlrc | 不读取启动文件（系统范围的gsqlrc或者用户的~/.gsqlrc都不读取）。<br>**说明**<br>启动文件默认为~/.gsqlrc，或通过PSQLRC环境变量指定。 | - |
| -1 ("one"), --single-transaction | 当gsql使用-f选项执行脚本时，会在脚本的开头和结尾分别加上START TRANSACTION/COMMIT用以把整个脚本当作一个事务执行。这将保证该脚本完全执行成功，或者脚本无效。<br>**说明**<br>如果脚本中已经使用了START TRANSACTION，COMMIT，ROLLBACK，则该选项无效。 | - |

| 参数 | 参数说明 | 取值范围 |
| --- | --- | --- |
| -y, --slash-command | 将当前语句终结并发送到内核执行或者重新执行已经执行过的语句（不包括gsql元命令）。<br>**说明**<br>该功能仅支持A兼容模式数据库，而且不支持-c, --command参数的使用场景。<br>A兼容模式数据库下默认普通语句分隔符为;，PL/SQL为/，使用该参数时不支持更换分隔符。 | - |
| -?, --help | 显示关于gsql命令行参数的帮助信息然后退出。 | - |
| -V, --version | 打印gsql版本信息然后退出。 | - |

**表 1-8** 输入和输出参数

| 参数 | 参数说明 | 取值范围 |
| --- | --- | --- |
| -a, --echo-all | 在读取行时向标准输出打印所有内容。<br>**注意**<br>使用此参数可能会暴露部分SQL语句中的敏感信息，如创建用户语句中的password信息等，请谨慎使用。 | - |
| -e, --echo-queries | 把所有发送给服务器的查询同时回显到标准输出。<br>**注意**<br>使用此参数可能会暴露部分SQL语句中的敏感信息，如创建用户语句中的password信息等，请谨慎使用。 | - |
| -E, --echo-hidden | 回显由\d和其他反斜杠命令生成的实际查询。 | - |
| -k, --with-key=KEY | 使用gsql对导入的加密文件进行解密。<br>**须知**<br>• 对于本身就是shell命令中的关键字符如单引号（'）或双引号（"），Linux shell会检测输入的单引号（'）或双引号（"）是否匹配。如果不匹配，shell认为用户没有输入完毕，会一直等待用户输入，从而不会进入到gsql程序。<br>• 不支持解密导入存储过程和函数。 | - |
| -L, --log-file=FILENAME | 除了正常的输出源之外，把所有查询输出记录到文件FILENAME中。<br>**注意**<br>• 使用此参数可能会暴露部分SQL语句中的敏感信息，如创建用户语句中的password信息等，请谨慎使用。<br>• 此参数只保留查询结果到相应文件中，主要目标是为了查询结果能够更好更准确地被其他调用者（例如自动化运维脚本）解析；而不是保留gsql运行过程中的相关日志信息。 | 绝对路径或相对路径，且满足操作系统路径命名规则。 |

  

| 参数 | 参数说明 | 取值范围 |
| --- | --- | --- |
| -m, --maintenance | 允许在两阶段事务恢复期间连接数据库。<br>**说明**<br>该选项是一个开发选项，禁止用户使用，只限专业技术人员使用，功能是：使用该选项时，gsql可以连接到备机，用于校验主备机数据的一致性。 | - |
| -n, --no-libedit | 关闭命令行编辑。 | - |
| -o, --output=FILENAME | 将所有查询输出重定向到文件FILENAME。 | 绝对路径或相对路径，且满足操作系统路径命名规则。 |
| -q, --quiet | 安静模式，执行时不会打印出额外信息。 | 缺省时gsql将打印许多其他输出信息。 |
| -s, --single-step | 单步模式运行。意味着每个查询在发往服务器之前都要提示用户，用这个选项也可以取消执行。此选项主要用于调试脚本。<br>**注意**<br>使用此参数可能会暴露部分SQL语句中的敏感信息，如创建用户语句中的password信息等，请谨慎使用。 | - |
| -S, --single-line | 单行运行模式，这时每个命令都将由换行符结束，像分号那样。 | - |
| -C，-C1, --enable-client-encryption=1 | 当使用-C参数连接本地数据库或者连接远程数据库时，可通过该选项打开密态数据库开关，此开关为密态等值查询基本能力开关。 | - |
| -C3，--enable-client-encryption=3 | 当使用-C参数连接本地数据库或者连接远程数据库时，可通过该选项打开内存解密逃生通道开关，支持密态等值查询基本能力以及内存解密逃生通道能力。 | - |

**表 1-9** 输出格式参数

| 参数 | 参数说明 | 取值范围 |
| --- | --- | --- |
| -A, --no-align | 切换为非对齐输出模式。 | 缺省为对齐输出模式。 |
| -F, --field-separator=STRING | 设置域分隔符（默认为“|”）。 | - |
| -H, --html | 打开HTML格式输出。 | - |

  

| 参数 | 参数说明 | 取值范围 |
| --- | --- | --- |
| -P, --pset=VAR[=ARG] | 在命令行上以\pset的风格设置打印选项。<br>**说明**<br>这里必须用等号而不是空格分隔名称和值。例如，把输出格式设置为LaTeX，可以键入-P format=latex | - |
| -R, --record-separator=STRING | 设置记录分隔符。 | - |
| -r | 开启在客户端操作中可以进行编辑的模式。 | 缺省为关闭。 |
| -t, --tuples-only | 只打印行。 | - |
| -T, --table-attr=TEXT | 允许声明放在HTML table标签里的选项。<br>使用时请搭配参数“-H,--html”，指定为HTML格式输出。 | - |
| -x, --expanded | 打开扩展表格式模式。 | - |
| -z, --field-separator-zero | 设置非对齐输出模式的域分隔符为空。<br>使用时请搭配参数“-A, --no-align”，指定为非对齐输出模式。 | - |
| -0, --record-separator-zero | 设置非对齐输出模式的记录分隔符为空。<br>使用时请搭配参数“-A, --no-align”，指定为非对齐输出模式。 | - |
| -2, --pipeline | 使用管道传输密码，禁止在终端使用，必须和-c或者-f参数一起使用。 | - |

  

**表 1-10** 连接参数

| 参数 | 参数说明 | 取值范围 |
| --- | --- | --- |
| -h, --host=HOSTNAME | 指定正在运行服务器的主机名、Unix域套接字的路径、或者域名。接受以“,”分割的字符串来指定多个主机地址，支持指定多个主机地址，支持指定IPv6主机地址。<br>当指定多个主机地址时，默认情况下会自动选择主节点地址进行连接。可通过设置PGTARGETSESSIONATTRS环境变量的值来选择连接到不同类型的节点，变量与节点类型对应的关系如下：<br>**PGTARGETSESSIONATTRS环境变量的值--选择连接的节点类型**<br>read-write--可读写的节点<br>read-only--只读节点<br>primary或者不设定--主节点<br>standby--备节点<br>prefer-standby--首选备节点，没有备节点则转为any<br>any--不进行角色检查<br>**说明**<br>当-h只指定一个域名，但是该域名对应多个IP时，无法触发自动选主功能。 | 如果省略主机名，gsql将通过Unix域套接字与本地主机的服务器相连，或者在没有Unix域套接字的机器上，通过TCP/IP与localhost连接。 |
| -p, --port=PORT | 指定数据库服务器的端口号。可以配置一个或多个，当配置一个时，所有的主机地址都使用同一个端口连接；当配置多个时，顺序与主机地址顺序相同，个数必须与主机地址数相等，当不相等时会报错。<br>可以通过port参数修改默认端口号。 | 默认端口可通过编译参数来指定，不指定的话默认为5432。 |
| -U, --username=USERNAME | 指定连接数据库的用户。<br>**说明**<br>• 通过该参数指定用户连接数据库时，需要同时提供用户密码用以身份验证。您可以通过交换方式输入密码，或者通过-W参数指定密码。<br>• 用户名中包含有字符$，需要在字符$前增加转义字符才可成功连接数据库。 | 字符串，默认使用与当前操作系统用户同名的用户。 |

  

| 参数 | 参数说明 | 取值范围 |
| --- | --- | --- |
| -W, --password=PASSWORD | 当使用-U参数连接本地数据库或者连接远端数据库时，可通过该选项指定密码。<br>**说明**<br>• 登录数据库主节点所在服务器后连接本地数据库主节点实例时，默认使用trust连接，会忽略此参数。<br>• 用户密码中包含特殊字符“\”和"`"时，需要增加转义字符才可成功连接数据库。<br>• 如果用户未输入该参数，但是数据库连接需要用户密码，这时将出现交互式输入，请用户输入当前连接的密码。该密码最长长度为999字节，受限于GUC参数password_max_length的最大值。 | 字符串。 |

# 1.5 元命令参考

介绍使用GaussDB数据库命令行交互工具登录数据库后，gsql所提供的元命令。所谓元命令就是在gsql里输入的任何以不带引号的反斜杠开头的命令。

## 注意事项

- 一个gsql元命令的格式是反斜杠后面紧跟一个动词，然后是任意参数。参数命令动词和其他参数以任意个空白字符间隔。
- 要在参数里面包含空白，必须用单引号把它引起来。要在这样的参数里包含单引号，可以在前面加一个反斜杠。任何包含在单引号里的内容都会被进一步进行类似C语言的替换：\n（新行）、\t（制表符）、\b（退格）、\r（回车）、\f（换页）、\digits（八进制表示的字符）、\xdigits（十六进制表示的字符）。
- 用""包围的内容被当做一个命令行传入shell。该命令的输出（删除了结尾的新行）被当做参数值。
- 如果不带引号的参数以冒号（:）开头，它会被当做一个gsql变量，并且该变量的值最终会成为真正的参数值。
- 有些命令以一个SQL标识的名称（比如一个表）为参数。这些参数遵循SQL语法关于双引号的规则：不带双引号的标识强制转换成小写，而双引号保护字母不进行大小写转换，并且允许在标识符中使用空白。在双引号中，成对的双引号在结果名称中分析成一个双引号。比如，FOO"BAR"BAZ解析成fooBARbaz；而"Aweird""name"解析成A weird"name。
- 对参数的分析在遇到另一个不带引号的反斜杠时停止。这里会认为是一个新的元命令的开始。特殊的双反斜杠序列（\\）标识参数的结尾并将继续分析后面的SQL语句（如果存在）。这样SQL和gsql命令可以自由的在一行里面混合。但是在任何情况下，一条元命令的参数不能延续超过行尾。
- M-Compatibility模式数据库不支持\h元命令。

## 元命令

元命令的详细说明请参见**表1-11**、**表1-12**、**表1-13**、**表1-14**、**表1-16**、**表1-18**、**表1-19**、**表1-20**、**表1-22**和**表1-23**。

  

**须知**

以下命令中所提到的FILE代表文件路径。此路径可以是绝对路径（如/home/gauss/file.txt），也可以是相对路径（file.txt，file.txt会默认在用户执行gsql命令所在的路径下创建）。

**表 1-11** 一般的元命令

| 参数 | 参数说明 | 取值范围 |
| --- | --- | --- |
| \copyright | 显示GaussDB的版本和版权信息。 | - |
| \g [FILE] or ; | 执行查询（并将结果发送到文件或管道）。 | - |
| \h(\help) [NAME] | 给出指定SQL语句的语法帮助。 | 如果没有给出NAME，gsql将列出可获得帮助的所有命令。如果NAME是一个星号（*），则显示所有SQL语句的语法帮助。 |
| \parallel [on [num]\|off] | 控制并发执行开关。<br>• on：打开控制并发执行开关，且最大并发数为num。<br>• off：关闭控制并发执行开关。<br>**说明**<br>• 不支持事务中开启并发执行以及并发中开启事务。<br>• 不支持\d这类元命令的并发。<br>• 并发select返回结果混乱问题，此为客户可接受，core、进程停止响应不可接受。<br>• 不推荐在并发中使用set语句，否则导致结果与预期不一致。<br>• 不支持创建临时表！如需使用临时表，需要在开启parallel之前创建好，并在parallel内部使用。parallel内部不允许创建临时表。<br>• \parallel执行时最多会启动num个独立的gsql进程连接服务器。<br>• \parallel中所有作业的持续时间不能超过session_timeout，否则可能会导致并发执行过程中断连。<br>• 在\parallel on 之后一条或多条命令，会等到\parallel off执行后才会执行，因而，\parallel on之后需要有对应的\parallel off，否则\parallel on后的命令都无法执行。 | num的默认值：1024。<br>**须知**<br>• 服务器能接受的最大连接数受max_connection及当前已有连接数限制。<br>• 设置num时请考虑服务器当前可接受的实际连接数合理指定。 |
| \q | 退出gsql程序。在一个脚本文件里，只在脚本终止的时候执行。 | - |

**表 1-12** 查询缓存区元命令

| 参数 | 参数说明 |
| --- | --- |
| \e [FILE] [LINE] | 使用外部编辑器编辑查询缓冲区（或者文件）。 |
| \ef [FUNCNAME [LINE]] | 使用外部编辑器编辑函数定义。如果指定了LINE（即行号），则光标会指到函数体的指定行。 |
| \p | 打印当前查询缓冲区到标准输出。 |
| \r | 重置（或清空）查询缓冲区。 |
| \w FILE | 将当前查询缓冲区输出到文件。 |

**表 1-13** 输入/输出元命令

| 参数 | 参数说明 |
| --- | --- |
| \copy { table [ ( column_list ) ] \| ( query ) } { from \| to } { filename \| stdin \| stdout \| pstdin \| pstdout }[LOAD] [LOAD_DISCARD 'string'] [ with ] [ binary ] [ oids ] [ delimiter [ as ] 'character' ] [ useeof ] [ null [ as ] 'string' ] [ csv [ header ] [ quote [ as ] 'character' ] [ escape [ as ] 'character' ] [ force quote column_list \| * ] [ force not null column_list ] ] [parallel integer] | 在任何gsql客户端登录数据库成功后可以执行导入导出数据，这是一个运行SQL COPY命令的操作，但不是读取或写入指定文件的服务器，而是读取或写入文件，并在服务器和本地文件系统之间路由数据。这意味着文件的可访问性和权限是本地用户的权限，而不是服务器的权限，并且不需要数据库初始化用户权限。<br>**说明**<br>• \COPY只适合小批量，格式良好的数据导入，导入数据应优先选择GDS或COPY。<br>• \COPY 可以指定数据导入时的客户端数量，从而实现数据文件的并行导入，目前并发数范围为[1, 8]。<br>• \COPY并行导入目前存在以下约束：临时表的并行导入不支持、在事务内的并行导入不支持、对二进制文件的并行导入不支持、数据导入支持AES128加密时不支持以及COPY选项中存在EOL。在这些情况下，即使指定了parallel参数，仍然会走非并行流程。<br>• \COPY的text格式和csv格式均支持header功能<br>• 其中LOAD功能为gs_loader进行语法转换后调用copy的标识，非主动调用功能。<br>• 其中LOAD_DISCARD功能为gs_loader解析后discard文件路径，非主动调用功能。 |
| \echo [STRING] | 把字符串写到标准输出。 |
| prompt [STRING] | 把字符串写到标准输出（等同于\echo）。 |
| \i FILE | 从文件FILE中读取内容，并将其当作输入，执行查询。 |
| \i+ FILE KEY | 执行加密文件中的命令。 |
| \ir FILE | 和\i类似，只是相对于存放当前脚本的路径。 |

  

| 参数 | 参数说明 |
| --- | --- |
| \ir+ FILE KEY | 和\i+类似，只是相对于存放当前脚本的路径。 |
| \o [FILE] | 把所有的查询结果发送到文件里。 |
| \qecho [STRING] | 把字符串写到查询结果输出流里。 |

**说明**

表1-14中的选项S表示显示系统对象，+表示显示对象附加的描述信息。PATTERN用来指定要被显示的对象名称。

**表 1-14** 显示信息元命令

| 参数 | 参数说明 | 取值范围 | 示例 |
| --- | --- | --- | --- |
| \d[S+] | 列出当前search_path中模式下所有的表、视图和序列。当search_path中不同模式存在同名对象时，只显示search_path中位置靠前模式下的同名对象。 | - | 列出当前search_path中模式下所有的表、视图和序列。<br>gaussdb=# \d |
| \d[S+] NAME | 列出指定表、视图和索引的结构。当search_path中不同模式存在同名对象时，只显示search_path中位置靠前模式下的同名对象。 | - | 假设存在表a，列出指定表a的结构。<br>gaussdb=# \dtable+ a |
| \d+ [PATTERN] | 列出所有表、视图和索引。当search_path中不同模式存在同名对象时，只显示search_path中位置靠前模式下的同名对象。 | 如果声明了PATTERN，只显示名称匹配PATTERN的表、视图和索引。 | 列出所有名称以f开头的表、视图和索引。<br>gaussdb=# \d+ f* |
| \da[S] [PATTERN] | 列出所有可用的聚集函数，以及它们操作的数据类型和返回值类型。当search_path中不同模式存在同名对象时，只显示search_path中位置靠前模式下的同名对象。 | 如果声明了PATTERN，只显示名称匹配PATTERN的聚集函数。 | 列出所有名称以f开头可用的聚集函数，以及它们操作的数据类型和返回值类型。<br>gaussdb=# \da f* |
| \db[+] [PATTERN] | 列出所有可用的表空间。 | 如果声明了PATTERN，只显示名称匹配PATTERN的表空间。 | 列出所有名称以p开头的可用表空间。<br>gaussdb=# \db p* |
| \dc[S+] [PATTERN] | 列出所有字符集之间的可用转换。当search_path中不同模式存在同名对象时，只显示search_path中位置靠前模式下的同名对象。 | 如果声明了PATTERN，只显示名称匹配PATTERN的转换。 | 列出所有字符集之间的可用转换。<br>gaussdb=# \dc * |

  

| 参数 | 参数说明 | 取值范围 | 示例 |
| --- | --- | --- | --- |
| \dC[+] [PATTERN] | 列出所有类型转换。<br>PATTERN需要使用实际类型名，不能使用别名。当search_path中不同模式存在同名对象时，只显示search_path中位置靠前模式下的同名对象。 | 如果声明了PATTERN，只显示名称匹配PATTERN的转换。 | 列出所有名称以c开头的类型转换。<br>gaussdb=# \dC c* |
| \dd[S] [PATTERN] | 显示所有匹配PATTERN的描述。当search_path中不同模式存在同名对象时，只显示search_path中位置靠前模式下的同名对象。 | 如果没有给出参数，则显示所有可视对象。“对象”包括：聚集、函数、操作符、类型、关系(表、视图、索引、序列、大对象)、规则。 | 列出所有可视对象。<br>gaussdb=# \dd |
| \ddp [PATTERN] | 显示所有默认的使用权限。 | 如果指定了PATTERN，只显示名称匹配PATTERN的使用权限。 | 列出所有默认的使用权限。<br>gaussdb=# \ddp |
| \dD[S+] [PATTERN] | 列出所有可用域。当search_path中不同模式存在同名对象时，只显示search_path中位置靠前模式下的同名对象。 | 如果声明了PATTERN，只显示名称匹配PATTERN的域。 | 列出所有可用域。<br>gaussdb=# \dD |
| \det[+] [PATTERN] | 列出所有的外部表。当search_path中不同模式存在同名对象时，只显示search_path中位置靠前模式下的同名对象。 | 如果声明了PATTERN，只显示名称匹配PATTERN的表。 | 列出所有的外部表。<br>gaussdb=# \det |
| \des[+] [PATTERN] | 列出所有的外部服务器。 | 如果声明了PATTERN，只显示名称匹配PATTERN的服务器。 | 列出所有的外部服务器。<br>gaussdb=# \des |
| \deu[+] [PATTERN] | 列出用户映射信息。 | 如果声明了PATTERN，只显示名称匹配PATTERN的信息。 | 列出用户映射信息。<br>gaussdb=# \deu |
| \dew[+] [PATTERN] | 列出封装的外部数据。 | 如果声明了PATTERN，只显示名称匹配PATTERN的数据。 | 列出封装的外部数据。<br>gaussdb=# \dew |

