

# 达梦技术手册

## DM8\_dmfldr 使用手册

Service manual of DM8\_dmfldr



# 前言

## 概述

本文档主要介绍如何使用 dmflldr 快速装载工具进行 DM 数据库中表数据的快速载入和导出。

## 读者对象

本文档主要适用于 DM 数据库的：

- 开发工程师
- 测试工程师
- 技术支持工程师
- 数据库管理员

## 通用约定

在本文档中可能出现下列标志，它们所代表的含义如下：

表 0.1 标志含义

标志	说明
 <b>警告：</b>	表示可能导致系统损坏、数据丢失或不可预知的结果。
 <b>注意：</b>	表示可能导致性能降低、服务不可用。
 <b>小窍门：</b>	可以帮助您解决某个问题或节省您的时间。
 <b>说明：</b>	表示正文的附加信息，是对正文的强调和补充。

在本文档中可能出现下列格式，它们所代表的含义如下：

表 0.2 格式含义

格式	说明
宋体	表示正文。
黑体	标题、警告、注意、小窍门、说明等内容均采用黑体。
Courier new	表示代码或者屏幕显示内容。
粗体	表示命令行中的关键字（命令中保持不变、必须照输的部分）或者正文中强调的内容。
<>	语法符号中，表示一个语法对象。
::=	语法符号中，表示定义符，用来定义一个语法对象。定义符左边为语法对象，右边为相应的语法描述。
	语法符号中，表示或者符，限定的语法选项在实际语句中只能出现一个。
{ }	语法符号中，大括号内的语法选项在实际的语句中可以出现 0...N 次 (N 为大于 0 的自然数)，但是大括号本身不能出现在语句中。
[ ]	语法符号中，中括号内的语法选项在实际的语句中可以出现 0...1 次，但是中括号本身不能出现在语句中。
关键字	关键字在 DM_SQL 语言中具有特殊意义，在 SQL 语法描述中，关键字以大写形式出现。但在实际书写 SQL 语句时，关键字既可以大写也可以小写。

## 访问相关文档

如果您安装了 DM 数据库，可在安装目录的“\doc”子目录中找到 DM 数据库的各种手册与技术丛书。

您也可以通过访问我们的网站 [www.dameng.com](http://www.dameng.com) 阅读或下载 DM 的各种相关文档。

## 联系我们

如果您有任何疑问或是想了解达梦数据库的最新动态消息，请联系我们：

网址：[www.dameng.com](http://www.dameng.com)

技术服务电话：400-991-6599

技术服务邮箱：[dmtech@dameng.com](mailto:dmtech@dameng.com)

# 目录

<b>1 概述</b>	<b>1</b>
1.1 功能简介	1
1.2 系统结构	1
<b>2 DMFLDR 入门</b>	<b>3</b>
2.1 启动 DMFLDR	3
2.2 查看 DMFLDR 参数	3
2.3 DMFLDR 参数简介	5
<b>3DMFLDR 实战</b>	<b>18</b>
3.1 DMFLDR 控制文件	18
3.2 指定数据文件	23
3.2.1 在控制文件中指定数据文件	23
3.2.2 使用 DATA 参数指定数据文件	24
3.3 数据转换与错误数据文件	25
3.4 服务器端错误数据处理	27
3.5 大字段数据处理	27
3.5.1 大字段数据的导出	27
3.5.2 DIRECT 为 TRUE 时大字段数据的载入	28
3.5.3 DIRECT 为 FALSE 时大字段数据的载入	29
3.6 日志文件及日志信息	31
3.7 自增列装载	33
3.8 数据排序	35
3.9 空值处理	37
3.10 类类型装载	38
3.11 条件过滤	41
3.12 多表装载	43
3.13 个性化设置	45
3.14 主备切换时的数据继续载入	46

3.15MPP 本地分发.....	47
3.16 提升 DMFLDR 性能 .....	49
3.17DMFLDR 使用限制.....	50

# 1 概述

## 1.1 功能简介

dmflldr (DM Fast Loader) 是 DM 提供的快速数据装载命令行工具。用户通过使用 dmflldr 工具能够把按照一定格式排序的文本数据以简单、快速、高效的方式载入到 DM 数据库中, 或把 DM 数据库中的数据按照一定格式写入文本文件。

## 1.2 系统结构

dmflldr 的系统结构如图 1.1 所示。

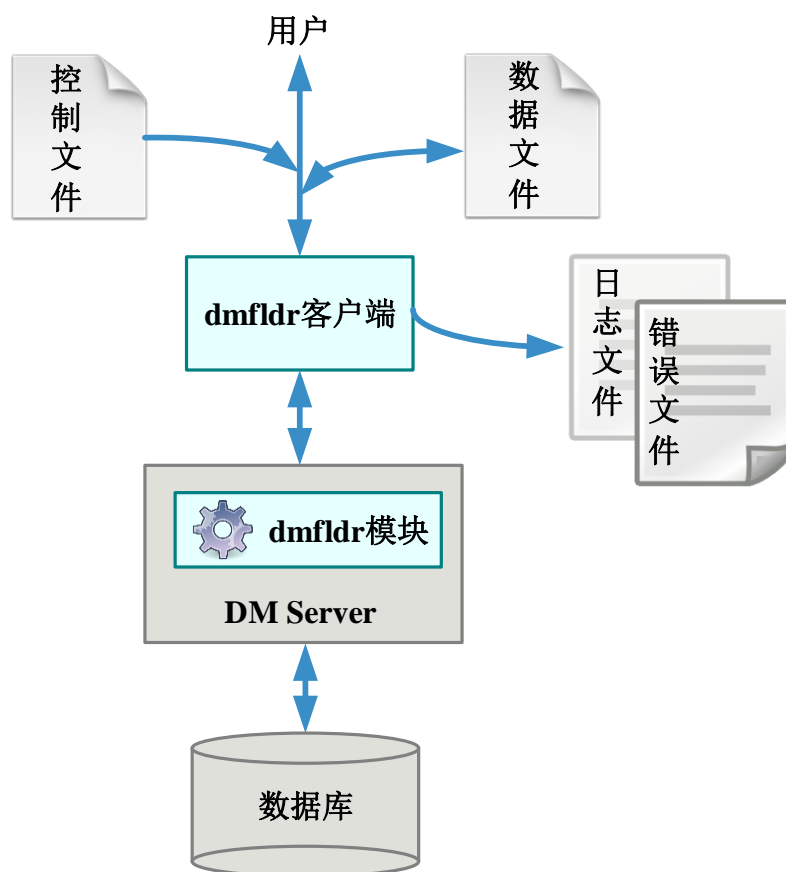


图 1.1 dmflldr 结构示意图

如图所示, dmflldr 实际上除了客户端工具, 还包含一个在数据库服务器中的 dmflldr 功能模块, 它们共同完成 dmflldr 的各项功能。

当进行数据载入时，dmfldr 客户端接收用户提交的命令与参数，分析控制文件与数据文件，将数据打包发送给服务器端的 dmfldr 模块，由服务器完成数据的真正装载工作。并分析服务器返回的消息，必要时根据用户参数指定生成日志文件与错误数据文件。

当进行数据导出时，dmfldr 客户端接收用户提交的命令与参数，分析控制文件，将用户要求转换成相应消息发送给服务器端的 dmfldr 模块。服务器解析并打包需要导出的数据，发送给 dmfldr 客户端，客户端将数据写入指定的数据文件，必要时根据用户参数指定生成日志文件。

## 2 dmfldr 入门

本章简单介绍如何启动 dmfldr 和 dmfldr 支持的参数简介。通过阅读本章，读者可以了解 dmfldr 通过各参数能提供的各项功能，不过要想熟练灵活地使用 dmfldr 还需要继续阅读下一章。

### 2.1 启动 dmfldr

安装好 DM 数据库管理系统后，在安装目录的“bin”子目录下可找到 dmfldr 执行文件。

启动操作系统的命令行窗口，进入“dmfldr”所在目录，可以准备启动 dmfldr 工具了。

dmfldr 的使用必须指定必要的参数，否则工具会报错“无效的参数个数”并退出。为 dmfldr 指定参数的格式为：

---

```
dmfldr keyword=value [keyword=value ...]
```

---

例如：

```
dmfldr USERID=SYSDBA/SYSDBA CONTROL='c:\fldr.ctl'
```

如例子所示，USERID 和 CONTROL 是启动 dmfldr 必须要指定的参数，且 USERID 必须是第一个参数，CONTROL 必须是第二个参数。

### 2.2 查看 dmfldr 参数

dmfldr 使用较为灵活，参数较多，用户可以使用“dmfldr help”查看各参数的简单信息。

```
dmfldr help
```

```
格式： DMFLDR    KEYWORD=value
```

```
例程： DMFLDR    SYSDBA/SYSDBA CONTROL='c:\fldr.ctl'
```

```
USERID 必须是命令行中的第一个参数
```

```
CONTROL 必须是命令行中的第二个参数
```



字符串类型参数必须以引号封闭

关键字	说明（默认值）
-----	
USERID	用户名/口令格式:USER/PWD@SERVER:PORT#SSL_PATH@SSL_PWD
CONTROL	控制文件，字符串类型
LOG	日志文件，字符串类型 (fldr.log)
BADFILE	错误数据记录文件，字符串类型 (fldr.bad)
SKIP	初始忽略逻辑行数 (0)
LOAD	需要装载的行数 (ALL)
ROWS	提交频次 (50000)，DIRECT 为 FALSE 有效
DIRECT	是否使用快速方式装载 (TRUE)
SET_IDENTITY	是否插入自增列 (FALSE)
SORTED	数据是否已按照聚集索引排序 (FALSE)
INDEX_OPTION	索引选项 (1)
	1 不刷新二级索引，数据按照索引先排序，装载完后再将排序的数据插入索引
	2 不刷新二级索引，数据装载完成后重建所有二级索引
	3 刷新二级索引，数据装载的同时将数据插入二级索引
ERRORS	允许的最大数据错误数 (100)
CHARACTER_CODE	字符编码，字符串类型 (GBK, GB18030, UTF-8, SINGLE_BYTE, EUC-KR)
MODE	装载方式，字符串类型 IN 表示载入，OUT 表示载出，OUTORA 表示载出 ORACLE (IN)
CLIENT_LOB	大字段目录是否在本地 (FALSE)
LOB_DIRECTORY	大字段数据文件存放目录
LOB_FILE_NAME	大字段数据文件名称，仅导出有效 (dmflldr.lob)
BUFFER_NODE_SIZE	读入文件缓冲区的大小 (10)，有效值范围 1~2048
READ_ROWS	工作线程一次最大处理的行数 (100000)，最大支持 $2^{26}$ -10000
NULL_MODE	载入时 NULL 字符串是否处理为 NULL
	载出时空值是否处理为 NULL 字符串 (FALSE)

