

使用之前，请确保系统中 \$PATH 环境变量中支持以下命令：

- mysqldump
- mysql
- etcdctl
- redis-cli

一、mysql 数据库之间的迁移

1、用法示例

```
./datamover mysql -h
Realize data migration commands between isomorphic mysql, support single-
threaded mode and multi-threaded mode

Usage:
  datamover mysql [flags]
  datamover mysql [command]

Available Commands:
  dump          dump data from mysql database
  online        migrate mysql database online from one mysql to another
  restore       restore data target mysql database

Flags:
  -f, --from string      from mysql connection string (default
"root:root@tcp(localhost:3306)")
  -h, --help             help for mysql
  -t, --target string    target mysql connection string (default
"root:root@tcp(localhost:3306)")
  -T, --thread          whether target enable multi-threaded mode

Use "datamover mysql [command] --help" for more information about a command.
```

2、命令行中的 flags 解释

标志全称	标志 简称	标志 类型	默认值	适用子命令域	解释说明
--from	-f	string	"root:root@tcp(localhost: 3306)"	datamover mysql (dump or online)	source database 连接串
--target	-t	string	"root:root@tcp(localhost:	datamover mysql (restore	target database 连接串

			3306)"	or online)	
--thread	-T	bool	false	datamover mysql (dump or restore)	是否开启多线程模式
--databases	-d	string	""	datamover mysql (全部子命令)	mysql 数据库名称
--output	-o	string	""	datamover mysql dump	要输出的文件或目录, 可以省略
--input	-i	string	""	datamover mysql restore	数据库恢复所需要的输入的文件或目录
--all-databases	-a	bool	false	datamover mysql (dump or online)	mysql 全部的数据库, 系统除外

3、mysql 数据库的导出

- 从源数据库导出sql文件, --output or -o 指明输出的文件目录, 当不指明时, 系统会自动生成相应的文件目录

```
$ ./datamover mysql dump -h
dump data from mysql database

Usage:
  datamover mysql dump [flags]

Flags:
  -a, --all-databases      all mysql databases
                           except(mysql|sys|performance_schema|information_schema)
  -d, --databases stringArray  the dump databases of mysql
  -h, --help                help for dump
  -o, --output string        the location that save the dump file or
                           directory

Global Flags:
  -f, --from string        from mysql connection string (default
                           "root:root@tcp(localhost:3306)")
  -t, --target string      target mysql connection string (default
                           "root:root@tcp(localhost:3306)")
  -T, --thread              whether target enable multi-threaded mode
```

- 1) 、单线程导出 sql文件, 只能支持一个database的导出, 用 flag --databases or -d 来指明具体的数据库名, 用法如下:

```
/datamover mysql dump --from "root:root@tcp(localhost:3306)" -d gep (-o
gep.sql)
```

- 2) 单线程导出全部数据库, 除了系统数据库不导出, 系统的数据库包括 mysql, sys, performance_schema, information_schema

```
/datamover mysql dump --from "user:password@tcp(host:port)" -a (-o all-databases.sql)
```

上述命令中，没有用 `-o` 指定的输出文件，系统会默认保存在 `all-databases.sql` 文件中

3)、多线程导出 sql 文件到目录，可以支持多个数据库的导出，多个每次都用 `-d` 指明，多线程模式下一定要加上 `-T` flag，用法如下：

```
/datamover mysql dump --from "root:root@tcp(localhost:3306)" -d gep -d exer -d safe (-o gep_exer_safe) -T
```

4)、多线程导出全部数据库，用法如下：

```
./datamover mysql dump --from "user:password@tcp(host:port)" -a -T (-o all-databases)
```

4、mysql 数据库的导入

```
$ ./datamover mysql restore -h
restore data target mysql database

Usage:
  datamover mysql restore [flags]

Flags:
  -h, --help            help for restore
  -i, --input string     the input sql file or directory for mysql restore

Global Flags:
  -f, --from string      from mysql connection string (default "root:root@tcp(localhost:3306)")
  -t, --target string    target mysql connection string (default "root:root@tcp(localhost:3306)")
  -T, --thread           whether target enable multi-threaded mode
```

1)、单线程导入sql文件，用法如下：

```
/datamover mysql restore --to "user:password@tcp(host:port)" -i gep.sql
```

注意：该用法可以修改数据库，把要 `update` or `drop` 数据的时候，可以写成 `xxx.sql`，然后用上面的命令执行即可，输入的文件改成该 `xxx.sql`

2)、多线程导入 sql 文件所在目录，用法如下，一定要加上多线程标志 `--thread` or `-T`：

```
/datamover mysql restore --to "user:password@tcp(host:port)" -i gep_exer_safe  
` -T (--thread)
```

5、mysql数据库的在线迁移

- 在线迁移，即源数据库和目标数据库同时在线，程序连接源数据库实例和目标数据库的实例进行迁移。
- 默认就是多线程模式，不需要用 --thread or -T 来表示，支持用 -d 表示多个和 -a 所有的数据库

```
migrate mysql database online from one mysql to another
```