  

| 参数 | 参数说明 | 取值范围 | 示例 |
| --- | --- | --- | --- |
| \df[antw][S+][PATTERN] | 列出所有可用函数，以及它们的参数和返回的数据类型。a代表聚集函数，n代表普通函数，t代表触发器，w代表窗口函数。当search_path中不同模式存在同名对象时，只显示search_path中位置靠前模式下的同名对象。 | 如果声明了PATTERN，只显示名称匹配PATTERN的函数。 | 列出所有可用函数，以及它们的参数和返回的数据类型。<br>gaussdb=# \df |
| \dF[+][PATTERN] | 列出所有的文本搜索配置信息。当search_path中不同模式存在同名对象时，只显示search_path中位置靠前模式下的同名对象。 | 如果声明了PATTERN，只显示名称匹配PATTERN的配置信息。 | 列出所有的文本搜索配置信息。<br>gaussdb=# \dF+ |
| \dFd[+][PATTERN] | 列出所有的文本搜索字典。当search_path中不同模式存在同名对象时，只显示search_path中位置靠前模式下的同名对象。 | 如果声明了PATTERN，只显示名称匹配PATTERN的字典。 | 列出所有的文本搜索字典。<br>gaussdb=# \dFd |
| \dFp[+][PATTERN] | 列出所有的文本搜索分析器。当search_path中不同模式存在同名对象时，只显示search_path中位置靠前模式下的同名对象。 | 如果声明了PATTERN，只显示名称匹配PATTERN的分析器。 | 列出所有的文本搜索分析器。<br>gaussdb=# \dFp |
| \dFt[+][PATTERN] | 列出所有的文本搜索模板。当search_path中不同模式存在同名对象时，只显示search_path中位置靠前模式下的同名对象。 | 如果声明了PATTERN，只显示名称匹配PATTERN的模板。 | 列出所有的文本搜索模板。<br>gaussdb=# \dFt |
| \dg[+][PATTERN] | 列出所有数据库角色。<br>**说明**<br>因为用户和群组的概念被统一为角色，所以这个命令等价于\du。为了和以前兼容，所以保留两个命令。 | 如果指定了PATTERN，只显示名称匹配PATTERN的角色。 | 列出名称为“j?e”所有数据库角色（“?”表示任一字符）。<br>gaussdb=# \dg j?e |
| \dl | \lo_list的别名，显示一个大对象的列表。 | - | 列出所有的大对象。<br>gaussdb=# \dl |
| \dL[S+][PATTERN] | 列出可用的程序语言。 | 如果指定了PATTERN，只列出名称匹配PATTERN的语言。 | 列出可用的程序语言。<br>gaussdb=# \dL |

  

| 参数 | 参数说明 | 取值范围 | 示例 |
| --- | --- | --- | --- |
| \dm[S+] [PATTERN] | 列出物化视图。当search_path中不同模式存在同名对象时，只显示search_path中位置靠前模式下的同名对象。 | 如果指定了PATTERN，只列出名称匹配PATTERN的物化视图。 | 列出物化视图。<br>gaussdb=# \dm |
| \dn[S+] [PATTERN] | 列出所有模式（名称空间）。如果向命令追加+，会列出每个模式相关的权限及描述。 | 如果声明了PATTERN，只列出名称匹配PATTERN的模式名。缺省时，只列出用户创建的模式。 | 列出所有名称以d开头的模式以及相关信息。<br>gaussdb=# \dn+ d* |
| \do[S] [PATTERN] | 列出所有可用的操作符，以及它们的操作数和返回的数据类型。当search_path中不同模式存在同名对象时，只显示search_path中位置靠前模式下的同名对象。 | 如果声明了PATTERN，只列出名称匹配PATTERN的操作符。缺省时，只列出用户创建的操作符。 | 列出所有可用的操作符，以及它们的操作数和返回的数据类型。<br>gaussdb=# \do |
| \dO[S+] [PATTERN] | 列出排序规则。当search_path中不同模式存在同名对象时，只显示search_path中位置靠前模式下的同名对象。 | 如果声明了PATTERN，只列出名称匹配PATTERN的规则。缺省时，只列出用户创建的规则。 | 列出排序规则。<br>gaussdb=# \dO |
| \dp [PATTERN] | 列出一列可用的表、视图以及相关的权限信息。当search_path中不同模式存在同名对象时，只显示search_path中位置靠前模式下的同名对象。<br>\dp显示结果如下：<br>rolename=xxxx/yyyy --赋予一个角色的权限<br>=xxxx/yyyy --赋予public的权限<br>xxxx表示赋予的权限，yyyy表示授予这个权限的角色。权限的参数说明请参见**表1-15**。 | 如果指定了PATTERN，只列出名称匹配PATTERN的表、视图。 | 列出一列可用的表、视图以及相关的权限信息。<br>gaussdb=# \dp |
| \drds [PATTERN1 [PATTERN2]] | 列出所有修改过的配置参数。这些设置可以是针对角色的、针对数据库的或者同时针对两者的。PATTERN1和PATTERN2表示要列出的角色PATTERN和数据库PATTERN。 | 如果声明了PATTERN，只列出名称匹配PATTERN的规则。缺省或指定*时，则会列出所有设置。 | 列出数据库所有修改过的配置参数。<br>gaussdb=# \drds * dbname |

| 参数 | 参数说明 | 取值范围 | 示例 |
| --- | --- | --- | --- |
| \dT[S+] [PATTERN] | 列出所有的数据类型。当search_path中不同模式存在同名对象时，只显示search_path中位置靠前模式下的同名对象。 | 如果指定了PATTERN，只列出名称匹配PATTERN的类型。 | 列出所有的数据类型。<br>gaussdb=# \dT |
| \du[+] [PATTERN] | 列出所有数据库角色。<br>**说明**<br>因为用户和群组的概念被统一为角色，所以这个命令等价于\dg。为了和以前兼容，所以保留两个命令。 | 如果指定了PATTERN，则只列出名称匹配PATTERN的角色。 | 列出所有数据库角色。<br>gaussdb=# \du |
| \dE[S+] [PATTERN]<br>\di[S+] [PATTERN]<br>\ds[S+] [PATTERN]<br>\dt[S+] [PATTERN]<br>\dv[S+] [PATTERN] | 这一组命令，字母E，i，s，t和v分别代表着外部表，索引，序列，表和视图。可以以任意顺序指定其中一个或者它们的组合来列出这些对象。当search_path中不同模式存在同名对象时，只显示search_path中位置靠前模式下的同名对象。例如：\dit列出所有的索引和表。在命令名称后面追加+，则每一个对象的物理尺寸以及相关的描述也会被列出。 | 如果指定了PATTERN，只列出名称匹配该PATTERN的对象。默认情况下只会显示用户创建的对象。通过PATTERN或者S修饰符可以把系统对象包括在内。 | 列出所有的索引和视图。<br>gaussdb=# \div |
| \dx[+] [PATTERN] | 列出安装数据库的扩展信息。 | 如果指定了PATTERN，则只列出名称匹配PATTERN的扩展信息。 | 列出安装数据库的扩展信息。<br>gaussdb=# \dx |
| \l[+] | 列出服务器上所有数据库的名称、所有者、字符集编码以及使用权限。 | - | 列出服务器上所有数据库的名称、所有者、字符集编码以及使用权限。<br>gaussdb=# \l |
| \sf[+] FUNCNAME | 显示函数的定义。<br>**说明**<br>对于带圆括号的函数名，需要在函数名两端添加双引号，并且在双引号后面加上参数类型列表。参数类型列表两端添加圆括号。<br>如果存在同名的函数，则会返回多个函数的定义。 | - | 假设存在函数function_a和函数名带圆括号的函数func()name，列出函数的定义。<br>gaussdb=# \sf function_a<br>gaussdb=# \sf "func()name"(argtype1, argtype2) |

  

| 参数 | 参数说明 | 取值范围 | 示例 |
| --- | --- | --- | --- |
| \z [PATTERN] | 列出数据库中所有表、视图和序列，以及它们相关的访问特权。 | 如果给出任何 pattern ，则被当成一个正则表达式，只显示匹配的表、视图、序列。 | 列出数据库中所有表、视图和序列，以及它们相关的访问特权。<br>gaussdb=# \z |

**表 1-15** 权限的参数说明

| 参数 | 参数说明 |
| --- | --- |
| r | SELECT：允许对指定的表、视图读取数据。 |
| w | UPDATE：允许对指定表更新字段。 |
| a | INSERT：允许对指定表插入数据。 |
| d | DELETE：允许删除指定表中的数据。 |
| D | TRUNCATE：允许清理指定表中的数据。 |
| x | REFERENCES：允许创建外键约束。 |
| t | TRIGGER：允许在指定表上创建触发器。 |
| X | EXECUTE：允许使用指定的函数，以及利用这些函数实现的操作符。 |
| U | USAGE：<br>● 对于过程语言，允许用户在创建函数时，指定过程语言。<br>● 对于模式，允许访问包含在指定模式中的对象。<br>● 对于序列，允许使用nextval函数。 |
| C | CREATE：<br>● 对于数据库，允许在该数据库里创建新的模式。<br>● 对于模式，允许在该模式中创建新的对象。<br>● 对于表空间，允许在其中创建表，以及允许创建数据库和模式的时候把该表空间指定为其缺省表空间。 |
| c | CONNECT：允许用户连接到指定的数据库。 |
| T | TEMPORARY：允许创建临时表。 |
| A | ALTER：允许用户修改指定对象的属性。 |
| P | DROP：允许用户删除指定的对象。 |
| m | COMMENT：允许用户定义或修改指定对象的注释。 |
| i | INDEX：允许用户在指定表上创建索引。 |

  

| 参数 | 参数说明 |
| --- | --- |
| v | VACUUM：允许用户对指定的表执行ANALYZE和VACUUM操作。 |
| * | 给前面权限的授权选项。 |

**表 1-16** 格式化元命令

| 参数 | 参数说明 |
| --- | --- |
| \a | 对齐模式和非对齐模式之间的切换。 |
| \C [STRING] | 把正在打印的表的标题设置为一个查询的结果或者取消这样的设置。 |
| \f [STRING] | 对于不对齐的查询输出，显示或者设置域分隔符。 |
| \H | • 若当前模式为文本格式，则切换为HTML输出格式。<br>• 若当前模式为HTML格式，则切换回文本格式。 |
| \pset NAME [VALUE] | 设置影响查询结果表输出的选项。NAME的取值见**表 1-17**。 |
| \t [on\|off] | 切换输出的字段名的信息和行计数脚注。 |
| \T [STRING] | 指定在使用HTML输出格式时放在table标签里的属性。如果参数为空，不设置。 |
| \x [on\|off\|auto] | 切换扩展行格式。 |

**表 1-17** 可调节的打印选项

| 选项 | 选项说明 | 取值范围 |
| --- | --- | --- |
| border | value必须是一个数字。通常这个数字越大，表的边界就越宽线就越多，但是这个取决于特定的格式。 | • 在HTML格式下，取值范围为大于0的整数。<br>• 在其他格式下，取值范围：<br>– 0：无边框<br>– 1：内部分隔线<br>– 2：台架 |

  

| 选项 | 选项说明 | 取值范围 |
| --- | --- | --- |
| expanded (或x) | 在正常和扩展格式之间切换。 | ● 当打开扩展格式时，查询结果用两列显示，字段名称在左、数据在右。这个模式在数据无法放进通常的“水平”模式的屏幕时很有用。<br>● 在正常格式下，当查询输出的格式比屏幕宽时，用扩展格式。正常格式只对aligned和wrapped格式有用。 |
| fieldsep | 声明域分隔符来实现非对齐输出。这样就可以创建其他程序希望的制表符或逗号分隔的输出。要设置制表符域分隔符，键入\pset fieldsep '\t'。缺省域分隔符是 '\|' (竖条符)。 | - |
| fieldsep_zero | 声明域分隔符来实现非对齐输出到零字节。 | - |
| footer | 用来切换脚注。 | - |
| format | 设置输出格式。允许使用唯一缩写（这意味着一个字母就够了）。 | 取值范围：<br>● unaligned：写一行的所有列在一条直线上中，当前活动字段分隔符分隔。<br>● aligned：此格式是标准的，可读性好的文本输出。<br>● wrapped：类似aligned，但是包装跨行的宽数据值，使其适应目标字段的宽度输出。<br>● html：把表输出为可用于文档里的对应标记语言。输出不是完整的文档。<br>● latex：把表输出为可用于文档里的对应标记语言。输出不是完整的文档。<br>● troff-ms：把表输出为可用于文档里的对应标记语言。输出不是完整的文档。 |
| null | 打印一个字符串，用来代替一个null值。 | 缺省是什么都不打印，这样很容易和空字符串混淆。 |
| numericlocale | 切换分隔小数点左边的数值的区域相关的分组符号。 | ● on：显示指定的分隔符。<br>● off：不显示分隔符。<br>忽略此参数，显示默认的分隔符。 |

| 选项 | 选项说明 | 取值范围 |
| --- | --- | --- |
| pager | 控制查询和gsql帮助输出的分页器。如果设置了环境变量PAGER，输出将被指向到指定程序，否则使用系统缺省。 | • on：当输出到终端且不适合屏幕显示时，使用分页器。<br>• off：不使用分页器。<br>• always：当输出到终端无论是否符合屏幕显示时，都使用分页器。 |
| recordsep | 声明在非对齐输出格式时的记录分隔符。 | - |
| recordsep_zero | 声明在非对齐输出到零字节时的记录分隔符。 | - |
| tableattr（或T） | 声明放在html输出格式中HTML table标签的属性（例如：cellpadding或bgcolor）。注意：这里可能不需要声明border，因为已经在\pset border里用过了。如果没有给出value，则不设置表的属性。 | - |
| title | 为随后打印的表设置标题。这个可以用于给输出一个描述性标签。如果没有给出value，不设置标题。 | - |
| tuples_only（或者t） | 在完全显示和只显示实际的表数据之间切换。完全显示将输出像列头、标题、各种脚注等信息。在tuples_only模式下，只显示实际的表数据。 | - |
| feedback | 切换是否输出结果行数 | - |

**表 1-18** 连接元命令

| 参数 | 参数说明 | 取值范围 |
| --- | --- | --- |
| \c[onnect] [DBNAME\|- USER\|- HOST\|- PORT\|-] | 连接到一个新的数据库。当数据库名称长度超过63个字节时，默认前63个字节有效，连接到前63个字节对应的数据库，但是gsql的命令提示符中显示的数据库对象名仍为截断前的名称。<br>**说明**<br>重新建立连接时，如果切换数据库登录用户，将可能会出现交互式输入，要求输入新用户的连接密码。该密码最长长度为999字节，受限于GUC参数password_max_length的最大值。 | - |
| \encoding [ENCODING] | 设置客户端字符编码格式。 | 不带参数时，显示当前的编码格式。 |
| \conninfo | 输出当前连接的数据库的信息。 | - |

  

**表 1-19** 操作系统元命令

| 参数 | 参数说明 | 取值范围 |
| --- | --- | --- |
| \cd [DIR] | 切换当前的工作目录。 | 绝对路径或相对路径，且满足操作系统路径命名规则。 |
| \setenv NAME [VALUE] | 设置环境变量NAME为VALUE，如果没有给出VALUE值，则不设置环境变量。 | - |
| \timing [on\|off] | 以毫秒为单位显示每条SQL语句的执行时间（不包括屏显打印时间）。 | • on表示打开显示。<br>• off表示关闭显示。 |
| \! [COMMAND] | 返回到一个单独的Unix shell或者执行Unix命令COMMAND。 | - |

**表 1-20** 变量元命令

| 参数 | 参数说明 |
| --- | --- |
| \prompt [TEXT] NAME | 提示用户用文本格式来指定变量名称。 |
| \set [NAME [VALUE]] | 设置内部变量NAME为VALUE或者如果给出了多于一个值，设置为所有这些值的连接结果。如果没有给出第二个参数，只设变量不设值。<br>有一些常用变量被gsql特殊对待，它们是一些选项设置，通常所有特殊对待的变量都是由大写字母组成(可能还有数字和下划线)。 表1-21是一个所有特殊对待的变量列表。 |
| \unset NAME | 不设置（或删除）gsql变量名。 |

**表 1-21** \set 常用命令

| 名称 | 命令说明 | 取值范围 |
| --- | --- | --- |
| \set VERBOSITY value | 这个选项可以设置为值default，verbose，terse之一以控制错误报告的冗余行。 | value取值范围：default，verbose，terse |
| \set ON_ERROR_STOP value | 如果设置了这个变量，脚本处理将马上停止。如果该脚本是从另外一个脚本调用的，该脚本也会按同样的方式停止。如果最外层的脚本不是从一次交互的gsql会话中调用的而是用-f选项调用的，gsql将返回错误代码3，以示这个情况与致命错误条件的区别（错误代码为1）。 | value取值范围为：on/off，true/false，yes/no，1/0 |

  

| 名称 | 命令说明 | 取值范围 |
| --- | --- | --- |
| \set AUTOCOMMIT [on\|off] | 设置当前gsql连接的自动提交行为，on为打开自动提交，off为关闭自动提交。默认情况下，gsql连接处于自动提交模式，每个单独的语句都被隐式提交。如果基于性能或者其它方面考虑，需要关闭自动提交时，需要用户自己显示的发出COMMIT命令来保证事务的提交。例如，在指定的业务SQL执行完之后发送COMMIT语句显式提交，特别是gsql客户端退出之前务必保证所有的事务已经提交。<br>**说明**<br>gsql默认使用自动提交模式，若关闭自动提交，将会导致后面执行的语句都受到隐式事务包裹，数据库中不支持在事务中执行的语句不能在此模式下执行。 | • on表示打开自动提交。<br>• off表示关闭自动提交。 |

**表 1-22** 大对象元命令

| 参数 | 参数说明 |
| --- | --- |
| \lo_list | 显示一个目前存储在该数据库里的所有GaussDB大对象和提供给大对象的注释。 |

**表 1-23** 全密态元命令

| 参数 | 参数说明 |
| --- | --- |
| \send_token | 全密态功能，传输密钥到服务端缓存，只在开启内存解密逃生通道的情况下使用。 |
| \st | 全密态功能，传输密钥到服务端缓存，只在开启内存解密逃生通道的情况下使用。 |
| \clear_token | 全密态功能，销毁服务端缓存的密钥，只在开启内存解密逃生通道的情况下使用。 |
| \ct | 全密态功能，销毁服务端缓存的密钥，只在开启内存解密逃生通道的情况下使用。 |
| \key_info KEY_INFO | 在全密态数据库特性中，用于设置访问外部密钥管理者的参数。 |

**说明**

M-Compatibility暂不支持全密态数据库。

## PATTERN

很多\d命令都可以用一个PATTERN参数来指定要被显示的对象名称。在最简单的情况下，PATTERN正好就是该对象的准确名称。在PATTERN中的字符通常会被变成小写形

  

式（就像在SQL名称中那样），例如\dt FOO将会显示名为foo的表。就像在SQL名称中那样，把PATTERN放在双引号中可以阻止它被转换成小写形式。如果需要在一个PATTERN中包括一个真正的双引号字符，则需要把它写成两个相邻的双引号，这同样是符合SQL引用标识符的规则。例如，\dt "FOO""BAR"将显示名为FOO"BAR（不是foo"bar）的表。和普通的SQL名称规则不同，不能只在PATTERN的一部分周围放上双引号，例如\dt FOO"FOO"BAR将会显示名为fooFOObar的表。