NULL_STR	载入时视为 NULL 值处理的字符串
SEND_NODE_NUMBER	运行时发送节点的个数 (20), 有效值范围 16~65535
TASK_THREAD_NUMBER	处理用户数据的线程数目, 默认与处理器核数量相同, 有效值范围 1~128
BLDR_NUM	服务器 BLDR 数目 (64), 有效值范围 1~1024
BDTA_SIZE	bdtA 的大小 (5000), 有效值范围 100~10000
COMPRESS_FLAG	是否压缩 bdtA (FALSE)
MPP_CLIENT	MPP 环境, 是否本地分发 (TRUE)
SINGLE_FILE	MPP 环境, 是否只生成单个数据文件 (FALSE)
LAN_MODE	MPP 环境, 是否以内网模式装载数据 (FALSE)
UNREP_CHAR_MODE	非法字符处理选项 (0), 为 0 时表示跳过该数据行, 为 1 时表示使用 (*) 替换错误字节
SILENT	是否静默方式装载数据 (FALSE)
BLOB_TYPE	BLOB 类型字段数据值的实际类型, 字符串类型 (HEX_CHAR) HEX 表示值为十六进制, HEX_CHAR 表示值为十六进制字符类型
仅在 direct=FALSE 有效	
OCI_DIRECTORY	OCI 动态库所在的目录
DATA	指定数据文件路径
ENABLE_CLASS_TYPE	允许用户导入 CLASS 类型数据 (FALSE)
FLUSH_FLAG	提交时是否立即刷盘 (FALSE)
IGNORE_BATCH_ERRORS	是否忽略错误数据继续导入 (FALSE)
SINGLE_HLDR_HP	是否使用单个 HLDR 装载 HUGE 水平分区表 (FALSE)
EP	指定需要发送数据的站点序号列表, 仅向 MPP 环境导入数据时有效
SQL	使用自定义查询语句, 仅导出模式有效
HELP	打印帮助信息

## 2.3 dmflldr 参数简介



**注意:** USERID 必须是第一个参数, CONTROL 必须是第二个参数, 这两个参数都不能省略; 其余参数均为可选参数, 可以不指定, 指定时也无顺序要求

## ■ USERID

用于指定登录数据库的信息，包含用户名、密码、服务器地址、端口号和安全证书存放路径。参数值格式为：

---

用户名/密码[@主库地址[:端口号]][#安全证书存放路径@安全证书密码]]

---

如果用户使用默认端口号 5236 登录则端口号可以不设置；如果登录的是本地的服务器则主库地址可以省略；如果不使用通信加密则安全证书相关设置可以省略。

此参数为必选参数且必须位于参数位置的第一个。

## ■ CONTROL

控制文件的路径，字符串类型。控制文件用于指定数据文件中数据的格式，在数据载入时，dmflldr 根据控制文件指定的格式来解析数据文件；导出数据时，dmflldr 也会根据控制文件指定的列分隔符、行分隔符等生成数据文件。控制文件中还可以指定其他的一些 dmflldr 参数值。对控制文件格式的详细介绍见下一章。

此参数为必选参数且必须位于参数位置的第二个。

## ■ LOG

dmflldr 的日志文件路径，字符串类型。默认日志文件名为 fldr.log。日志文件记录了 dmflldr 运行过程中的工作信息、错误信息以及统计信息。

用户也可以通过设置控制文件中的 OPTIONS 选项来指定日志路径，参数值的优先选择顺序为先 OPTIONS 选项，后参数选项。

此参数为可选参数。

## ■ NULL\_STR

指定数据文件中 NULL 值的表示字符串，字符串类型，默认忽略此参数。

若设置了 NULL\_STR，则此参数值将成为数据文件中 NULL 值的唯一表示方式。

NULL\_STR 区分字符串大小写，并且长度不允许超过 128 个字节。

用户可以通过控制文件中的 OPTIONS 选项来设置 NULL\_STR 的值，参数值的优先顺序为先 OPTIONS 选项，后参数选项。

此参数为可选参数。

## ■ BADFILE

记录错误数据的文件路径，字符串类型。默认的错误文件名为 fldr.bad。文件记录的信息为数据文件中存在格式错误的行数据以及转换出错的行数据。

用户也可以通过设置控制文件中的 `OPTIONS` 选项来指定错误数据文件的路径，同时也可以控制文件的 `LOAD` 节点中指定错误数据文件的路径。`BADFILE` 选项的最终值的优先选择顺序为 `LOAD` 节点选项，`OPTIONS` 选项，参数选项，用户可以同时对三种设置方式中的一个或多个设置，但最终的值只取一个。

此参数为可选参数，作用于 `MODE` 为 `IN` 的情况下，当 `MODE` 为 `OUT` 时无效。

#### ■ SKIP

跳过数据文件起始的逻辑行数，整型数值。默认的跳过起始行数为 0 行。如果用户指定了多个文件，且起始文件中的行数不足 `SKIP` 所指定的行数，则 `dmflldr` 工具会扫描下一个文件直至累加的行数等于 `SKIP` 所设置的行数或者所有文件都已扫描结束。

用户也可以通过设置控制文件中的 `OPTIONS` 选项来指定 `SKIP` 参数，参数值的优先选择顺序为先 `OPTIONS` 选项，后参数选项。

#### ■ LOAD

装载的最大行数，整型数值。默认的最大装载行数为数据文件中的所有行数。`LOAD` 指定的值不包括 `SKIP` 指定的跳过的行数。

用户也可以通过设置控制文件中的 `OPTIONS` 选项来设置 `LOAD` 的值，参数值的优先选择顺序为先 `OPTIONS` 选项，后参数选项。

#### ■ ROWS

每次提交的行数，整形数值。默认的提交行数为 50000 行。提交行数的值表示提交到服务器的行数，并不一定代表按照数据文件中的数据顺序的行数。用户可以根据实际情况调整每次提交的行数，以达到性能的最佳点。

用户也可以通过设置控制文件中的 `OPTIONS` 选项来设置 `ROWS` 值，参数值的优先选择顺序为先 `OPTIONS` 选项，后参数选项。

此参数为可选参数，作用于 `MODE` 为 `IN` 且 `DIRECT` 为 `FALSE` 的情况下，其他情况下无效。

#### ■ DIRECT

数据装载方式，布尔值。默认为 `TRUE`。`DIRECT` 指定了装载的方式，当为 `TRUE` 时，`dmflldr` 选择快速的载入模式，通过数据的转换和数据的封装直接对 B 树进行操作，省去了普通插入方式下各个操作符之间的跳转，提升了装载的效率，但对于约束的检查等由用户保证，`dmflldr` 将不处理有约束冲突的数据。当为 `FALSE` 时，`dmflldr` 选择普通的插入方式装载数据，可以保证数据的正确性和约束的有效性，效率比前者要低。

用户也可以通过设置控制文件中的 `OPTIONS` 选项来设置 `DIRECT` 值，参数值的优先选择顺序为先 `OPTIONS` 选项，后参数选项。

此参数为可选参数，作用于 `MODE` 为 `IN` 的情况下，当 `MODE` 为 `OUT` 时无效。

#### ■ **SET\_IDENTITY**

设置自增列选项，布尔值。默认为 `FALSE`。如果指定 `SET_IDENTITY` 选项值为 `TRUE`，则 `dmflldr` 将从数据文件中读取的自增列值作为目标值插入数据库表中，用户应当保证每一行的自增列的值符合自增列的规则。如果 `SET_IDENTITY` 选项值设置为 `FALSE`，则 `dmflldr` 将忽略数据文件中对应自增列的值，服务器将自动生成自增列的值插入每一行的对应列。

用户也可以通过设置控制文件中的 `OPTIONS` 选项来设置 `SET_IDENTITY` 的值，参数值的优先选择顺序为先 `OPTIONS` 选项，后参数选项。

此参数为可选参数，作用于 `MODE` 为 `IN` 且 `DIRECT` 为 `TRUE` 的情况下，当 `MODE` 为 `OUT` 或 `DIRECT` 为 `FALSE` 时无效。

#### ■ **SORTED**

数据是否已经按照聚集索引排序，布尔值。默认为 `FALSE`。如果设置为 `TRUE`，则用户必须保证数据已按照聚集索引排序完成，并且如果表中存在数据，需要插入的数据索引值要比表中数据的索引值大，服务器在做插入操作时顺序进行插入。如果设置为 `FALSE`，则服务器对于每条记录进行定位插入。

用户也可以通过设置控制文件中的 `OPTIONS` 选项来设置 `SORTED` 的值，参数值的优先选择顺序为先 `OPTIONS` 选项，后参数选项。

此参数为可选参数，作用于 `MODE` 为 `IN` 且 `DIRECT` 为 `TRUE` 的情况下，对于其他情况此参数无效。

#### ■ **INDEX\_OPTION**

索引的设置选项，整形数值。默认为 1。`INDEX_OPTION` 的可选项有 1、2 和 3。1 代表服务器装载数据时先不刷新二级索引，而是将新数据按照索引预先排序，在装载完成后，再将排好序的数据插入索引；2 代表服务器在快速装载过程中不刷新二级索引数据，只在装载完成时重建所有二级索引；3 代表服务器使用追加模式来进行二级索引的插入，在数据装载的过程中，同时进行二级索引的插入。

用户也可以通过设置控制文件中的 `OPTIONS` 选项来设置 `INDEX_OPTION` 的值，参数值的优先选择顺序为先 `OPTIONS` 选项，后参数选项。

此参数为可选参数，作用于 MODE 为 IN 且 DIRECT 为 TRUE 的情况下，对于其他情况此参数无效。

## ■ ERRORS

最大的容错个数，整形数值。默认为 100。当 dmflldr 在插入过程中出现错误的个数超过了 ERRORS 所设置的数目，则 dmflldr 会停止载入，当前时间点的所有正确数据将会被提交到服务器。如果载入过程中不允许出现错误则可以设置 ERRORS 为 0，如果允许所有的错误出现，则可以设置 ERRORS 为一个非常大的数。ERRORS 所统计的错误包含在数据转换和数据插入过程中所产生的数据错误。

用户也可以通过设置控制文件中的 OPTIONS 选项来设置 ERRORS 的值，参数值的优先选择顺序为先 OPTIONS 选项，后参数选项。

此参数为可选参数，作用于 MODE 为 IN 的情况下，当 MODE 为 OUT 时无效。

## ■ CHARACTER\_CODE

数据文件中数据的编码格式，字符串类型。默认为 GBK。CHARACTER\_CODE 的可选项有 GBK、GB18080、UTF-8、SINGLE\_BYTE 和 EUC-KR 五种：GBK 和 GB18030 对应中文编码；UTF-8 对应 UTF-8 国际编码；SINGLE\_BYTE 对应单字节字符编码；EUC-KR 对应韩文字符集。

在 MODE 为 IN 的情况下，指定编码格式为 SINGLE\_BYTE 时，dmflldr 将不做字符完整性检查，按单字节顺序读取每个字符。指定编码格式为 GBK、GB18030 或 UTF-8 时，dmflldr 将对每一个字符做字符的完整性检查，确保数据的正确性。

用户在使用 dmflldr 时可以根据不同的数据文件调整编码的格式确保装载的正确性，同时如果可以确保数据文件中的数据为单字节字符，则指定 SINGLE\_BYTE 的载入效率将优于指定其他字符集的情况。

当 MODE 为非 IN 时，不支持 SINGLE\_BYTE。

用户也可以通过指定控制文件中的 OPTIONS 选项来设置 CHARACTER\_CODE 的值，参数值的优先选择顺序为先 OPTIONS 选项，后参数选项。

## ■ MODE

dmflldr 的装载模式，字符串类型。默认值为 IN。MODE 的可选项有 IN、OUT 和 OUTORA 三种。

IN 模式指从数据文件中将数据装载入数据库，这种模式下控制文件的格式对应为数据文件中现有数据的格式；OUT 模式指从数据库中将数据导出到数据文件，这种模式下控制

文件所指定的格式为数据存放在数据文件中的格式。需要说明的是在 OUT 模式下，如果指定了多个数据文件，则 dmfldr 最终只会将数据写入第一个数据文件。视图对象只支持导出，不支持导入。

