**备份恢复命令**

**使用说明**

作者 ： 陈淼

版本 ： 1.6

命令版本 ： 1.5

日期 ： 2018/12/18

# 命令介绍

gpmcbackup与gpmcrestore命令配合使用，这两个脚本用于备份和恢复GP数据库数据。命令使用PERL编写，需在Master上执行。

支持增量备份，对于增量的识别，AO表的增量识别与gpcrondump的方式一致，通过最后操作时间和操作计数器来识别数据表从上次备份到本次备份是否发生过变化；Heap表的增量识别，通过识别底层数据文件从上次备份到本次备份是否发生过变化。要完整恢复一个增量备份的数据，需要配合gpmcrestore一起使用。

如果没有指定备份的目录，数据将备份到实例的db\_dumps目录下，以14位时间戳作为时间子目录，每天可以备份多次。以前每天只能有一个备份的限制现已取消，每日备份次数请根据需求自行决定。

指定增量备份时，如果在此之前没有进行过全量备份，此时将自动进行全量备份，当次的全量备份将作为之后增量备份的全量依据。在已有全量备份的情况下，如果需要重新开始一个新的全量备份，不指定--increment参数即可使当次备份作为一个新的全量。

备份可以指定备份表清单，即便是增量模式，也可以指定备份表清单，不过，如果之后在此基础上进行增量备份，而且又没有使用相同的表清单，将会自动扩大备份的范围。比如，系统中有很多表，希望通过多次备份来完成一个全备，则可以分批指定表清单进行备份。

如果仅希望保留一个完整的备份，而增量保留的数量有限，可以选择增量合并功能，在每次增量备份开始时即会检查上次备份到当前备份之间的增量备份的个数，如果超过了指定的数量，则会将全量和多余份数的增量进行合并，以得到新的全量，这样可以避免过多的增量备份无限制的占据磁盘存储空间。

备份脚本会自动展开所有的分区表，每个叶子分区作为一个独立的备份单元进行备份，备份先将数据备份到一个临时的文件中，待成功之后才修改为 或者 覆盖 正式的备份文件名称。

备份可以指定条件进行备份，对于那些数据表经常会发生变化的场景，不适合使用增量模式的情况，可以自行设计某些条件通过逻辑上变相实现增量备份。

恢复时，需要指定一个时间戳，命令会自动根据给定的时间戳查找最新的备份时间点，同时根据此时间点向前追溯，找到距离该时间点最新的一份全量备份。

备份的日志信息对备份和恢复都非常有用，不可随意修改或删除，否则可能会影响功能的正常使用，对于指定了增量合并的情况，过期的数据和日志会自动清理，无需手动干预。

为了不与GP自带的gpdbrestore命令名称冲突，自1.5版本开始，备份恢复命令改名为gpmc开头。

从1.5版本开始，使用gpddlbackup和gpddlrestore来备份和恢复ddl信息，可以极大改善ddl备份和恢复的性能，不过，这两个命令暂时还没有经过大规模的验证，但，这并不影响数据备份，而实际上，理论上来说，ddl信息并不是完全不可恢复。

# 退出码说明

退出值含义：

0 全部任务SUCCESS

1 脚本报错退出

11 无表需要备份

33 备份的表中有异常(未全SUCCESS)

# 命令部署

将下述4个文件文件复制到**$GPHOME/bin**目录下。修改owner为gpadmin和mod为755。

两个文件皆为平面文本，可以直接将内容复制粘帖。

|  |
| --- |
| [gpadmin@mdw bin]$ cd $GPHOME/bin  [gpadmin@mdw bin]$ ls -1 gp{mc,ddl}\*  gpddlbackup  gpddlrestore  gpmcbackup  gpmcrestore |