不使用PATTERN参数时，\d命令会显示当前schema搜索路径中可见的全部对象——这等价于用*作为PATTERN。所谓对象可见是指可以直接用名称引用该对象，而不需要用schema来进行限定。要查看数据库中所有的对象而不管它们的可见性，可以把*.*用作PATTERN。

如果放在一个PATTERN中，*将匹配任意字符序列（包括空序列），而?会匹配任意的单个字符（这种记号方法就像 Unix shell 的文件名PATTERN一样）。例如，\dt int*会显示名称以int开始的表。但是如果被放在双引号内，*和?就会失去这些特殊含义而变成普通的字符。

包含一个点号（.）的PATTERN被解释为一个schema名称模式后面跟上一个对象名称模式。例如，\dt foo*.*bar*会显示名称以foo开始的schema中所有名称包括bar的表。如果没有出现点号，那么模式将只匹配当前schema搜索路径中可见的对象。同样，双引号内的点号会失去其特殊含义并且变成普通的字符。

高级用户可以使用字符类等正则表达式记法，如[0-9]可以匹配任意数字。所有的正则表达式特殊字符都按照POSIX正则表达式所说的工作。以下字符除外：

- .会按照上面所说的作为一种分隔符。
- *会被翻译成正则表达式记号.*。
- ?会被翻译成.。
- $则按字面意思匹配。

根据需要，可以通过书写?、(*R*+|)、(*R*|)和*R*?来分别模拟PATTERN字符.、*R**和*R*?。$不需要作为一个正则表达式字符，因为PATTERN必须匹配整个名称，而不是像正则表达式的常规用法那样解释（换句话说，$会被自动地追加到PATTERN上）。如果不希望该PATTERN的匹配位置被固定，可以在开头或者结尾写上*。注意在双引号内，所有的正则表达式特殊字符会失去其特殊含义并且按照其字面意思进行匹配。另外，在操作符名称PATTERN中（即\do的PATTERN参数），正则表达式特殊字符也按照字面意思进行匹配。

## DELIMITER

更改SQL语句之间分隔符命令，分隔符默认值为“;”。

DELIMITER命令用来为客户端设置一个分隔符。当用户设置分隔符后，gsql客户端识别到分隔符时，会立即将SQL语句发送到服务端执行，但是服务端仍然将“;”看做SQL语句分隔符，并相应的处理SQL语句。

注意事项：

- delimiter符号目前不是自由设定的，结束符范围有限制，目前接受大小写字母组合或特殊字符组合（~/!/@/#/^/&/`/?/+/-/*//（除号）/%/</>/=），其中常见的用法是“//”。
- 符号组合中尽量使用无歧义符号组合，特殊符号组合（注释符：“*”、“--”，以加号“+”或减号“-”结尾的符号组合）目前不支持用于delimiter命名。
- delimiter长度范围：0~15。

  

- 设置的结束符的级别是会话级别的，当切换数据库时delimiter_name会设置为默认值“;”。
- 用户如果想使用其他字符组合例如“adbc $$”，可以使用引号包含，例如delimiter "adbc $$"，但使用时也需要使用引号包含，例如：select 1"adbc $$"。
- delimiter分隔符只有sql_compatibility = 'B'时支持。

# 1.6 常见问题处理

## 连接性能问题

- 开启log_hostname，但是配置错误的DNS导致的连接性能问题。

  连接数据库，通过“show log_hostname”语句，检查数据库中是否开启log_hostname参数。

  如果开启了相关参数，那么数据库内核通过DNS反查客户端所在机器的主机名。如果数据库配置了不正确的/不可达的DNS服务器，导致数据库建立连接过程较慢。此参数的更多信息详见GUC参数**log_hostname**。

- 数据库内核执行初始化语句较慢导致的性能问题。

  此种情况定位较难，可以尝试使用Linux的命令：strace。

  ```
  strace gsql -U MyUserName -d gaussdb -h 127.0.0.1 -p 23508 -r -c '\q'
  Password for MyUserName:
  ```

  在屏幕上打印出数据库的连接过程。比如较长时间停留在下面的操作上：

  ```
  sendto(3, "Q\0\0\0\25SELECT VERSION()\0", 22, MSG_NOSIGNAL, NULL, 0) = 22
  poll([{fd=3, events=POLLIN|POLLERR}], 1, -1) = 1 ([{fd=3, revents=POLLIN}])
  ```

  可以确定是数据库执行“SELECT VERSION()”语句较慢。

  在连接上数据库后，便可以通过执行“explain performance select version()”语句确定初始化语句执行较慢的原因。更多信息，详见《开发指南》中“SQL调优指南 > SQL执行计划介绍”章节。

  另外还有一种场景不太常见：由于DN所在机器的磁盘满或故障，此时所查询等受影响，无法进行用户认证，导致连接过程挂起。解决此问题清理DN的数据盘空间便可。

- TCP连接创建较慢问题。

  此问题可以参考上面的初始化语句较慢排查的做法，通过strace侦听，如果长时间停留在：

  ```
  connect(3, {sa_family=AF_FILE, path="/home/test/tmp/gaussdb_llt1/.s.PGSQL.61052"}, 110) = 0
  ```

  或者

  ```
  connect(3, {sa_family=AF_INET, sin_port=htons(61052), sin_addr=inet_addr("127.0.0.1")}, 16) = -1
  EINPROGRESS (Operation now in progress)
  ```

  说明客户端与数据库端建立物理连接过慢，此时应当检查网络是否存在不稳定、网络吞吐量太大的问题。

- 资源负载满导致连接较慢的问题。

  原因分析：当CPU、内存、I/O负载中的任意一项接近100%时，会出现gsql连接慢的现象。

  问题解决：

  a. 通过top命令等确认CPU使用率；通过free命令确认内存使用情况；通过iostat命令确认I/O负载；还可以通过cm_agent中的监控日志，以及数据库运维平台中的监测记录进行检查。

  

b. 针对短时间内大量慢查询导致的峰值负载场景，可通过[数据库服务器的端口号+1]端口连接，查询pg_stat_activity视图；针对慢查询，可以使用系统函数pg_terminate_backend进行查杀会话。

c. 针对业务量长期超负载情况（即无明显慢查询，或慢查询查杀后但新的查询依然会变成慢查询），应考虑降低业务负载、增加数据库资源的方式进行优化。

## 创建连接故障

- gsql: could not connect to server: No route to host

  此问题一般是指定了不可达的地址或者端口导致的。请检查-h参数与-p参数是否添加正确。

- gsql: FATAL: Invalid username/password,login denied.

  此问题一般是输入了错误的用户名和密码导致的，请联系数据库管理员，确认用户名和密码的正确性。

- gsql: FATAL: Forbid remote connection with trust method!

  数据库由于安全问题，禁止远程登录时使用trust模式。这时需要修改gs_hba.conf里的连接认证信息。请联系管理员处理。

  **说明**

  请不要修改gs_hba.conf中数据库主机的相关设置，否则可能导致数据库功能故障。建议业务应用部署在数据库之外，而非数据库内部。

- 在DN连接数据库，添加“-h 127.0.0.1”可以连接，去掉后无法连接问题。

  通过执行SQL语句“show unix_socket_directory”检查DN使用的Unix套接字目录，是否与shell中的环境变量$PGHOST一致。

  如果检查结果不一致，那么修改PGHOST环境变量到GUC参数unix_socket_directory指向的目录。

  关于unix_socket_directory的更多信息，请联系管理员获取。

- The "libpq.so" loaded mismatch the version of gsql, please check it.

  此问题是由于环境中使用的libpq.so的版本与gsql的版本不匹配导致的，请通过“ldd gsql”命令确认当前加载的libpq.so的版本，并通过修改LD_LIBRARY_PATH环境变量来加载正确的libpq.so。

- gsql: symbol lookup error: xxx/gsql: undefined symbol: libpqVersionString

  此问题是由于环境中使用的libpq.so的版本与gsql的版本不匹配导致的（也有可能是环境中存在PostgreSQL的libpq.so），请通过“ldd gsql”命令确认当前加载的libpq.so的版本，并通过修改LD_LIBRARY_PATH环境变量来加载正确的libpq.so。

- gsql: connect to server failed: Connection timed out

  Is the server running on host "xx.xxx.xxx.xxx" and accepting TCP/IP connections on port xxxx?

  此问题是由于网络连接故障造成。请检查客户端与数据库服务器间的网络连接。如果发现从客户端无法PING到数据库服务器端，则说明网络连接出现故障。请联系网络管理人员排查解决。

  ```
  ping -c 4 10.10.10.1
  PING 10.10.10.1 (10.10.10.1) 56(84) bytes of data.
  From 10.10.10.1: icmp_seq=2 Destination Host Unreachable
  From 10.10.10.1 icmp_seq=2 Destination Host Unreachable
  From 10.10.10.1 icmp_seq=3 Destination Host Unreachable
  From 10.10.10.1 icmp_seq=4 Destination Host Unreachable
  ```

  

```
--- 10.10.10.1 ping statistics ---
4 packets transmitted, 0 received, +4 errors, 100% packet loss, time 2999ms
```

- gsql: FATAL: permission denied for database "gaussdb"

  DETAIL: User does not have CONNECT privilege.

  此问题是由于用户不具备访问该数据库的权限，可以使用如下方法解决。

  a. 使用管理员用户dbadmin连接数据库。

  ```
  gsql -d gaussdb -U dbadmin -p 8000
  ```

  b. 赋予该用户访问数据库的权限。

  GRANT CONNECT ON DATABASE gaussdb TO user1;

  **说明**

  实际上，常见的许多错误操作也可能产生用户无法连接上数据库的现象。如用户连接的数据库不存在，用户名或密码输入错误等。这些错误操作在客户端工具也有相应的提示信息。

  ```
  gsql -d gaussdb -p 8000
  gsql: FATAL:  database "gaussdb" does not exist

  gsql -d gaussdb -U user1 -p 8000
  Password for user user1:
  gsql: FATAL:  Invalid username/password,login denied.
  ```

- gsql: FATAL: sorry, too many clients already，active/non-active: 197/3.

  此问题是由于系统连接数量超过了最大连接数量。请联系数据库DBA进行会话连接数管理，释放无用会话。

  关于查看用户会话连接数的方法如表1-24。

  会话状态可以在视图PG_STAT_ACTIVITY中查看。无用会话可以使用函数pg_terminate_backend进行释放。

  ```
  select datid,pid,state from pg_stat_activity;
   datid |       pid       | state
  -------+-----------------+--------
   13205 | 139834762094352 | active
   13205 | 139834759993104 | idle
  (2 rows)
  ```

  其中pid的值即为该会话的线程ID。根据线程ID结束会话。

  ```
  SELECT PG_TERMINATE_BACKEND(139834759993104);
  ```

  显示类似如下信息，表示结束会话成功。

  ```
  PG_TERMINATE_BACKEND
  ----------------------
   t
  (1 row)
  ```

**表 1-24** 查看会话连接数

| 描述 | 命令 |
| --- | --- |
| 查看指定用户的会话连接数上限。 | 执行如下命令查看连接到指定用户USER1的会话连接数上限。其中-1表示没有对用户user1设置连接数的限制。<br>SELECT ROLNAME,ROLCONNLIMIT FROM PG_ROLES WHERE ROLNAME='user1';<br> rolname \| rolconnlimit<br>---------+--------------<br> user1 \| -1<br>(1 row) |

  

| 描述 | 命令 |
| --- | --- |
| 查看指定用户已使用的会话连接数。 | 执行如下命令查看指定用户USER1已使用的会话连接数。其中，1表示USER1已使用的会话连接数。<br>SELECT COUNT(*) FROM dv_sessions WHERE USERNAME='user1';<br><br>count<br>-------<br>1<br>(1 row) |
| 查看指定数据库的会话连接数上限。 | 执行如下命令查看连接到指定数据库gaussdb的会话连接数上限。其中-1表示没有对数据库gaussdb设置连接数的限制。<br>SELECT DATNAME,DATCONNLIMIT FROM PG_DATABASE WHERE DATNAME='gaussdb';<br><br>datname \| datconnlimit<br>----------+--------------<br>gaussdb \| -1<br>(1 row) |
| 查看指定数据库已使用的会话连接数。 | 执行如下命令查看指定数据库gaussdb上已使用的会话连接数。其中，1表示数据库gaussdb上已使用的会话连接数。<br>SELECT COUNT(*) FROM PG_STAT_ACTIVITY WHERE DATNAME='gaussdb';<br>count<br>-------<br>1<br>(1 row) |
| 查看所有用户已使用会话连接数。 | 执行如下命令查看所有用户已使用的会话连接数。<br>SELECT COUNT(*) FROM dv_sessions;<br><br>count<br>-------<br>10<br>(1 row) |

- gsql: wait xxx.xxx.xxx.xxx:xxxx timeout expired

  gsql在向数据库发起连接的时候，会有5分钟超时机制，如果在这个超时时间内，数据库未能正常的对客户端请求进行校验和身份认证，那么gsql会退出当前会话的连接过程，并报出如上错误。

  一般来说，此问题是由于连接时使用的-h参数及-p参数指定的连接主机及端口有误（即错误信息中的xxx部分），导致通信故障；极少数情况是网络故障导致。要排除此问题，请检查数据库的主机名及端口是否正确。

- gsql: could not receive data from server: Connection reset by peer.

  同时，检查DN日志中出现类似如下日志“ FATAL: cipher file "/data/coordinator/server.key.cipher" has group or world access”，一般是由于数据目录或部分关键文件的权限被误操作篡改导致。请参照其他正常实例下的相关文件权限修改。

- gsql: FATAL: GSS authentication method is not allowed because XXXX user password is not disabled.

  目标DN的gs_hba.conf里配置了当前客户端IP使用"gss"方式来做认证，该认证算法不支持用作客户端的身份认证，请修改到"sha256"后再试。具体操作请联系管理员处理。

  

**说明**

- 请不要修改gs_hba.conf中数据库主机的相关设置，否则可能导致数据库功能故障。
- 建议业务应用部署在数据库之外，而非数据库内部。

## 其他故障

- 出现因“总线错误”（Bus error）导致的core dump或异常退出。

  一般情况下出现此种问题，是进程运行过程中加载的共享动态库（在Linux为.so文件）出现变化；或者进程二进制文件本身出现变化，导致操作系统加载机器的执行码或者加载依赖库的入口发生变化，操作系统出于保护目的将进程终止，产生core dump文件。

  解决此问题，请重试。同时请尽可能避免在升级等运维操作过程中，在数据库内部运行业务程序，避免升级时因替换文件产生此问题。

  **说明**

  此故障的core dump文件的可能堆栈是dl_main及其子调用，它是操作系统用来初始化进程做共享动态库加载的。如果进程已经初始化，但是共享动态库还未加载完成，严格意义上来说，进程并未完全启动。

  

# 2 gs_loader

## 概述

gs_loader工具用于进行数据导入。gs_loader将控制文件支持的语法转换为\COPY语法，然后利用已有的\COPY功能，做主要数据导入工作，同时gs_loader将\COPY结果记录到日志中。

使用gs_loader前请确保gs_loader版本与gsql版本、数据库版本保持一致。

**说明**

gs_loader工具当前不支持M-Compatibility数据库。

gs_loader工具当前支持PDB。

## 安装部署

在存放数据源文件的服务器上，安装并配置gs_loader客户端工具，方便使用gs_loader工具进行数据的导入。

**步骤1** 创建用于存放gs_loader工具包的目录。

```
mkdir -p /opt/bin
```

**步骤2** 将gsql工具包上传至新创建的目录中。

以上传EULER Linux版本的工具包为例，将软件安装包中的gsql工具包“GaussDB-Kernel_数据库版本号_操作系统版本号_64bit_gsql.tar.gz”上传至新创建的目录中。

**步骤3** 在工具包所在的目录下，解压工具包。

```
cd /opt/bin
tar -zxvf GaussDB-Kernel_数据库版本号_操作系统版本号_64bit_gsql.tar.gz
source gsql_env.sh
```

**步骤4** 验证工具位置及版本信息。

```
which gs_loader
```

**步骤5** 验证客户端版本信息。

gs_loader工具版本与gsql工具版本相对应，直接查询gsql客户端版本即可验证客户端版本信息。

```
gsql -V
```

**步骤6** 验证数据库版本信息，确保与客户端工具版本保持一致。

使用gsql工具成功连接数据库后输入：

```
select version();
```

**----结束**

## 日志等级配置

设置日志级别，可以供开发者查看。设置后会在控制台打印工具运行的相应信息。

```
export gs_loader_log_level=debug
export gs_loader_log_level=info
export gs_loader_log_level=warning
export gs_loader_log_level=error
```

## 使用权限

使用场景分为三权分立场景下及非三权分立场景下的使用。使用者可以选择将guc参数enableSeparationOfDuty设置为on或者off来控制三权分立功能的开启或关闭。

GUC参数enable_copy_error_log是控制是否使用错误表pgxc_copy_error_log的参数，默认为off、即不使用错误表，错误记录直接记录到gs_loader的bad文件中。如果该参数设置为on，则会使用错误表pgxc_copy_error_log，将错误记录插入错误表。

默认场景，不开启三权分立（即enableSeparationOfDuty=off）时，使用者可以是数据库普通用户或管理员用户。当使用者为普通用户的时候，需要管理员用户对普通用户赋权。管理员账户可以直接使用。错误表pgxc_copy_error_log通过GUC参数enable_copy_error_log控制开启和关闭，默认关闭。

1. 使用管理员用户创建新的用户：
   ```
   CREATE USER load_user WITH PASSWORD '************';
   ```
2. 给新用户授权pg_catalog.gs_copy_summary表：

   **说明**

   gs_loader使用的public.gs_copy_summary变更为pg_catalog.gs_copy_summary，原public.gs_copy_summary表废弃。由于public.gs_copy_summary可能与用户表同名，因此需要用户进行识别，确认不是用户表后，将数据迁移到pg_catalog.gs_copy_summary表里之后，删除public.gs_copy_summary。由于存在pg_catalog.gs_copy_summary，gsql元命令\d(+)优先找到pg_catalog.gs_copy_summary，而因为是系统表的缘故不显示，因此列表中找不到gs_copy_summary，但是实际上public.gs_copy_summary是存在的。可以直接通过public.gs_copy_summary引用。

   ```
   GRANT INSERT,SELECT ON  pg_catalog.gs_copy_summary To load_user;
   ```
3. （可选）给新用户授权错误表pgxc_copy_error_log：

   **说明**

   gs_loader使用的public.pgxc_copy_error_log变更为pg_catalog.pgxc_copy_error_log，原public.pgxc_copy_error_log表废弃。由于public.pgxc_copy_error_log可能与用户表同名，因此需要用户进行识别，确认不是用户表后，将数据迁移到pg_catalog.pgxc_copy_error_log表里之后，删除public.pgxc_copy_error_log。由于存在pg_catalog.pgxc_copy_error_log，gsql元命令\d(+)优先找到pg_catalog.pgxc_copy_error_log，而因为是系统表的缘故不显示，因此列表中找不到pgxc_copy_error_log，但是实际上public.pgxc_copy_error_log是存在的。可以直接通过public.pgxc_copy_error_log引用。

   ```
   GRANT INSERT,SELECT,DELETE ON  pg_catalog.pgxc_copy_error_log To load_user;
   ```

开启三权分立（即enableSeparationOfDuty=on）时，使用者可以是数据库普通用户或管理员用户。使用前需要到各自的schema下创建pgxc_copy_error_log表以及gs_copy_summary这两张表并添加索引，不需要再进行授权。

  

**说明**

因为pg_catalog模式下始终存在pgxc_copy_error_log以及gs_copy_summary两张表，而pg_catalog模式在search_path中优先级高于用户模式，因此在三权分立场景下无法直接通过gsql元命令\d(+)查询到，使用这两张表需要显式指定用户模式。