OUT 模式下的控制文件与 IN 模式下的控制文件格式相同，用户可以通过使用同一个文件进行载入和导出数据。

OUTORA 模式表示导出 ORACLE 表的数据，此模式下暂不支持带有大字段表的导出。

用户也可以通过指定控制文件中的 OPTIONS 选项来设置 MODE 的值，参数值的优先选择顺序为先 OPTIONS 选项，后参数选项。

#### ■ CLIENT\_LOB

指明 LOB\_DIRECTORY 表示的目录是否是客户端本地目录，布尔类型，默认值为 FALSE。

CLIENT\_LOB 仅在 MODE 为 IN 且 DIRECT 为 TRUE 时有效，指明在载入大字段对象数据时，LOB\_DIRECTORY 参数指定的目录是否是客户端本地目录。若为 TRUE，表示目录为客户端本地目录；若为 FALSE，表示目录为 DM 服务器所在主库的目录。

#### ■ LOB\_DIRECTORY

指明 dmfldr 使用的大字段数据存放的目录，字符串类型。

当 MODE 为 OUT 时，dmfldr 生成大字段对应的数据文件，文件名由 LOB\_FILE\_NAME 指定，并存放于 LOB\_DIRECTORY 指定的目录，如果未指定 LOB\_DIRECTORY 则存放于指定的导出数据文件同一目录。对于 MODE 为 OUT 的情况，此参数为可选参数。

当 MODE 为 IN 且 DIRECT 为 TRUE 时，此时数据载入若涉及到大字段对象，需要用户指定大字段数据文件。若 CLIENT\_LOB 为 TRUE，LOB\_DIRECTORY 应指定大字段数据文件所在的客户端本地目录；若 CLIENT\_LOB 为 FALSE，用户必须先把相关文件传送到 DM 服务器所在主库，然后使用 LOB\_DIRECTORY 指明存放目录。此时此参数为必选参数。

当 MODE 为 IN 且 DIRECT 为 FALSE 时，此参数无效。

用户也可以通过指定控制文件中的 OPTIONS 选项来设置 LOB\_DIRECTORY 的值，参数值的优先顺序为先 OPTIONS 选项，后参数选项。

#### ■ LOB\_FILE\_NAME

指明 dmfldr 导出大字段数据的文件名，字符串类型，默认为 dmfldr.lob。

当 MODE 为 OUT 时, dmfldr 生成大字段对应的数据文件名由 LOB\_FILE\_NAME 指定, 若未指定默认为 dmfldr.lob, 文件存放于 LOB\_DIRECTORY 指定的目录。

用户也可以通过设置控制文件中的 OPTIONS 选项来设置 LOB\_FILE\_NAME 的值, 参数值的优先选择顺序为先 OPTIONS 选项, 后参数选项。

此参数为可选参数, 作用于 MODE 为 OUT 的情况下, 当 MODE 为 IN 时无效。

#### ■ BLOB\_TYPE

指定 BLOB 数据值的实际类型, 字符串类型。可选项为 HEX 表示值为十六进制, HEX\_CHAR 表示值为十六进制字符, 默认为 HEX\_CHAR。

用户也可以通过设置控制文件中的 OPTIONS 选项来设置 BLOB\_TYPE 的值, 参数值的优先顺序为先 OPTIONS 选项, 后参数选项。

此参数为可选参数, 只在 DIRECT 参数为 FALSE 的情况下有效。

#### ■ BUFFER\_NODE\_SIZE

指定读取文件缓冲区页大小, 整数类型, 单位为 MB, 范围为 1~2048, 默认为 10。

BUFFER\_NODE\_SIZE 的值越大, 缓冲区的页越大, 每次读取的数据就越多, 每次发送到服务器的数据也就越多, 效率越高。但其大小受 dmfldr 客户端内存大小限制。

用户也可以通过设置控制文件中的 OPTIONS 选项来设置 BUFFER\_NODE\_SIZE 的值, 参数值的优先顺序为先 OPTIONS 选项, 后参数选项。

#### ■ READ\_ROWS

指定读取缓冲区每次读取的最大行数, 整数类型, 范围为 0~( $2^{26}$ -10000) 默认为 100000。

在某些情况下, 1MB 的 BUFFER\_NODE\_SIZE 读入的数据行数很大, 而后续操作处理不了这么大的行数, 此时可以用 READ\_ROWS 来限制处理的行数。dmfldr 取 READ\_ROWS 和 BUFFER\_NODE\_SIZE 中较小的值作为一次处理的行数。

用户也可以通过设置控制文件中的 OPTIONS 选项来设置 READ\_ROWS 的值, 参数值的优先顺序为先 OPTIONS 选项, 后参数选项。

#### ■ NULL\_MODE

指定载入和导出数据时对 NULL 字符串和空值的处理方式, 布尔类型, 默认为 FALSE。

若 NULL\_MODE 为 TRUE, 数据载入时将 NULL 字符串处理为空值, 数据导出时则空值处理为 NULL 字符串; 若设置为 FALSE, 数据载入时将 NULL 字符串处理为字符串, 数据导出时将空值处理为空串。



用户也可以通过设置控制文件中的 `OPTIONS` 选项来设置 `NULL_MODE` 的值，参数值的优先顺序为先 `OPTIONS` 选项，后参数选项。

#### ■ `SEND_NODE_NUMBER`

指定 `dmfldr` 在数据载入时发送节点的个数，整数类型，范围为 16~65535，默认有 20 个数据节点。

用户也可以通过设置控制文件中的 `OPTIONS` 选项来设置 `SEND_NODE_NUMBER` 的值，参数值的优先顺序为先 `OPTIONS` 选项，后参数选项。

此参数为可选参数，只在 `MODE` 为 `IN` 的情况下有效。

#### ■ `TASK_THREAD_NUMBER`

指定 `dmfldr` 在数据载入时处理用户数据的线程数目，整数类型，范围为 1~128。默认情况下，`dmfldr` 将该参数值设为系统 CPU 的个数，但无论设定值是多少，`dmfldr` 至少会创建 2 个任务线程。在多核 CPU 环境下，增大 `TASK_THREAD_NUMBER` 值可以提升 `dmfldr` 装载性能。



在导出模式下，当 `TASK_THREAD_NUMBER` 设置为大于 16 而小于 128 时，  
说明: `dmfldr` 不会报错，但会将 `TASK_THREAD_NUMBER` 自动置为 16。

用户也可以通过设置控制文件中的 `OPTIONS` 选项来设置 `TASK_THREAD_NUMBER` 的值，参数值的优先顺序为先 `OPTIONS` 选项，后参数选项。

#### ■ `BLDR_NUM`

水平分区表装载时，指定服务器 `BLDR` 的最大个数，整数类型，范围为 1~1024，默认为 64。

服务器的 `BLDR` 保存水平分区子表相关信息，`BLDR_NUM` 的设置也就指定了服务器能同时载入的水平分区子表的个数。

用户也可以通过设置控制文件中的 `OPTIONS` 选项来设置 `BLDR_NUM` 的值，参数值的优先顺序为先 `OPTIONS` 选项，后参数选项。

此参数为可选参数，作用于 `MODE` 为 `IN` 的情况下，当 `MODE` 为 `OUT` 时无效。

#### ■ `BDTA_SIZE`

`BDTA` (Batch Data) 的大小，整数类型，范围为 100~10000，默认为 5000。

`BDTA` 代表 DM 数据库批量数据处理机制中一个批量，在内存、CPU 允许的条件下，增大 `BDTA_SIZE` 能加快装载速度；在网络是装载性能瓶颈时，增大 `BDTA_SIZE` 影响不大。

用户也可以通过设置控制文件中的 `OPTIONS` 选项来设置 `BDTA_SIZE` 的值，参数值的优先顺序为先 `OPTIONS` 选项，后参数选项。

此参数为可选参数，作用于 `MODE` 为 `IN` 的情况下，当 `MODE` 为 `OUT` 时无效。

#### ■ **COMPRESS\_FLAG**

指定是否压缩 `BDTA`，布尔类型，默认为 `FALSE`。

压缩 `BDTA` 能节省网络带宽，但同时也会加重 `CPU` 的负担，用户应根据具体应用情况考虑是否制定压缩。

用户也可以通过设置控制文件中的 `OPTIONS` 选项来设置 `COMPRESS_FLAG` 的值，参数值的优先顺序为先 `OPTIONS` 选项，后参数选项。

#### ■ **MPP\_CLIENT**

指定当服务器环境为 `MPP` 环境时 `dmflldr` 的数据装载模式，布尔类型，默认为 `TRUE`。

当服务器环境为 `MPP` 环境时，若 `MPP_CLIENT` 为 `TRUE`，为客户端分发模式，数据在 `dmflldr` 客户端分发好后直接往指定站点发送数据；若 `MPP_CLIENT` 为 `FALSE`，为本地分发模式，`dmflldr` 客户端将数据全部发往连接的 `MPP EP` 站点。

`MPP` 环境下要配置 `dmmal.ini` 文件中的 `MAL_INST_HOST` 和 `MAL_INST_PORT` 参数。

用户也可以通过设置控制文件中的 `OPTIONS` 选项来设置 `MPP_CLIENT` 的值，参数值的优先顺序为先 `OPTIONS` 选项，后参数选项。

此参数只有当服务器环境为 `MPP` 环境时才有效。

#### ■ **SINGLE\_FILE**

指定当服务器环境为 `MPP` 环境时，`dmflldr` 的导出数据文件是否为单个文件，布尔类型，默认为 `FALSE`。

此参数只在 `MPP` 环境下且 `MODE` 为 `OUT` 时有效。参数值为 `TRUE` 表示仅生成一个数据导出文件，`MPP` 各个站点的数据将导出到同一个数据文件；值为 `FALSE` 表示可以生成多个数据文件，每一个 `MPP` 站点都有专门的数据文件接收该站点的数据。

用户也可以通过设置控制文件中的 `OPTION` 选项来设置 `SINGLE_FILE` 的值，参数值的优先顺序为先 `OPTION` 选项，后参数选项。

#### ■ **LAN\_MODE**

指定当服务器环境为 `MPP` 环境时，`dmflldr` 导入/导出数据是否使用局域网，布尔类型，默认为 `FALSE`。

此参数只在 MPP 环境下有效。值为 TRUE 表示使用局域网，此时服务器的 MAL 系统必须配置了局域网 IP，否则 dmflldr 依然采用服务器对外服务的外网 IP；值为 FALSE 表示不使用局域网。

用户也可以通过设置控制文件中的 OPTION 选项来设置 LAN\_MODE 的值，参数值的优先顺序为先 OPTION 选项，后参数选项。

#### ■ UNREP\_CHAR\_MODE

指定是否将不完整的字符用 “\*” 替换。1 是，0 否，默认为 0。

设置为 1 时，dmflldr 装载过程中遇到包含不完整字符的数据时，将用 "\*" 替换不完整的字符；为 0 时，该行数据可能会被丢弃。

用户可以通过设置控制文件中的 OPTIONS 选项来设置 UNREP\_CHAR\_MODE 的值，参数值的优先顺序为先 OPTIONS 选项，后参数选项

#### ■ SILENT

指定是否以静默方式进行数据装载，布尔类型，默认为 FALSE。

当设置为 TRUE 时，dmflldr 装载过程中将忽略反馈式的消息，例如装载进度提示信息、错误信息等，但仍然会在装载的开始和结束阶段打印一些静态/统计信息。静默方式只作于客户端的屏幕打印，dmflldr 日志依然会完整地记录装载过程中的详细信息。

