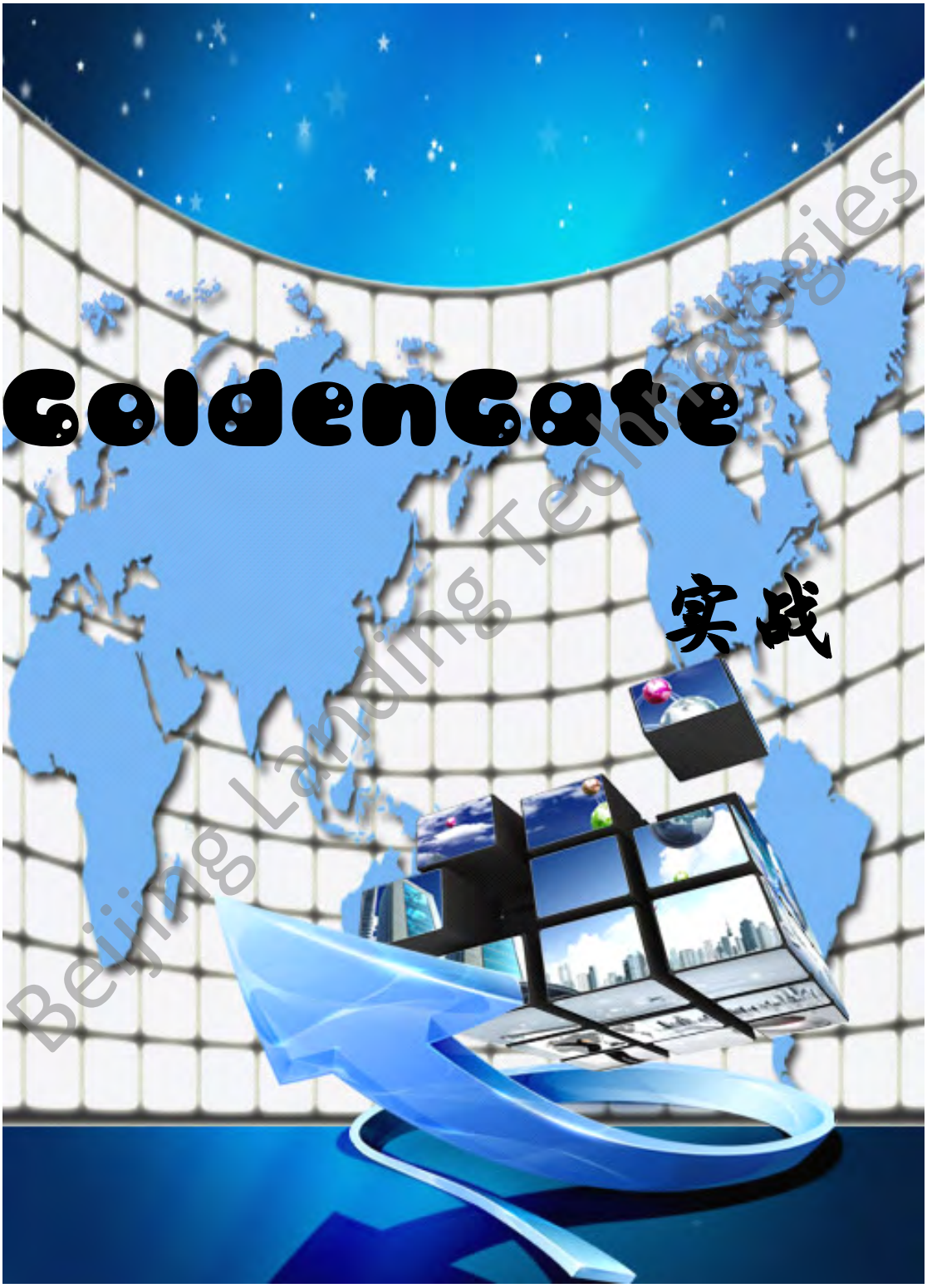
联动北方
LANDING

企业级 IT 运维宝典系列



联动北方 出品

目　录

目　录 2

第 4 章　目标端数据初始化 3
4.1　目标端数据库初始化同步的方法及比较 3
4.1.1　GoldenGate 初始化数据的方法 3
4.1.2　在初始化同步之前需要做的准备 3
4.2　数据库自带工具初始化 6
4.3　Oracle 的 RMAN 在线初始化 9
4.4　GoldenGate initial load 直接传输初始化 13
4.4.1　源端批量抽取的配置 13
4.4.2　目标端批量复制的配置 14
4.4.3　启动批量更新同步 14
4.5　GoldenGate initial load 使用文件传输初始化 15
4.5.1　配置 initial load extract 进程组 16
4.5.2　执行 initial load 捕获进程 16
4.5.3　配置 initial load replicat 进程组 16
4.5.4　执行 initial load 复制进程 16

第 1 章 目标端数据初始化

前面提到过，GoldenGate 作为数据复制的工具，他的原理是将数据从一个数据库容灾到另一个数据库中，由于他是基于交易的复制，是通过读取生产端的 redo 或 archive 日志文件来获取数据的变化，然后在容灾端还原 SQL，从而使生产端和容灾端的数据库数据一致。很多时候在配置 GoldenGate 软件的时候，生产端的数据库中已经存在了大量的数据。那么这些数据我们就需要同步到容灾端。

数据同步可以在一个活动的生产库上实现，在同步数据的同时，生成库还有数据的更新，这样就需要配置一组进程来实时的获取更新。同步也可以在一个不活动的生产库上实现，当然这样容易的多。

1.1 目标端数据库初始化同步的方法及比较

1.1.1 GoldenGate 初始化数据的方法

一般有以下几种方法：

- 使用数据库工具进行初始化，这种方法最简单最有效；