1. 使用初始用户创建新用户：
```
CREATE USER load_user WITH PASSWORD '********';
```
2. 从初始用户切换为新用户：
```
\c - load_user
```
3. 创建gs_copy_summary表并添加索引：
```
CREATE TABLE load_user.gs_copy_summary(relname varchar, begintime timestamptz, endtime timestamptz, id bigint, pid bigint, readrows bigint, skiprows bigint, loadrows bigint, errorrows bigint, whenrows bigint, allnullrows bigint, detail text);
CREATE INDEX gs_copy_summary_idx ON load_user.gs_copy_summary(id);
```
4. （可选）创建pgxc_copy_error_log表并添加索引：

   **说明**

   1. 如果guc参数enable_copy_error_log未设置（默认为off），或者设置为off，则无需使用错误表，无需创建。否则需要创建该错误表。

```
CREATE TABLE load_user.pgxc_copy_error_log (relname varchar, begintime timestamptz, filename varchar, lineno int8, rawrecord text, detail text);
CREATE INDEX copy_error_log_relname_idx ON load_user.pgxc_copy_error_log(relname);
```

## 使用环境

由用户自己将工具路径添加到PATH中。gs_loader支持SSL加密通信，使用方式同gsql方式。

## 使用指导

**步骤1** （非三权分立）仅对于普通用户。

1. （在管理员用户下）创建用户：
```
CREATE USER load_user WITH PASSWORD '************';
```
2. （在管理员用户下）给用户授权gs_copy_summary表：
```
GRANT INSERT,SELECT ON  pg_catalog.gs_copy_summary To load_user;
```
3. 切换用户。
```
\c - load_user
```

**步骤2** （三权分立）对于普通用户和管理员用户。

1. （在初始用户下）创建用户：
```
CREATE USER load_user WITH PASSWORD '********';
```
2. （在初始用户下）切换为load_user用户：
```
\c - load_user
```
3. 创建gs_copy_summary表并添加索引。
```
CREATE TABLE load_user.gs_copy_summary(relname varchar, begintime timestamptz, endtime timestamptz, id bigint, pid bigint, readrows bigint, skiprows bigint, loadrows bigint, errorrows bigint, whenrows bigint, allnullrows bigint, detail text);
CREATE INDEX gs_copy_summary_idx ON load_user.gs_copy_summary(id);
```

**步骤3** 创建表和控制文件，准备数据文件。

创建表loader_tbl。

```
CREATE TABLE  loader_tbl
(
   ID   NUMBER,
```

  

```
    NAME VARCHAR2(20),
    CON  VARCHAR2(20),
    DT   DATE
);
```

（在gs_loader客户端机器上）创建控制文件loader.ctl。

```
LOAD DATA
truncate into table loader_tbl
WHEN (2:2) = ','
fields terminated by ','
trailing nullcols
(
    id integer external,
    name char(32),
    con ":id || '-' || :name",
    dt date
)
```

（在gs_loader客户端机器上）创建guc参数文件guc.txt。

```
set a_format_copy_version='s1';
```

（在gs_loader客户端机器上）创建数据文件data.csv。

```
1,OK,,2007-07-8
2,OK,,2008-07-8
3,OK,,2009-07-8
4,OK,,2007-07-8
43,DISCARD,,2007-07-8
,,,
32,DISCARD,,2007-07-8
a,ERROR int,,2007-07-8
8,ERROR date,,2007-37-8
,,,,,
 ,
8,ERROR fields,,2007-37-8
,,,
5,OK,,2021-07-30
```

**步骤4** 进行导入。

（在gs_loader客户端机器上）执行导入前，先确认gs_loader工具有可执行权限。确保当前路径有文件写入权限（gs_loader在处理过程中会生成一些临时文件，导入完成后自动删除）。

```
gs_loader control=loader.ctl data=data.csv db=testdb bad=loader.bad guc_param=guc.txt errors=5
port=8000 passwd=********** user=load_user
```

执行结果：

```
gs_loader: version 0.1

 5 Rows successfully loaded.

log file is:
 loader.log
```

**----结束**

**说明**

gs_copy_summary用于记录调用的copy语法及其详细情况，*[badfile]_bad.log*文件用于记录错误数据及其详细情况。为防止上一次导入时记录的错误数据以及详细情况被覆盖，建议每次执行导入时使用不同的bad参数。如果使用错误表pgxc_copy_error_log记录错误数据以及详细情况，请开启GUC参数enable_copy_error_log。如需删除表中的数据，可以对上述表执行truncate或者delete操作。

  

## 参数说明

**表 2-1** gs_loader 参数说明

| 参数 | 参数说明 | 参数类型：取值范围 |
| --- | --- | --- |
| help | 查看帮助信息。 | - |
| user | 数据库链接用户（与-U等价）。 | 字符串 |
| -U | 数据库链接用户（与user等价）。 | 字符串 |
| passwd | 用户密码（与-W等价）。 | 字符串 |
| -W | 用户密码（与passwd等价）。 | 字符串 |
| db | 数据库名称（必选，与-d等价）。 | 字符串 |
| -d | 数据库名称（必选与db等价）。 | 字符串 |
| host | 指定正在运行服务器的主机名、Unix域套接字的路径、或者域名。接受以“,”分隔的字符串来指定多个主机地址，支持指定多个主机地址（与-h等价）。<br>当指定多个主机地址时，默认选择连接到主节点。 | 参考gsql --host参数 |
| -h | 指定正在运行服务器的主机名、Unix域套接字的路径、或者域名。接受以“,”分隔的字符串来指定多个主机地址，支持指定多个主机地址（与host等价）。<br>当指定多个主机地址时，默认选择连接到主节点。 | 参考gsql --host参数 |
| port | 指定数据库服务器的端口号。可以配置一个或多个，当配置一个时，所有的IP都使用同一个端口连接；当配置多个时，顺序与IP顺序相同，个数必须与IP数相等，当不相等时会报错（与-p等价）。 | 参考gsql --port参考 |
| -p | 指定数据库服务器的端口号。可以配置一个或多个，当配置一个时，所有的IP都使用同一个端口连接；当配置多个时，顺序与IP顺序相同，个数必须与IP数相等，当不相等时会报错（与port等价）。 | 参考gsql --port参考 |
| create | 是否创建pgxc_copy_error_log和gs_copy_summary表。由于当前版本会默认创建这两张表，因此该参数失去意义。保留仅为了兼容性。 | [true, false]，默认true |
| clean | 是否清除本次错误记录。 | [true, false]，默认false |
| data | 数据文件，可以指定多个，或者通配符多字符通配(*)以及单字符通配(?)（必选）。 | 字符串 |

  

| 参数 | 参数说明 | 参数类型：取值范围 |
| --- | --- | --- |
| control | 控制文件名称（必选）。 | 字符串 |
| log | 日志文件名称。 | 字符串 |
| bad | 出错行以及详细情况记录文件名称，也可以指定目录，不指定时根据数据文件名生成。 | 字符串 |
| discard | WHEN匹配失败行记录文件名称，也可以指定目录，根据数据文件名生成。 | 字符串 |
| errors | 允许数据文件中出现多少出错行。 | 整数，默认0 |
| skip | 允许跳过数据文件的前多少行。 | 整数，默认0 |
| limit | 指定最多导入的行数。 | 整数，默认无限大 |
| bindsize | 仅做语法兼容不实现功能。 | - |
| rows | 多行提交参数，指定导入多少行数据后进行一次提交。 | 整数，取值范围[1, 2147483647] |
| compatible_nul | 是否开启数据中nul字符(0x00)兼容，开启后当数据文件中存在nul字符时，会先将nul字符转换成为空格字符' '(0x20)，再进行判断，加工和导入。 | [true, false]，默认true。 |
| compatible_illegal_chars | 是否开启非法字符容错功能，与COPY语法中的COMPATIBLE_ILLEGAL_CHARS容错规则和限制相同，详见《开发指南》的“SQL 语法 > COPY”章节的COPY_OPTION中COMPATIBLE_ILLEGAL_CHARS参数说明。 | [true, false]，默认false。 |
| parallel | 指定并行导入的并发度。当并发度大于1时，表示开启并行导入模式。当并发度等于1时，转化为串行导入。并发度最大值限定不超过客户端CPU核数的两倍。当客户端运行在容器中时，由于获取到的是主机的cpu数量，可能比容器实际能使用的多，并发度建议由用户自行控制在实际能使用的客户端cpu数量的两倍范围内。另外该能力通过多线程并发多个事务实现，实际并发度会受限于服务端的线程池模型，也同时会给服务端增加压力，请按实际情况合理设置并发度。 | 整数，取值范围[1, CPU核数两倍]，默认1 |
| binary | 数据文件是否为通过copy的binary模式导出的二进制文件。 | [true, false]，默认false。 |

  

**⚠ 注意**

- 参数均为小写，不支持大写，同时兼容gsql登录方式：-p端口号，-h主机，-d数据库，-U用户名，-W密码方式。
- 使用rows参数时，提交次数不要超过1000次，否则会对性能产生影响。提交次数约等于数据文件中数据行数除以rows参数取值。不指定rows参数时，rows无默认取值，表现为只进行一次提交，即所有数据都导入表中后进行一次事务提交。
- 小数据量频繁的提交会影响导入数据的性能，推荐合理配置rows参数的取值，保证每次提交的数据量大于5MB。对于常用的16U128G规格机器，一主两备部署场景下，向5个字段的表内导入13GB数据，排除网络影响，多次提交和单次提交（每次提交5MB数据）的速率基本持平，为10MB/s左右。
- compatible_nul参数实际控制guc参数loader_support_nul_character值的设置：
  - compatible_nul=true对应session级set loader_support_nul_charchter='s2'。
  - compatible_nul=false对应session级set loader_support_nul_character='s1'。

    建议通过命令行设置此参数且通过compatible_nul设置优先级高于guc_param中设置。
- 当前gs_loader仅支持数据文件中存在nul字符时的兼容，不支持ctl控制文件中存在nul字符。ctl文件中存在nul字符会存在不可预期的问题。
- gs_loader导入过程中，不需要转码场景下，单行数据最大不超过1GB-1；转码场景下单行数据最大不超过256MB-1。
- 如果是基本的数据迁移场景，建议先删除表上的索引，禁用表上的触发器，待数据迁移完成后再重建索引，恢复表上的触发器。对导入性能会有所提升。
- 指定binary参数为true后，有以下行要求：
  - 数据文件必须为通过\COPY中BINARY模式导出的二进制格式数据文件，但是该模式导出的数据文件通常兼容性及可移植性较差，建议直接使用\COPY语句进行导入。
  - gs_loader会将控制文件中语法转换为\COPY中BINARY模式下最简单的语法，即\COPY table_name FROM 'binary_file_path' BINARY; 语句。只解析控制文件中导入模式，表名信息和命令行中的control、data、binary、guc_param及数据库连接参数信息，不对其他参数语法进行解析和生效。
  - 对于gs_loader的命令行及控制文件中有以下要求：
    - 不支持字符集配置。
    - 不支持WHEN条件过滤及DISCARD生成。
    - 不支持enable_copy_error_log = off下将错误数据直接写入bad文件，enable_copy_error_log=on下将错误数据直接写入错误表。
    - 不支持配置CSV模式，不支持指定分隔符及包裹符，不支持TRAILING NULLCOLS语法。
    - 不支持数据类型配置、POSITION配置及列表达式使用。
    - 不支持FILLER、CONSTANT、SEQUENCE、NULLIF参数。
    - 不支持skip、rows、compatible_nul、compatible_illegal_chars参数。
- 指定parallel大于1时：
  - 当同时设置binary参数为true时，parallel参数失效，按串行导入。
  - 不支持在控制文件中设置OPTIONALLY ENCLOSED BY或者FIELDS CSV。

  

  - 不支持在控制文件中设置SEQUENCE列。
  - 无法保证数据按数据文件中的顺序导入。如果表中存在自增列，导入后自增列值的顺序无法保证与数据文件中顺序一致。
  - 同时使用errors参数时，errors参数的意义为每个子任务允许出现的最大错误行数。
  - 同时使用skip参数时，skip参数的意义是在整个数据文件开头跳过的行数。
  - 同时使用rows参数时，分批提交的批次各子任务独立计算。
  - 在客户端CPU、内存和服务端CPU、内存、空闲线程以及网络带宽不存在瓶颈，并且bad/discard总数不超过1%时，相比于串行导入，并发度为2/4/8时的性能提升不低于1.5/3/5倍。
  - 并发度每增加1，大约增加客户端10MB内存，服务端大约35MB内存。
- 当设置GUC参数support_zero_character为on时，表示数据库支持0x00字符的写入和读取，gs_loader导入数据时，会将0x00按照原始样式导入，而不是受其他兼容性参数影响转换成0x20。对于0字符处理场景，参数优先级为support_zero_character > copy_special_character_version > compatible_illegal_chars > loader_support_nul_character。
- 当设置了copy_special_character_version='no_error'时，优先级高于compatible_illegal_chars，非法编码的数据会原样导入，而不是发生转换。

## 控制文件

- 语法说明：

```
LOAD [ DATA ]
[CHARACTERSET char_set_name]
[INFILE [directory_path] [filename ] ]
[BADFILE [directory_path] [filename ] ]
[OPTIONS(name=value)]
[{ INSERT | APPEND | REPLACE | TRUNCATE }]
INTO TABLE table_name
[{ INSERT | APPEND | REPLACE | TRUNCATE }]
[FIELDS CSV]
[TERMINATED [BY] { 'string' }]
[OPTIONALLY ENCLOSED BY { 'string' }]
[TRAILING NULLCOLS]
[ WHEN { (start:end) | column_name } {= | !=} ‘string’ ]
[(
col_name [ [ POSITION ({ start:end }) ]  ["sql_string"] ] | [ FILLER [column_type [external] ] ] |
[ CONSTANT "string" ] | [ SEQUENCE ( { COUNT | MAX | integer } [, incr] ) ]|[NULLIF (COL=BLANKS)]
[, ...]
)]
```

- 参数说明：
  - **CHARACTERSET**

    字符集。

    取值范围：字符串，目前可指定为，'AL32UTF8','zhs16gbk','zhs32gb18030'。

    注意：控制文件中**CHARACTERSET**指定的字符集，应该和文件的编码格式保持一致，否则会报错或者导入数据乱码。
  - **INFILE**

    当前关键字无效，并在控制文件中需要单独占一行，运行时候会忽略该关键字。需要用户在gs_loader命令行参数中指定对应的数据文件。
  - **BADFILE**

  

当前关键字无效，运行时候会忽略该关键字，如果gs_loader 命令行参数没有指定badfile，则会根据对应控制文件名称生成对应的badfile文件。

- **OPTIONS**

  其中只有skip和rows功能生效，skip=n为导入时跳过前n条数据，rows=n为导入多少行数据后进行一次提交。命令行和控制文件同时指定时，命令行优先级更高。

- **INSERT | APPEND | REPLACE | TRUNCATE**

  导入模式。

  **INSERT**：如果表中有数据，则报错。

  **APPEND**：直接插入数据。

  **REPLACE**：如果表中有数据，则全部删除，然后再插入。

  **TRUNCATE**：如果表中有数据，则全部删除，然后再插入。

  **说明**

  - 在写控制文件(.ctl)文件时，在INTO TABLE table_name语句前后都可以指定（导入模式，INSERT | APPEND | REPLACE | TRUNCATE），使用优先级为：在INTO TABLE table_name语句后面指定导入模式优先级高于在INTO TABLE table_name语句前面指定导入模式，在INTO TABLE table_name语句后面指定导入模式会覆盖在前面指定的导入模式。
  - 当开启多个gs_loader会话，并发地向同一张表中导入数据时，推荐以APPEND的方式进行导入，以INSERT|REPLACE|TRUNCATE的方式会出现导入报错或数据导入不全的问题。

- **FIELDS CSV**

  标识使用copy的CSV模式。在CSV模式下分隔符缺省值为逗号，引号字符的缺省值为双引号。

  **说明**

  - 当前CSV模式下，被双引号包含的换行符被视为字段数据的一部分。
  - CSV模式下，设置了GUC参数a_format_copy_version为's1'时，会跳过字段开头空格。并且当某个字段第一个非空格字符不是enclosed字符时，忽略enclosed设置。当未匹配到关闭enclosed字符，先匹配到行末时，会进行报错。
  - CSV模式下，在不打开0字符GUC开关support_zero_characters时，如果使用了compatible_nul或者compatible_illegal_chars参数对0x00字符进行兼容，由于0x00转换0x20的行为发生在跳过开头空格的行为之前，因此字段开头的0x00字符会被当作0x20处理被删除掉。

- **table_name**

  表的名称（可以有模式修饰）。

  取值范围：已存在的表名。

- **TERMINATED [BY] { 'string' }**

  在文件中分隔各个字段的字符串，分隔符最大长度不超过10个字节。

  取值范围：不允许包含\.abcdefghijklmnopqrstuvwxyz0123456789中的任何一个字符。不支持将nul字符设置为分隔符。

  缺省值：在文本模式下，缺省是水平制表符，在CSV模式下是一个逗号。

  

**注意**

开启nul字符兼容，即compatible_nul=true，如果指定分隔符为' '空格字符(0x20)时需要注意，所判断的分隔符为数据文件中已存在的空格字符，并非nul字符转换而来的空格字符。

- **OPTIONALLY ENCLOSED BY { 'string' }**

  CSV格式文件下的引号字符。

  仅在使用FIELDS CSV参数明确说明的CSV模式下缺省值：双引号。

  其余模式下无缺省值。

  **说明**

  - 设置**OPTIONALLY ENCLOSED BY { 'string' }**时，数据左边可以不带引号字符，如果有引号字符，数据左右都必须为奇数个，但个数不必相等。
  - 当前仅CSV模式支持**OPTIONALLY ENCLOSED BY { 'string' }**。当指定**OPTIONALLY ENCLOSED BY { 'string' }**时，默认进入CSV模式。

- **TRAILING NULLCOLS**

  当数据加载时，若数据源文件中一行的多个字段缺失的处理方式。

  当一行数据的最后存在一个或多个字段为空时，按照空值处理将其导入到表中。不设置则会报错字段为空，将这行数据当作错误数据处理。

- **WHEN { (start:end) | column_name } {= | !=}**

  对行中的start到end之间的字符串，或者根据列名进行行过滤。

  取值范围：字符串。

  **说明**

  - 当GUC参数enable_copy_when_filler=on（默认）时，支持根据FILLER类型列进行过滤。当GUC参数enable_copy_when_filler=off时，则不支持。
  - WHEN条件后的常量字符串中不支持'\0'、'\r'等特殊字符。

- **POSITION ({ start:end })**

  对列进行处理，根据start到end范围获取对应字符串。

**须知**

使用POSITION用法表示在转换后的\COPY语句中使用FIXED FORMATTER的定长模式，会无法正确处理字段数据中的换行符。

- **"sql_string"**

  对列进行处理，列表达式，根据表达式计算列的取值。详见**•列表达式**。

  取值范围：字符串。

- **FILLER**

  对列进行处理，如果出现FILLER，则这个字段跳过。

  **说明**

  当前不支持FILLER与POSITION **({ start:end })**同时使用。

- **column_type [external]**

在导入数据时，根据不同的数据类型对数据进行处理。详见•**数据类型**。

- **CONSTANT**

  对列进行处理，将插入的对应字段设置为常量。

  取值范围：字符串。

- **SEQUENCE ( { COUNT | MAX | integer } [, incr] )**

  对列进行处理，生成对应的序列值。

  - COUNT：表示根据表中数据的行数开始计算。
  - MAX：表示根据表中这一列的最大值开始计算。
  - integer：表示从用户指定的值开始计算。
  - incr：表示每次递增多少。

- **NULLIF**

  在设置a_format_copy_version等于's1'时，当指定列的数据只包含空白字符时返回NULL，否则返回trim(COL)，等价于列表达式 "nullif(trim(COL), '')"。

