

云数据库 GaussDB

# 主备版工具参考

文档版本            01  
发布日期            2023-11-23



版权所有 © 华为云计算技术有限公司 2024。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

## 商标声明



HUAWEI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

---

<b>1 客户端工具</b> .....	<b>1</b>
1.1 gsql.....	1
1.1.1 gsql 概述.....	1
1.1.2 使用指导.....	8
1.1.3 获取帮助.....	10
1.1.4 命令参考.....	12
1.1.5 元命令参考.....	16
1.1.6 常见问题处理.....	31
1.2 gs_ktool.....	35
1.3 gs_loader.....	38

# 1 客户端工具

数据库在部署成功后，需要通过一些工具来便捷地连接数据库，对数据库进行各种调试和操作。GaussDB提供了一些数据库连接工具。通过这些工具可以方便地连接数据库并对数据库进行数据操作。

工具包获取请参考《开发指南》的“应用程序开发教程 > 驱动包获取”章节。

## 1.1 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。

### 1.1.1 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《开发者指南》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	<p>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：</p> <pre>\set varname value</pre> <p>删除由\set命令设置的变量请使用如下方式：</p> <pre>\unset varname</pre> <p><b>说明</b></p> <ul style="list-style-type: none"> <li>变量只是简单的名称/值对，这里的值可以是任意长度。</li> <li>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。</li> <li>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。</li> <li>可以使用不带参数的\set来显示所有变量的值。</li> </ul> <p>变量的示例和详细说明请参见<a href="#">变量</a>。</p>
SQL代换	<p>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。</p> <p>SQL代换的示例和详细说明请参见<a href="#">SQL代换</a>。</p>
自定义提示符	<p>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。</p> <p>这三个变量的值可以由用户自定义，也可以使用gsql预定义的值。详情请参见<a href="#">提示符</a>。</p>
客户端操作历史记录	<p>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。</p> <p><b>说明</b></p> <ul style="list-style-type: none"> <li>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。</li> <li>出于安全考虑，将包含PASSWORD、IDENTIFIED、GS_ENCRYPT_AES128、GS_DECRYPT_AES128、GS_ENCRYPT、GS_DECRYPT、PG_CREATE_PHYSICAL_REPLICATION_SLOT_EXTERN、SECRET_ACCESS_KEY、SECRETKEY、CREATE_CREDENTIAL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。</li> </ul>

• 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	<code>\set DBNAME dbname</code>	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	<code>\set ECHO all   queries</code>	<ul style="list-style-type: none"> <li>• 如果设置为all，只显示查询信息。等效于使用gsq连接数据库时指定-a参数。</li> <li>• 如果设置为queries，显示命令行和查询信息。等效于使用gsq连接数据库时指定-e参数。</li> </ul>
ECHO_HIDDEN	<code>\set ECHO_HIDDEN on   off   noexec</code>	<p>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：</p> <ul style="list-style-type: none"> <li>• 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gsq连接数据库时指定-E参数。</li> <li>• 设置为off，则只显示查询结果。</li> <li>• 设置为noexec，则只显示查询信息，不执行查询操作。</li> </ul>
ENCODING	<code>\set ENCODING encoding</code>	当前客户端的字符集编码。
FETCH_COUNT	<code>\set FETCH_COUNT variable</code>	<ul style="list-style-type: none"> <li>• 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。</li> <li>• 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。</li> </ul> <p><b>说明</b> 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。</p>
HISTCONTROL	<code>\set HISTCONTROL ignorespace   ignoredups   ignoreboth   none</code>	<ul style="list-style-type: none"> <li>• ignorespace：以空格开始的行将不会写入历史列表。</li> <li>• ignoredups：与以前历史记录里匹配的行不会写入历史记录。</li> <li>• 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。</li> </ul> <p><b>说明</b> none表示不设置HISTCONTROL。</p>
HISTFILE	<code>\set HISTFILE filename</code>	此文件用于存储历史名列表。缺省值是 <code>~/.bash_history</code> 。