# 备份命令用法

|  |
| --- |
| gpmcbackup --database database name  [--port database port]  [-t <schema.relation> [-t <schema.relation>] ...]  [-T <schema.relation> [-T <schema.relation>] ...]  [-f file include table name]  [-F file exclude table name]  [-s <schema> [-s <schema>] ...]  [-S <schema> [-S <schema>] ...]  [-B batch\_size]  [--directory directory]  [--where condition]  [--time-flag time\_flag]  [--increment]  [--residue number]  [--force-redo]  [-a]  [--truncate]  [--no-error]  [--encoding encoding]  [--fast]  [--parameter-file filename]  [-h|--help]  [--version] |

# 备份命令参数说明

**如果不指定任何表名，则缺省备份指定数据库中的全部用户表。**

* **--database database name**

必选参数，需要备份的数据库名称。

* **--port database port**

可选参数，数据库端口。

如果不指定，使用缺省值5432。

* **-t table\_name [-t tables\_name] ...**

可选参数，指定需要备份的表名。

以schema.relname的方式提供，命令会自动检查该表是否存在，如不存在，会忽略该表并给出提示信息。

可以是root分区表，也可以是分区叶子表，可根据情况决定，但中间层次的分区表名称不被支持。

* **-T table\_name [-t tables\_name] ...**

可选参数，指定被排除的表名。

以schema.relname的方式提供，命令会自动检查该表是否存在，如不存在，会忽略该表并给出提示信息。

可以是root分区表，也可以是分区叶子表，可根据情况决定，但中间层次的分区表名称不被支持。

* **-f**

需要备份的表清单。

可以被接受的格式为：

schema\_a.relname\_a[;some\_code='021']

分别表示：

表名;过滤条件

过滤条件，是可选项，无需写where，无需使用双引号引起，与--where有别，--where使用双引号的原因是shell识别问题，文件中指定是文本读取。

不过，当使用增量备份时，所有的过滤条件无效。

* **-F**

需要被排除的表清单，此文件中出现的表将不会被备份。

可以被接受的格式为：

schema\_a.relname\_a

* **-s schema\_name [-s schema\_name] ...**

可选参数，指定需要备份的模式名称。

* **-S schema\_name [-s schema\_name] ...**

可选参数，指定需要被排除的模式名，该模式名下的表将不会被备份。

* **-B**

可选参数，并发数量，同时在执行备份的表的个数。

* **--directory directory**

可选参数，备份文件存储的目录，缺省存储在实例的db\_dumps目录下，如果指定了该参数，命令会自动在指定的目录下建立相关子目录，例如：--directory /data

备份文件的存储路径将可能是

/data/gpseg0/db\_dumps/20170101120055/dbnm^0^scma.tbnm.gz

* **--where**

可选参数，备份数据时的过滤条件。此为全局条件，对于在-f中指定了条件的情况，此参数不能覆盖。-f表清单中指定的条件优先级高于--where指定的条件。

注意条件部分不能包含where关键字，且条件需用双引号引起来，例如：

--where “somecode=’201’ and date=’20160101’”

当使用增量备份时，所有的过滤条件无效。

* **--time-flag time\_flag**

可选参数，备份的时间戳，缺省为当前时间年月日时分秒的14位数字格式，如20170101120055。这个参数请不要胡乱设置，因为增量备份和增量合并都基于一个假设：日期越早的备份是越旧的备份。最早版本会识别备份信息中的时间戳，不过，由于后来增加了对Heap表的识别，无法进行可靠的时间戳识别，所以，必须基于这个通常不会有问题的假设。

* **--increment**

可选参数，将表的当前状态与之前的备份进行比较，只备份发生了变化的表，Heap表也可以识别是否发生变化，但Heap表的变化识别效率不是特别高，所以，对于Heap表的使用要遵守最佳实践的要求，目前是鼓励普遍使用AO表。

* **--residue**

可选参数，指定增量保留的个数，以备份开始之前的情况计算，如果已有1份全量，10份增量，指定--residue为5，则会将1份全量和紧接着的5分增量进行合并，合并之后的时间戳目录为第5个增量的时间戳。