  在设置a_format_copy_version不等于's1'时，对列进行处理，在多行导入场景中，若列名后未指定sysdate、constant、position、列表达式等运算时，执行导入操作，表现为未指定NULLIF关键字的列字段设置为空。

  当前只支持COL POSITION() CHAR NULLIF (COL=BLANKS)语法。具体使用详见•**NULLIF使用用例**。

**⚠ 注意**

- 不支持OPTIONS、INFILE、BADFILE，仅在特定场景下不报语法错误。
- gs_loader使用bad文件来记录出错数据，如果设置guc参数enable_copy_error_log开启错误表，该数据来自错误表的rawrecord字段，由于错误表对于以某种编码无法读起的错误不记录rawrecord，因此bad文件中遇到此情况时记录空行。
- gs_loader在设置guc参数a_format_load_with_constraints_violation开启支持约束冲突不回滚场景时，如果表带有BEFORE/AFTER ROW INSERT触发器，则每次提交行数不能超过1000万行。
- gs_loader在设置guc参数a_format_load_with_constraints_violation开启支持约束冲突不回滚场景时，不支持语句级触发器。

- bad文件对应数据为空的需要对应错误表的内容参考源文件和行号（不识别某种编码序列，不写bad文件内容，只记录空行）。

```
loader=# select * from pgxc_copy_error_log;
      relname       |          begintime          | filename | lineno | rawrecord |
detail
----------------------+-----------------------------+----------+--------+-----------
+-------------------------------------------------
 public.test_gsloader | 2023-02-09 09:20:33.646843-05 | STDIN    |      1 |           | invalid byte sequence
for encoding "UTF8": 0xb4
(1 row)
//如上例子对于loader对应的文件，查找数据文本第一行找出源数据
```

- NULLIF使用用例

```
// 建表
create table gsloader_test_nullif(
```

  

```
col1   varchar2(100) not null enable,
col2   number(5,0) not null enable,
col3   varchar2(200) not null enable,
col4   varchar2(34) not null enable,
col5   varchar2(750),
col6   number(20,0),
col7   varchar2(4000),
col8   varchar2(200)
);
// 数据文件 test.csv
6007 17060072021-09-0360070001102010000000230        1       600700010000218         0
1       1       229465       3
6007 17060072021-09-0360070001102010000000299        1       600700010000282         0
1       1       230467       3
6007 17060072021-09-0360070001102010000000242        1       600700010000255         0
1       1       226400       3
6007 17060072021-09-0360070001102010000000202        1       600700010000288         0
1       1       219107       3
6007 17060072021-09-0360070001102010000000294        1       600700010000243         0
1       1       204404       3
6007 17060072021-09-0360070001102010000000217        1       600700010000270         0
1       1       226644       3
// 控制文件 test.ctl
LOAD DATA
CHARACTERSET UTF8
TRUNCATE
INTO TABLE gsloader_test_nullif
TRAILING NULLCOLS
(COL1 POSITION(1:10) CHAR NULLIF (COL1 = BLANKS),
COL2  POSITION(11:14) CHAR NULLIF (COL2 = BLANKS),
COL3  POSITION(21:30) CHAR NULLIF (COL3 = BLANKS),
COL4  POSITION(31:40) CHAR NULLIF (COL4 = BLANKS),
COL5  sysdate,
COL6,
COL7,
COL8 POSITION(71:80) CHAR NULLIF (COL8 = BLANKS))
// 执行导入
gs_loader -p xxx host=xxx control=test.ctl  data=test.csv -d testdb -W xxx
// 导入结果：导入成功
loader=# select * from gsloader_test_nullif;
   col1    | col2 |    col3    |    col4    |        col5         | col6 | col7 |   col8
-----------+------+------------+------------+---------------------+------+------+-----------
 6007 17060 |  720 | 0360070001 | 1020100000 | 2023-05-17 20:32:03 |      |      | 010000218
 6007 17060 |  720 | 0360070001 | 1020100000 | 2023-05-17 20:32:03 |      |      | 010000282
 6007 17060 |  720 | 0360070001 | 1020100000 | 2023-05-17 20:32:03 |      |      | 010000255
 6007 17060 |  720 | 0360070001 | 1020100000 | 2023-05-17 20:32:03 |      |      | 010000288
 6007 17060 |  720 | 0360070001 | 1020100000 | 2023-05-17 20:32:03 |      |      | 010000243
 6007 17060 |  720 | 0360070001 | 1020100000 | 2023-05-17 20:32:03 |      |      | 010000270
(6 rows)
```

从导入表中的数据可以看出在使用NULLIF关键字后，除指定NULLIF运算和sysdate运算的列执行导入操作后导入字段正常，其余未指定运算的列表现为导入字段为空。

- 列表达式

  gs_loader支持对指定列进行表达式转换和场景扩展：

  ```
  ({ column_name [ data_type ] [ AS transform_expr ] } [, ...])
  ```

  其中data_type指定该列在表达式参数中的数据类型；transform_expr为目标表达式，返回与表中目标列数据类型一致的结果值。

  示例：

  – ctl文件中不指定列类型，源数据不满足表中列限制（数据类型限制、数据长度限制）。

    ```
    // 建表
    create table t_test(id int, text varchar(5));
    // 数据文件 test.csv
    addf2,bbbbaaa,20220907,
    ```

  

```
// 控制文件 test.ctl
Load Data
TRUNCATE INTO TABLE t_test
fields terminated by ','
TRAILING NULLCOLS(
id "length(trim(:id))",
text "replace(trim(:text),'bbbb','aa')"
)
// guc_param file
set a_format_copy_version='s1';
// 执行导入
gs_loader -p xxx host=xxx control=test.ctl  data=test.csv -d testdb -W xxx guc_param=test_guc.txt
// 导入结果：导入成功
select * from t_test;
 id | text
----+-------
  5 | aaaaa
(1 row)
```

- ctl文件中不指定列类型，隐式类型转换（涉及隐式类型转换，建议加上兼容性参数）。

```
// 建表
create table test(mes int, mes1 text, mes2 float8, mes3 timestamp with time zone, mes4
INTEGER);
// 数据文件
cat load_support_transform.data
1,mmoo,12.6789,Thu Jan 01 15:04:28 1970 PST,32767
2,yyds,180.883,Thu Jun 21 19:00:00 2012 PDT,32768
// 控制文件
cat load_support_transform.ctl
Load Data
TRUNCATE INTO TABLE test
fields terminated by ','
TRAILING NULLCOLS(
mes,
mes1 "mes1 || mes2",
mes2 "mes2 + 1",
mes3 "date_trunc('year', mes3)",
mes4
)
// guc_param file
cat test_guc.txt
set a_format_copy_version='s1';
set a_format_dev_version='s2';
set a_format_version='10c';
//执行导入
gs_loader -p xxx host=xxx control=load_support_transform.ctl data=load_support_transform.data
-d testdb -W xxx guc_param=test_guc.txt
// 导入结果：导入成功
select * from test;
 mes |    mes1     |  mes2  |          mes3          | mes4
-----+-------------+--------+------------------------+-------
   1 | mmoo12.6789 | 13.6789 | 1970-01-01 00:00:00+08 | 32767
   2 | yyds180.883 | 181.883 | 2012-01-01 00:00:00+08 | 32768
```

- 数据类型

  对应控制文件中的column_type [external]，在加载数据时，根据不同的数据类型对数据进行处理。gs_loader中可以将数据类型分为普通数据类型和特殊数据类型。

  - 普通数据类型

    - CHAR [(length)]：

      按照字段分隔符读取数据，并转换使用CHAR类型来保存值。length表示单条数据的最大长度，以字节为单位，通常一个字符占用一个字节，并且可以缺省，分为以下几种场景：

  

- 缺省对length长度的声明时，length的值会根据POSITION的声明来继承最大长度值。
- 声明了length的长度，则它会覆盖POSITION中对于最大长度的声明。
- 缺省了length的声明，同时也缺省了POSITION的声明，length的长度会根据分隔符间长度进行设置。
- 对于长度声明的优先级：length > POSITION > 分隔符。
- 缺省length，POSITION，分隔符的声明时，会从当前位置读到行结束符为止。
- 如果实际数据长度超过了length声明的最大长度，会报错。

■ INTEGER external [(length)]：

按照字段分隔符读取数据，并转换使用INTEGER类型来保存值。length的使用规则与CHAR类型中相同。

■ FLOAT external [(length)]：

按照字段分隔符读取数据，并转换使用FLOAT类型来保存值。length的使用规则与CHAR类型中相同。

■ DECIMAL external (length)：

按照字段分隔符读取数据，并转换使用DECIMAL类型来保存值。length的使用规则与CHAR类型中相同

■ TIMESTAMP：

按照字段分隔符读取数据，并转换使用TIMESTAMP类型来保存值。

■ DATE：

按照字段分隔符读取数据，并转换使用DATE类型来保存值。

■ DATE external：

按照字段分隔符读取数据，并转换使用DATE类型来保存值。

■ SYSDATE：

在数据库执行对应的插入时，取系统时间。该字段对应对应的值无法被引用使用，被引用使用的内容为SYSDATE字符串。

– 特殊数据类型

■ INTEGER：

无视字段分隔符读取四个字节长度的字符，按小端存储逻辑保存，然后将每个字符解析成十六进制ASCII码值，最后将整体转换为十进制数来保存值。

■ SMALLINT：

无视字段分隔符读取两个字节长度的字符，按小端存储逻辑保存，然后将每个字符解析成十六进制ASCII码值，最后将整体转换为十进制数来保存值。

示例：

```
// 建表
create table t_spec(col1 varchar(10), col2 varchar(10));
// 数据文件
```

  

```
cat t_spec.txt
1234,5678,
// 控制文件
cat t_spec.ctl
Load Data
TRUNCATE INTO TABLE t_spec
fields terminated by ','
TRAILING NULLCOLS(
col1 position(2:6) integer,
col2 position(5:8) smallint
)
// guc_param file
cat test_guc.txt
set a_format_copy_version='s1';
set a_format_dev_version='s2';
set a_format_version='10c';
// 执行导入
gs_loader -p xxx host=xxx control=t_spec.ctl data=t_spec.txt -d testdb -W xxx
guc_param=test_guc.txt
// 导入结果：导入成功
select * from t_spec;
   col1    | col2
-----------+-------
 741618482 | 13612
(1 row)
```

- RAW：

  会把每个字符解析成ASCII码值保存，转义字符“\”不执行转义操作。

  限制：RAW不能使用分隔符。

  示例：

```
// 建表
create table t_raw(col raw(50));
// 数据文件
cat t_raw.txt
12\n\x78!<~?'k^(%s)>/c[$50]
// 控制文件
cat t_raw.ctl
Load Data
TRUNCATE INTO TABLE t_raw
TRAILING NULLCOLS(
col position(1:50) raw
)
// guc_param file
cat test_guc.txt
set a_format_copy_version='s1';
set a_format_dev_version='s2';
set a_format_version='10c';
// 执行导入
gs_loader -p xxx host=xxx control=t_raw.ctl data=t_raw.txt -d testdb -W xxx
guc_param=test_guc.txt
// 导入结果：导入成功
select * from t_raw;
                          col
------------------------------------------------------------
 31325C6E5C783738213C7E3F276B5E282573293E2F635B2435305D
(1 row)
```

**⚠ 注意**

- 在多列导入场景中，不指定guc参数时，部分position与分隔符不能同时使用。
- 在多列导入场景中，SYSDATE和CONSTANT运算不能和POSITION运算同时使用。
- 指定数据类型导入时，包含普通数据类型需要通过guc_param设置a_format_copy_version参数，包含特殊数据类型则需要通过guc_param设置a_format_copy_version和a_format_dev_version及a_format_version参数。
- 列表达式涉及到系统函数时，需要根据对应功能通过guc_param设置合适的a_format_dev_version及a_format_version参数。
- 带length数据类型的使用，length需指定为大于0的整数；RAW数据类型作为特殊类型，RAW(length)的使用区别于普通类型的使用，如INTEGER EXTERNAL(length)的使用，当不指定position时，INTEGER EXTERNAL(length)表现为，当length小于文本文件（.csv/.txt等)中对应列数据长度时报错；当length大于文本文件（.txt)中对应列数据长度时，输出INTEGER EXTERNAL类型的结果。RAW(length)当不指定position时表现为读取length个字符。
- POSITION使用时，POSITION(start:end)，start需设置为大于0的整数，且end值应大于等于start的值。
- 指定POSITION时，在处理字段内容时不会省略尾部的空格；不指定POSITION时，处理字段内容时会省略尾部的空格，如果需要保留空格，需要在guc_param所指定的文件中，已设置好a_format_version的前提下，添加 set behavior_compat_options='char_coerce_compat'。
- 并发导入时，若多个gs_loader的discard文件名或bad文件名指向同一目录同名文件，则后一个执行的gs_loader会中止报错。若前一个已经导入完成，则文件被覆盖。

  报错如下：

  ```
  ERROR: An error occurred. Please check logfile.
  ```

  log文件中：

  ```
  …lock failed: Resource temporarily unavailable…
  ```

- 控制文件中对于字段值的部分若不为空且不使用本字段内容，则不占用数据文件的位置。

  比如控制文件如下：

  ```
  Load Data
  TRUNCATE INTO TABLE gsloader
  fields terminated by ','
  TRAILING NULLCOLS(
  id "trim(:id)",
  text "to_char(SYSDATE,'yyyymmdd')",
  gmt_create  "trim(:gmt_create)",
  create_str "trim(:create_str)"
  )
  ```

  数据文件如下：

  ```
  11,22,33,
  ```

  导入结果为：

  ```
  loader=# select * from gsloader;
  id |  text |     gmt_create      | create_str
  ```

```
----+---------+---------------------+-------------
11 | 2023-02-08 16:00:54 | 22 |  33
```

# 3 gs_dump

## 背景信息

gs_dump是GaussDB用于导出数据库相关信息的工具，用户可以自定义导出一个数据库或其中的对象（模式、表、视图等）。支持导出的数据库可以是默认数据库postgres，也可以是自定义数据库。

gs_dump支持导出PDB。当导出PDB时，用户可以自定义导出一个PDB或其中的对象（模式、表、视图等）。

gs_dump工具在进行数据导出时，其他用户可以访问数据库（读或写）。

gs_dump工具支持导出完整一致的数据。例如，T1时刻启动gs_dump导出A数据库，那么导出数据结果将会是T1时刻A数据库的数据状态，T1时刻之后对A数据库的修改不会被导出。

gs_dump时生成列不会被转储。

gs_dump时HTAP表创建的IMCV元信息（gs_imcv系统表）不会被转储。

gs_dump支持导出兼容v1数据库的文本格式文件。

gs_dump支持将数据库信息导出至纯文本格式的SQL脚本文件或其他归档文件中。

- 纯文本格式的SQL脚本文件：包含将数据库恢复为其保存时的状态所需的SQL语句。通过**gsql**运行该SQL脚本文件，可以恢复数据库。即使在其他主机和其他数据库产品上，只要对SQL脚本文件稍作修改，也可以用来重建数据库。
- 归档格式文件：包含将数据库恢复为其保存时的状态所需的数据，可以是tar格式、目录归档格式或自定义归档格式，详见**表3-1**。

gs_dump支持SSL加密通信，使用方式同gsql方式。

使用gs_dump前请确保gs_dump版本与数据库版本保持一致，高版本gs_dump不保证完全兼容低版本内核数据。

gs_dump不适合库中对象数量过多的场景。当库中对象数量过多，或者对象间依赖关系过于复杂时，gs_dump导出时间会很长。

## 主要功能

gs_dump可以创建四种不同的导出文件格式，通过**[**-F或者--format=**]**选项指定，具体如**表3-1**所示。

**表 3-1** 导出文件格式

| 格式名称 | -F的参数值 | 说明 | 建议 | 对应导入工具 |
| --- | --- | --- | --- | --- |
| 纯文本格式 | p | 纯文本脚本文件包含SQL语句和命令。命令可以由gsql命令行终端程序执行，用于重新创建数据库对象并加载表数据。 | 小型数据库，一般推荐纯文本格式。 | 使用**gsql**工具恢复数据库对象前，可根据需要使用文本编辑器编辑纯文本导出文件。 |
| 自定义归档格式 | c | 一种二进制文件。支持从导出文件中恢复所有或所选数据库对象。 | 中型或大型数据库，推荐自定义归档格式。 | 使用**gs_restore**可以选择要从自定义归档/目录归档/tar归档导出文件中导入相应的数据库对象。 |
| 目录归档格式 | d | 该格式会创建一个目录，该目录包含两类文件，一类是目录文件，另一类是每个表和blob对象对应的数据文件。 | - | |
| tar归档格式 | t | tar归档文件支持从导出文件中恢复所有或所选数据库对象。tar归档格式不支持压缩且对于单独表大小应小于8GB。 | - | |

**说明**

- 可以使用gs_dump工具将文件压缩为目录归档或自定义归档导出文件，减少导出文件的大小。生成目录归档或自定义归档导出文件时，默认进行中等级别的压缩。gs_dump程序无法压缩已归档导出文件。
- M-Compatibility模式数据库下，禁止在lower_case_table_names参数不同的实例之间进行导入导出，否则可能引起数据丢失。

## 注意事项

- 禁止修改-F c/d/t 格式导出的文件和内容，否则可能无法恢复成功。对于-F p 格式导出的文件，如有需要，可谨慎编辑导出的文件。
- 为了保证数据一致性和完整性，gs_dump会对需要转储的表设置共享锁。如果表在别的事务中设置了共享锁，gs_dump会等待锁释放后锁定表。如果无法在指定时间内锁定某个表，转储会失败。用户可以通过指定--lock-wait-timeout选项，自定义等待锁超时时间。
- 不支持加密导出存储过程和函数。
- 对于物化视图，本工具仅支持物化视图定义的导出，在导入后需手动执行REFRESH命令来进行数据恢复。
- 对于临时对象，本工具仅支持导出全局临时表。
- 本工具不支持在备机上使用。

- gs_dump导出分区索引时，部分索引分区的属性无法导出，比如索引分区的unusable状态。可以通过查询系统表PG_PARTITION或者查询视图ADM_IND_PARTITIONS/ADM_IND_SUBPARTITIONS获取索引分区的具体属性，通过ALTER INDEX命令可以手动设置索引分区属性。
- 对于定时任务，本工具仅支持导出在B兼容性数据库中，通过CREATE EVENT创建的定时任务或通过高级包创建的非周期性定时任务。
- gs_dump不支持导出自定义Tokenweight分词词典，可以根据报错WARNING: dictionary xx cannot be automatically exported, please create it manually手动创建对应分词词典。
- 在多租场景下，使用gs_dump导出时，不支持导出模板PDB，也不支持导出关闭的PDB。
- 在多租场景下，普通用户使用gs_dump导出时，只能导出该用户有权限的数据库对象和数据。
- 当未开启多租时，gs_dump不支持导出PDB及其内的对象。
- gs_dump时HTAP表创建IMCV元信息(gs_imcv系统表)不会被转储。
- 如果数据库中存在初始用户创建的表且表上有含用户自定义函数的表达式索引，系统管理员使用gs_dump导出后，需要使用初始用户通过gsql或gs_restore进行导入。否则会因为安全原因，导致创建索引失败。

**说明**

普通用户不支持导出DIRECTORY、SYNONYM，若普通用户进行相关导出，会提示“WARNING: xx not dumped because current user is not a superuser”。

## 语法

```
gs_dump [OPTION]... [DBNAME]
```

**说明**

“dbname”前面不需要加短或长选项。“dbname”指定要连接的数据库。

例如：

不需要-d，直接指定“dbname”。

```
gs_dump -p port_number  testdb -f dump1.sql
```

或者

```
export PGDATABASE=testdb
gs_dump -p port_number -f dump1.sql
```

环境变量：PGDATABASE

## 参数说明

通用参数：

- -f, --file=<FILE_NAME>

  将输出发送至指定文件或目录。如果省略该参数，则使用标准输出。如果输出格式为(-F c/-F d/-F t)时，必须指定-f参数。如果-f的参数值含有目录，要求目录对当前用户具有读写权限。