变量	设置方法	变量说明
HISTSIZE	<code>\set HISTSIZE size</code>	保存在历史命令里命令的个数。缺省值是500。
HOST	<code>\set HOST hostname</code>	已连接的数据库主机名称。
IGNOREEOF	<code>\set IGNOREEOF variable</code>	<ul style="list-style-type: none"> <li>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。</li> <li>若设置此变量为非数值，则缺省为10。</li> <li>若删除此变量，则向交互的gsql会话发送一个EOF终止应用。</li> </ul>
LASTOID	<code>\set LASTOID oid</code>	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	<code>\set ON_ERROR_ROLLBACK on   interactive   off</code>	<ul style="list-style-type: none"> <li>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。</li> <li>如果是interactive，这样的错误只是在交互的会话里忽略。</li> <li>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。</li> </ul>
ON_ERROR_STOP	<code>\set ON_ERROR_STOP on   off</code>	<ul style="list-style-type: none"> <li>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。</li> <li>off（缺省）：命令执行错误时将会跳过错误继续执行。</li> </ul>
PORT	<code>\set PORT port</code>	正连接数据库的端口号。
USER	<code>\set USER username</code>	当前用于连接的数据库用户。
VERBOSITY	<code>\set VERBOSITY terse   default   verbose</code>	<p>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。</p> <ul style="list-style-type: none"> <li>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）。</li> <li>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）。</li> <li>verbose：返回所有的错误信息。</li> </ul>

- 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。 PROMPT1的默认值为： %/%R%#	使用变量PROMPT1切换提示符： <ul style="list-style-type: none"> <li>提示符变为[local]： gaussdb=&gt; \set PROMPT1 %M [local:/tmp/gaussdba_mppdb]</li> <li>提示符变为name： gaussdb=&gt; \set PROMPT1 name name</li> <li>提示符变为=： gaussdb=&gt; \set PROMPT1 %R =</li> </ul>
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： gaussdb=# \set PROMPT2 TEST gaussdb=# select * from HR.areaS TEST; area_id   area_name -----+----- <ul style="list-style-type: none"> <li>1   Europe</li> <li>2   Americas</li> <li>4   Middle East and Africa</li> <li>3   Asia</li> </ul> (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： gaussdb=# \set PROMPT3 '>>>>' gaussdb=# copy HR.areaS from STDIN; Enter data to be copied followed by a newline. End with a backslash and a period on a line by itself. >>>>1 aa >>>>2 bb >>>>\.

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	<ul style="list-style-type: none"> <li>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“!”。</li> <li>对于PROMPT2该序列被“-”、“*”、单引号、双引号或“\$”（取决于gsq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。</li> </ul>
%x	事务状态： <ul style="list-style-type: none"> <li>如果不在事务块里，则是一个空字符串。</li> <li>如果在事务块里，则是“*”。</li> <li>如果在一个失败的事务块里则是“!”。</li> <li>如果无法判断事务状态时为“?”（比如没有连接）。</li> </ul>
%digits	指定字节值的字符将被替换到该位置。
%:name	gsq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， gaussdb=> \set PROMPT1 '%[%033[1;33;40m%]n@%/%R%[%033[0m%]#' 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。 <b>说明</b> 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： PSQL_EDITOR_LINENUMBER_ARG = '+' PSQL_EDITOR_LINENUMBER_ARG='--line ' 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。

## 1.1.2 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《开发者指南》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用户连接到本机gaussdb数据库的8000端口。

```
gsql -d gaussdb -p 8000
```

示例2，使用jack用户连接到远程主机gaussdb数据库的8000端口。

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

示例3，参数gaussdb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 gaussdb omm -p 8000
```

#### 等效于

```
gsql -d gaussdb -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。

```
CREATE DATABASE human_staff;  
CREATE DATABASE
```

通常，输入的命令在遇到分号的时候结束。如果输入的命令没有错误，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=#
```

## 1.1.3 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** 使用如下命令连接数据库。

```
gsql -d gaussdb -p 8000
```

gaussdb为需要连接的数据库名称，8000为CN的端口号。

连接成功后，系统显示类似如下信息：

