DataEX 软件命令手册

Version 6.0

2024 年 3 月

TOMcoding.com

配置命令

增加数据库

命令

```
add db db_name
(
  storage_asm=[yes|no],
  remote=[yes|no],
  cdb=cdb_name,
  pdb=pdb_name,
  user=username,
  password=password,
  (
    instance_name=instance_name1,
    thread=thread#,
    service=service_name,
    host=host,
    port=port1,
    agent_port=port2,
    tns=tns_name
  )
  (
    instance_name=instance_name2,
    ......
  )
);
```

参数描述

命令 add db 向 dataex 系统中添加一个数据库，命令参数分为两部分，前面的是数据库参数，后面的是实例参数，用括号包括起来，可以配置多个实例，用于 RAC 实例匹配。

db_name 是 dataex 系统中的数据库名称，唯一标识一个数据库，不能重复。

storage_asm 参数指示数据库的存储是否使用了 ASM 存储管理,取值为 *yes* 或 *no*，不区分大小写。

remote 参数指示数据库是否远程服务器，即与 dataex 不在同一台机器上，取值为 *yes* 或 *no*，不区分大小写。如果是远程服务器，需要在数据库服务器安装读取代理程序。

cdb 参数指示 CDB 容器名称，Oracle 12c 以后的数据库架构发生变化，在多租户模式下 cdb 指定 CDB 容器的名称。

pdb 参数指示 PDB 数据库的名称，Oracle 12c 以后的数据库，数据抽取在 pdb 数据库上进行。

user 参数定义数据库的用户名称，用于获取数据库字典信息，或应用数据到目标数据库中，*username* 是用户名称。

password 参数定义数据库用户的登录密码，加密存储，*password* 是用户密码。

instance_name 指示数据库实例的名称，*instance_name1* 是实例名称，同一数据库中不能重复。

thread 参数指示实例的线程号，*thread#*是线程号。

service 参数定义数据库的服务名称，*service_name* 是服务名称。

host 参数定义数据库所在的主机名称或 IP 地址，*host* 为主机名称或 IP 地址。

port 参数定义数据库监听的端口，*port1* 为端口值，默认为 1521。

agent_port 参数定义部署在数据库服务器的读取代理程序的端口，*port2* 为端口值。

例子

添加一个名称为 src_db 的数据库，ASM 存储，本地读取，一个数据库实例。

```
add db src_db
(
  storage_asm=yes,
  remote=no,
  user=dxcap,
  password=dxcap,
  (
    instance_name=inst1,
    thread=1,
    service=ora11g,
    host=192.168.21.116,
    port=1521,
```

```
    agent_port=9001
  )
);
```

修改数据库

命令

```
modify db db_name
(
  storage_asm=[yes|no],
  remote=[yes|no],
  cdb=cdb_name,
  pdb=pdb_name,
  user=username,
  password=password
);
```

参数描述

修改数据库命令，修改已配置数据库的相关参数，*db_name* 在 dataex 系统中已存在。

storage_asm 参数指示数据库的存储是否使用了 ASM 存储管理,取值为 *yes* 或 *no*，不区分大小写。

remote 参数指示数据库是否远程服务器，即与 dataex 不在同一台机器上，取值为 *yes* 或 *no*，不区分大小写。如果是远程服务器，需要在数据库服务器安装读取代理程序。

cdb 参数指示 CDB 容器名称，Oracle 12c 以后的数据库架构发生变化，在多租户模式下 cdb 指定 CDB 容器的名称。

pdb 参数指示 PDB 数据库的名称，Oracle 12c 以后的数据库，数据抽取在 pdb 数据库上进行。

user 参数定义数据库的用户名称，用于获取数据库字典信息，或应用数据到目标数据库中，*username* 是用户名称。

password 参数定义数据库用户的登录密码，加密存储，*password* 是用户密码。

例子

修改数据库 src_db 的属性，ASM 存储，远程读取日志。

```
modify db src_db
(
  storage_asm=yes,
  remote=yes,
);
```

增加数据库实例

命令

```
modify db db_name add instance
(
  instance_name=instance_name2,
  thread=thread#
  service=service_name,
  host=host,
  port=port1,
  agent_port=port2
);
```