- -F, --format=c|d|t|p

  选择输出格式。格式如下：

  - p|plain：输出一个文本SQL脚本文件（默认）。

  

  - c|custom：输出一个自定义格式的归档，并且以目录形式输出，作为gs_restore输入信息。该格式是最灵活的输出格式，因为能手动选择，而且能在恢复过程中将归档项重新排序。该格式默认状态下会被压缩。
  - d|directory：该格式会创建一个目录，该目录包含两类文件，一类是目录文件，另一类是每个表和blob对象对应的数据文件。
  - t|tar：输出一个tar格式的归档形式，作为gs_restore输入信息。tar格式与目录格式兼容；tar格式归档形式在提取过程中会生成一个有效的目录格式归档形式。但是，tar格式不支持压缩且对于单独表有8GB的大小限制。此外，表数据项的相应排序在恢复过程中不能更改。

- -v, --verbose

  指定verbose模式。该选项将导致gs_dump向转储文件输出详细的对象注解和启动/停止次数，向标准错误流输出处理信息。

- -V, --version

  打印gs_dump版本，然后退出。

- -Z, --compress=0-9

  指定使用的压缩比级别。

  取值范围：0~9

  - 0表示无压缩。
  - 1表示压缩比最小，处理速度最快。
  - 9表示压缩比最大，处理速度最慢。

  针对自定义归档格式，该选项指定单个表数据片段的压缩，默认方式是以中等级别进行压缩。tar归档格式和纯文本格式目前不支持压缩。

- --lock-wait-timeout=TIMEOUT

  请勿在转储刚开始时一直等待以获取共享表锁。如果无法在指定时间内锁定某个表，就选择失败。可以以任何符合SET statement_timeout的格式指定超时时间。

- -?, --help

  显示gs_dump命令行参数帮助，然后退出。

转储参数：

- -a, --data-only

  只输出数据，不输出模式(数据定义)。转储表数据、大对象和序列值。

- -b, --blobs

  该参数为扩展预留接口，不建议使用。

- -c, --clean

  在将创建数据库对象的指令输出到备份文件之前，先将清理（删除）数据库对象的指令输出到备份文件中。（如果目标数据库中没有任何对象，gsql或gs_restore工具可能会输出一些提示性的错误信息。）

  该选项只对文本格式有意义。针对归档格式，可以在调用gs_restore时指定选项。

- -C, --create

  备份文件以创建数据库和连接到创建的数据库的命令开始（如果命令脚本是这种方式执行，无所谓在运行脚本之前连接的是哪一个数据库）。

  该选项只对文本格式有意义。针对归档格式，可以在调用gs_restore时指定选项。

  

**说明**

- 在多租场景下，使用gs_dump导出指定PDB时，不支持使用该选项。
- 在M-Compatibility模式数据库下，不支持使用该选项。必须先在目标实例上创出M-Compatibility模式数据库，然后在源实例上导出，最后在目标实例连接新创出的M-Compatibility模式数据库导入。

- -E, --encoding=ENCODING

  以指定的字符集编码创建转储。默认情况下，以数据库编码创建转储。（得到相同结果的另一个办法是将环境变量“PGCLIENTENCODING”设置为所需的转储编码）

  **说明**

  当指定转储编码存在转码场景时，且表中的数据存在非法编码的数据，导出会报错invalid byte sequence，建议使用gs_dump的-s参数只导出定义，并单独使用COPY打开编码容错进行数据的导出与导入。

- -n, --schema=SCHEMA

  只转储与模式名称匹配的模式，此选项包括模式本身和所有它包含的对象。如果该选项没有指定，所有在目标数据库中的非系统模式将会被转储。写入多个-n选项来选择多个模式。此外，根据gsql的\d命令所使用的相同规则，模式参数可被理解成一个pattern，所以多个模式也可以通过在该pattern中写入通配符来选择。使用通配符时，注意给pattern打引号，防止shell扩展通配符。

  **说明**

  - 当-n已指定时，gs_dump不会转储已选模式所附着的任何其他数据库对象。因此，无法保证某个指定模式的转储结果能够自行成功地储存到一个空数据库中。
  - 当-n指定时，非模式对象不会被转储。
  - M-Compatibility兼容模式下， 通过CREATE DATABASE带templatem创建的数据库直接通过指定db_name导出数据； 而通过CREATE DATABASE db_name创建出来的DATABASE与Schema等效，只能通过-n导出数据。
  - GaussDB会自动将对象名称中的大写字母转为小写，当模式名称中包含大写字母时，需要添加额外的引号，如：-n '"Sch1"' 或 -n "\"Sch1\""
  - M-Compatibility兼容模式下，该参数值受GUC参数lower_case_table_names影响。比如在大小写敏感模式下（lower_case_table_names=0），该参数值也需要大小写敏感，如果包含大写字母，则需要添加额外的引号，否则效果相当于小写。在大小写不敏感模式下（lower_case_table_names=1），该参数值需要传小写名称。

  转储支持多个模式的转储。多次输入-n schemaname转储多个模式。

  例如：

  ```
  gs_dump -h host_name -p port_number testdb -f backup/bkp_shl2.sql -n sch1 -n sch2
  ```

  在上面这个例子中，sch1和sch2会被转储。

- -N, --exclude-schema=SCHEMA

  不转储任何与模式pattern匹配的模式。Pattern将参照针对-n的相同规则来理解。可以通过输入多次-N，不转储与任何pattern匹配的模式。

  当同时输入-n和-N时，会转储与至少一个-n选项匹配、与-N选项不匹配的模式。如果有-N没有-n，则不转储常规转储中与-N匹配的模式。

  转储过程支持排除多个模式。

  在转储过程中，输入-N exclude schema name排除多个模式。

  例如：

  ```
  gs_dump -h host_name -p port_number testdb -f backup/bkp_shl2.sql -N sch1 -N sch2
  ```

  在上面这个例子中，sch1和sch2在转储过程中会被排除。

  

**说明**

GaussDB会自动将对象名称中的大写字母转为小写，当模式名称中包含大写字母时，需要添加额外的引号，如：-N '"Sch1"' 或 -N "\"Sch1\""

- -o, --oids

  转储每个表的对象标识符（OIDs），作为表的一部分数据。该选项用于应用以某种方式（例如：外键约束方式）参照了OID列的情况。如果不是以上这种情况，请勿使用该选项。

- -O, --no-owner

  不输出设置对象的归属这样的命令，以匹配原始数据库。默认情况下，gs_dump会发出ALTER OWNER或SET SESSION AUTHORIZATION语句设置所创建的数据库对象的归属。如果脚本正在运行，该语句不会执行成功，除非是由系统管理员触发（或是拥有脚本中所有对象的同一个用户）。通过指定-O，编写一个任何用户都能存储的脚本，且该脚本会授予该用户拥有所有对象的权限。

  该选项只对文本格式有意义。针对归档格式，可以在调用gs_restore时指定选项。

- -s, --schema-only

  只转储对象定义（模式），而非数据。

- -S, --sysadmin=NAME

  该参数为扩展预留接口，不建议使用。

- -t, --table=TABLE

  指定转储的表（或视图、或序列、或外表）对象列表，可以使用多个-t选项来选择多个表，也可以使用通配符指定多个表对象。

  当使用通配符指定多个表对象时，注意给pattern打引号，防止shell扩展通配符。

  当使用-t时，-n和-N没有任何效应，这是因为由-t选择的表的转储不受那些选项的影响。

  **说明**

  - -t参数选项个数必须小于等于100。
  - 如果-t参数选项个数大于100，建议使用参数--include-table-file来替换。
  - 当-t已指定时，gs_dump不会转储已选表所附着的任何其他数据库对象。因此，无法保证某个指定表的转储结果能够自行成功地储存到一个空数据库中。
  - -t tablename只转储在默认搜索路径中可见的表。-t *.tablename转储数据库下所有模式下的tablename表。-t schema.table转储特定模式中的表。
  - -t tablename不会导出表上的触发器信息。
  - 对于表名中包含大写字母的表，在使用-t参数指定导出时需对表名添加\"来导出。如对于表"abC"，导出需指定-t \"abC\"；如对于表schema."abC"，导出需指定-t schema.\"abC\"。
  - -t "" 不匹配任何表
  - M-Compatibility兼容模式下，该参数值受GUC参数lower_case_table_names影响。比如在大小写敏感模式下（lower_case_table_names=0），该参数值也需要大小写敏感，如果包含大写字母，则需要添加额外的引号，否则效果相当于小写。在大小写不敏感模式下（lower_case_table_names=1），该参数值需要传小写名称。

  例如：

  ```
  gs_dump -h host_name -p port_number testdb -f backup/bkp_shl2.sql -t schema1.table1 -t schema2.table2
  ```

  在上面这个例子中，schema1.table1和schema2.table2会被转储。

- --include-table-file=<FILE_NAME>

  

指定需要dump的表文件。

- -T, --exclude-table=TABLE

  不转储的表（视图、序列、外表）对象列表，可以使用多个-T选项来选择多个表，也可以使用通配符指定多个表对象。

  当同时输入-t和-T时，会转储在-t列表中，而不在-T列表中的表对象。

  例如：

  ```
  gs_dump -h host_name -p port_number testdb -f backup/bkp_shl2.sql -T table1 -T table2
  ```

  在上面这个例子中，table1和table2在转储过程中会被排除。

- --exclude-table-file=<FILE_NAME>

  指定不需要dump的表文件。

  **说明**

  同--include-table-file，其内容格式如下：
  schema1.table1
  schema2.table2
  .....

- -x, --no-acl

  防止转储访问权限（授权/撤销命令），只影响acl对象，不影响privilege对象。

- -q, --target

  指定导出兼容其他版本数据库的文本文件，目前支持v1和v5参数，指定其他参数不会报错，但不会生效。v1参数用于导出GaussDB v5版本数据库的数据为兼容GaussDB v1版本的文本文件。v5参数用于导出GaussDB v5版本数据库的数据为GaussDB v5版本数据库格式的文本文件，减少了导入GaussDB v5版本数据库时的可能的报错情况。

  在使用v1参数时，建议和--exclude-guc="enable_cluster_resize"，--exclude-function，--exclude-with等选项共用，否则导入到GaussDB v1版本数据库时可能报错。

- -g, --exclude-guc

  该参数为扩展预留接口，不建议使用。

- --exclude-function

  不导出函数和存储过程。

- --exclude-with

  导出的表定义，末尾不添加WITH(orientation=row，compression=on）这样的描述。

- --binary-upgrade

  该参数为扩展预留接口，不建议使用。

  **说明**

  M-Compatibility模式数据库不支持该选项。

- --binary-upgrade-usermap="USER1=USER2"

  该参数为扩展预留接口，不建议使用。

- --column-inserts/--attribute-inserts

  以INSERT命令带列名（INSERT INTO表（列、…）值…）方式导出数据。这会导致恢复缓慢。但是由于该选项会针对每行生成一个独立分开的命令，所以在重新加载某行时出现的错误只会导致对应的一行数据丢失，而非整个表内容。

  

**说明**

M-Compatibility模式数据库不支持该选项。

- --disable-dollar-quoting

  该选项将禁止在函数体前使用美元符号$，并强制使用SQL标准字符串语法对其进行引用。

- --include-alter-table

  dump后的表删除列。

- --disable-triggers

  该参数为扩展预留接口，不建议使用。

- --exclude-table-data=TABLE

  指定不转储任何匹配表pattern的表这方面的数据。依照针对-t的相同规则理解该pattern。

  可多次输入--exclude-table-data来排除匹配任何pattern的表。当用户需要特定表的定义但不需要其中的数据时，这个选项很有帮助。

  排除数据库中所有表的数据，请参见**-s, --schema-only**。

- --inserts

  发出INSERT命令（而非COPY命令）时转储数据。这会导致恢复缓慢。

  但是由于该选项会针对每行生成一个独立分开的命令，所以在重新加载某行时出现的错误只会导致对应的一行数据丢失，而非整个表内容。注意如果重排列顺序，可能会导致恢复整个失败。列顺序改变时，--column-inserts选项不受影响，虽然会更慢。

  **说明**

  M-Compatibility模式数据库不支持该选项。

- --no-security-labels

  该参数为扩展预留接口，不建议使用。

- --no-tablespaces

  不输出选择表空间的命令。使用该选项，无论默认表空间是哪一个，在恢复过程中所有对象都会被创建。

  该选项只对文本格式有意义。针对归档格式，可以在调用gs_restore时指定选项。

- --no-unlogged-table-data

  该参数为扩展预留接口，不建议使用。

- --non-lock-table

  该参数仅供软件间接口调用。

- --quote-all-identifiers

  强制对所有标识符加引号。为了向后续版本迁移，且其中可能涉及引入额外关键词，在转储相应数据库时该选项会有帮助。

  **说明**

  M-Compatibility模式数据库不支持该选项。

- --section=SECTION

  指定已转储的名称区段（pre-data、data、和post-data）。

  

- --serializable-deferrable

  转储过程中使用可串行化事务，以确保所使用的快照与之后的数据库状态一致；要实现该操作需要在无异常状况的事务流中等待某个点，因为这样才能保证转储成功，避免引起其他事务出现serialization_failure要重新再做。

  但是该选项对于灾难恢复没有益处。对于在原始数据库进行升级的时候，加载一个数据库的复制作为报告或其他只读加载共享的转储是有帮助的。没有这个选项，转储会反映一个与任何事务最终提交的序列化执行不一致的状态。

  如果当gs_dump启动时，读写事务仍处于非活动状态，即便使用该选项也不会对其产生影响。如果读写事务处于活动状态，转储的开始时间可能会延迟一段不确定的时间。

- --use-set-session-authorization

  输出符合SQL标准的SET SESSION AUTHORIZATION命令而不是ALTER OWNER命令来确定对象所有权。这样令转储更加符合标准，但是如果转储文件中的对象的历史有些问题，那么可能不能正确恢复。并且，使用SET SESSION AUTHORIZATION的转储需要数据库系统管理员的权限才能转储成功，而ALTER OWNER需要的权限则低得多。但是SET SESSION AUTHORIZATION部分支持直接使用密文密码，因为使用此参数导出的脚本可能无法正常恢复，不建议使用此参数导出。

  **说明**

  SET SESSION AUTHORIZATION使用范围：

  - 系统管理员可以通过SET SESSION AUTHORIZATION语句切换到普通用户，无法切换到初始用户, 其他sysadmin、opradmin、monadmin、poladmin和auditadmin。
  - 其他用户无法通过SET SESSION AUTHORIZATION语句切换用户。

- --with-encryption=AES128

  指定转储数据需用AES128进行加密。

- --with-key=KEY

  AES128密钥规则如下：

  - 密钥长度为8~16个字符。
  - 至少包含大写字母（A-Z），小写字母（a-z），数字（0-9），非字母数字字符（限定为~!@#$%^&*()-_=+\|[{}];:,<.>/?）四类字符中的三类字符。

  **说明**

  不支持加密导出存储过程和函数。

- --with-salt=RANDVALUES

  gs_dumpall使用此参数传递随机值。

- --include-extensions

  在转储中包含扩展。

  **须知**

  扩展功能为内部使用功能，不建议用户使用。

- --include-depend-objs

  备份结果包含依赖于指定对象的对象信息。该参数需要同-t/--include-table-file参数关联使用才会生效。

  

- --exclude-self

  备份结果不包含指定对象自身的信息。该参数需要同-t/--include-table-file参数关联使用才会生效。

- --pipeline

  使用管道传输密码，禁止在终端使用。

- --dont-overwrite-file

  文本、tar、以及自定义格式情况下会重写现有文件。这对目录格式不适用。

  例如：

  设想这样一种情景，即当前目录下backup.sql已存在。如果在输入命令中输入-f backup.sql选项时，当前目录恰好也生成backup.sql，文件就会被重写。

  如果备份文件已存在，且输入--dont-overwrite-file选项，则会报告附带‘转储文件已经存在’信息的错误。

  ```
  gs_dump -p port_number testdb -f backup.sql -F plain --dont-overwrite-file
  ```

**说明**

- -s/--schema-only和-a/--data-only不能同时使用。
- -c/--clean和-a/--data-only不能同时使用。
- --inserts/--column-inserts和-o/--oids不能同时使用，因为INSERT命令不能设置OIDS。
- --role和--rolepassword必须一起使用。
- --binary-upgrade-usermap和--binary-upgrade必须一起使用。
- --include-depend-objs/--exclude-self需要同-t/--include-table-file参数关联使用才会生效。
- --exclude-self必须同--include-depend-objs一起使用。
- --with-encryption=AES128仅支持-F p/plain。
- --with-key=KEY仅支持-F p/plain。
- --with-salt=RANDVALUES由gs_dumpall调用，不需要用户手动输入。

连接参数：

- -h, --host=HOSTNAME

  指定主机名称。如果数值以斜杠开头，则被用作到Unix域套接字的路径。缺省从PGHOST环境变量中获取（如果已设置），否则，尝试一个Unix域套接字连接。

  该参数只针对数据库外，对数据库内本机IPv4用127.0.0.1，IPv6用::1。

  例如：主机名

  环境变量：PGHOST

- -p, --port=PORT

  指定主机端口。在开启线程池情况下，建议使用 pooler port，即主机端口+1。

  环境变量：PGPORT

- -U, --username=NAME

  指定所连接主机的用户名，跨节点执行不支持使用初始用户。

  环境变量：PGUSER

- -w, --no-password

  不出现输入密码提示。如果主机要求密码认证并且密码没有通过其它形式给出，则连接尝试将会失败。 该选项在批量工作和不存在用户输入密码的脚本中很有帮助。

  

- -W, --password=PASSWORD

  指定用户连接的密码。如果主机的认证策略是trust，则不会对系统管理员进行密码验证，即无需输入-W选项；如果不加此参数，并且不是系统管理员，则会提示交互式输入，为了系统安全，推荐使用交互式输入密码方式。

- --role=ROLENAME

  指定创建转储使用的角色名。选择该选项，会使gs_dump连接数据库后，发起一个SET ROLE角色名命令。当所授权用户（由-U指定）没有gs_dump要求的权限时，该选项会起到作用，即切换到具备相应权限的角色。某些安装操作规定不允许直接以超系统管理员身份登录，而使用该选项能够在不违反该规定的情况下完成转储。

- --rolepassword=ROLEPASSWORD

  指定角色名的密码。

## 说明

如果某数据库有任何本地数据要添加到template1数据库，请谨慎将gs_dump的输出恢复到一个真正的空数据库中，否则可能会因为被添加对象的定义被复制，出现错误。要创建一个无本地添加的空数据库，需从template0而非template1复制，例如：

```
CREATE DATABASE foo WITH TEMPLATE template0;
```

tar归档形式的文件大小不得超过8GB（tar文件格式的固有限制）。tar文档整体大小和任何其他输出格式没有限制，操作系统可能对此有要求。

由gs_dump生成的转储文件不包含优化程序用来做执行计划决定的统计数据。因此，建议从某转储文件恢复之后运行ANALYZE以确保最佳效果。转储文件不包含任何ALTER DATABASE…SET命令，这些设置由gs_dumpall转储，还有数据库用户和其他完成安装设置。

## 示例

使用gs_dump转储数据库为SQL文本文件或其它格式的操作，如下所示。

示例中“backup/MPPDB_backup.sql”表示导出的文件，其中backup表示相对于当前目录的相对目录；“37300”表示数据库服务器端口；“testdb”表示要访问的数据库名。