- Extract 进程把数据抽取到文件然后 Replicat 进程投递到容灾端数据库，这是方法的缺点是比较慢。这种方式相对较简单，但是需要熟悉 ETL 工具；
- Extract 进程把抽取数据到到一个外部表，然后通过外部工具导入到容灾库中。Extract 进程通过 TCP/IP 协议直接连接 Replicat 进程，不需要 Collect 进程或文件。通过数据库引擎把数据导入到容灾端，这种方式对网络可靠性以及速度要求比较高；
- Extract 进程把数据抽取到一个外部表文件，然后通过 SQL*Loader 工具把数据导入到容灾端，这种方法比较快。

初始化同步的方法多种多样，但其原理基本相同，都是先把数据库恢复到一个 SCN 或一个时间点以后，然后利用一组抽取和投递进程组来获取数据的变化，最终达到数据同步的目的。

例如利用 Oracle 自带的 RMAN 在线初始化同步数据库，由于这种方法对源端数据库影响较小，并且无需停机进行，而且是基于 SCN 的复制，数据一致性保护比较好。比其他的初始化同步方法有很多的优势，所以在实际 Oracle-Oracle 容灾项目中，使用这种方式实现初始化同步的较多，另外还有 exp/imp、expdp/impdp、Transportatable tablespace 等，也可以使用其基于 SCN 的导出的方式进行。

具体可以参考 Oracle 数据库方面的知识。

注意：某些工具可以配置多个通道或者并行来提高初始化的速度。

1.1.2 在初始化同步之前需要做的准备

a）在执行初始化之前，如果配置了 DDL 需要禁用掉，GG 在配置是通过在 Extract 和 Replicat 参数文件中配置一下参数来实现同步 DDL 的，所以禁用 DDL 只需要把参数文件中和 DDL 有关的参数去掉就可以了；

b) 如果容灾端已经创建表，需要确保容灾端的表是空的。否则可能导入重复的数据记录使容灾库出现重复数据；禁用表的外键、约束和触发器，外键可能导致错误而约束会减慢插入的速度；删除表的索引，如果表存在索引，数据库在插入数据的时候需要同时更新索引，这会减慢插入的速度；