参数描述

增加数据库实例命令，用于在已配置的数据库中增加一个实例配置。

instance_name 指示数据库实例的名称，*instance_name2* 是实例名称，同一数据库中不能重复。

thread 参数指示实例的线程号，*thread#*是线程号。

service 参数定义数据库的服务名称，*service_name* 是服务名称。

host 参数定义数据库所在的主机名称或 IP 地址，*host* 为主机名称或 IP 地址。

port 参数定义数据库监听的端口，*port1* 为端口值，默认为 1521。

agent_port 参数定义部署在数据库服务器的读取代理程序的端口，*port2* 为端口值。

例子

在 src_db 数据库中添加一个名称为 inst2 的实例。

```
modify db src_db add instance
(
  instance_name=inst2,
  thread=2,
  service=ora11g,
  host=192.168.21.117,
  port=1521,
  agent_port=9001
);
```

删除数据库实例

命令

modify db *db_name* delete instance *instance_name*;

参数描述

删除数据库实例命令，用于在配置的数据库 *db_name* 中删除一个实例。*instance_name* 指示实例名称。

例子

在数据库 src_db 中删除实例 inst2。

```
modify db src_db delete instance inst2;
```

删除数据库

命令

delete db *db_name*;

参数描述

删除数据库命令用于删除一个 dataex 中配置的数据库 *db_name*，该命令会把数据库及所属的各个实例一并删除。

例子

删除数据库 src_db。

```
delete db src_db;
```

增加复制单元

命令

add unit *unit_name*;

参数描述

unit_name 是复制单元名称，在 dataex 系统中唯一标识一个复制单元。所有 dataex 组件必须隶属于一个复制单元，在一个复制单元内，组件的名称唯一。

例子

增加一个名称为 unit1 的复制单元。

```
add unit unit1;
```

删除复制单元

命令

delete unit *unit_name*;

参数描述

unit_name 是复制单元名称，删除复制单元命令会把复制单元下的所以组件一起删除掉。

例子

删除名称为 unit1 的复制单元。

```
delete unit unit1;
```

增加抽取组件

命令

```
add extract extract_name
(
  unit=unit_name,
  db=db_name,
  max_trans=max_transactions,
  trans_size=transaction_data_size,
  interval=read_interval,
  net_timeout=time_out,
  tables=('schema1'.'*', 'schema2'.'*'),
  exclude=('schema1'.'table1',"Schema2"."Table2"),
  mapping=('schema1'.'*'='schema2'.'*','schema3'.'table1'='schema4'.'table2')
);
```

参数描述

extract_name 字符类型，抽取组件的名称，在一个复制单元内唯一。

unit 字符类型，指定复制单元的名称，表明抽取组件属于哪个复制单元。*unit_name* 是复制单元名称。

db 字符类型，指定抽取组件从哪个数据库中抽取数据，*db_name* 是数据库名称。

max_trans 数字类型，指定数据库中最大的并发事务数，*max_transactions* 是最大事务数。

trans_size 数字类型，一个事务数据的大小，小于这个值的数据存放在内存中，大于这个值的数据存放在文件中，取值以字节为单位，也可以后缀为 K，M 等单位。

interval 数字类型，读取日志的时间间隔，如果日志文件中有数据，抽取组件会一直读取，当读到文件没有新数据时，抽取组件会停顿一段时间间隔，然后再读取数据，本参数就是定义时间间隔的，单位为秒。

net_timeout 数字类型，当抽取组件读取远程的数据库日志时，等待网络超时的时间，单位为秒。

tables 字符类型，抽取组件要抽取的表的集合。集合中指定的 schema 名称和表名都是大小写敏感的，不加引号或单引号包括的 schema 和表名都会转换成大写，双引号包括的 schema 和表名保持引号内大小写的原样。配置中星号（*）表示所有的对象，比如 *schema1.**表示 *schema1* 下所有的表，也包括在 *schema1* 下新创建的表，还有针对这个 schema 的 DDL 语句。这里对 schema 和表的说明也适用于下面的 exclude 和 mapping 集合。多个 schema 或表的配置，之间用逗号隔开。

exclude 字符类型，定义排除表的名称集合，在这个集合中的表都不会被抽取数据，不管是否已经在 tables 集合中包含。

mapping 字符类型，定义 schema 名称和表名映射的集合，在这个集合中的元素通过等号（=）隔开，等号后面的信息会去替换前面的信息，比如 *schema3.table1=schema4.table2*，在抽取完生成改变记录时，其中的 *schema3* 会被替换成 *schema4*，*table1* 会被替换成 *table2*，如果是 *schema1.*=schema2.**，表示只进行 schema 替换，所有 *schema1* 的记录中，都会被替换成 *schema2*。多个映射关系之间用逗号分割。