```
gsql ((GaussDB Kernel 503.1.XXX build f521c606) compiled at 2021-09-16 14:55:22 commit 2935 last mr
6385 release)
Non-SSL connection (SSL connection is recommended when requiring high-security)
Type "help" for help.
```

---结束

**步骤1** 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	<p>查看GaussDB支持的SQL语句的帮助</p> <p>例如，查看GaussDB支持的所有SQL语句：</p> <pre>gaussdb=# \h Available help: ABORT ALTER AGGREGATE ... ..</pre> <p>例如，查看CREATE DATABASE命令的参数可使用下面的命令：</p> <pre>gaussdb=# \help CREATE DATABASE Command: CREATE DATABASE Description: create a new database Syntax: CREATE DATABASE database_name [ [ WITH ] {[ OWNER [=] user_name ]} [ TEMPLATE [=] template ]] [ ENCODING [=] encoding ]] [ LC_COLLATE [=] lc_collate ]] [ LC_CTYPE [=] lc_ctype ]] [ DBCOMPATIBILITY [=] compatibility_type ]] [ TABLESPACE [=] tablespace_name ]] [ CONNECTION LIMIT [=] connlimit ]}{... };</pre>
查看gsql命令的帮助	<p>例如，查看gsql支持的命令：</p> <pre>gaussdb=# \? General \copyright          show GaussDB Kernel usage and distribution terms \g [FILE] or ;     execute query (and send results to file or  pipe) \h(\help) [NAME]  help on syntax of SQL commands, * for all commands \q                quit gsql ... ..</pre>

---结束

## 1.1.4 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。 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。 <b>注意</b> 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。 变量的示例和详细说明请参见 <a href="#">变量</a> 。	-
-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）。 <b>说明</b> 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。 <b>说明</b> 如果脚本中已经使用了START TRANSACTION, COMMIT, ROLLBACK，则该选项无效。	-
-, --help	显示关于gsql命令行参数的帮助信息然后退出。	-

参数	参数说明	取值范围
-V, --version	打印gsqll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。 <b>注意</b> 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。 <b>注意</b> 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gsqll对导入的加密文件进行解密。 <b>须知</b> <ul style="list-style-type: none"> <li>对于本身就是shell命令中的关键字如单引号（'）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gsqll程序。</li> <li>不支持解密导入存储过程和函数。</li> </ul>	-
-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。 <b>注意</b> <ul style="list-style-type: none"> <li>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。</li> <li>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gsqll运行过程中的相关日志信息。</li> </ul>	绝对路径或相对路径，且满足操作系统路径命名规则。
-m, --maintenance	允许在两阶段事务恢复期间连接数据库。 <b>说明</b> 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，gsqll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。	-



参数	参数说明	取值范围
-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gsqll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。 <b>注意</b> 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。	-

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。	-
-P, --pset=VAR[=ARG]	在命令行上以\pset的风格设置打印选项。 <b>说明</b> 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编辑的模式。	缺省为关闭。
-t, --tuples-only	只打印行。	-

参数	参数说明	取值范围
-T, --table-attr=TEXT	允许声明放在HTML table标签里的选项。 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。 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。 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。	-
-g,	打印来自文件的所有SQL。	-

表 1-10 连接参数

参数	参数说明	取值范围
-h, --host=HOSTNAME	<p>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。</p> <p>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：</p> <p><b>PGTARGETSESSIONATTRS环境变量的值--选择连接的节点类型</b></p> <p>read-write--可读写的节点 read-only--只读节点 primary或者不设定--主节点 standby--备节点 prefer-standby--首选备节点，没有备节点则转为any any--不进行角色检查</p> <p><b>说明</b> 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。</p>	<p>如果省略主机名，gsqll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>

参数	参数说明	取值范围
-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。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=USER NAME	指定连接数据库的用户。 <b>说明</b> <ul style="list-style-type: none"><li>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。</li><li>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。</li></ul>	字符串，默认使用与当前操作系统用户同名的用户。
-W, --password=PASS WORD	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。 <b>说明</b> <ul style="list-style-type: none"><li>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。</li><li>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。</li><li>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。</li></ul>	字符串。

## 1.1.5 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。

## 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](#)。

### 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。

参数	参数说明	取值范围
\parallel [on [num]]off]	<p>控制并发执行开关。</p> <ul style="list-style-type: none"> <li>on: 打开控制并发执行开关, 且最大并发数为num。</li> <li>off: 关闭控制并发执行开关。</li> </ul> <p><b>说明</b></p> <ul style="list-style-type: none"> <li>不支持事务中开启并发执行以及并发中开启事务。</li> <li>不支持\d这类元命令的并发。</li> <li>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。</li> <li>不推荐在并发中使用set语句, 否则导致结果与预期不一致。</li> <li>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。</li> <li>\parallel执行时最多会启动num个独立的gsq进程连接服务器。</li> <li>\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。</li> <li>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。</li> </ul>	<p>num的默认值: 1024。</p> <p><b>须知</b></p> <ul style="list-style-type: none"> <li>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。</li> <li>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。</li> </ul>
\q	退出gsq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
<code>\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]</code>	<p>在任何gsq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。</p> <p><b>说明</b></p> <ul style="list-style-type: none"> <li>• \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。</li> <li>• \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]。</li> <li>• \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。</li> <li>• \COPY的text格式和csv格式均支持header功能</li> <li>• 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。</li> <li>• 其中LOAD_DISCARD功能为gs_loader解析后discard文件路径，非主动调用功能。</li> </ul>
<code>\echo [STRING]</code>	把字符串写到标准输出。
<code>\i FILE</code>	从文件FILE中读取内容，并将其当作输入，执行查询。
<code>\i+ FILE KEY</code>	执行加密文件中的命令。
<code>\ir FILE</code>	和\i类似，只是相对于存放当前脚本的路径。
<code>\ir+ FILE KEY</code>	和\i+类似，只是相对于存放当前脚本的路径。
<code>\o [FILE]</code>	把所有的查询结果发送到文件里。