**说明**

导出操作时，请确保该目录存在并且当前的操作系统用户对其具有读写权限。

示例1：执行gs_dump，导出testdb数据库全量信息，导出的MPPDB_backup.sql文件格式为纯文本格式。

```
gs_dump -U omm -f backup/MPPDB_backup.sql -p 37300 testdb -F p
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 09:49:17]: The total objects number is 356.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 09:49:17]: [100.00%] 356 objects have been dumped.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 09:49:17]: dump database testdb successfully
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 09:49:17]: total time: 1274  ms
```

使用gsql程序从纯文本导出文件中导入数据。

示例2：执行gs_dump，导出testdb数据库全量信息，导出的MPPDB_backup.tar文件格式为tar格式。

  

```
gs_dump -U omm -f backup/MPPDB_backup.tar -p 37300 testdb -F t
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:02:24]: The total objects number is 1369.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:02:53]: [100.00%] 1369 objects have been dumped.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:02:53]: dump database testdb successfully
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:02:53]: total time: 50086  ms
```

示例3：执行gs_dump，导出testdb数据库全量信息，导出的MPPDB_backup.dmp文件格式为自定义归档格式。

```
gs_dump -U omm -f backup/MPPDB_backup.dmp -p 37300 testdb -F c
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:05:40]: The total objects number is 1369.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:06:03]: [100.00%] 1369 objects have been dumped.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:06:03]: dump database testdb successfully
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:06:03]: total time: 36620  ms
```

示例4：执行gs_dump，导出testdb数据库全量信息，导出的MPPDB_backup文件格式为目录格式。

```
gs_dump -U omm -f backup/MPPDB_backup -p 37300  testdb -F d
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:16:04]: The total objects number is 1369.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:16:23]: [100.00%] 1369 objects have been dumped.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:16:23]: dump database testdb successfully
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:16:23]: total time: 33977  ms
```

示例5：执行gs_dump，导出testdb数据库信息，但不导出/home/MPPDB_temp.sql中指定的表信息。导出的MPPDB_backup.sql文件格式为纯文本格式。

```
gs_dump -U omm -p 37300 testdb --exclude-table-file=/home/MPPDB_temp.sql -f backup/MPPDB_backup.sql
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:37:01]: The total objects number is 1367.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:37:22]: [100.00%] 1367 objects have been dumped.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:37:22]: dump database testdb successfully
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-27 10:37:22]: total time: 37017  ms
```

示例6：执行gs_dump，仅导出依赖于指定表testtable的视图信息。然后创建新的testtable表，再恢复依赖其上的视图。

备份仅依赖于testtable的视图

```
gs_dump -U omm -s -p 37300 testdb -t PUBLIC.testtable --include-depend-objs --exclude-self -f backup/MPPDB_backup.sql -F p
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-15 14:12:54]: The total objects number is 331.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-15 14:12:54]: [100.00%] 331 objects have been dumped.
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-15 14:12:54]: dump database testdb successfully
gs_dump[user='omm'][localhost][port='37300'][testdb][2018-06-15 14:12:54]: total time: 327  ms
```

修改testtable名称

```
gsql -p 37300 testdb -r -c "ALTER TABLE PUBLIC.testtable RENAME TO testtable_bak;"
```

创建新的testtable表

```
CREATE TABLE PUBLIC.testtable(a int, b int, c int);
```

  

还原依赖于testtable的视图

```
gsql -p 37300 testdb -r -f backup/MPPDB_backup.sql
```

示例7：在多租场景下，执行gs_dump，导出名称为testpdb的PDB的全量信息，导出的backup_pdb.sql文件格式为纯文本格式。

```
gs_dump -U omm testpdb -f backup/backup_pdb.sql -p 20000 -F p
gs_dump[user='omm'][localhost][port='20000'][testpdb][2024-04-26 17:41:21]: The total objects number is 459.
gs_dump[user='omm'][localhost][port='20000'][testpdb][2024-04-26 17:41:21]: [100.00%] 459 objects have been dumped.
gs_dump[user='omm'][localhost][port='20000'][testpdb][2024-04-26 17:41:21]: dump database testpdb successfully
gs_dump[user='omm'][localhost][port='20000'][testpdb][2024-04-26 17:41:21]: total time: 5427  ms
```

示例8：在多租场景下，执行gs_dump，导出名称为testpdb的PDB的全量信息，导出的backup_pdb_t.tar文件格式为tar格式。

```
gs_dump -U omm testpdb -p 20000 -f backup/backup_pdb_t.tar -F t
gs_dump[user='omm'][localhost][port='20000'][testpdb][2024-04-26 17:02:40]: The total objects number is 459.
gs_dump[user='omm'][localhost][port='20000'][testpdb][2024-04-26 17:02:40]: [100.00%] 459 objects have been dumped.
gs_dump[user='omm'][localhost][port='20000'][testpdb][2024-04-26 17:02:40]: dump database testpdb successfully
gs_dump[user='omm'][localhost][port='20000'][testpdb][2024-04-26 17:02:40]: total time: 5506  ms
```

示例9：在多租场景下，执行gs_dump，导出名称为testpdb的PDB的全量信息，导出的backup_pdb_c文件格式为自定义归档格式。

```
gs_dump -U omm testpdb -p 20000 -f backup/backup_pdb_c -F c
gs_dump[user='omm'][localhost][port='20000'][testpdb][2024-04-26 16:57:19]: The total objects number is 459.
gs_dump[user='omm'][localhost][port='20000'][testpdb][2024-04-26 16:57:19]: [100.00%] 459 objects have been dumped.
gs_dump[user='omm'][localhost][port='20000'][testpdb][2024-04-26 16:57:19]: dump database testpdb successfully
gs_dump[user='omm'][localhost][port='20000'][testpdb][2024-04-26 16:57:19]: total time: 5622  ms
```

示例10：在多租场景下，执行gs_dump，导出名称为testpdb的PDB的全量信息，导出的backup_pdb_dir文件格式为目录格式。

```
gs_dump -U omm testpdb -p 20000 -f backup/backup_pdb_dir -F d
gs_dump[user='omm'][localhost][port='20000'][testpdb][2024-04-26 17:05:46]: The total objects number is 459.
gs_dump[user='omm'][localhost][port='20000'][testpdb][2024-04-26 17:05:46]: [100.00%] 459 objects have been dumped.
gs_dump[user='omm'][localhost][port='20000'][testpdb][2024-04-26 17:05:46]: dump database testpdb successfully
gs_dump[user='omm'][localhost][port='20000'][testpdb][2024-04-26 17:05:46]: total time: 5680  ms
```

示例11：在多租场景下，执行gs_dump，使用-C, --create选项导出名称为testpdb的PDB的全量信息，gs_dump报错退出

```
gs_dump -U omm testpdb -C -p 20000 -f backup/backup_pdb_dir -F d
gs_dump unsupport the '-C, --create' option for pdb.
```

## 相关命令

**gs_dumpall**

  

# 4 gs_dumpall

## 背景信息

gs_dumpall是GaussDB用于导出所有数据库相关信息工具，它可以导出数据库的所有数据，包括默认数据库postgres的数据、自定义数据库的数据、以及所有数据库公共的全局对象。

gs_dumpall工具在进行数据导出时，其他用户可以访问数据库（读或写）。

gs_dumpall工具支持导出完整一致的数据。例如，T1时刻启动gs_dumpall导出整个数据库，那么导出数据结果将会是T1时刻该数据库的数据状态，T1时刻之后对数据库的修改不会被导出。

gs_dumpall时生成列不会被转储。

gs_dump时HTAP表创建IMCV元信息（gs_imcv系统表）不会被转储。

gs_dumpall在导出所有数据库时分为两部分：

- gs_dumpall自身对所有数据库公共的全局对象进行导出，包括有关数据库用户和组，表空间以及属性（例如，适用于数据库整体的访问权限）信息。
- gs_dumpall通过调用gs_dump来完成数据库中各数据库的SQL脚本文件导出，该脚本文件包含将数据库恢复为其保存时的状态所需要的全部SQL语句。

以上两部分导出的结果为纯文本格式的SQL脚本文件，使用**gsql**运行该脚本文件可以恢复数据库。

gs_dumpall支持SSL加密通信，使用方式同gsql方式。

使用gs_dumpall前请确保gs_dumpall版本与gs_dump版本、数据库版本保持一致，高版本gs_dump、gs_dump不保证完全兼容低版本内核数据。

**说明**

M-Compatibility模式数据库下，禁止在lower_case_table_names参数不同的实例之间进行导入导出，否则可能引起数据丢失。

## 注意事项

- 禁止修改导出的文件和内容，否则可能无法恢复成功。
- 为了保证数据一致性和完整性，gs_dumpall会对需要转储的表设置共享锁。如果某张表在别的事务中设置了共享锁，gs_dumpall会等待此表的锁释放后锁定此

  

表。如果无法在指定时间内锁定某张表，转储会失败。用户可以通过指定--lock-wait-timeout选项，自定义等待锁超时时间。

- 由于gs_dumpall读取所有数据库中的表，因此必须以数据库管理员身份进行连接，才能导出完整文件。在使用gsql执行脚本文件导入时，同样需要管理员权限，以便添加用户和组，以及创建数据库。导入备份前，验证其安全性，防止管理员权限被利用。
- 使用gs_dumpall导出所有数据库对象，并希望在新的实例环境上进行导入时，需要保证导出和导入时使用用户的名称和权限相同，否则会出现名称不一致或权限不足的报错。
- gs_dumpall导出的文件，在新的实例环境上进行导入时，连接的数据库必须是默认database。否则可能出现语法不兼容的情况，导致导入出错。
- 由于M-compatibility数据库全局只能创建一个，当导出的数据库中包含M-compatibility数据库时，须保证导入的目标实例环境上不存在其他的M-compatibility模式数据库。
- 对于定时任务，本工具仅支持导出在B兼容性数据库中，通过CREATE EVENT创建的定时任务或通过高级包创建的非周期性定时任务。
- gs_dumpall不支持导出自定义Tokenweight分词词典，可以根据报错WARNING: dictionary xx cannot be automatically exported, please create it manually手动创建对应分词词典。
- 当导出的数据库中包含PDB时，使用gs_dumpall不会导出模板PDB及创建的PDB。
- gs_dump时HTAP表创建IMCV元信息（gs_imcv系统表）不会被转储。
- 当指定转储编码存在转码场景时，且表中的数据存在非法编码的数据，导出会报错invalid byte sequence，建议使用gs_dump的-s参数只导出定义，并单独使用COPY打开编码容错进行数据的导出与导入。

## 语法

**gs_dumpall** [OPTION]...

## 参数说明

通用参数：

- -f, --filename=<FILE_NAME>

  将输出发送至指定文件。如果这里省略，则使用标准输出。

- -v, --verbose

  指定verbose模式。该选项将导致gs_dumpall向转储文件输出详细的对象注解和启动/停止次数，向标准错误流输出处理信息。

- -V, --version

  打印gs_dumpall版本，然后退出。

- --lock-wait-timeout=TIMEOUT

  请勿在转储刚开始时一直等待以获取共享表锁。如果无法在指定时间内锁定某个表，就选择失败。可以以任何符合SET statement_timeout的格式指定超时时间。

- -?, --help

  显示gs_dumpall命令行参数帮助，然后退出。

转储参数：

  

- -a, --data-only

  只转储数据，不转储模式（数据定义）。

- -c, --clean

  在重新创建数据库之前，执行SQL语句清理（删除）这些数据库。针对角色和表空间的转储命令已添加。

- -g, --globals-only

  只转储全局对象（角色和表空间），无数据库。

- -o, --oids

  转储每个表的对象标识符（OIDs），作为表的一部分数据。该选项用于应用以某种方式（例如：外键约束方式）参照了OID列的情况。如果不是以上这种情况，请勿使用该选项。

- -O, --no-owner

  不输出设置对象的归属这样的命令，以匹配原始数据库。默认情况下，gs_dumpall会发出ALTER OWNER或SET SESSION AUTHORIZATION语句设置所创建的模式元素的所属。如果脚本正在运行，该语句不会执行成功，除非是由系统管理员触发（或是拥有脚本中所有对象的同一个用户）。通过指定-O，编写一个任何用户都能存储的脚本，且该脚本会授予该用户拥有所有对象的权限，因为没有使用ALTER OWNER或SET SESSION AUTHORIZATION语句，导入时一直使用执行用户权限，所以在导入前请确认转储文件内是否存在风险。例如，检查转储文件中是否包含提权语句，且该语句管理员是否知晓。

- -r, --roles-only

  只转储角色，不转储数据库或表空间。

- -s, --schema-only

  只转储对象定义（模式），而非数据。

- -S, --sysadmin=NAME

  该参数为扩展预留接口，不建议使用。

- -t, --tablespaces-only

  只转储表空间，不转储数据库或角色。

- -x, --no-privileges

  防止转储访问权限（授权/撤销命令）。

- --column-inserts/--attribute-inserts

  以INSERT命令带列名（INSERT INTO表（列、…）值…）方式导出数据。这会导致恢复缓慢。但是由于该选项会针对每行生成一个独立分开的命令，所以在重新加载某行时出现的错误只会导致失败语句对应的数据丢失，而非整个表内容。

  **说明**

  M-Compatibility模式数据库不支持该选项，导出M-Compatibility模式数据库时会跳过该选项。

- --disable-triggers

  该参数为扩展预留接口，不建议使用。

- --inserts

  发出INSERT命令（而非COPY命令）时转储数据。这会导致恢复缓慢。注意如果重排列顺序，可能会导致恢复整个失败。--column-inserts选项更加安全，虽然可能更慢些。

  

**说明**

M-Compatibility模式数据库不支持该选项，导出M-Compatibility模式数据库时会跳过该选项。

- --no-security-labels

  该参数为扩展预留接口，不建议使用。

- --no-tablespaces

  请勿输出创建表空间的命令，也请勿针对对象选择表空间。使用该选项，无论默认表空间是哪一个，在恢复过程中所有对象都会被创建。

- --no-unlogged-table-data

  该参数为扩展预留接口，不建议使用。

- --include-alter-table

  导出表中已删除的列信息。

- --quote-all-identifiers

  强制对所有标识符加引号。为了向后续版本迁移，且其中可能涉及引入额外关键词，在转储相应数据库时该选项会有帮助。

  **说明**

  M-Compatibility模式数据库不支持该选项，导出M-Compatibility模式数据库时会跳过该选项。

- --dont-overwrite-file

  不重写当前文件。

- --use-set-session-authorization

  输出符合SQL标准的SET SESSION AUTHORIZATION命令而不是ALTER OWNER命令来确定对象所有权。这样令转储更加符合标准，但是如果转储文件中的对象的历史有些问题，那么可能不能正确恢复。并且，使用SET SESSION AUTHORIZATION的转储需要数据库系统管理员的权限才能转储成功，而ALTER OWNER需要的权限则低得多。但是SET SESSION AUTHORIZATION部分支持使用密文密码随意切换用户及权限，因为使用此参数导出的脚本可能无法正常恢复，不建议使用此参数导出。

  **说明**

  SET SESSION AUTHORIZATION使用范围：

  - 系统管理员可以通过SET SESSION AUTHORIZATION语句切换到普通用户，无法切换到初始用户, 其他sysadmin、opradmin、monadmin、poladmin和auditadmin。
  - 其他用户无法通过SET SESSION AUTHORIZATION语句切换用户。

- --with-encryption=AES128

  指定转储数据需用AES128进行加密。

- --with-key=KEY

  AES128密钥规则如下：

  - 密钥长度为8~16个字符。
  - 至少包含大写字母（A-Z），小写字母（a-z），数字（0-9），非字母数字字符（限定为~!@#$%^&*()-_=+\|[{}];:,<.>/?）四类字符中的三类字符。

- --include-extensions

  如果include-extensions参数被设置，将备份所有的CREATE EXTENSION语句。

  

**须知**

扩展功能为内部使用功能，不建议用户使用。

- --include-templatedb
  转储过程中包含模板库。
- --binary-upgrade
  该参数为扩展预留接口，不建议使用。

  **说明**

  M-Compatibility数据库不支持该选项，导出M-Compatibility数据库时会跳过该选项。
- --binary-upgrade-usermap="USER1=USER2"
- 该参数为扩展预留接口，不建议使用。
- --non-lock-table
  该参数仅供软件间接口调用。
- --tablespaces-postfix
  该参数为扩展预留接口，不建议使用。
- --parallel-jobs
  指定备份进程并发数，取值范围为1~1000。
- --pipeline
  使用管道传输密码，禁止在终端使用。

**说明**

- -g/--globals-only和-r/--roles-only不能同时使用。
- -g/--globals-only和-t/--tablespaces-only不能同时使用。
- -r/--roles-only和-t/--tablespaces-only不能同时使用。
- -s/--schema-only和-a/--data-only不能同时使用。
- -r/--roles-only和-a/--data-only不能同时使用。
- -t/--tablespaces-only和-a/--data-only不能同时使用。
- -g/--globals-only和-a/--data-only不能同时使用。
- --tablespaces-postfix和--binary-upgrade必须一起使用。
- --binary-upgrade-usermap和--binary-upgrade必须一起使用。
- --parallel-jobs和-f/--file必须一起使用。

连接参数：

- -h, --host=HOSTNAME
  指定主机的名称。如果取值是以斜线开头，它将用作Unix域套接字的目录。默认值取自PGHOST环境变量；如果没有设置，将启动某个Unix域套接字建立连接。
  该参数只针对数据库外，对数据库内本机IPv4用127.0.0.1，IPv6用::1。
  环境变量：PGHOST
- -l, --database=DATABASENAME
  指定所连接的转储全局对象的数据库名称，并去寻找还有其他哪些数据库需要被转储。如果没有指定，会使用postgres数据库，如果postgres数据库不存在，会使用template1。

  

**说明**

无论指定哪一个数据库作为导出时连接的数据库，在新实例环境上使用gsql进行导入时，必须连接postgres，否则可能出现语法不兼容的情况，导致导入出错。特别地，如果错误地连接M-compatibility数据库进行导入时，由于M-compatibility数据库下CREATE DATABASE等价与CREATE SCHEMA会导致创建失败。

- -p, --port=PORT

  指定服务器所侦听的TCP端口或本地Unix域套接字后缀，以确保连接。默认值设置为PGPORT环境变量。

  在开启线程池情况下，建议使用 pooler port，即侦听端口+1。

  环境变量：PGPORT

- -U, --username=NAME

  所连接的用户名，跨节点执行不支持使用初始用户。

  环境变量：PGUSER

- -w, --no-password

  不出现输入密码提示。如果服务器要求密码认证并且密码没有通过其它形式给出，则连接尝试将会失败。 该选项在批量工作和不存在用户输入密码的脚本中很有帮助。

- -W, --password=PASSWORD

  指定用户连接的密码。如果主机的认证策略是trust，则不会对系统管理员进行密码验证，即无需输入-W选项；如果不加此参数，并且不是系统管理员，则会提示交互式输入，为了系统安全，推荐使用交互式输入密码方式。

- --role=ROLENAME

  指定创建转储使用的角色名。选择该选项，会使gs_dumpall连接数据库后，发起一个SET ROLE角色名命令。当所授权用户（由-U指定）没有gs_dumpall要求的权限时，该选项会起到作用，即切换到具备相应权限的角色。某些安装操作规定不允许直接以系统管理员身份登录，而使用该选项能够在不违反该规定的情况下完成转储。

- --rolepassword=ROLEPASSWORD

  指定具体角色用户的角色密码。

## 说明

- 由于gs_dumpall内部调用gs_dump，所以一些诊断信息请参见**gs_dump**。
- 一旦恢复，建议在每个数据库上运行ANALYZE，优化程序提供有用的统计数据。
- gs_dumpall恢复前需要所有必要的表空间目录为空；否则，对于处在非默认位置的数据库，数据库创建会失败。