用户可以通过设置控制文件中的 OPTIONS 选项来设置 SILENT 的值，参数值的优先顺序为先 OPTIONS 选项，后参数选项

#### ■ OCI\_DIRECTORY

指定 ORACLE OCI 动态库所在的目录，字符串类型。

该参数与 MODE 参数配合使用，当 MODE 指定为 OUTORA 时，使用 OCI\_DIRECTORY 指定 OCI 的动态库来构建导出环境。

用户也可以通过设置控制文件中的 OPTIONS 选项来设置 OCI\_DIRECTORY 的值，参数值的优先顺序为先 OPTIONS 选项，后参数选项。

#### ■ DATA

指定数据文件路径，字符串类型。

一般情况下数据文件路径在控制文件中指定。DATA 参数值的优先顺序为先控制文件，后参数选项。如果控制文件中数据文件路径指定为 ‘\*’，在命令行通过 DATA 参数指定数据文件路径，DATA 所指定的文件路径会替换 ‘\*’。

#### ■ ENABLE\_CLASS\_TYPE

指定是否以 BLOB 方式载入或导出 CLASS 类型列数据，布尔类型，默认为 FALSE。

CLASS 类型是 DM 数据库服务器内部实现的数据类型，实际以 BLOB 类型存储。如果通过交互式工具或 DM 提供的接口等正常途径创建的 CLASS 类型数据，内部会转换成 BLOB 存储。而通过 dmfldr 直接导 CLASS 数据，没有进行转换，有可能出现载入的大对象数据无法转换成对应的 CLASS 类型。因此，当将 ENAME\_CLASS\_TYPE 设为 TRUE 时，用户要保证对应的 BLOB 数据能正确转换成对应的 CLASS 类型。

用户也可以通过设置控制文件中的 OPTIONS 选项来设置 ENAME\_CLASS\_TYPE 的值，参数值的优先顺序为先 OPTIONS 选项，后参数选项。

#### ■ FLUSH\_FLAG

指定提交数据时服务器的处理方式，布尔类型，默认为 FALSE。

该参数用于 dmfldr 向服务器发送 commit 请求时，是否要求服务器提供可靠的服务响应，若设置为 TRUE，则服务器会等数据写入磁盘后才将响应结果返回给 dmfldr，此种方式会降低数据装载的效率，但提供可靠的服务，不存在数据丢失的情况；若为 FALSE，则服务器在确认数据正确无误后便将响应结果返回，随后再写入磁盘，此种方式装载效率高，但若遇到掉电、机器故障、服务器崩溃等灾难性情况下，有可能会造成数据来不及写入磁盘而导致数据丢失。

用户可以通过设置控制文件中的 OPTIONS 选项来设置 FLUSH\_FLAG 的值，参数选项优先顺序为 OPTIONS 选项，后参数选项。

此参数为可选参数，作用于 MODE 为 IN 的情况下，当 MODE 为 OUT 时无效。

#### ■ SINGLE\_HLDR\_HP

是否使用单个 HLDR 装载 HUGE 水平分区表主表。取值 TRUE 或 FALSE，默认值为 FALSE。TRUE 表示使用单个 HLDR 装载 HUGE 水平分区表主表。FALSE 表示装载涉及到的每个叶子子表都各使用一个 HLDR。

HLDR 是服务器端装载 HUGE 表时用到的处理装置。SINGLE\_HLDR\_HP 的设置相当于指定了服务器装载 HUGE 水平分区主表时可以使用资源的模式。

TRUE 的方式更节约空间，FALSE 的方式装载速度更快，用户需要根据自己得需要权衡哪种方式更适合自己。

#### ■ IGNORE\_BATCH\_ERRORS

指定用户在数据装载过程中遇到数据错误时，是否忽略错误继续装载，还是报错返回，布尔类型，默认为 FALSE。

该参数只在 DIRECT 为 FALSE 时有效。在普通数据装载方式下，以绑定插入的方式向服务器发送数据，服务器检查 IGNORE\_BATCH\_ERRORS 参数，发现为 TRUE 时，将处理所有接受到的数据，如果有不合要求的数据，服务器保存该数据的错误信息，直到当前批次所有数据处理完毕后，再将执行的结果返回 dmfldr，dmfldr 根据服务器返回的错误信息向错误数据文件写入错误数据。当 IGNORE\_BATCH\_ERRORS 参数值为 FALSE 时，服务器一旦遇到不合要求的数据，则终止当前批次数据的装载，向 dmfldr 返回错误信息，dmfldr 则直接废弃当前批次的所有数据。

用户可以通过设置控制文件中的 OPTIONS 选项来设置 IGNORE\_BATCH\_ERRORS 的值，参数选项优先顺序为 OPTIONS 选项，后参数选项。

此参数为可选参数，作用于 MODE 为 IN 且 DIRECT 为 FALSE 的情况下，当 MODE 为 OUT 时无效。

#### ■ EP

指定 dmfldr 只能向指定的 MPP 站点发送数据，对于其他站点的数据直接丢弃。

该参数仅在 MPP 环境有效，dmfldr 只能向 EP 指定的站点上的表上锁并发送数据，其他站点的表操作与其无关。需要注意的是，指定 EP 方式对于随机分布表无效，并且 dmfldr 当前连接站点的数据发送不能由其他 dmfldr 客户端发送，换言之，客户端 A 连接 EP1，客户端 B 连接 EP2，那么 EP1 的表数据只能有 A 来发送，不能由 B 发送，否则会导致锁等待或锁超时错误。

用户可以通过开展文件中的 OPTIONS 选项来设置，OPTIONS 选项的优先级高于参数选项。

此参数为可选参数，在 MODE 为 IN 且 DIRECT 为 TRUE 的情况下有效。

#### ■ SQL

用户指定 SQL 查询语句，用于导出自定义查询结果数据，仅导出模式有效。

自定义 SQL 查询语句会将 SQL 语句发送到服务器进行分析执行，若发现 SQL 语句不是查询语句，dmfldr 将报错返回。SQL 语句可以实现用户的个性化数据导出需要，可以选择需要导出的列，需要过滤的结果。

用户可在控制文件中设置此参数值，命令行中难以正确设置此参数，因为一般 SQL 语句中都含有空格，而命令行解析参数是以空格作为分隔符的，因此完整的 SQL 语句可能会被错误的分割。

此参数为可选参数，在 MODE 为 OUT 的情况下有效。

■ **HELP**

获取帮助信息。

## 3dmfldr 实战

在上一章简单了解了 dmfldr 的各参数含义后，本章介绍如何通过设置不同参数值使用 dmfldr 进行不同情况的数据装载。并介绍了如何通过设置参数提高 dmfldr 的性能以及使用限制。



本章中皆以在 Linux 操作系统下进行操作为例，在 Windows 操作说明：系统下目录分隔符与单引号转义的处理有些许区别，不再赘述

### 3.1 dmfldr 控制文件

控制文件 CONTROL 是启动 dmfldr 必须要指定的参数，用于指定数据文件中数据的格式。在数据载入时，dmfldr 根据控制文件指定的格式来解析数据文件；导出数据时，dmfldr 也会根据控制文件指定的列分隔符、行分隔符等生成数据文件。

控制文件中还可以指定其他 dmfldr 参数值。

dmfldr 控制文件的语法如下所示：

---

```
[OPTIONS (  
  <id>=<value>  
  .....  
)]  
  
LOAD [DATA]  
  
INFILE  < <file_option>|<directory_option> >  
  
[BADFILE <path_name>]  
  
[APPEND|REPLACE|INSERT]  
  
<into_table_clause>  
  
<id> ::=参数  
  
<value> ::=值
```

---

<file\_option> ::= [LIST] <path\_name> [<row\_term\_option>] [, <path\_name>  
[<row\_term\_option>]]

<directory\_option> ::= DIRECTORY <path\_name> [<row\_term\_option>]

<path\_name> ::= 文件地址

<row\_term\_option> ::= STR [X] <delimiter>

<into\_table\_clause> ::= <into\_table\_single>{<into\_table\_single>}

<into\_table\_single> ::= INTO TABLE [<schema>.]<tablename>  
[EP <ep\_option>]  
[WHEN <field\_conditions>]  
[FIELDS [TERMINATED BY] [X] <delimiter>]  
[<enclosed\_option>]  
[<coldef\_option>]

<schema> ::= 模式名

<tablename> ::= 表名

<ep\_option> ::= (<ep\_list>)

<ep\_list> ::= 整型数字列表, 以逗号分隔

<field\_conditions> ::= <field\_condition>{ AND <field\_condition>}

<field\_condition> ::= [(] <cmp\_exp><cmp\_ops><cmp\_data>[)]

<cmp\_exp> ::= <colid> | (p1:p2)

<cmp\_ops> ::= = | <> | !=

<cmp\_data> ::= [X] '<字符串常量>' | BLANKS | WHITESPACE

<delimiter> ::= '<字符串常量>'

<coldef\_option> ::= (<col\_def>{ ,<col\_def>})

<col\_def> ::= <col\_id> [FILLER] [<property\_option>] [<fmt\_option>] [<term\_option>]  
[<enclosed\_option>] [<constant\_option>] [<fun\_option>]

<col\_id> ::= 列名

<property\_option> ::= <position\_option> | NULL

<position\_option> ::= position(p1:p2) | position(p1)

<fmt\_option> ::= DATE FORMAT '<时间日期格式串>'

<term\_option> ::= TERMINATED [BY] <wx\_option>

---



---

```
<wx_option> ::= WHITESPACE|[X] <delimiter>
```

```
<enclosed_option> ::= [OPTIONALLY] ENCLOSE [BY] [X] <delimiter>
```

```
<constant_option> ::= CONSTANT "<常量>"
```

```
<fun_option> ::= "函数名称()"
```

---

对于上述控制文件语法，需要说明的是：

- dmflldr 在处理数据文件中换行符时 windows 默认为 0x0D0A(\r\n)，非 windows 默认为 0x0A(\n)，用户应该根据现有的数据文件中的换行符做相应的调整。对应选项为<row\_term\_option>，若指定的<value>值为十六进制的字符串值需要指明[X]选项，<value>值不再需要以 0x 开头。若没有指明[X]选项，则<value>值为指定的字符串；
- 关于列分隔符，用户应当指定 FIELDS 或者 coldef\_option 中的至少一种。若两者均存在，则以 coldef\_option 中的设置为准，若分隔符指明[X]选项，则表明此分隔符为十六进制格式的字符串；
- 关于 file\_option，用来指定单个文件；
- 关于 directory\_option，用来指定整个文件夹。指定此选项后，dmflldr 会自动扫描指定文件夹下的所有文件，这些文件数据将被导入到服务器；
- 关于 LIST 选项，INFILE 使用 LIST 选项时，表明实际的数据文件路径存储在 INFILE 指定的文件中，该文件可以存储多个实际的数据文件路径，使用逗号或者换行分割；
- 关于 APPEND|REPLACE|INSERT 选项，表示将数据装载时采用的加载方式。INSERT，插入方式，向空表插入新记录（如果不是空表则会报错无效的装载模式）；APPEND，追加方式，为缺省方式，在表中追加新记录；REPLACE，替代方式，先清空表再插入新记录；