<code>\qecho [STRING]</code>	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。 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。 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。 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。 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。 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。 gaussdb=# \dc *
\dC[+] [PATTERN]	列出所有类型转换。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。  PATTERN需要使用实际类型名，不能使用别名。	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。 gaussdb=# \dC c*

参数	参数说明	取值范围	示例
\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。 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。 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。 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。 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。 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。 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。 gaussdb=# \dew
\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。 gaussdb=# \df



参数	参数说明	取值范围	示例
\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。 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。 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。 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。 gaussdb=# \dFt
\dg[+] [PATTERN]	列出所有数据库角色。 <b>说明</b> 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）。 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。 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。 gaussdb=# \dL
\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。 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。 gaussdb=# \dn+ d*

参数	参数说明	取值范围	示例
\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。 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。 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。  \dp显示结果如下： rolename=xxxx/yyyy --赋予一个角色的权限 =xxxx/yyyy --赋予public的权限  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。 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。 gaussdb=# \drds * dbname
\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。 gaussdb=# \dT
\du[+] [PATTERN]	列出所有数据库角色。 <b>说明</b> 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。 gaussdb=# \du

参数	参数说明	取值范围	示例
\dE[S+] [PATTERN] \di[S+] [PATTERN] \ds[S+] [PATTERN] \dt[S+] [PATTERN] \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。 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。 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。 gaussdb=# \l
\sf[+] FUNCNAME	显示函数的定义。 <b>说明</b> 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。	-	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。 gaussdb=# \sf function_a gaussdb=# \sf "func()name"(argtype1, argtype2)
\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。 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。

参数	参数说明
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: <ul style="list-style-type: none"> <li>对于过程语言, 允许用户在创建函数时, 指定过程语言。</li> <li>对于模式, 允许访问包含在指定模式中的对象。</li> <li>对于序列, 允许使用nextval函数。</li> </ul>
C	CREATE: <ul style="list-style-type: none"> <li>对于数据库, 允许在该数据库里创建新的模式。</li> <li>对于模式, 允许在该模式中创建新的对象。</li> <li>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。</li> </ul>
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。
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。

参数	参数说明
\f [STRING]	对于不对齐的查询输出，显示或者设置域分隔符。
\H	<ul style="list-style-type: none"> <li>若当前模式为文本格式，则切换为HTML输出格式。</li> <li>若当前模式为HTML格式，则切换回文本格式。</li> </ul>
\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。	<ul style="list-style-type: none"> <li>在HTML格式下，取值范围为大于0的整数。</li> <li>在其他格式下，取值范围： <ul style="list-style-type: none"> <li>- 0: 无边框</li> <li>- 1: 内部分隔线</li> <li>- 2: 台架</li> </ul> </li> </ul>
expanded (或x)	在正常和扩展格式之间切换。	<ul style="list-style-type: none"> <li>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。</li> <li>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。</li> </ul>
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_z ero	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。	-

选项	选项说明	取值范围
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： <ul style="list-style-type: none"> <li>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。</li> <li>aligned：此格式是标准的，可读性好的文本输出。</li> <li>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。</li> <li>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。</li> <li>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。</li> <li>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。</li> </ul>
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。	<ul style="list-style-type: none"> <li>on：显示指定的分隔符。</li> <li>off：不显示分隔符。</li> </ul> 忽略此参数，显示默认的分隔符。
pager	控制查询和gsqhelp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。	<ul style="list-style-type: none"> <li>on：当输出到终端且不适合屏幕显示时，使用分页器。</li> <li>off：不使用分页器。</li> <li>always：当输出到终端无论是否符合屏幕显示时，都使用分页器。</li> </ul>
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。	-

选项	选项说明	取值范围
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 -]	<p>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gsq的命令行提示符中显示的数据库对象名仍为截断前的名称。</p> <p><b>说明</b> 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，受限与GUC参数password_max_length的最大值。</p>	-
\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）。	<ul style="list-style-type: none"> <li>on表示打开显示。</li> <li>off表示关闭显示。</li> </ul>
\! [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]]	<p>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。</p> <p>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。</p>
\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，那个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
\set AUTOCOMMIT [on off]	<p>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。</p> <p><b>说明</b> 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。</p>	<ul style="list-style-type: none"> <li>on表示打开自动提交。</li> <li>off表示关闭自动提交。</li> </ul>



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大对象和提供他们的注释。

## PATTERN

很多\dt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，\dt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，gsq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。

## 1.1.6 常见问题处理

### 连接性能问题

- 开启了log\_hostname，但是配置了错误的DNS导致的连接性能问题。  
在连接上数据库，通过“show log\_hostname”语句，检查数据库中是否开启了log\_hostname参数。  
如果开启了相关参数，那么数据库内核会通过DNS反查客户端所在机器的主机名。这时如果数据库配置了不正确的/不可达的DNS服务器，那么会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 gsq -U MyUserName -d gaussdb -h 127.0.0.1 -p 23508 -r -c '\q'  
Password for MyUserName:
```