c) 在容灾端参数文件中加入 HANDLECOLLISIONS 参数来解决数据冲突，但需要表有主键或唯一索引，如果没有主键和唯一索引，则需要在生产端 table 和容灾端 map 配置文件加入 KEYCOLS 参数；

d) 如果生产端和容灾端的数据库不一致，需要配置定义 DEFGEN 文件来实现转换；

e) 配置管理进程：

```
edit params mgr

PORT 7839
DYNAMICPORTLIST 7840-7849
--AUTOSTART EXTRACT *
--AUTORESTART EXTRACT *,RETRIES 5,WAITMINUTES 3
PURGEOLDEXTRACTS ./dirdat/*,usecheckpoints, minkeepdays 3
LAGREPORTHOURS 1
LAGINFOMINUTES 30
LAGCRITICALMINUTES 45
```

示例 4-1

f) 为了防止在初始化目标端数据库的过程中数据库发生交易数据的变化，需要配置一组在线 Extract 和 Replicat 进程组来获取数据的变化。GoldenGate 只抽取在 Extract 进程启动以后开始的交易。在 Extract 启动之前开始的交易将被跳过。所以在执行初始化之前一定要把保证数据库当前的交易都是在 Extract 启动之后开始的。

```
-----配置 Extract 进程-----
edit params extya

EXTRACT extya
userid GoldenGate, password GoldenGate
SETENV (NLS_LANG="AMERICAN_AMERICA.ZHS16GBK") --此处数据库字符集设为一致
--SETENV (ORACLE_SID = "epmsc1")
GETTRUNCATES
REPORTCOUNT EVERY 1 MINUTES, RATE
numfiles 5000
DISCARDFILE ./dirrpt/extya.dsc,APPEND,MEGABYTES 1000
WARNLONGTRANS 2h,CHECKINTERVAL 3m
EXTTRAIL ./dirdat/ya
TRANLOGOPTIONS  CONVERTUCS2CLOBS
THREADOPTIONS MAXCOMMITPROPAGATIONDELAY 60000
DBOPTIONS ALLOWUNUSEDCOLUMN
TRANLOGOPTIONS altarchivelogdest primary instance epmsc1  /oraarch1
altarchivelogdest  instance  epmsc2 /oraarch2
DYNAMICRESOLUTION
table scott.* ;
--------添加 Extract 进程组---------
```

```
add extract extya, tranlog, begin now
add exttrail ./dirdat/ya, extract extya, megabytes 500

--------配置 Pump 进程参数
edit params dpeya

EXTRACT dpeya
RMTHOST ip_addr , MGRPORT 7839, compress
PASSTHRU
numfiles 50000
RMTTRAIL ./dirdat/ya
DYNAMICRESOLUTION
table scott.* ;
---------添加 Pump 进程组---------
ADD EXTRACT dpeya ,EXTTRAILSOURCE ./dirdat/ya
ADD RMTTRAIL ./dirdat/ya,EXTRACT dpeya,MEGABYTES 500

---------配置 Replicat 进程--------
edit params repya

REPLICAT repya
USERID GoldenGate, PASSWORD GoldenGate
setenv (NLS_LANG="AMERICAN_AMERICA.UTF8")
--REPORT AT 01:59
REPORTCOUNT EVERY 30 MINUTES, RATE
REPERROR DEFAULT, abend
numfiles 50000
DBOPTIONS ALLOWUNUSEDCOLUMN
MAXTRANSOPS   500000
GROUPTRANSOPS 10000
CHECKPOINTSECS 40
--HANDLECOLLISIONS
assumetargetdefs
DISCARDFILE ./dirrpt/repya.dsc, APPEND, MEGABYTES 1000
GETTRUNCATES
ALLOWNOOPUPDATES

MAP scott.* , target scott.* ;
--------添加 GLOBALS 参数文件，加入检查电表--------------
Edit params ./GLOBALS
checkpointtable GoldenGate.ckpt

dblogin userid GoldenGate, password GoldenGate
add checkpointtable GoldenGate.ckpt

--------添加 Replicat 进程组-----------------------------
ADD REPLICAT repya ,EXTTRAIL ./dirdat/ya  checkpointtable GoldenGate.ckpt
```