当 dmflldr 处于导出数据模式时，设置为 APPEND 时，dmflldr 会检查导出数据文件是否存在，若存在，则以追加的方式写入数据；若不存在，则新建数据文件；设置为其他值时，dmflldr 将直接创建新数据文件。选项默认值为 APPEND；

- 关于 OPTIONS 选项，该选项支持命令行参数中除 userid, control, help 以外的所有参数的指定，每个参数值对使用空格或者换行分割。对于 option 中出

现的参数，在 dmflldr 的指定执行参数中也出现的，dmflldr 会选择 option 中对应参数的值执行；

- 关于 col\_def, FILLER 表示跳过处理数据文件中指定列的值；
  - 关于 property\_option 选项
    - ✓ position(p1:p2): 从数据文件中每行数据的第 p1 个字节到第 p2 个字节为该列值，包含边界 p1,p2；
    - ✓ position(p1): 从数据文件中每行数据的第 p1 个字节开始，到下一个列分隔符之间的数据为该列值，包含边界 p1；
    - ✓ position 选项对大字段数据无效，若对大字段类型指定此选项会报错；
    - ✓ NULL: 指定的值为 NULL，忽略数据文件中的值；
- property\_option 参数仅对导入有效；
- 关于 term\_option 选项，该选项用来指定数据文件中指定列的结束标志。列的结束标志可以是 WHITESPACE（空格）或者用户自定义的字符串或十六进制串。指定了 term\_option 后，该列不需要用 FIELDS 分隔；
  - 关于 enclosed\_option 选项，此参数指定封闭符，为可选参数，默认不存在封闭符。若在 into\_table\_clause 和 coldef\_option 中均设置了封闭符，则以 coldef\_option 中的设置为准，若封闭符前指定 [X] 选项，则表明此封闭符为十六进制格式的字符串；
  - 分隔符或封闭符字符串的长度均不能超过 255 个字节；
  - 关于 constant\_option 选项，指定 constant 关键字后，数据文件中不需要为该列准备数据，如果指定了，该列数据将作为下一字段数据装载而导致数据混乱。constant 选项对大字段类型无效；
  - 关于 fun\_option 选项，目前只支持 trim() 和 replace(colname, srcStr, destStr) 函数。trim() 函数用于去除列数据的前后空格；replace() 函数用于将 colname 列名指定的列数据中的 srcStr 替换为 destStr，其中 srcStr 和 destStr 参数可使用 chr(int) 函数将数字转换成字符串，要求 chr() 的参数和返回值均不超过 4 个字节；
  - 关于 ep\_option 选项，用于指定数据将要发送的目的站点，仅适用于 MPP 环境。

例：一个 dmflldr 控制文件的例子

```
OPTIONS

(
    SKIP = 0

    ROWS = 50000

    DIRECT = TRUE

    INDEX_OPTION = 2
)

LOAD DATA

INFILE '/opt/data/test1.txt' STR X '0A'

BADFILE '/opt/data/test1.bad'

INTO TABLE test1

FIELDS '|'

(
    F1,

    F2 DATE FORMAT 'YYYY-MM-DD',

    F3 NULL,

    F4 TERMINATED BY WHITESPACE ENCLOSE BY '(',

    F5 CONSTANT "test",

    F6 "trim()"
)

LOAD DATA

INFILE '/opt/data/test2.txt' STR X '0A'

BADFILE '/opt/data/test2.bad'

INTO TABLE test2

FIELDS '|'

(
    C1 TERMINATED BY ' ',

    C2,

    C3 DATE FORMAT 'yyyy-mm-dd'
```

)

在这个例子中：

- OPTIONS 选项中定义了 SKIP、ROWS、DIRECT 和 INDEX\_OPTION 参数
- 有两个 LOAD 选项，表明有一次装载同时处理两个表，在每个 LOAD 选项中：
  - 指定了数据文件和数据文件的换行符，还指定了 BADFILE 文件
  - 指定了操作的数据库基表、列分隔符以及列定义

## 3.2 指定数据文件

当 dmflldr 工作在 IN 模式时，从数据文件中读取数据并载入数据库；当工作在 OUT 模式时，从数据库中将指定数据导出到数据文件。

数据文件通常为文本文件，列与列之间由列分隔符隔开，行与行之间由行分隔符隔开。数据文件中的列分隔符和行分隔符由用户指定，并在控制文件中设置为与数据文件中的一致。

### 3.2.1 在控制文件中指定数据文件

可以在控制文件的 LOAD 节点中指定数据文件。

例如：

1) 建表 TEST

```
DROP TABLE TEST;

CREATE TABLE TEST(C1 INT,C2 INT,C3 DATE);
```

2) 编辑数据文件 test.txt，存放路径为/opt/data/test.txt，文件内容如下

```
1 1|2015-11-06
2 2|2015-11-05
3 3|2015-11_04
```

3) 编辑控制文件 test.ctrl，存放路径为/opt/data/test.ctrl，内容如下：

```
LOAD DATA

INFILE '/opt/data/test.txt'

INTO TABLE test

FIELDS '|'

(
```

```
C1 TERMINATED BY ' ',
C2,
C3 DATE FORMAT 'yyyy-mm-dd'
)
```

#### 4) 使用 dmfldr 进行数据载入

```
./dmfldr userid=SYSDBA/SYSDBA@localhost:5236 control=\\'/opt/data/test.ctrl\\'
```

### 3.2.2 使用 DATA 参数指定数据文件

也可以使用 DATA 参数指定 dmfldr 的数据文件，数据文件路径的优先选择顺序为先控制文件，后参数选项。如果控制文件中数据文件路径指定为 ‘\*’，在命令行通过 DATA 参数指定数据文件路径，DATA 所指定的文件路径会替换 ‘\*’。

例如：

#### 1) 建表 TEST

```
DROP TABLE TEST;

CREATE TABLE TEST(C1 INT,C2 INT,C3 DATE);
```

#### 2) 编辑数据文件 test.txt，存放路径为/opt/data/test.txt，文件内容如下

```
1 1|2015-11-06
2 2|2015-11-05
3 3|2015-11_04
```

#### 3) 编辑控制文件 test.ctrl，存放路径为/opt/data/test.ctrl，内容如下：

```
LOAD DATA
INFILE *
INTO TABLE test
FIELDS '|'
(
C1 TERMINATED BY ' ',
C2,
C3 DATE FORMAT 'yyyy-mm-dd'
)
```

## 4) 使用 dmflldr 进行数据载入

```
./dmflldr userid=SYSDBA/SYSDBA@localhost:5236 control='/opt/data/test.ctrl\'  
data='/opt/data/test.txt \'
```

### 3.3 数据转换与错误数据文件

dmflldr 使用的数据文件都是文本格式的，其中的列值都是以字符串的方式保存在数据文件中。要想将这些数据载入数据库表中，需要将字符串转换成数据库表各列对应的数据类型。dmflldr 支持所有 DM 数据库支持的列定义类型，包括字符串、数值、时间日期、时间日期间隔、大字段类型等。

若数据文件的编码方式与 DM 数据库服务器的编码方式不一样，dmflldr 还需要进行字符编码的转换。dmflldr 支持 UTF8、GBK 和 GB18030 编码之间的相互转换。

数据类型和编码转换工作由 dmflldr 客户端进行，在这个过程中如果出现错误，dmflldr 会跳过该行继续后面的工作，并记录错误行到 BADFILE 指定的文件。常见的出错的情况有以下几种：

- 编码转换失败
- 目标列为字符串类型时，数据长度大于列定义长度
- 目标列为数值类型时，数据包含非法字符或者转换后超出该数值的范围
- 目标列为日期类型时，dmflldr 默认按 yyyy-mm-dd hh:mi:ss 的格式解析，如果数据不是这样的格式，需要指定对应列的时间日期 fmt 格式而未指定

dmflldr 错误数据的文件路径由 BADFILE 参数设置，默认的错误文件名为 flldr.bad。用户也可以通过设置控制文件中的 OPTIONS 选项来指定错误数据文件的路径，同时也可以控制文件的 LOAD 节点中指定错误数据文件的路径。错误数据文件路径最终值的优先选择顺序为 LOAD 节点选项，OPTIONS 选项，参数选项。用户可以同时对三种设置方式中的一个或多个设置，但最终的值只取一个。BADFILE 仅作用于 dmflldr 的工作 MODE 为 IN 的情况下，MODE 为 OUT 时无效。

当数据类型和编码转换中存在错误数据，而错误数在允许的最大错误数范围内时，dmflldr 会将出错的数据记录到错误数据文件中。文件记录的信息为执行程序、时间、目标表、数据文件中存在格式错误的行数据以及转换出错的行数据。

允许的最大容错个数由 ERRORS 选项设置，默认为 100。当 dmfldr 客户端在数据类型和编码转换过程中出现的错误个数超过了 ERRORS 所设置的数目，dmfldr 会停止载入，当前时间点的所有正确数据将会被提交到服务器端。如果载入过程中不允许出现错误则可以将 ERRORS 设置为 0；如果允许所有的错误出现，则可以将 ERRORS 设置为一个非常大的数。

例如：

#### 1) 建表 TEST

```
DROP TABLE TEST;

CREATE TABLE TEST(C1 INT,C2 INT,C3 DATE);
```

#### 2) 编辑数据文件 test.txt，存放路径为/opt/data/test.txt，文件内容如下

```
1 1|2015-11-06
2 2|2015-11-05
3 3|2015-11_04
44|aaaa-bbb-ccc
```

#### 3) 编辑控制文件 test.ctrl，存放路径为/opt/data/test.ctrl，内容如下：

```
LOAD DATA

INFILE '/opt/data/test.txt'

INTO TABLE test

FIELDS '|'

(

C1 TERMINATED BY ' ',

C2,

C3 DATE FORMAT 'yyyy-mm-dd'

)
```

#### 4) 使用 dmfldr 进行数据载入

```
./dmfldr userid=SYSDBA/SYSDBA@localhost:5236 control='/opt/data/test.ctrl'
badfile='/opt/data/test.bad'
```

#### 5) 查看错误数据文件/opt/data/test.bad，其内容如下

```
dmfldr: 2015-11-09 16:56:52 SYSDBA->TEST 4|4 aaaa-bbb-ccc
```

## 3.4 服务器端错误数据处理

dmfldr 客户端将载入的数据进行数据转换和编码转换后，将转换正确的数据发往 DM 服务器的 dmfldr 模块，也就是 dmfldr 的服务器端，由其进行真正的数据载入工作。

dmfldr 客户端每次向服务器端发送一批数据，在服务器端插入数据的过程中，由于目的表上可能存在约束等原因，导致某些数据无法插入成功，此时服务器端会将这一批数据全部回滚，并将这批数据全部记为错误数据，但服务器端插入时的错误数据并不会记录到 BADFILE 中。

ERRORS 所统计的错误包含在数据转换和数据插入过程中所产生的数据错误，因此当服务器端插入数据记录的错误数据数加上客户端数据转换时的错误数据数超过 ERRORS 参数的指定值时，dmfldr 服务器会停止插入数据。