此时便会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}})
```

此时便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()”语句来确定初始化语句执行较慢的原因。更多信息，详见《开发者指南》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，表现为假死。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)
```

那么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。

## 创建连接故障

- **gsq: 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。
- **gsq: 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。
- **gsq: 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pg\_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pg\_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指向的目录便可。  
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gsq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。

### 📖 说明

请参照下面示例，修改LD\_LIBRARY\_PATH环境变量。其中\${path\_to\_correct\_libpq\_dir}表示实际环境中正确libpq.so所在目录：

```
export LD_LIBRARY_PATH=${path_to_correct_libpq_dir}:$LD_LIBRARY_PATH
```

- **gsq: symbol lookup error: xxx/gsq: undefined symbol: libpqVersionString**  
此问题是由于环境中使用的libpq.so的版本与gsq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。
- **gsq: 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1-23](#)。

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 1-23 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。	<p>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。</p> <pre>SELECT ROLNAME,ROLCONNLIMIT FROM PG_ROLES WHERE ROLNAME='user1'; rolname   rolconnlimit -----+----- user1              -1 (1 row)</pre>
查看指定用户已使用的会话连接数。	<p>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。</p> <pre>SELECT COUNT(*) FROM dv_sessions WHERE USERNAME='user1';  count ----- 1 (1 row)</pre>
查看指定数据库的会话连接数上限。	<p>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。</p> <pre>SELECT DATNAME,DATCONNLIMIT FROM PG_DATABASE WHERE DATNAME='gaussdb';  datname   datconnlimit -----+----- gaussdb            -1 (1 row)</pre>
查看指定数据库已使用的会话连接数。	<p>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。</p> <pre>SELECT COUNT(*) FROM PG_STAT_ACTIVITY WHERE DATNAME='gaussdb'; count ----- 1 (1 row)</pre>
查看所有用户已使用会话连接数。	<p>执行如下命令查看所有用户已使用的会话连接数。</p> <pre>SELECT COUNT(*) FROM dv_sessions;  count ----- 10 (1 row)</pre>