Usage:

```
datamover mysql online [flags]
```

Flags:

```
-a, --all-databases      all mysql databases  
except(mysql|sys|performance_schema|information_schema)  
-d, --databases stringArray the dump databases of mysql  
-h, --help               help for online
```

Global Flags:

```
-f, --from string      from mysql connection string (default  
"root:root@tcp(localhost:3306)")  
-t, --target string    target mysql connection string (default  
"root:root@tcp(localhost:3306)")  
-T, --thread           whether target enable multi-threaded mode
```

1)、多个 databases 的在线迁移

```
/datamover mysql online --from "user1:password1@tcp(host1:port1)" --target  
"user2:password2@tcp(host2:port2)" -d exer -d safe -d ...
```

2)、所有 databases (except mysql, sys, performance_schema, information_schema) 的在线迁移

```
/datamover mysql online -f "user1:password1@tcp(host1:port1)" -t  
"user2:password2@tcp(host2:port2)" -a
```

二、etcd 之间的迁移

1、特性说明及用法

1)、特性

- etcd 之间的迁移，先通过命令行工具从源 etcd 导出 xxx.db 文件，然后再用命令行工具将 xxx.db 文件导入到另外一个 etcd 集群
- etcd 的子命令包含了 save 和 restore，跟 etcdctl 保留一致
- etcd 之间的数据迁移实现是通过 etcdctl 命令来实现的，如果大家对 etcdctl 更熟悉，那就用 etcdctl 来进行迁移会更好。

2)、用法

```
$ ./datamover etcd -h
Realize data migration commands between isomorphic etcd, support save,
restore and online move

Usage:
  datamover etcd [flags]
  datamover etcd [command]

Available Commands:
  restore    etcd snapshot restore command
  save       etcd snapshot save command

Flags:
  --cacert string    the cacert path of the etcd endpoints
  --cert string      the cert path of the etcd endpoints
  --endpoints string the endpoints of the etcd cluster (default
"http://127.0.0.1:2379")
  -h, --help         help for etcd
  --key string       the key path of the etcd endpoints

Use "datamover etcd [command] --help" for more information about a command.
```

2、etcd 数据库的导出

- 从源 etcd 中导出 xxx.db 文件，【db_file_name】如果不填，则默认输出的是 etcd-snapshot-YYYY-MM-DD HH:mm:ss.db 文件

```
$ ./datamover etcd save -h
etcd snapshot save command

Usage:
  datamover etcd save [dump_file_name] [flags]

Flags:
  -h, --help    help for save

Global Flags:
  --cacert string    the cacert path of the etcd endpoints
  --cert string      the cert path of the etcd endpoints
  --endpoints string the endpoints of the etcd cluster (default
"http://127.0.0.1:2379")
  --key string       the key path of the etcd endpoints
```

1)、不用 tls 的情况下, dump 出文件

- 加入参数[dump_file_name]

```
./datamover etcd save etcd-node1.db --endpoints http://127.0.0.1:2379
```

- 不加参数

```
./datamover etcd save --endpoints http://127.0.0.1:2379
```

则默认在当前路径生成类似 etcd-snapshot-2023-07-26_15:23:35.db的文件

2)、使用 tls 的情况下

如果是用到了 tls, 则命令行中还要明确 --cacert, --cert, --key 指明 tls 所需要的文件路径, 例如:

```
./datamover etcd save etcd-node2.db --cacert=/opt/etcd/ssl/ca.pem --  
cert=/opt/etcd/ssl/server.pem --key=/opt/etcd/ssl/server-key.pem --  
endpoints="https://192.168.1.61:2379"
```

3、etcd 数据库的导入

- 将 xxx.db 文件导入到新的 etcd 集群, 此命令中要用 --data-dir 指明要导入的新etcd集群的数据目录, 而且该数据目录必须为空

```
$ ./datamover etcd restore -h  
etcd snapshot restore command  
  
Usage:  
  datamover etcd restore <dump_file_name> [flags]  
  
Flags:  
  --data-dir string          path to the data directory  
  -h, --help                help for restore  
  --initial-advertise-peer-urls string  list of this member's peer URLs  
to advertise to the rest of the cluster  
  --initial-cluster string    Initial cluster configuration  
for restore bootstrap  
  --name string              human-readable name for this  
member  
  
Global Flags:  
  --cacert string          the cacert path of the etcd endpoints  
  --cert string            the cert path of the etcd endpoints  
  --endpoints string        the endpoints of the etcd cluster (default  
"http://127.0.0.1:2379")  
  --key string              the key path of the etcd endpoints
```

1)、简短命令

```
./datamover etcd restore etcd-node1.db --data-dir new-etcd-node1
```

2)、当 etcd restore 命令行中要出现 --name 的时候，必须同时指明 --initial-cluster 和 initial-advertise-peer-urls 这两个标志位

```
./datamover etcd restore etcd-node2.db --data-dir new-etcd-node2 --name node2  
--initial-cluster node2=http://127.0.0.1:2380 --initial-advertise-peer-urls  
http://127.0.0.1:2380
```