例子

在复制单元 unit1 中增加名称为 t_cap 的抽取组件。

```
add extract t_cap
(
  unit=unit1,
  db=src_db,
  max_trans=100,
  trans_size=32768,
```

```
  interval=5,
  net_timeout=10,
  tables=(crema.*),
  exclude=(crema.t_ext),
  mapping=(crema.*=jake.*)
);
```

修改抽取组件

命令

```
modify extract unit_name.extract_name
(
  unit=unit_name,
  db=db_name,
  max_trans=max_transactions,
  trans_size=transaction_data_size,
  interval=read_interval,
  net_timeout=time_out
);
```

参数描述

在抽取组件名字之前加上复制单元的名称，这样指定哪个复制单元下的抽取组件，否则配置程序找不到抽取组件所属的复制单元。

unit 字符类型，指定复制单元的名称，表明抽取组件属于哪个复制单元。*unit_name* 是复制单元名称。

db 字符类型，指定抽取组件从哪个数据库中抽取数据，*db_name* 是数据库名称。

max_trans 数字类型，指定数据库中最大的并发事务数，*max_transactions* 是最大事务数。

trans_size 数字类型，一个事务数据的大小，小于这个值的数据存放在内存中，大于这个值的数据存放在文件中，取值以字节为单位，也可以后缀为 K，M 等单位。

interval 数字类型，读取日志的时间间隔，如果日志文件中有数据，抽取组件会一直读取，当读到文件没有新数据时，抽取组件会停顿一段时间间隔，然后再读取数据，本参数就是定义时间间隔的，单位为秒。

net_timeout 数字类型，当抽取组件读取远程的数据库日志时，等待网络超时的时间，单位为秒。

例子

修改复制单元 unit1 下的抽取组件 t_cap 的参数。

```
modify extract unit1.t_cap
(
  max_trans=150,
  interval=10
);
```

增加抽取组件的包含表

命令

modify extract *unit_name.extract_name* add tables (*schema3.**);

参数描述

在抽取组件中增加包含的表，或 schema 下的所有表，如果没有包含表集合时，先创建集合再添加，如果有包含表集合时，追加表到集合中，保留以前集合中存在的表。

例子

在复制单元 unit1 中的抽取组件 t_cap 中，增加抽取数据库用户 scott 下的所有表。

```
modify extract unit1.t_cap add tables (scott.*);
```

增加抽取组件的排除表

命令

modify extract *unit_name.extract_name* add exclude (*schema3.table1*);

参数描述

在抽取组件中增加排除表，多个表名通过逗号分割。

例子

在复制单元 unit1 下的抽取组件 t_cap 中，添加 crema 用户下排除表 tmp_ct1，tmp_ct2。

```
modify extract unit1.t_cap add exclude (crema.tmp_ct1,crema.tmp_ct2);
```

增加抽取组件的表映射

命令

```
modify extract unit_name.extract_name
add mapping (schema1.table1=schema3.table3);
```

参数描述

在抽取组件中增加表的映射关系，多个映射关系通过逗号分割。

例子

在复制单元 unit1 下的抽取组件 t_cap 中，增加用户映射关系，用 andy 用户替换 scott 用户。

```
modify extract unit1.t_cap add mapping (scott.*=andy.*);
```

删除抽取组件的包含表

命令

```
modify extract unit_name.extract_name
delete tables (schema1.table1,schema1.table2);
```

参数描述

删除抽取组件包含表集合中的指定表，或 schema 下的所有表。

例子

删除复制单元 unit1 下的抽取组件 t_cap 中的包含表，用户 scott 下的所有表。

```
modify extract unit1.t_cap delete tables (soctt.*);
```