- `mysql: wait xxx.xxx.xxx.xxx:xxxx timeout expired`

mysql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mysql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。
- `mysql: 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，修改回来便可。

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

目标DN的`pg_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`pg_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杀死，产生core dump文件。

解决此问题，重试便可。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。

## 1.2 gs\_ktool

当前特性是实验室特性，使用时请联系华为技术工程师提供技术支持。

### 前提条件

`gs_ktool`工具前向兼容数据库，若需使用与数据库同步增加的新特性，须升级数据库。

### 背景

在使用`gs_ktool`前，需先了解全密态数据库等值查询特性，该特性使用三级加密模型，旨在提供客户端加密的解决方案：

- 使用客户端加密密钥（Client Encryption Key, CEK）加密数据；
- 使用客户端加密主密钥（Client Master Key, CMK）加密CEK；
- 使用根密钥（Root Key, RK）加密CMK。

### 功能介绍

`gs_ktool`为客户端提供CMK明文，以保证其能够对CEK的进行加解密；与此同时，`gs_ktool`还为用户提供人机交互接口，以支持以下功能：

- 支持管理CMK：创建、删除、查询、导入和导出CMK。其中，查询功能不支持查询CMK的明文或者密文。目前，gs\_ktool不支持更新CMK，用户仅能通过删除旧的密钥，并创建新的密钥以达到同等的效果。

**警告**

- 全密态当前仅支持华为云KMS，gs\_ktool不具备自动备份密钥能力，一旦密钥丢失，则数据无法恢复，建议gs\_ktool仅作为验证与调试使用。
- 如果用户需要将gs\_ktool中密钥进行备份，请手工执行密钥导出命令。并注意，在进行密钥导入时，会覆盖当前gs\_ktool中保存的所有密钥，请谨慎使用。

- 支持管理RK：查询、更新RK。
- 支持查询操作日志：针对上述对密钥进行管理的操作，gs\_ktool均会记录相关的操作日志，用户可查看日志文件。

## 环境变量

GS\_KTOOL\_FILE\_PATH：gs\_ktool运行时需要读取配置文件gs\_ktool\_conf.ini，使用GS\_KTOOL\_FILE\_PATH来设置配置文件的路径。默认情况下，运行gs\_ktool时生成的密钥存储文件和日志文件也会存储在该路径下。

### 说明

解压GaussDB-Kernel\_数据库版本号\_操作系统版本号\_64bit\_gsql.tar.gz包后，可在gs\_ktool\_file路径下找到gs\_ktool\_conf.ini配置文件，该文件中所有配置项已设置缺省值；另外，通过指令"source gsql\_env.sh"可配置环境变量\$GS\_KTOOL\_FILE\_PATH。

表 1-24 gs\_ktool\_conf.ini 配置文件说明

配置项	配置项说明	参数类型：取值范围
rk_validity	生成RK时，默认的有效期（该配置项仅在生成RK时有效，即生成密钥文件时）。	整型：[1, 10000] days
cmk_validity	生成CMK时，默认的有效期。	整型：[1, 3660] days
cmk_length	生成CMK时，默认的密钥长度。	整型：[16, 112] bytes
pri_ksf	密钥存储文件。	字符型：绝对路径/密钥文件名
sec_ksf	备份的密钥存储文件。	字符型：绝对路径/密钥文件名
export_ksf	导出密钥时，默认的密钥存储文件。	字符型：绝对路径/密钥文件名
is_logmodule_open	是否打开日志记录功能。	字符型：{ON, OFF}
log_file	日志文件。	字符型：绝对路径/日志文件名