* **--force-redo**

可选参数，强制重做已经成功的表。缺省情况下，同一个备份，如果发现某些表在之前的备份中已经成功，将会忽略这些表。这样可以避免重复备份相同的表，但如果希望在同一个时间戳标记下做增量备份，应该指定此参数，以确保可以将发生变化的表再次备份。即便指定了--force-redo参数，在--increment模式下，对于没有发生变化的表依然会忽略，对于非--increment模式下的情况则真的一定会重做。

* **-a**

可选参数，仅备份表中的数据，不备份数据库的DDL信息。

实际上，多数时候，数据库的DDL是可以找回的。

* **--truncate**

可选参数，在一张表备份成功之后，将其清空。但在增量备份模式下，该参数是无效的。

* **--no-error**

可选参数，不备份外部表的error表。很多时候这些表中的数据没有什么备份价值。

* **--encoding encoding**

可选参数，指定备出数据的编码格式。缺省使用UTF8编码，该编码为GP数据库库内编码，对于大多数用户来说，此编码已经足够，如果你的数据库使用了李巍大神的delimiter外部表，那么你可能需要用到此参数，因为库内可能会有中文乱码，此时，建议将附件中的2个文件选择一个覆盖数据库中的原始so包文件，该文件修改了原有的编码函数，可以确保数据乱码不会报错，但首先需要确保该编码方式在项目中永远不会被使用，否则可能会影响正常转码需求，不过好在一般都用不到此编码函数。.cm的文件是由作者尝试创作，.liwei文件由李巍大神提供，至于你喜欢用哪个，我就不管了。

/usr/local/greenplum-db-4.3.7.1/lib/postgresql/utf8\_and\_iso8859\_1.so



或者使用如下文件进行base64解码得到so文件



文件MD5值为

f714c8658f36c7d202f174c5795bca6d

也可以使用如下两个文件进行手工编译，make.sh为编译命令，2个文件放在一个目录，执行sh make.sh即可



* **--fast**

备份文件使用gzip -1进行快速压缩，但会增加大约30%左右的存储空间。

如果希望备份为未压缩文件，请使用gpddbackup和gpddrestore命令，这两个命令备份数据文件将不压缩，适合将数据备份到DataDomain压缩去重存储上。

* **--parameter-file**

通过文件的方式指定参数，建议将通常不会发生变化的参数使用参数文件的方式指定。

此前的所有参数均可以通过参数文件的方式指定。

格式为：

database=postgres

port=4371

truncate

...

在命令行指定的参数优于参数文件中指定的参数。

对于多选类参数，不存在覆盖特性，两者都是有效值。

对于开关类参数，不存在覆盖特性，因为这些参数缺省是关闭的，唯一可选择的就是打开，因此，有一处打开即为打开。

* **-h|--help**

显示帮助信息。

* **--version**

显示当前命令的版本。

# 恢复命令用法

|  |
| --- |
| gpmcrestore --database database name  [--backup-database database name]  [--port database port]  [-t <schema.relation> [-t <schema.relation>] ...]  [-T <schema.relation> [-T <schema.relation>] ...]  [-f file include table name]  [-F file exclude table name]  [-s <schema> [-s <schema>] ...]  [-S <schema> [-S <schema>] ...]  [-B batch\_size]  [--directory directory]  [--where condition]  --time-to time  [--force-redo]  [-a]  [--truncate]  [--no-error]  [--encoding encoding]  [--parameter-file filename]  [-h|--help]  [--version] |