删除抽取组件的排除表

命令

modify extract *unit_name.extract_name* delete exclude (*schema1.table3*);

参数描述

删除抽取组件排除表集合中的指定表。

例子

在复制单元 unit1 下的抽取组件 t_cap 中，删除排除表 crema.tmp_ct1。

```
modify extract unit1.t_cap delete exclude (crema.tmp_ct1);
```

删除抽取组件的表映射

命令

modify extract *unit_name.extract_name*
delete mapping (*shcema1.*=schema3.**);

参数描述

删除抽取组件中的映射关系。

例子

在复制单元 unit1 下的抽取组件 t_cap 中，删除映射关系 scott.*=andy.*。

```
modify extract unit1.t_cap delete mapping (scott.*=andy.*);
```

删除抽取组件

命令

```
delete extract unit_name.extract_name;
```

参数描述

删除抽取组件，同时删除组件参数，包含表集合，排除表集合，映射关系集合。

例子

删除复制单元 unit1 下的抽取组件 t_cap。

```
delete extract unit1.t_cap;
```

增加应用组件

命令

```
add apply apply_name
(
  unit=unit_name,
  db=db_name,
  prev=com_name,
  interval=read_interval,
  net_timeout=time_out,
  tables=('schema1.*', 'schema2.*'),
  exclude=('schema1.table1,"Schema2"."Table2"),
  mapping=('schema1.*='schema2.*,'schema3.'table1'='schema4.'table2')
);
```

参数描述

apply_name 字符类型，应用组件的名称，在一个复制单元内唯一。

unit 字符类型，指定复制单元的名称，表明应用组件属于哪个复制单元。*unit_name* 是复制单元名称。

db 字符类型，指定应用组件向哪个数据库中装载数据，*db_name* 是数据库名称。

prev 字符类型，指定应用组件的前一个组件名称，用于确定前一个组件产生的跟踪文件名称。

interval 数字类型，读取跟踪文件的时间间隔，如果跟踪文件中没有新数据到达，应用组件下次读取跟踪文件的间隔。单位为秒。

net_timeout 数字类型，当应用组件装载数据时，网络超时的时间，单位为秒。

tables 字符类型，应用组件要装载的表的集合。集合中指定的 schema 名称和表名都是大小写敏感的，不加引号或单引号包括的 schema 和表名都会转换成大写，双引号包括的 schema 和表名保持引号内大小写的原样。配置中星号（*）表示所有的对象，比如 *schema1.**表示 *schema1* 下所有的表。多个 schema 或表的配置，之间用逗号隔开。如果没有包含表的集合，那么应用组件会装载抽取组件产生的所有数据。

exclude 字符类型，定义排除表的名称集合，在这个集合中的表都不会被装载数据，不管是否已经在 tables 集合中包含。

mapping 字符类型，定义 schema 名称和表名映射的集合，在这个集合中的元素通过等号（=）隔开，等号后面的信息会去替换前面的信息，比如 *schema3.table1=schema4.table2*，在抽取完生成改变记录时，其中的 *schema3* 会被替换成 *schema4*，*table1* 会被替换成 *table2*，如果是 *schema1.*=schema2.**，表示只进行 schema 替换，所有 *schema1* 的记录中，都会被替换成 *schema2*。多个映射关系之间用逗号分割。

例子

在复制单元 unit1 下增加应用组件 t_app。

```
add apply t_app
```

```
(
  unit=unit1,
  db=tgt_db,
  prev=t_cap,
  interval=5,
  net_timeout=10,
  tables=(crema.*)
);
```

修改应用组件

命令

```
modify apply unit_name.apply_name
(
  db=db_name,
  prev=com_name,
  interval=read_interval,
  net_timeout=time_out
);
```

参数描述

修改应用组件的参数值，在应用组件名字之前加上复制单元的名称，这样指定哪个复制单元下的应用组件。

例子

在复制单元 unit1 下的应用组件 t_app 中，修改参数。

```
modify apply unit1.t_app
(
  interval=10,
  net_timeout=20
);
```

增加应用组件的包含表

命令

modify apply *unit_name.apply_name* add tables (*schema1.**);