## 参数说明

表 1-25 gs\_ktool 参数说明

参数	参数说明	参数类型：取值范围
-g	生成单个密钥，可在配置文件中配置默认的CMK长度。	-
-l CMK_LENGTH	配合-g使用，可指定生成的CMK长度。	整型：[16, 112] bytes
-d CMK_ID	删除指定CMK。	整型：[1, 4095]; 字符型：all
-s CMK_ID	查询指定CMK的信息，不包含CMK明文或密文。	整型：[1, 4095]; 字符型：all
-e	导出当前gs_ktool中的全部密钥到文件中，可使用-f指定导出文件。	-
-i	从指定密钥文件中导入全部密钥，必须使用-f指定密钥文件；请注意该操作的覆盖特性。	-
-f FILE	配合-i和-e使用，指定密钥文件的位置（可指定绝对路径或相对路径，但必须是已存在的路径；当配合-e使用时，如果指定的文件不存在，则新建，如果已存在，则会覆盖）。	字符型：绝对路径/日志文件名
-p FILE_PASSWORD	配和-i和-e使用，输入密钥文件的口令（如果未使用该参数，则默认使用交互式口令输入方式）。	字符型：长度区间为[8, 255]，且至少包含3种类型的字符。
-R -u	轮转（更新）RK；该操作会更新CMK的密文。	-
-R -s	查询RK的信息，不包含RK的明文或密文。	-
-v	查看版本信息。	-
-h, -?	查看使用帮助。	-

### 须知

在使用gs\_ktool -d all命令删除全部密钥时，命令执行的过程中，如果同时使用gs\_ktool -g命令生成密钥，新生成的密钥可能会被删除。



## 1.3 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。

### 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，以便后续使用gs\_loader工具进行数据的导入。

**步骤1** 创建存放gs\_loader工具包的目录。

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

**步骤2** 将gs\_loader工具包上传至上一步所创建的目录中。

以上传EULER Linux版本的工具包为例，将软件安装包中的gs\_loader工具包"GaussDB-Kernel\_数据库版本号\_操作系统版本号\_64bit\_gsql.tar.gz"上传至上一步所创建的目录中。

**步骤3** 在工具包所在目录下，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gs\_loader工具版本相对应，直接查询gs\_loader客户端版本即可。

```
gs_loader -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gs\_loader工具连接到数据库后输入：

```
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。你可以通过将GUC参数enableSeparationOfDuty设置为on或者off，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，错误数据直接记录到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，将错误数据同时写入bad文件和错误表中。

- 默认场景，关闭三权分立（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。
  - a. （在管理员用户下）创建用户。

```
CREATE USER load_user WITH PASSWORD '*****';
```
  - b. （在管理员用户下）将public schema权限给用户。

```
GRANT ALL ON SCHEMA public TO load_user;
```
  - c. （在管理员用户下）创建并给用户授权gs\_copy\_summary表。

```
SELECT copy_summary_create() WHERE NOT EXISTS(SELECT * FROM pg_tables WHERE schemaname='public' AND tablename='gs_copy_summary');  
GRANT ALL PRIVILEGES ON public.gs_copy_summary To load_user;
```
  - d. （可选，在管理员用户下）创建并给用户授权错误表pgxc\_copy\_error\_log。

#### 📖 说明

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。

```
SELECT copy_error_log_create() WHERE NOT EXISTS(SELECT * FROM pg_tables WHERE schemaname='public' AND tablename='pgxc_copy_error_log');  
GRANT ALL PRIVILEGES ON public.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。
  - a. （在初始用户下）创建用户。

```
CREATE USER load_user WITH PASSWORD '*****';
```
  - b. （在初始用户下）切换为load\_user用户

```
\c - load_user
```
  - c. （在自建用户下）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);
```
  - d. （可选，在自建用户下）创建pgxc\_copy\_error\_log表并添加索引。

#### 📖 说明

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gsq方式。

## 新增系统表

新增gs\_copy\_summary表，记录COPY执行结果汇总，包括成功行数，出错行数，忽略行数，空行数。

新增函数：copy\_summary\_create用于创建gs\_copy\_summary表。

gs\_copy\_summary表格式如下：

```

relname | public.sqlldr_tbl
begintime | 2021-09-03 16:00:11.7129-04
endtime | 2021-09-03 16:00:15.259908-04
id | 21870
pid | 47582725060352
readrows | 100000
skiprows | 0
loadrows | 111
errorrows | 0
whenrows | 99889
allnullrows | 0
detail | 111 Rows successfully loaded.
| 0 Rows not loaded due to data errors.
| 99889 Rows not loaded because all WHEN clauses were failed.
| 0 Rows not loaded because all fields were null.
    
```