# 恢复命令参数说明

* **--database database name**

必选参数，需要恢复的数据库名称。

* **--backup-database**

备份时的数据库名称，用于将从其他数据库的备份数据库中恢复数据。

* **--port database port**

可选参数，数据库端口。

如果不指定，使用缺省值5432。

* **-t table\_name [-t tables\_name] ...**

可选参数，指定需要恢复的表名。

以schema.relname的方式提供，命令会自动检查该表是否存在，如不存在，会忽略该表并给出提示信息。

可以是root分区表，也可以是分区叶子表，可根据情况决定，但中间层次的分区表名称不被支持。

* **-T table\_name [-t tables\_name] ...**

可选参数，指定被排除的表名。

以schema.relname的方式提供，命令会自动检查该表是否存在，如不存在，会忽略该表并给出提示信息。

可以是root分区表，也可以是分区叶子表，可根据情况决定，但中间层次的分区表名称不被支持。

* **-f**

需要恢复的表清单。

可以被接受的格式为：

schema\_a.relname\_a[;some\_code='021']

分别表示：

表名;过滤条件

过滤条件，是可选项，无需写where，无需使用双引号引起，与--where有别，--where使用双引号的原因是shell识别问题，文件中指定是文本读取。

不过，当使用增量备份时，所有的过滤条件无效。

* **-F**

需要被排除的表清单，此文件中出现的表将不会被恢复。

可以被接受的格式为：

schema\_a.relname\_a

* **-s schema\_name [-s schema\_name] ...**

可选参数，指定需要恢复的模式名称。

* **-S schema\_name [-s schema\_name] ...**

可选参数，指定需要被排除的模式名，该模式名下的表将不会被恢复。

* **-B**

可选参数，并发数量，同时在执行恢复的表的个数。

* **--directory directory**

可选参数，备份文件存储的目录，缺省存储在实例的db\_dumps目录下，如果指定了该参数，命令会在指定的目录下查找备份信息

* **--where**

可选参数，恢复数据时的过滤条件。此为全局条件，对于在-f中指定了条件的情况，此参数不能覆盖。-f表清单中指定的条件优先级高于--where指定的条件。

注意条件部分不能包含where关键字，且条件需用双引号引起来，例如：

--where “somecode=’201’ and date=’20160101’”

* **--time-to time**

必选参数，恢复的数据结束时间戳，值应为14位数字的时间戳，命令自动根据该值找到最后一次增量备份和最后一次全量备份，将数据恢复到该时间戳最近的时间。

* **--force-redo**

可选参数，强制重做已经成功的表。缺省情况下，相同日期标记的恢复，如果发现某些表在之前的恢复中已经成功，将会忽略这些表。这样可以避免重复恢复相同的表。

* **-a**

可选参数，仅恢复表中的数据，不恢复数据库的DDL信息。

* **--truncate**

可选参数，在恢复一张表之前，先将其清空。

* **--no-error**

可选参数，不备份外部表的error表。很多时候这些表中的数据没有什么备份价值。

* **--encoding encoding**

可选参数，指定备出数据的编码格式。参见之前的备份命令此参数说明。

* **--parameter-file**

可选参数，以参数文件的形式指定参数值。参见之前的备份命令此参数说明。

* **-h|--help**

显示帮助信息。

* **--version**

显示当前命令的版本。

# 备份命令示例

假如，按计划，每天对dw库进行增量备份，且仅保留7个增量，备份使用的目录为/backup，由于没有使用特殊的乱码处理程序，使用UTF8编码，由于日常运行是增量备份，使用缺省并发度，或者更小的并发度，DDL不在备份脚本中执行。于是可以每天执行如下命令：

gpmcbackup --database dw --directory /backup --residue 7 --increment --B 4 --encoding UTF8 -a

在正式部署之前，请进行必要的测试以确保工作正常，可以在测试环境测试或者选择个别表进行测试。

# 恢复命令示例

假如，将dw库的备份数据恢复到newdw数据库中，恢复的备份周期为20170101到20170108，备份数据的目录是/backup，为尽快恢复，并发度选择8。于是执行如下命令：

gpmcrestore --database newdw --backup-database dw --directory /backup --time-to 20140825082500 -B 8 --encoding UTF8 –a

在正式部署之前，请进行必要的测试以确保工作正常，可以在测试环境测试或者选择个别表进行测试。