参数描述

在应用组件的包含表集合中增加包含表，第一次增加包含表，先创建包含表集合，再向集合中添加包含表，星号（*）表示 schema 下的所有表，多个表之间用逗号分割。

tables 字符类型，应用组件要装载的表的集合。集合中指定的 schema 名称和表名都是大小写敏感的，不加引号或单引号包括的 schema 和表名都会转换成大写，双引号包括的 schema 和表名保持引号内大小写的原样。

例子

在复制单元 unit1 下的应用组件 t_app 中增加包含表 scott.*。

modify apply unit1.t_app add tables (scott.*);

增加应用组件的排除表

命令

modify apply *unit_name.apply_name* add exclude (*schema1.table1*);

参数描述

在应用组件的排除表集合中增加排除表，第一次增加排除表，先创建排除表集合，再向集合中添加排除表，星号（*）表示 schema 下的所有表，多个表之间用逗号分割。

exclude 字符类型，定义排除表的名称集合，在这个集合中的表都不会被装载数据，不管是否已经在 tables 集合中包含。

例子

在复制单元 unit1 下的应用组件 t_app 中，增加排除表 scott.tmp_ct1 和 scott.tmp_ct2。

modify apply unit1.t_app add exclude (scott.tmp_ct1,scott.tmp_ct2);

增加应用组件的表映射

命令

modify apply *unit_name.apply_name* add mapping (*schema1.*=schema3.*);

参数描述

在应用组件的映射关系集合中增加映射关系，星号（*）表示只映射 schema 名称，否则 schema 名称和表名一起映射。

mapping 字符类型，定义 schema 名称和表名映射的集合，在这个集合中的元素通过等号（=）隔开，等号后面的信息会去替换前面的信息，比如 *schema3.table1=schema4.table2*，在抽取完生成改变记录时，其中的 *schema3* 会被替换成 *schema4*，*table1* 会被替换成 *table2*，如果是 *schema1.*=schema2.**，表示只进行 schema 替换，所有 *schema1* 的记录中，都会被替换成 *schema2*。多个映射关系之间用逗号分割。

例子

在复制单元 unit1 下的应用组件 t_app 中，增加映射关系 scott.*=andy.*。

modify apply unit1.t_app add mapping (scott.*=andy.*);

删除应用组件的包含表

命令

modify apply *unit_name.apply_name* delete tables (*schema1.*);

参数描述

删除应用组件包含表集合中的包含表，星号（*）表示删除 schema 下的所有表。

例子

删除复制单元 unit1 下的应用组件 t_app 中的包含表，用户 scott 下的所有表。

```
modify apply unit1.t_app delete tables (scott.*);
```

删除应用组件的排除表

命令

modify apply *unit_name.apply_name* delete exclude (*schema1.table1*);

参数描述

删除应用组件排除表集合中的排除表。

例子

在复制单元 unit1 下的应用组件 t_app 中，删除排除表 scott.tmp_ct1。

```
modify apply unit1.t_app delete exclude (scott.tmp_ct1);
```

删除应用组件的表映射

命令

modify apply *unit_name.apply_name* delete mapping (*schema1.*=schema3.**);

参数描述

删除应用组件映射关系集合中的映射关系。

例子

在复制单元 unit1 下的应用组件 t_app 中，删除映射关系 scott.*=andy.*。

modify apply unit1.t_app delete mapping(scott.*=andy.*);

删除应用组件

命令

delete apply *unit_name.apply_name*;

参数描述

删除应用组件，同时删除与应用组件相关的参数，包含表集合，排除表集合，映射关系集合。

例子

删除复制单元 unit1 下的应用组件 t_app。

delete apply unit1.t_app;

其他命令

保存配置

添加、修改、删除命令执行后会改变配置数据，系统不会自动保存这些改变，需要执行命令保存配置。配置数据改变后，命令行提示符会出现一个星号（*）指示数据改变，保存后星号消失。命令文本如下：

save config

重新装载

如果修改了配置而不想保存，可以使用该命令把配置文件中修改前的信息重新装载到内存中。命令文本如下：

reload config

退出命令

退出控制台命令行程序，使用该命令。命令文本如下：

exit

显示组件命令

显示所有组件

命令

list

参数描述

不带参数的 list 命令，显示所有组件名称，首先显示所有数据库的名称，然后按照复制单元，逐一显示复制单元下的所有组件名称，并注明组件类型。

例子

list

显示数据库

命令

list [database|db]