## 3.5 大字段数据处理

dmfldr 支持对 DM 数据库的大字段类型数据的载入和导出，DM 数据库支持的大字段数据类型有 TEXT、LONGVARCHAR、IMAGE、LONGVARBINARY、BLOB 以及 CLOB。

### 3.5.1 大字段数据的导出

当 dmfldr 工作在导出模式即 MODE 为 OUT 时，dmfldr 生成大字段对应的数据文件名由 LOB\_FILE\_NAME 指定，若未指定默认为 dmfldr.lob，文件存放于 LOB\_DIRECTORY 指定的目录，如果未指定 LOB\_DIRECTORY 则存放于指定的导出数据文件同一目录。

例如：

#### 1) 建表 TEST

```
DROP TABLE TEST;  
  
CREATE TABLE TEST(C1 INT,C2 BLOB,C3 CLOB);
```

#### 2) 插入数据

```
INSERT INTO TEST VALUES(1,0XAB121032DE,'abcdefg');  
  
INSERT INTO TEST VALUES(2,0XAB121032DE,'abcdefg');  
  
COMMIT;
```



3) 编辑控制文件 test.ctrl, 存放路径为/opt/data/test.ctrl, 内容如下:

```
LOAD DATA
INFILE '/opt/data/test.txt'
INTO TABLE test
FIELDS '|'
(
C1,
C2,
C3
)
```

4) 使用 dmflldr 进行导出数据

```
./dmflldr userid=SYSDBA/SYSDBA@localhost:5236 control='/opt/data/test.ctrl'
LOB_DIRECTORY='/opt/data/' mode='out'
```

在这个例子中, 指定了 LOB\_DIRECTORY, 而没有指定 LOB\_FILE\_NAME, 导出的大字段数据文件将存放在 LOB\_DIRECTORY 指定的/opt/data 目录, 文件名为 dmflldr.lob。

### 3.5.2 DIRECT 为 TRUE 时大字段数据的载入

当 MODE 为 IN 且 DIRECT 为 TRUE 时, 此时数据载入若涉及到大字段对象, 需要用户指定大字段数据文件。若 CLIENT\_LOB 为 TRUE, LOB\_DIRECTORY 应指定大字段数据文件所在的客户端本地目录; 若 CLIENT\_LOB 为 FALSE, 用户必须先把相关文件传送到 DM 服务器所在主库, 然后使用 LOB\_DIRECTORY 指明存放目录

大字段数据文件在数据文件中指定, 可以是任意格式的文件。在数据文件中, 大字段以“文件名: 起始偏移: 长度”的形式记录在数据文件中。指定的文件名无效时, dmflldr 会报错, 装载失败。对于 CLOB 类型字段, 当指定的偏移、长度范围内带有不完整字符时, dmflldr 将装载失败。

例如:

1) 建表 TEST

```
DROP TABLE TEST;
```

```
CREATE TABLE TEST(C1 INT,C2 BLOB,C3 CLOB);
```

2) 编辑数据文件 test.txt, 存放路径为 DM 服务器所在主库的

/opt/data/test.txt, 文件内容如下

```
1|testblob.txt:0:10|testclob.txt:0:10
2|testblob.txt:10:20|testclob.txt:10:20
3|testblob.txt:20:30|testclob.txt:20:30
```

其中, testblob.txt、testclob.txt 为文本文件, 长度大于 30 字节, 存放路径为 /opt/data。

3) 编辑控制文件 test.ctrl, 存放路径为 /opt/data/test.ctrl, 内容如下:

```
LOAD DATA
INFILE '/opt/data/test.txt'
INTO TABLE test
FIELDS '|'
(
C1,
C2,
C3
)
```

4) 使用 dmfldr 进行导入数据

```
./dmfldr userid=SYSDBA/SYSDBA@localhost:5236 control='/opt/data/test.ctrl'
LOB_DIRECTORY='/opt/data/'
```

### 3.5.3 DIRECT 为 FALSE 时大字段数据的载入

当 MODE 为 IN 且 DIRECT 为 FALSE 时, 数据文件中大字段列数据即字段内容。

BLOB\_TYPE 参数指定 BLOB 列内容为十六进制或者字符串:

- BLOB\_TYPE 为 HEX\_CHAR 时, 数据文件中 BLOB 列当作为十六进制内容;
- BLOB\_TYPE 为 HEX 时, 数据文件中 BLOB 列为字符串形式内容, 导入后会转换为十六进制。

BLOB\_TYPE 参数只对 DIRECT 为 FALSE 时有效, 默认为 HEX\_CHAR。

例 1:

1) 建表 TEST

```
DROP TABLE TEST;

CREATE TABLE TEST(C1 INT,C2 BLOB,C3 CLOB);
```

2) 编辑数据文件 test.txt, 存放路径为/opt/data/test.txt, 文件内容如下

```
1|0x12d3c8a7|abcdefg
2|0x12a4cbac|hijklmn
3|0x22d3c8b3|defhjd
```

3) 编辑控制文件 test.ctrl, 存放路径为/opt/data/test.ctrl, 内容如下:

```
LOAD DATA

INFILE '/opt/data/test.txt'

INTO TABLE test

FIELDS '|'

(

C1,

C2,

C3

)
```

4) 使用 dmflldr 进行导入数据, BLOB\_TYPE 为 HEX\_CHAR

```
./dmflldr userid=SYSDBA/SYSDBA@localhost:5236 control='/opt/data/test.ctrl\'
direct=false blob_type='hex_char\'
```

5) 查询表数据

```
SELECT * FROM TEST;
```

行号	C1	C2	C3
1	1	0x12D3C8A7	abcdefg
2	2	0x12A4CBAC	hijklmn
3	3	0x22D3C8B3	defhjd

例 2:

步骤 1)、2)、3) 同例 1

## 4) 使用 dmflldr 进行导入数据, BLOB\_TYPE 为 HEX

```
./dmflldr userid=SYSDBA/SYSDBA@localhost:5236 control='/opt/data/test.ctrl\'
direct=false blob_type='hex\'
```

## 5) 查询表数据

```
select * from test;
```

行号	C1	C2	C3
1	1	0x30783132643363386137	abcdefg
2	2	0x30783132613463626163	hijklmn
3	3	0x30783232643363386233	adefhjd

### 3.6 日志文件及日志信息

dmflldr 的日志文件路径由 LOG 参数设置, 默认日志文件名为 flldr.log。文件记录了装载过程中的装载信息和错误信息以及统计信息。用户也可以通过设置控制文件中的 OPTIONS 选项来指定日志路径。如果参数及 OPTION 中同时指定了日志路径则其将以 OPTION 中指定的路径为最终路径。

例如:

## 1) 建表 TEST

```
DROP TABLE TEST;

CREATE TABLE TEST(C1 INT,C2 INT,C3 DATE);
```

## 2) 编辑数据文件 test.txt, 存放路径为/opt/data/test.txt, 文件内容如下

```
1|1 2018-11-06
2|2 2018-11-05
3|3 2018-11_04
```

## 3) 编辑控制文件 test.ctrl, 存放路径为/opt/data/test.ctrl, 内容如下:

```
LOAD DATA

INFILE '/opt/data/test.txt'

INTO TABLE test

FIELDS '|'
```

```
(
C1,
C2 TERMINATED BY ' ',
C3 DATE FORMAT 'yyyy-mm-dd'
)
```

#### 4) 使用 dmfldr 进行数据载入。

```
./dmfldr userid=SYSDBA/SYSDBA@localhost:5236 control='/opt/data/test.ctrl\'
log='/opt/data/test.log\'
```

#### 5) 查看日志文件/opt/data/test.log，其内容如下：

```
dmfldr: 2018-12-03 13:46:33 dmfldr:
Copyright (c) 2011, 2015, Dameng. All rights reserved.
控制文件:
加载行数:全部
每次提交服务器行数:50000
跳过行数:0
允许错误数:100
是否直接加载:Yes
是否插入自增列:No
数据是否已按照聚集索引排序:No
字符集:GBK
dmfldr: 2018-12-03 13:46:33
数据文件共 1 个:
d:\test.txt
错误文件:fldr.bad
目标表:TEST
```

列名	包装数据类型	终止
C1	CHARACTER	
C2	CHARACTER	WHT
C3	yyyy-mm-dd	

行缓冲区数量: 4

```
任务线程数量: 4  
dmfldr: 2018-12-03 13:46:33  
目标表:TEST  
3 行加载成功。  
由于数据格式错误,0 行 丢弃。  
由于数据错误,0 行 没有加载。  
跳过的逻辑记录总数:0  
读取的逻辑记录总数:3  
拒绝的逻辑记录总数:0  
用时:20.522 (ms)
```

### 3.7 自增列装载

自增列是比较特殊的列，为了保证数据库中自增列列值的正确性，用户在进行数据载入时需要特别注意。

当 DIRECT 参数为 FALSE 时，dmfldr 将把从数据文件中读取的自增列值作为目标值插入数据库表中，用户应当保证每一行的自增列的值符合自增列的规则，否则将造成数据混乱。

当 DIRECT 参数为 TRUE 时，dmfldr 提供了 SET\_IDENTITY 参数（默认为 FALSE）对数据载入时自增列的处理进行设置：

- ✓ 如果指定 SET\_IDENTITY 选项值为 TRUE，则 dmfldr 将把从数据文件中读取的自增列值作为目标值插入数据库表中，用户应当保证每一行的自增列的值符合自增列的规则，否则将造成数据混乱；
- ✓ 如果 SET\_IDENTITY 选项值设置为 FALSE，则 dmfldr 将忽略数据文件中对应自增列的值，服务器将根据自增列定义和表中已有数据自动生成自增列的值插入每一行的对应列。

例 1：

1) 建表 TEST，并插入两行数据

```
DROP TABLE TEST;  
  
CREATE TABLE TEST(C1 INT IDENTITY(1,1),C2 VARCHAR);
```

```
INSERT INTO TEST(C2) VALUES('AAA');

INSERT INTO TEST(C2) VALUES('BBB');

COMMIT;
```

此时表 TEST 中的数据为:

行号	C1	C2
1	1	AAA
2	2	BBB

2) 编辑数据文件 test.txt, 存放路径为/opt/data/test.txt, 文件内容如下

```
2|aaa
3|bbb
4|ccc
```

3) 编辑控制文件 test.ctrl, 存放路径为/opt/data/test.ctrl, 内容如下:

```
LOAD DATA

INFILE '/opt/data/test.txt'

INTO TABLE test

FIELDS '|'

(

C1,

C2

)
```

4) 使用 dmfldr 进行数据载入

```
./dmfldr userid=SYSDBA/SYSDBA@localhost:5236 control='/opt/data/test.ctrl'
direct=true set_identity=false
```

5) 查看表 TEST 的数据

```
SELECT * FROM TEST;
```

行号	C1	C2
1	1	AAA
2	2	BBB

3	3	aaa
4	4	bbb
5	5	ccc

在这个例子中，表 TEST 中已有两行数据，由于 SET\_IDENTITY 置为 FALSE，因此在数据载入时 dmfldr 根据 C1 列的定义和表中已有数据，为 C1 列重新插入合适的值。

我们再看看如果将 SET\_IDENTITY 置为 TRUE 结果会怎样。

例 2:

重复例 1 的 1)、2)、3) 步骤

4) 使用 dmfldr 进行数据载入

```
./dmfldr userid=SYSDBA/SYSDBA@localhost:5236 control='/opt/data/test.ctrl\'
direct=true set_identity=true
```

5) 查看表 TEST 的数据

```
SELECT * FROM TEST;
```

行号	C1	C2
1	1	AAA
2	2	BBB
3	2	aaa
4	3	bbb
5	4	ccc

## 3.8 数据排序

SORTED 参数用来设置数据是否已经按照聚集索引排序，默认为 FALSE。

如果设置为 TRUE，则用户必须保证数据已按照聚集索引排序完成，并且如果表中存在数据，需要插入的数据索引值要比表中数据的索引值大，服务器在做插入操作时顺序进行插入。若数据并未按照索引排序，则 dmfldr 会报错，装载失败。

如果设置为 FALSE，则服务器对于每条记录进行定位插入。



用户也可以通过设置控制文件中的 OPTIONS 选项来设置 SORTED 的值。SORTED 参数值的优先选择顺序为 OPTIONS 选项，参数选项。此参数为可选参数，作用于 MODE 为 IN 且 DIRECT 为 TRUE 的情况下，对于其他情况此参数无效。

在数据量大，并且确定数据已按照聚集索引排序完成的情况下，将 SORTED 参数设置为 TRUE，可以提升装载性能。

例如：

1) 建表 TEST

```
DROP TABLE TEST;  
  
CREATE TABLE TEST(C1 INT CLUSTER PRIMARY KEY,C2 VARCHAR);
```

2) 编辑数据文件 test.txt，存放路径为/opt/data/test.txt，文件内容如下

```
2|aaa  
3|bbb  
4|ccc  
5|ddd  
1|zzz
```

3) 编辑控制文件 test.ctrl，存放路径为/opt/data/test.ctrl，内容如下：

```
LOAD DATA  
  
INFILE '/opt/data/test.txt'  
  
INTO TABLE test  
  
FIELDS '|'   
  
(  
  
C1,  
  
C2  
  
)
```

4) 使用 dmfldr 进行数据载入

```
./dmfldr userid=SYSDBA/SYSDBA@localhost:5236 control=\'/opt/data/test.ctrl\'  
sorted=true
```

由于本例中数据文件中的数据并没有按照 C1 列排序，dmfldr 将会报错。

不能使用 NOSORT 选项，数据非有序

### 3.9 空值处理

dmflldr 通过设置 NULL\_MODE 参数来处理空值。

- 设置为 TRUE，载入时 NULL 字符串处理为 NULL，载出时空值处理为 NULL 字符串
- 设置为 FALSE，载入时 NULL 字符串处理为字符串，载出时空值处理为空串

例 1:

1) 建表 TEST

```
DROP TABLE TEST;

CREATE TABLE TEST(C1 INT,C2 VARCHAR);
```

2) 编辑数据文件 test.txt，存放路径为/opt/data/test.txt，文件内容如下

```
1|aaa
2|NULL
3|null
```

3) 编辑控制文件 test.ctrl，存放路径为/opt/data/test.ctrl，内容如下:

```
LOAD DATA
INFILE '/opt/data/test.txt'
INTO TABLE test
FIELDS '|'
(
C1,
C2
)
```

4) 使用 dmflldr 进行数据载入

```
./dmflldr userid=SYSDBA/SYSDBA@localhost:5236 control='/opt/data/test.ctrl\'
null_mode=true
```

5) 查看表 TEST 的数据，数据文件中 C2 列的“NULL”和“null”字符串都被处理为空值

```
SELECT C1, IFNULL(C2, 'NULL VALUE') FROM TEST;
```

行号	C1	IFNULL(C2, 'NULLVALUE')
----	----	-------------------------

```

-----
1          1          aaa
2          2          NULL VALUE
3          3          NULL VALUE

```

例 2:

步骤 1)、2)、3) 与例 1 相同

4) 使用 dmflldr 进行数据载入

```

./dmflldr userid=SYSDBA/SYSDBA@localhost:5236 control=\\'/opt/data/test.ctrl\\'
null_mode=false

```

5) 查看表 TEST 的数据，数据文件中 C2 列的“NULL”和“null”字符串直接被作为字符串值插入表中

```
SELECT C1, IFNULL(C2, 'NULL VALUE') FROM TEST;
```

```

行号      C1          IFNULL(C2, 'NULLVALUE')
-----

```

```

1          1          aaa
2          2          NULL
3          3          null

```

### 3.10 类类型装载

dmflldr 支持对 CLASS 类型数据的装载。CLASS 类型装载和 LOB 类型一样，载入时需要指定 LOB\_DIRECTORY，另外需要指定 ENABLE\_CLASS\_TYPE 为 TRUE；导出时，默认导出目录和数据文件所在目录一致，导出类类型形成的大字段文件默认为 dmflldr.lob，导出目录和大字段文件可通过 LOB\_DIRECTORY 和 LOB\_FILE\_NAME 参数设置。



**CLASS 类型大字段数据文件无法通过手动创建，只有从表中导出到大字段文件中，并使用数据文件 CLASS 字段对应的偏移、字长进行导入。**

例如：

1) 建表 TEST

数据准备，创建类 mycls 的类头与类体，之后创建表 TEST，其 C2 列类型为 mycls 类类型，并向 TEST 中插入两行数据。

```
--类头创建

CREATE CLASS mycls
AS

TYPE rec_type IS RECORD (c1 INT, c2 INT);      --类型声明

id      INT;                                --成员变量

r       rec_type;                          --成员变量

FUNCTION f1(a INT, b INT) RETURN rec_type;    --成员函数

FUNCTION mycls(id INT , r_c1 INT, r_c2 INT) RETURN mycls;--用户自定义构造函数

END;

/

--类体创建

CREATE OR REPLACE CLASS BODY mycls
AS

FUNCTION f1(a INT, b INT) RETURN rec_type
AS
BEGIN

    r.c1 = a;

    r.c2 = b;

    RETURN r;

END;

FUNCTION mycls(id INT, r_c1 INT, r_c2 INT) RETURN mycls
AS
BEGIN

    this.id = id;        --可以使用 this.来访问自身的成员

    r.c1 = r_c1;         --this 也可以省略

    r.c2 = r_c2;

    RETURN this;        --使用 return this 返回本对象
```

```

END;

END;

/

--建表 TEST

DROP TABLE TEST;

CREATE TABLE TEST(C1 INT,C2 mycls);

--插入数据:

INSERT INTO TEST VALUES(1,mycls(1,1,1));

INSERT INTO TEST VALUES(2,mycls(2,2,2));

COMMIT;

```

2) 编辑控制文件 test.ctrl, 存放路径为/opt/data/test.ctrl, 内容如下:

```

LOAD DATA

INFILE '/opt/data/test.txt'

INTO TABLE test

FIELDS '|'

(

C1,

C2

)

```

3) 使用 dmfldr 进行数据导出

```

./dmfldr userid=SYSDBA/SYSDBA@localhost:5236 control='/opt/data/test.ctrl'
mode='out' lob_directory='/opt/data'

```

4) 在指定的 LOB\_DIRECTORY 目录会生成大字段文件 dmfldr.lob, 在指定的数据文件路径生成的数据文件内容如下:

```

1|dmfldr.lob:0:70

2|dmfldr.lob:70:70

```

5) 创建一张新表 TEST2, 表的两列都为 mycls 类类型

```

CREATE TABLE TEST2(C1 MYCLS,C2 MYCLS);

```

- 6) 编辑数据文件 test2.txt，存放路径为/opt/data/test2.txt，文件内容如下：

```
dmfldr.lob:70:70|dmfldr.lob:0:70
dmfldr.lob:0:70|dmfldr.lob:70:70
```

- 7) 编辑控制文件 test2.ctrl，存放路径为/opt/data/test2.ctrl，内容如下：

```
LOAD DATA
INFILE '/opt/data/test2.txt'
INTO TABLE test2
FIELDS '|'
(
C1,
C2
)
```

- 8) 使用 dmfldr 进行数据导入

```
./dmfldr userid=SYSDBA/SYSDBA@localhost:5236 control='/opt/data/test2.ctrl\'
lob_directory='/opt/data/' enable_class_type=true
```

- 9) 查看表 TEST2 的数据

行号	C1	C2
1	SYSDBA.MYCLS(2,REC_TYPE(2,2))	SYSDBA.MYCLS(1,REC_TYPE(1,1))
2	SYSDBA.MYCLS(1,REC_TYPE(1,1))	SYSDBA.MYCLS(2,REC_TYPE(2,2))

### 3.11 条件过滤

通过在控制文件中指定 WHEN <field\_conditions>子句，可以在装载过程中对数据进行过滤，符合 field\_conditions 条件的数据才会被装载。

对于条件过滤的使用需注意以下几点：

- 判断条件中的操作符仅支持比较相等和不相等，即=、!=和<>这三个比较操作符；
- 目前仅支持使用 AND 连接多个过滤条件；
- BLANKS 和 WHITESPACE 表示若干个空格；

- 判断条件若使用 (p1:p2) 作为比较表达式，其意义与在 POSITION 子句中的意义相同，表示从该行指定位置获取数据进行比较，起始位置和结束位置表示的都是字节位置，包含边界 p1,p2；
- 如果判断条件中使用 colid 作为比较表达式，该列必须在 INTO 表的 coldef\_option 中进行说明；
- 如果判断条件中使用 colid 作为比较表达式，判断条件中使用的列仅用于过滤，并没有对应表中的某个实际列，应在 col\_def 中指明 FILLER 属性表示装载时跳过该列；
- 如果判断条件中比较数据是字符常量值，其长度小于比较表达式长度，则在其之后补充空格；如果判断条件中比较数据是二进制串常量，其长度小于比较表达式长度，则在之后补充 0。

例如：

1) 建表 TEST

```
DROP TABLE TEST;

CREATE TABLE TEST(C1 INT,C2 INT);
```

2) 编辑数据文件 test.txt，存放路径为/opt/data/test.txt，文件内容如下

```
12
23
32
48
91
```

3) 编辑控制文件 test.ctrl，存放路径为/opt/data/test.ctrl，内容如下：

```
LOAD DATA
INFILE '/opt/data/test.txt'
INTO TABLE test
WHEN C1 != '2'
(
C1 position (1:1),
C2 position (2:2)
)
```

## 4) 使用 dmfldr 进行数据载入

```
./dmfldr userid=SYSDBA/SYSDBA@localhost:5236 control='/opt/data/test.ctrl\'
```

## 5) 查看表 TEST 的数据，可以看到数据文件中的行 2,3 被过滤掉了。

```
SELECT * FROM TEST;
```

行号	C1	C2
1	1	2
2	3	2
3	4	8
4	9	1

### 3.12 多表装载

通过在控制文件中指定多个 INTO TABLE 子句，可以将一批数据同时向多个表进行装载。每个 INTO TABLE 子句中都可以指定 WHEN 过滤条件、FIELDS 子句和列定义子句。

对于多表装载的使用需注意以下几点：

- 每个 INTO TABLE 子句的目标表必须是不同的表；
- 多表装载时不支持直接装载分区表子表；
- 对于第二个及其之后的 INTO TABLE 子句，在其 coldef\_option 中，必须为第一列指定 POSITION 选项；