3)、etcd restore 命令行中同时也可以带上 --endpoints，如下所示：

```
./datamover etcd restore etcd-node1.db --data-dir new-etcd-node1 --endpoints  
http:127.0.0.1:2378
```

```
./datamover etcd restore etcd-node2.db --data-dir new-etcd-node2 --name node2  
--initial-cluster node2=http://127.0.0.1:2380 --initial-advertise-peer-urls  
http://127.0.0.1:2380 --endpoints http://127.0.0.1:2378
```

4)、etcd restore 命令行中也支持 tls，例如：

```
./datamover etcd restore etcd-node2.db --data-dir new-etcd-node2 --  
cacert=/opt/etcd/ssl/ca.pem --cert=/opt/etcd/ssl/server.pem --  
key=/opt/etcd/ssl/server-key.pem --endpoints="https://192.168.1.61:2379"
```

三、redis 之间的迁移

1、用法

```
$ ./datamover redis -h  
Realize data migration commands between redis, support move from one to  
another  
  
Usage:  
  datamover redis [flags]  
  datamover redis [command]  
  
Available Commands:  
  online      move redis data from source cluster target the target cluster  
  save        redis generates rdb snapshot files and outputs them to the  
              specified directory
```

```
Flags:
  -h, --help    help for redis

Use "datamover redis [command] --help" for more information about a command.
```

2、redis在线迁移

```
$ ./datamover redis online -h
move redis data from source cluster target the target cluster
```

```
Usage:
  datamover redis online [flags]
```

```
Flags:
  -f, --from string          source redis cluster url
  --from-db int              source redis db number
  --from-password string
  -h, --help                  help for online
  -t, --target string         target redis cluster url
  --target-db int             target redis db number
  --target-password string
```

```
./datamover redis online --from [host1:port1] --target [host2:port2] --from-
password <pwd1> --from-db <db1> --target-password <pwd2> --target-db <db2>`
```

- 以上命令中的flag，--from 和 --target 是必须的，其他的可以省略，默认为空或者是0
- 注意： 以上命令行中的 url 不要写成 <http://127.0.0.1:6379>，不需要带 http://，否则会报错

具体用法：

```
./datamover redis online --from 127.0.0.1:6379 --target 192.168.3.137:6379
```

or

```
./datamover redis online -f 127.0.0.1:6379 -t 192.168.3.137:6379 --from-db 0
--target-db 0 --from-password root1@123 --target-password root2@456
```

3、redis 数据库的导出

- dump出rdb文件，通过重启redis加载该rdb文件进行迁移


```
$ ./datamover redis save -h
redis generates rdb snapshot files and outputs them to the specified
directory

Usage:
  datamover redis save [db_file_name] [flags]

Flags:
  -h, --help           help for save
  -u, --url string      redis server url (default "redis://127.0.0.1:6379")
```

- 参数[db_file_name]是可选的，如果不填，则默认输出类似 dump-2023-07-26_17:45:56.db
- -u 和 --from 的形式不同，-u 后面的格式如下：

```
redis://[user:password@] [host:port] [/database]
```

其中，用户名和密码可以省略当没有设置的时候

```
./datamover redis save dump.db -u redis://localhost:6379
```

运行以上命令，会在当前目录中生成 dump.db 文件，然后将该 dump.db 文件放在目标 redis 服务的数据目录中(注意路径一定要准确，和redis.conf文件中的设置保持一致)，然后重新启动目标 redis 服务即可完成数据的迁移。

四、zookeeper 之间的迁移

- 只支持在线迁移

1、用法

```
$ ./datamover zookeeper online -h
move zookeeper data from source cluster to the target cluster

Usage:
  datamover zookeeper online [flags]

Flags:
  -f, --from string      source zookeeper cluster url
  -h, --help             help for online
  -t, --target string     target zookeeper cluster url
```

2、示例

```
./datamover zookeeper online --from 127.0.0.1:2181 --target 192.168.34.165:2181`
```

or

```
./datamover zookeeper online -f 127.0.0.1:2181 -t 192.168.34.165:2181
```

五、kafka 之间的迁移

- 只支持在线迁移

1、用法

```
$ ./datamover kafka online -h
move kafka data from source cluster target the target cluster

Usage:
  datamover kafka online [flags]

Flags:
  -f, --from string    source kafka cluster url
  -h, --help           help for online
  -t, --target string  target kafka cluster url
```

2、示例

```
./datamover kafka online --from 127.0.0.1:9092 --target 192.168.34.165:9092
```

or

```
./datamover kafka online -f 127.0.0.1:9092 -t 192.168.34.165:9092
```

3、注意

kafka 本来就和 zookeeper一起使用，当遇到kafka迁移的时候，用上面kafka的迁移命令同时能完成 zookeeper 的数据迁移，这种情况下不需要单独迁移 zookeeper。