参数描述

list 命令后面带参数 database 或简写 db，显示系统配置的所有数据库名称。

例子

显示配置的所有数据库名称。

```
list db
```

显示复制单元中的组件

命令

```
list unit [unit_name]
```

参数描述

list 命令后面带参数 unit，显示复制单元 *unit_name* 下的所有组件名称和类型，如果 *unit_name* 为空或不指定，则显示所有复制单元的名称。

例子

显示复制单元 unit1 下的所有组件名称。

```
list unit unit1
```

显示组件详细信息命令

显示数据库信息

命令

```
desc db db_name
```

参数描述

desc 命令后面带参数 db，显示 *db_name* 指定数据库的详细信息。

例子

显示数据库 src_db 的详细信息。

desc db src_db

显示抽取组件信息

命令

desc extract *unit_name.extract_name*

参数描述

desc 命令后面带参数 extract，显示复制单元 *unit_name* 下的抽取组件 *extract_name* 的详细信息，包括参数，包含表集合，排除表集合，映射关系集合。

例子

显示复制单元 unit1 下的抽取组件 t_cap 的详细信息。

desc extract unit1.t_cap

显示应用组件信息

命令

desc apply *unit_name.apply_name*

参数描述

desc 命令后面带参数 apply，显示复制单元 *unit_name* 下的应用组件 *apply_name* 的详细信息，包括参数，包含表集合，排除表集合，映射关系集合。

例子

显示复制单元 unit1 下的应用组件 t_app 的详细信息。

```
desc apply unit1.t_app
```

启动命令

启动管理进程

命令

```
start manager
```

参数

该命令没有参数。

例子

启动管理进程。

```
start manager
```

启动抽取组件

命令

start extract *uint_name*.*extract_name* [options]

参数

启动复制单元 *unit_name* 下的抽取组件 *extract_name*。options 是命令行选项，选项详细信息如下。

-sscn 指定启动的开始 SCN，抽取组件从这个 SCN 开始分析后面的日志。

-fscn 指定过滤事务的提交 SCN，小于这个 SCN 的事务数据不会写入跟踪文件中。

-fxid 指定过滤事务的 XID，与-fscn 联合使用，确定一个唯一的事务，这个事务之后的数据才会写入跟踪文件。

例子

启动复制单元 unit1 下的抽取组件 t_cap，从 SCN 为 345356 的点开始分析日志。

```
start extract unit1.t_cap -sscn 345356
```

启动应用组件

命令

start apply *unit_name*.*apply_name* [options]

参数

启动复制单元 *unit_name* 下的应用组件 *apply_name*。options 是命令行选项，命令行选项的详细信息如下。

-scn 指定装载的 SCN，只有大于这个 SCN 的事务数据才被装载到数据库中。

-xid 指定装载的事务 XID，与-scn 联合使用，确定一个唯一事务，这个事务之后的数据才会被装载到数据库中。

-fno 指定跟踪文件的文件号，与-scn 和-xid 联合使用，指示在哪个文件中查找事务。

-reset 指示应用组件从跟踪文件的第一个文件的第一个事务开始装载。

例子

启动复制单元 unit1 下的应用组件 t_app，从头开始装载数据。
start apply unit1.t_app -reset

停止命令

停止应用组件

命令

stop apply *unit_name*.*apply_name*

参数

停止复制单元 *unit_name* 下的应用组件 *apply_name*。

例子

停止复制单元 unit1 下的应用组件 t_app。

stop apply unit1.t_app

停止抽取组件

命令

stop extract *unit_name*.*extract_name*

参数

停止复制单元 *unit_name* 下的抽取组件 *extract_name*。

例子

停止复制单元 unit1 下的抽取组件 t_cap。

```
stop extract unit1.t_cap
```

停止管理进程

命令

```
stop manager
```

参数

停止管理进程，没有参数。

例子

停止管理进程。

```
stop manager
```