例如：

## 1) 建表 TEST1、TEST2

```
DROP TABLE TEST1;

DROP TABLE TEST2;

CREATE TABLE TEST1(C1 INT,C2 INT);

CREATE TABLE TEST2(C1 INT,C2 INT);
```

## 2) 编辑数据文件 test.txt，存放路径为/opt/data/test.txt，文件内容如下

```
1,2

2,3

3,2
```



4,8

9,1

3) 编辑控制文件 test.ctrl, 存放路径为/opt/data/test.ctrl, 内容如下:

```
LOAD DATA
INFILE '/opt/data/test.txt'
INTO TABLE test1
WHEN C1 != '1'
FIELDS ','
(
  c1 position (1:1),
  c2
)
INTO TABLE test2
WHEN (3:3) = '2' AND c1 != '3'
FIELDS ','
(
  c1 position (1:1),
  c2
)
```

4) 使用 dmflldr 进行数据载入

```
./dmflldr userid=SYSDBA/SYSDBA@localhost:5236 control=\\'/opt/data/test.ctrl\\'
```

5) 查看表 TEST1 和 TEST2 的数据如下

```
SELECT * FROM TEST1;
```

行号	C1	C2
1	2	3
2	3	2
3	4	8
4	9	1

```
SELECT * FROM TEST2;
```

行号	C1	C2
1	1	2

### 3.13 个性化设置

用户通过设置 dmfldr 的 SKIP、LOAD、ROWS 参数，可以根据自己的需求调整装载的起始行、装载最大行数以及每次提交的行数。

SKIP 参数用来设置跳过数据文件起始的逻辑行数，整形数值。默认的跳过起始行数为 0 行。如果用户指定了多个文件，且起始文件中的行数不足 SKIP 所指定的行数，则 dmfldr 工具会扫描下一个文件直至累加的行数等于 SKIP 所设置的行数或者所有文件都已扫描结束。

LOAD 参数用来设置装载的最大行数，整形数值。默认的最大装载行数为数据文件中的所有行数。LOAD 指定的值不包括 SKIP 指定的跳过的行数。

ROWS 参数用来设置每次提交的行数，整形数值。默认的提交行数为 50000 行。提交行数的值表示提交到服务器的行数，并不一定代表按照数据文件中的数据顺序的行数。用户可以根据实际情况调整每次提交的行数，以达到性能的最佳点。ROWS 参数作用于 MODE 为 IN 的情况下，当 MODE 为 OUT 时无效。

例如：

#### 1) 建表 TEST

```
DROP TABLE TEST;  
  
CREATE TABLE TEST(C1 INT,C2 VARCHAR);
```

#### 2) 编辑数据文件 test.txt，存放路径为/opt/data/test.txt，文件内容如下

```
1|aaa  
2|bbb  
3|ccc  
4|ddd  
5|eee  
6|fff  
7|ggg
```

```
8|hhh
9|iii
10|jjj
```

3) 编辑控制文件 test.ctrl，存放路径为/opt/data/test.ctrl，内容如下：

```
LOAD DATA
INFILE '/opt/data/test.txt'
INTO TABLE test
FIELDS '|'
(
C1,
C2
)
```

4) 使用 dmfldr 进行数据载入

```
./dmfldr userid=SYSDBA/SYSDBA@localhost:5236 control='/opt/data/test.ctrl\'
skip=3 load=5
```

5) 查看表 TEST 的数据，载入时跳过了数据文件的前 3 行数据，且只载入了 5 条数据。

```
SELECT * FROM TEST;
```

行号	C1	C2
1	4	ddd
2	5	eee
3	6	fff
4	7	ggg
5	8	hhh

### 3.14 主备切换时的数据继续载入

在 DM 数据守护主备环境下，如果 dmfldr 数据载入中发生主备切换，dmfldr 支持主备切换完成后自动继续装载数据，并且能保持数据正确。

要使用此项功能，USERID 参数需要使用主备服务名方式进行配置，如：

```
dmfldr USERID=SYSDBA/SYSDBA@dw CONTROL='c:\fldr.ctl'
```

同时在 dm\_svc.conf 配置文件中配置主备服务名，如：

```
dw=(192.168.0.101:5236, 192.168.0.102:5236)
```

目前主备切换时的数据继续载入功能还存在以下功能限制：

- 目前仅支持单机的主备，不支持 MPP 主备。

但是，若在 MPP 主备环境中使用 MPP\_CLIENT=FALSE，等同于单机，也是支持的。

- 目前不支持分区表装载

### 3.15MPP 本地分发

MPP\_CLIENT 参数用来设置在 DM MPP 环境下使用 dmfldr 进行数据装载时的数据分发方式。

当 DM 数据库服务器环境为单站点时此参数无效。当服务器环境为 MPP 环境时，若 MPP\_CLIENT 为 TRUE，dmfldr 采用客户端分发模式；若 MPP\_CLIENT 为 FALSE，则采用本地分发模式。

客户端分发模式下，数据在 dmfldr 客户端分发好后直接往指定站点发送数据。

本地分发模式下，导入时，dmfldr 直接将数据发送到连接的站点，数据最终在连接的站点；导出时，dmfldr 只导出连接站点的数据。

MPP 环境下要配置 dmmal.ini 文件中的 MAL\_INST\_HOST 和 MAL\_INST\_PORT 参数。

例如：

当前 DM 数据库服务器环境为 MPP 环境，两个节点，SEQNO 号分别为 0、1。使用 dmfldr 采用本地分发方式进行数据装载，连接的节点为 SEQNO 为 0 的节点。

#### 1) 建表 TEST

```
DROP TABLE TEST;

CREATE TABLE TEST(C1 INT,C2 INT);
```

#### 2) 编辑数据文件 test.txt，存放路径为/opt/data/test.txt，文件内容如下

```
1|1
2|2
```

```

3|3
4|4
5|5
6|6
7|7
8|8
9|9
10|10

```

3) 编辑控制文件 test.ctrl, 存放路径为/opt/data/test.ctrl, 内容如下:

```

LOAD DATA

INFILE '/opt/data/test.txt'

INTO TABLE test

FIELDS '|'

(

C1,

C2

)

```

4) 使用 dmfldr 进行数据载入

```

./dmfldr userid=SYSDBA/SYSDBA@localhost:5236 control='/opt/data/test.ctrl\'
mpp_client=false

```

5) 查询 SEQNO 为 0 的节点上 TEST 表的数据

```
SELECT * FROM TEST WHERE SF_GET_EP_SEQNO(ROWID)=0;
```

行号	C1	C2
1	1	1
2	2	2
3	3	3
4	4	4
5	5	5
6	6	6

7	7	7
8	8	8
9	9	9
10	10	10

6) 查询 SEQNO 为 1 的节点上 TEST 表的数据

```
SELECT * FROM TEST WHERE SF_GET_EP_SEQNO(ROWID)=0;
```

未选定行

## 3.16 提升 dmfldr 性能

用户在使用 dmfldr 时根据系统和数据的具体情况对一些参数进行调整，可以获得更好的性能，这些参数包括：

### ■ BUFFER\_NODE\_SIZE

BUFFER\_NODE\_SIZE 设置读取文件缓冲区页大小，值越大，缓冲区的页越大，每次读取的数据就越多，每次发送到服务器的数据也就越多，效率越高。但其大小受 dmfldr 客户端内存大小限制。

### ■ READ\_ROWS

在某些情况下，BUFFER\_NODE\_SIZE 读入的数据行数很大，而后续操作处理不了这么大的行数，此时可以用 READ\_ROWS 来限制处理的行数。dmfldr 取 READ\_ROWS 和 BUFFER\_NODE\_SIZE 中较小的值作为一次处理的行数。

### ■ SEND\_NODE\_NUMBER

指定 dmfldr 在数据载入时发送节点的个数，默认由系统计算一个初始值。若在数据载入时发现发送节点不够用，系统会动态增加分配。在系统内存足够的情况下，可以适当设大 SEND\_NODE\_NUMBER 值，提升 dmfldr 载入性能。

### ■ TASK\_THREAD\_NUMBER

指定 dmfldr 在数据载入时处理用户数据的线程数目。默认情况下，dmfldr 将该参数值设为系统 CPU 的个数，但当 CPU 个数大于 8 时，默认值都被置为 8。在 dmfldr 客户端所在机器 CPU 大于 8 环境中，提高 TASK\_THREAD\_NUMBER 值可以提升 dmfldr 装载性能。

### ■ BLDR\_NUM

水平分区表装载时，指定服务器 BLDR 的最大个数，默认为 64。

服务器的 BLDR 保存水平分区子表相关信息，BLDR\_NUM 的设置也就指定了服务器能同时载入的水平分区子表的个数。若 BLDR\_NUM 设置太大，当水平分区子表数过多时，可能会导致服务器内存不足。当载入时实际需要的 BLDR 个数超出 BLDR\_NUM 设置时，会淘汰指定子表的 BLDR，并替换为新的子表 BLDR。

#### ■ BDTA\_SIZE

BDTA (Batch Data) 的大小，默认为 5000。

BDTA 代表 DM 数据库批量数据处理机制中一个批量，在内存、CPU 允许的条件下，增大 BDTA\_SIZE 能加快装载速度；在网络是装载性能瓶颈时，增大 BDTA\_SIZE 影响不大。

#### ■ INDEX\_OPTION

索引的设置选项，默认为 1。INDEX\_OPTION 的可选项有 1、2 和 3。

1 代表服务器装载数据时先不刷新二级索引，而是将新数据按照索引预先排序，在装载完成后，再将排好序的数据插入索引。如果在数据载入前，目标表中已有较多数据，建议 INDEX\_OPTION 置为 1。

2 代表服务器在快速装载过程中不刷新二级索引数据，只在装载完成时重建所有二级索引。如果在数据载入前，目标表中没有数据或数据量较小，建议 INDEX\_OPTION 置为 2。

3 代表服务器使用追加模式来进行二级索引的插入，在数据装载的过程中，同时进行二级索引的插入，当原有数据量远大于插入数据量时，建议 INDEX\_OPTION 置为 3。

### 3.17 dmflldr 使用限制

dmflldr 的使用存在以下一些限制：

- 不支持向临时表、外部表装载数据
- 不支持向系统表装载数据
- 不支持向带有位图索引的表装载数据
- 不支持向带有函数索引的表装载数据
- 不支持向带有全文索引的表装载数据
- 不支持向 DCP 代理装载数据
- dmflldr 装载时，对约束进行检查，对各种约束的处理机制如下表所示

表 3.1 dmflldr 的约束检查机制

约束	数据不满足时	数据插入与否	约束是否有效
非空约束 (NOT NULL)	报错	不插入	有效
聚集索引 (CLUSTER PRIMARY KEY)	报错	不插入	有效
唯一约束 (UNIQUE, PRIMARY KEY)	报错	插入	失效
引用约束 (FOREIGN KEY)	不报错	插入	有效
CHECK 约束 (CHECK)	不报错	插入	有效



咨询热线：400-991-6599

技术支持：dmtech@dameng.com

官网网址：www.dameng.com



**武汉达梦数据库有限公司**  
**Wuhan Dameng Database Co.,Ltd.**

地址：武汉市东湖新技术开发区高新大道999号未来科技大厦C3栋16—19层

16th-19th Floor, Future Tech Building C3, No.999 Gaoxin Road, Donghu New Tech Development Zone,Wuhan,Hubei Province,China

电话：(+86) 027-87588000 传真：(+86) 027-87588810

---