## 系统表 gs\_copy\_summary 字段说明

表 1-26 gs\_copy\_summary 字段说明

字段名称	描述
relname	本次导入的目标表名。
begintime	导入任务开始时间。
endtime	导入任务结束时间。
id	本次导入的事务id。
pid	本次导入工作线程id。
readrows	本次导入任务共计读取的数据行数。
skiprows	本次导入任务共计跳过的数据行数。
loadrows	本次导入任务成功导入的数据行数。
errorrows	本次导入任务的错误数据行数。
whenrows	本次导入任务违反when过滤条件的数据行数。
allnullrows	全部字段都为空的数据行数。
detail	本次导入任务的导入情况总结。主要包括成功导入行数、错误数据行数、违反when条件行数和字段全空行数等。

## 使用指导

**步骤1** （非三权分立）仅对于普通用户

- （在管理员用户下）创建用户。  
CREATE USER load\_user WITH PASSWORD '\*\*\*\*\*';
- （在管理员用户下）将public schema权限赋给普通用户。  
GRANT ALL ON SCHEMA public TO load\_user;

- （在管理员用户下）创建并给用户授权gs\_copy\_summary表。  

```
SELECT copy_summary_create() WHERE NOT EXISTS(SELECT * FROM pg_tables WHERE
schemaname='public' AND tablename='gs_copy_summary');
GRANT ALL PRIVILEGES ON public.gs_copy_summary To load_user;
```
- （可选，在管理员用户下）创建并给用户授权错误表pgxc\_copy\_error\_log。

#### 📖 说明

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。

```
SELECT copy_error_log_create() WHERE NOT EXISTS(SELECT * FROM pg_tables WHERE
schemaname='public' AND tablename='pgxc_copy_error_log');
GRANT ALL PRIVILEGES ON public.pgxc_copy_error_log To load_user;
```

- （在管理员用户下）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。

- （在初始用户下）创建用户。  

```
CREATE USER load_user WITH PASSWORD '*****';
```
- （在初始用户下）切换为load\_user用户。  

```
\c - load_user
```
- （在自建用户下）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);
```
- （在自建用户下，可选）创建pgxc\_copy\_error\_log表并添加索引。

#### 📖 说明

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);
```

### 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）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）创建guc参数guc.txt

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

（在gs\_loader客户端）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。

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 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

#### 注意

gs\_copy\_summary记录调用的copy语法以及详细情况，[badfile]\_bad.log文件记录错误数据以及详细情况，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。pgxc\_copy\_error\_log默认关闭，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 1-27 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

参数	参数说明	参数类型：取值范围
-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）。 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）。 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。	[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
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

参数	参数说明	参数类型：取值范围
skip	允许跳过数据文件的前多少行。	整数，默认0
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]

### ⚠ 注意

- 参数均为小写，不支持大写，同时兼容gsqll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，提交次数不能超过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。
- 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。

## 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'或'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。

 **注意**

当前CSV模式下，被引号包裹的换行符被视为字段数据的一部分。

- **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。



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。

- **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。

---

 **注意**

- 设置**OPTIONALLY ENCLOSED BY { 'string' }**时，要么数据左边不带引号字符，如果有引号字符，左右都必须为奇数个，但不必相等。
- 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。

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

对列进行处理，根据start到end范围获取对应字符串。

- **"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。

- **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**

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，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。
  - 单次导入的数据文件约束冲突的行数较多时，如数据库服务端机器内存较小（如32G），约束冲突数据量预估超200万时；或者内存128G以上，约束冲突超过1000万行时，可能会占用较大缓存，报"ERROR: memory is temporarily unavailable"而导入失败，所以此时不建议使用约束冲突不回滚特性。
- 
- bad文件对应数据为空的需要对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 |
-----+-----+-----+-----+-----
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),'bbb','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，length的默认长度为1。
- 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，指定GUC参数时，不支持部分列使用POSITION运算。
- 在多列导入场景中，普通数据类型与特殊数据类型混用时，需全部指定POSITION。
- 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, a\_format\_dev\_version及a\_format\_version参数。
- 列表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。
- 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,HELLO,33,
```

导入结果为：

```
loader=# select * from gsloader;
id | text | gmt_create | create_str
---+-----+-----+-----
11 | 2023-02-08 16:00:54 | HELLO | 33
```