## 示例

使用gs_dumpall一次导出数据库的所有数据库。

**说明**

gs_dumpall仅支持纯文本格式导出。所以只能使用gsql恢复gs_dumpall导出的转储内容。

```
gs_dumpall -U omm -f backup/bkp2.sql -p 37300
gs_dump[user='omm'][localhost][port='37300'][dbname='testdb'][2018-06-27 09:55:09]: The total objects number is 2371.
gs_dump[user='omm'][localhost][port='37300'][dbname='testdb'][2018-06-27 09:55:35]: [100.00%] 2371
```

  

```
objects have been dumped.
gs_dump[user='omm'][localhost][port='37300'][dbname='testdb'][2018-06-27 09:55:46]: dump database
dbname='testdb' successfully
gs_dump[user='omm'][localhost][port='37300'][dbname='testdb'][2018-06-27 09:55:46]: total time: 55567
ms
gs_dumpall[user='omm'][localhost][port='37300'][2018-06-27 09:55:46]: dumpall operation successful
gs_dumpall[user='omm'][localhost][port='37300'][2018-06-27 09:55:46]: total time: 56088  ms
```

其中：backup/bkp2.sql表示导出的文件，37300表示数据库服务器端口，omm为用户名。

## 相关命令

**gs_dump**

  

# 5 gs_restore

## 背景信息

gs_restore是GaussDB提供的针对gs_dump导出数据的导入工具。通过此工具可将gs_dump导出生成的文件进行导入。

主要功能包含：

- 导入到数据库

  如果连接参数中指定了数据库，则数据将被导入到指定的数据库中。其中，并行导入必须指定连接的密码。导入时生成列会自动更新，并像普通列一样保存。

  如果连接参数中指定了PDB，则数据将被导入到指定的PDB中。

  **说明**

  - 如果导入至指定的PDB，则不支持使用-C, --create选项。
  - 如果导入指定的PDB处于close状态时，导入操作会执行失败。
  - 如果导入指定的PDB设置事务为只读事务时，导入操作会执行失败。
  - M-Compatibility模式数据库下，禁止在lower_case_table_names参数不同的实例之间进行导入导出，否则可能引起数据丢失。

- 导入到归档文件

  如果参数指定"-l"，则生成归档文件，其中包含简略的数据总结。

gs_restore支持SSL加密通信，使用方式同gsql方式。

使用gs_restore前请确保gs_restore版本与gs_dump版本、数据库版本保持一致。

## 命令格式

```
gs_restore [OPTION]... FILE
```

  

**说明**

- FILE没有短选项或长选项。用来指定归档文件所处的位置。
- 作为前提条件，需输入dbname或-l选项。不允许用户同时输入dbname和-l选项。
- gs_restore默认是以追加的方式进行数据导入。为避免多次导入造成数据异常，在进行导入时，建议选择使用“-c”和“-e”参数。“-c”表示在重新创建数据库对象前，清理（删除）已存在于将要还原的数据库中的数据库对象；“-e”表示当发送SQL语句到数据库时如果出现错误请退出，默认状态下会继续，且在导入后会显示一系列错误信息。
- 在进行导入时，如果schema对象的owner拥有OPRADMIN的系统权限，那么在导入时需要使用初始用户。
- 在进行导入时，如果数据中存在非法编码的数据并且不需要进行转码，可以配置数据库兼容性参数copy_special_character_version为'no_error'将非法编码的数据进行导入，否则会报错，但不会中断导入。

## 参数说明

通用参数：

- -d, --dbname=NAME
  连接数据库dbname并直接导入到该数据库中。
- -f, --file=<FILE_NAME>
  指定生成归档的输出文件，使用-l时列表的输出文件。
  默认是标准输出。

  **说明**

  -f不能同-d一起使用。
- -F, --format=c|d|t
  指定归档格式。由于gs_restore会自动决定格式，因此不需要指定格式。
  取值范围：
  - c/custom：该归档形式为gs_dump的自定义格式。
  - d/directory：该归档形式是一个目录归档形式。
  - t/tar：该归档形式是一个tar归档形式。
- -l, --list
  列出归档形式内容。这一操作的输出可用作-L选项的输入。注意如果像-n或-t的过滤选项与-l使用，过滤选项将会限制列举的项目（即归档形式内容）。
- -v, --verbose
  指定verbose模式。
- -V, --version
  打印gs_restore版本，然后退出。
- -?, --help
  显示gs_restore命令行参数帮助，然后退出。

导入参数：

- -a, -data-only
  只导入数据，不导入模式（数据定义）。gs_restore的导入是以追加方式进行的。
- -c, --clean

  

在重新创建数据库对象前，清理（删除）已存在于将要还原的数据库中的数据库对象。如果目标数据库中没有删除操作涉及的对象，可能会输出一些提示性的错误信息。

- -C, --create

  导入数据库之前会先使用CREATE DATABASE创建数据库（指定该选项后，-d指定的数据库仅用以执行CREATE DATABASE命令，所有数据依然会导入到创建的数据库中）。

  **说明**

  - 在多租场景下，使用gs_restore导入数据至PDB时，不支持使用该选项。
  - 在M-Compatibility模式数据库下，不支持使用该选项。必须先在目标实例上创出M-Compatibility模式数据库，然后在源实例上导出，最后在目标实例连接新创出的M-Compatibility模式数据库导入。

- -e, --exit-on-error

  当发送SQL语句到数据库时如果出现错误，请退出。默认状态下会继续，且在导入后会显示一系列错误信息。

- -I, --index=NAME

  只导入已列举的index的定义。允许导入多个index。如果多次输入-I index导入多个index。

  例如：

  ```
  gs_restore -h host_name -p port_number -d testdb -I Index1 -I Index2 backup/MPPDB_backup.tar
  ```

  在上面这个例子中，Index1和Index2会被导入。

- -j, --jobs=NUM

  运行gs_restore最耗时的部分（如加载数据、创建index、或创建约束）使用并发任务。该选项能大幅缩短导入时间，即将一个大型数据库导入到某一多处理器的服务器上。

  每个任务可能是一个进程或一个线程，这由操作系统决定。每个任务与服务器进行单独连接。

  该选项的最优值取决于服务器的硬件设置、客户端、以及网络。还包括这些因素，如CPU核数量、硬盘设置。建议是从增加服务器上的CPU核数量入手，更大的值（服务器上CPU核数量）在很多情况下也能导致数据文件更快地被导入。相应的，过高的值会由于超负荷反而导致性能降低。

  该选项只支持自定义归档格式。输入文件必须是常规文件（不能是像pipe的文件）。如果是通过脚本文件，而非直接连接数据库服务器，该选项可忽略。而且，多任务不能与--single-transaction选项一起使用。

  **说明**

  - 此参数适用于多表/多索引/多约束的情况。实际使用过程中，创建的进程数（或线程数）与表、索引、约束等的数量有关，最高并发不会超过给定的jobs数。
  - 当gs_restore导入目标数据库是PDB时，该选项的最优值取决于分配给该PDB的资源规格。

- -L, --use-list=<FILE_NAME>

  只导入列举在list-file中的那些归档形式元素，导入顺序以它们在文件中的顺序为准。注意如果像-n或-t的过滤选项与-L使用，它们将会进一步限制导入的项目。

  一般情况下，list-file是通过编辑前面提到的某个-l参数的输出创建的。文件行的位置可更改或直接删除行，也可使用分号（;）在行的开始注出。见下文的举例。

- -n, --schema=NAME

  

只导入已列举的模式中的对象。

该选项可与-t选项一起用以导入某个指定的表。

多次输入-n *schemaname*可以导入多个模式。

例如：

```
gs_restore -h host_name -p port_number -d testdb -n sch1 -n sch2 backup/MPPDB_backup.tar
```

在上面这个例子中，sch1和sch2会被导入。

**说明**

- M-Compatibility兼容模式下，通过CREATE DATABASE带templatem创建的数据库通过-d指定db_name导入数据，而通过CREATE DATABASE db_name创建出来的database等价于Schema，只能通过-n导入数据。
- 指定的模式名必须存在于作为输入的归档文件中，如果指定了归档文件中不存在的模式名，该模式的导入不生效。
- 与gs_dump不同，在gs_restore中该参数为全匹配，当模式名称中包含大写字母时，不需要添加额外的引号，如：-n "Sch1"
- M-Compatibility兼容模式下，该参数值受GUC参数lower_case_table_names影响。比如在大小写敏感模式下（lower_case_table_names=0），该参数值也需要大小写敏感，如果包含大写字母，不需要添加额外的引号。在大小写不敏感模式下（lower_case_table_names=1），该参数值需要传小写名称。

- -O, --no-owner

  不输出设置对象的归属这样的命令，以匹配原始数据库。默认情况下，gs_restore会发出ALTER OWNER或SET SESSION AUTHORIZATION语句设置所创建的模式元素的所属。除非是由系统管理员（或是拥有脚本中所有对象的同一个用户）进行数据库首次连接的操作，否则语句会失败。使用-O选项，任何用户名都可用于首次连接，且该用户拥有所有已创建的对象。

- -P, --function=NAME(args)

  只导入已列举的函数。请按照函数所在转储文件中的目录，准确拼写函数名称和参数。

  当-P单独使用时，表示导入文件中所有'function-name(args)'函数；当-P同-n一起使用时，表示导入指定模式下的'function-name(args)'函数；多次输入-P，而仅指定一次-n，表示所有导入的函数默认都是位于-n模式下的。

  可以多次输入-n schema-name -P 'function-name(args)'同时导入多个指定模式下的函数。

  例如：

  ```
  ./gs_restore -h host_name -p port_number -d testdb -n test1 -P 'Func1(integer)' -n test2 -P 'Func2(integer)' backup/MPPDB_backup.tar
  ```

  在上面这个例子中，test1模式下的函数Func1(i integer)和test2模式下的函数Func2(j integer)会被一起导入。

- -s, --schema-only

  只导入模式（数据定义），不导入数据（表内容）。当前的序列值也不会导入。

- -S, --sysadmin=NAME

  该参数为扩展预留接口，不建议使用。

- -t, --table=NAME

  只导入已列举的表定义、数据或定义和数据。该选项与-n选项同时使用时，用来指定某个模式下的表对象。-n参数不输入时，默认为PUBLIC模式。多次输入-n <schemaname> -t <tablename>可以导入指定模式下的多个表。

  例如：

  

导入PUBLIC模式下的table1

```
gs_restore -h host_name -p port_number -d testdb -t table1 backup/MPPDB_backup.tar
```

导入test1模式下的test1和test2模式下test2

```
gs_restore -h host_name -p port_number -d testdb -n test1 -t test1 -n test2 -t test2 backup/MPPDB_backup.tar
```

导入PUBLIC模式下的table1和test1 模式下test1

```
gs_restore -h host_name -p port_number -d testdb -n PUBLIC -t table1 -n test1 -t table1 backup/MPPDB_backup.tar
```

**须知**

- -t不支持schema_name.table_name的输入格式，指定此格式不会报错，但不会生效。
- 当-t已指定时，gs_restore不会导入已选表所附着的任何其他数据库对象。因此，无法保证某个指定表的转储结果能够自行成功地导入到一个空数据库中。
- -t tablename不会导入表上的触发器信息。
- 与gs_dump不同，在gs_restore中该参数为全匹配，当表名称中包含大写字母时，不需要添加额外的引号，如：-t "Tbl1"
- M-Compatibility兼容模式下，该参数值受GUC参数lower_case_table_names影响。比如在大小写敏感模式下（lower_case_table_names=0），该参数值也需要大小写敏感，如果包含大写字母，不需要添加额外的引号。在大小写不敏感模式下（lower_case_table_names=1），该参数值需要传小写名称。

- -T, --trigger=NAME

  该参数为扩展预留接口。

- -x, --no-privileges/--no-acl

  防止导入访问权限（grant/revoke命令）。

- -1, --single-transaction

  执行导入作为一个单独事务（即把命令包围在BEGIN/COMMIT中）。

  该选项确保要么所有命令成功完成，要么没有改变应用。该选项意为--exit-on-error。

- --disable-triggers

  该参数为扩展预留接口，不建议使用。

- --no-data-for-failed-tables

  默认状态下，即使创建表的命令失败（如表已经存在），表数据仍会被导入。使用该选项，像这种表的数据会被跳过。如果目标数据库已包含想要的表内容，这种行为会有帮助。

  该选项只有在直接导入到某数据库中时有效，不针对生成SQL脚本文件输出。

- --no-security-labels

  该参数为扩展预留接口，不建议使用。

- --no-tablespaces

  不输出选择表空间的命令。使用该选项，无论默认表空间是哪一个，在导入过程中所有对象都会被创建。

- --section=SECTION

  

导入已列举的区段（如pre-data、data、或post-data）。

- --use-set-session-authorization

  该选项用来进行文本格式的备份。

  输出SET SESSION AUTHORIZATION命令，而非ALTER OWNER命令，用以决定对象归属。该选项使转储更加兼容标准，但通过参考转储中对象的记录，导入过程可能会有问题。使用SET SESSION AUTHORIZATION的转储要求必须是系统管理员，同时在导入前还需参考"SET SESSION AUTHORIZATION"，手工对导出文件的密码进行修改验证，只有这样才能进行正确的导入操作，相比之下，ALTER OWNER对权限要求较低。

- --pipeline

  使用管道传输密码，禁止在终端使用。

**须知**

- 如果安装过程中有任何本地数据要添加到template1数据库，请谨慎将gs_restore的输出载入到一个真正的空数据库中；否则可能会因为被添加对象的定义被复制，而出现错误。要创建一个无本地添加的空数据库，需从template0而非template1复制，例如：

```
CREATE DATABASE foo WITH TEMPLATE template0;
```

- gs_restore不能选择性地导入大对象；例如只能导入那些指定表的对象。如果某个归档形式包含大对象，那所有大对象都会被导入。如果此归档对象通过-L、-t或其他选项被排除，那么所有大对象一个都不会被导入。

**说明**

1.-d/--dbname 和 -f/--file 不能同时使用。
2. -s/--schema-only 和 -a/--data-only不能同时使用。
3. -c/--clean 和 -a/--data-only不能同时使用。
4. 使用--single-transaction时，-j/--jobs必须为单任务。
5. --role 和 --rolepassword必须一起使用。

连接参数：

- -h, --host=HOSTNAME

  指定的主机名称。如果取值是以斜线开头，他将用作Unix域套接字的目录。默认值取自PGHOST环境变量；如果没有设置，将启动某个Unix域套接字建立连接。

  该参数只针对数据库外，对数据库内本机IPv4用127.0.0.1，IPv6用::1。

  环境变量：PGHOST

- -p, --port=PORT

  指定服务器所侦听的TCP端口或本地Unix域套接字后缀，以确保连接。默认值设置为PGPORT环境变量。

  在开启线程池情况下，建议使用 pooler port，即侦听端口+1。

  环境变量：PGPORT

- -U, --username=NAME

  所连接的用户名，跨节点执行不支持使用初始用户。

  环境变量：PGUSER

  

- -w, --no-password

  不出现输入密码提示。如果服务器要求密码认证并且密码没有通过其它形式给出，则连接尝试将会失败。 该选项在批量工作和不存在用户输入密码的脚本中很有帮助。

- -W, --password=PASSWORD

  指定用户连接的密码。如果主机的认证策略是trust，则不会对系统管理员进行密码验证，即无需输入-W参数；如果不加此参数，并且不是系统管理员，则会提示交互式输入，为了系统安全，推荐使用交互式输入密码方式。

- --role=ROLENAME

  指定导入操作使用的角色名。选择该参数，会使gs_restore连接数据库后，发起一个SET ROLE角色名命令。当所授权用户（由-U指定）没有gs_restore要求的权限时，该参数会起到作用，即切换到具备相应权限的角色。某些安装操作规定不允许直接以初始用户身份登录，而使用该参数能够在不违反该规定的情况下完成导入。

- --rolepassword=ROLEPASSWORD

  指定具体角色用户的角色密码。

## 示例

特例：执行gsql程序，使用如下选项导入由gs_dump/gs_dumpall生成导出文件夹（纯文本格式）的MPPDB_backup.sql文件到testdb数据库。

```
gsql -d testdb -p 8000 -f /home/omm/test/MPPDB_backup.sql
SET
SET
SET
SET
SET
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
ALTER TABLE
CREATE INDEX
CREATE INDEX
CREATE INDEX
SET
CREATE INDEX
REVOKE
REVOKE
GRANT
GRANT
total time: 30476  ms
```

示例中“-f”后的是导出的文件，“8000”表示数据库服务器端口；“testdb”表示要访问的数据库名。

gs_restore用来导入由gs_dump生成的导出文件。

示例1：执行gs_restore，将导出的MPPDB_backup.dmp文件（自定义归档格式）导入到testdb数据库。

```
gs_restore backup/MPPDB_backup.dmp -p 8000 -d testdb
restore operation successful
total time: 13053  ms
```

示例2：执行gs_restore，将导出的MPPDB_backup.tar文件（tar格式）导入到testdb数据库。

```
gs_restore backup/MPPDB_backup.tar -p 8000 -d testdb
restore operation successful
total time: 21203  ms
```

  

示例3：执行gs_restore，将导出的MPPDB_backup文件（目录格式）导入到testdb数据库。

```
gs_restore backup/MPPDB_backup -p 8000 -d testdb
restore operation successful
total time: 21003  ms
```

示例4：执行gs_restore，使用自定义归档格式的MPPDB_backup.dmp文件来进行如下导入操作。 导入PUBLIC模式下所有对象的定义和数据。在导入时会先删除已经存在的对象，如果原对象存在跨模式的依赖则需手工强制干预。

```
gs_restore backup/MPPDB_backup.dmp -p 8000 -d testdb -e -c -n PUBLIC
Error while PROCESSING TOC:
Error from TOC entry 313; 1259 337399 TABLE table1 gaussdba
could not execute query: ERROR:  cannot drop table table1 because other objects depend on it
DETAIL:  view t1.v1 depends on table table1
HINT:  Use DROP ... CASCADE to drop the dependent objects too.
    Command was: DROP TABLE IF EXISTS public.table1;
```

手工删除依赖，导入完成后再重新创建。

```
gs_restore backup/MPPDB_backup.dmp -p 8000 -d testdb -e -c -n PUBLIC
restore operation successful
total time: 2203  ms
```

示例5：执行gs_restore，使用自定义归档格式的MPPDB_backup.dmp文件来进行如下导入操作。只导入PUBLIC模式下表table1的定义。

```
gs_restore backup/MPPDB_backup.dmp -p 8000 -d testdb -e -c -s -n PUBLIC -t table1
restore operation successful
total time: 21000  ms
```

示例6：执行gs_restore，使用自定义归档格式的MPPDB_backup.dmp文件来进行如下导入操作。只导入PUBLIC模式下表table1的数据。

```
gs_restore backup/MPPDB_backup.dmp -p 8000 -d testdb -e -a -n PUBLIC -t table1
restore operation successful
total time: 20203  ms
```

示例7：在多租场景下，执行gs_restore，导入指定PDB_backup.dmp文件（自定义归档格式）到名称为testpdb的PDB。

```
gs_restore backup/PDB_backup.dmp -p 20000 -d testpdb
restore operation successful
total time: 371  ms
```

示例8：在多租场景下，执行gs_restore，导入指定PDB_backup.tar文件（tar归档格式）到名称为testpdb的PDB。

```
gs_restore backup/PDB_backup.tar -p 20000 -d testpdb
restore operation successful
total time: 367  ms
```

示例9：在多租场景下，执行gs_restore，导入指定PDB_backup目录文件（目录归档格式）到名称为testpdb的PDB。

```
gs_restore backup/PDB_backup -p 20000 -d testpdb
restore operation successful
total time: 370  ms
```

示例10：在多租场景下，执行gs_retsore，使用-C, --create选项来进行导入操作，gs_restore报错退出。

```
gs_restore backup/PDB_backup -C -p 20000 -d testpdb
gs_restore unsupport the '-C, --create' option for pdb.
```

  

## 相关命令

**gs_dump**，**gs_dumpall**