示例 4-2

1.2 数据库自带工具初始化

下面为自动工具停机初始化的原理图：

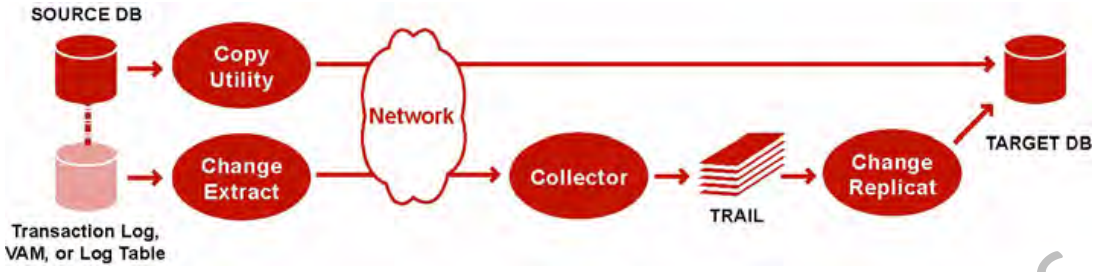


图 4-1

显而易见，这是方法的是利用数据库自带工具 copy 生产端的数据库然后再容灾端建立数据库。在 copy 数据库的过程中 GoldenGate 启动一组抽取进程来获取在 copy 过程中生产端数据库交易的变化，copy 完成启动容灾端的 Replicat 进程把在 copy 数据库过程中的交易再在容灾端执行一遍。初始化同步完成以后，Extract 和 Replicat 进程持续运行来保证生产端和容灾端数据一致。

g) 在生产端和容灾 GoldenGate 安装目录下，执行 ./ggsci，然后 start mgr：

```
cd /GoldenGate
./ggsci

GGSCI (target) 1> start mgr
Manager started.
```

示例 4-3

h) 启动生产端的 Extract 进程 start extya ：

```
GGSCI (target) 2> start extya
Sending START request to MANAGER ...
EXTRACT EXTYA starting.
```

示例 4-4

i) 拷贝生产库到容灾端直到完成，并记录完成时候的时间；

j) 确认容灾端参数文件中加入了 HANDLECOLLISIONS 参数：

```
GGSCI (target) 3>view params repya
REPLICAT repya
USERID ggs , PASSWORD ggs
--SETENV (NLS_LANG = "American_America.ZHS16GBK")
--REPORT AT 01:59
REPORTCOUNT EVERY 30 MINUTES, RATE
REPERROR DEFAULT, ABEND
numfiles 50000
DBOPTIONS ALLOWUNUSEDCOLUMN
MAXTRANSOPS   500000
GROUPTRANSOPS 10000
CHECKPOINTSECS 40
HANDLECOLLISIONS
```

```
    assumetargetdefs
    DISCARDFILE ./dirrpt/repya.dsc, APPEND, MEGABYTES 1000
    GETTRUNCATES
    ALLOWNOOPUPDATES
    DDL INCLUDE MAPPED
    DDLOPTIONS REPORT
    map scott.* , target scott.* ;
```

示例 4-5

k) 禁用外键约束和 check 约束，还需要禁用掉 trigger，因为外键约束可能会导致插入错误，而 check 约束又会减慢插入的速度。数据库中的 trigger 还可能导致冗余数据的产生，例如：GG 插入一份数据，trigger 也插入一份数据。

```
    declare
    v_sql varchar2(2000);
    CURSOR c_trigger IS SELECT 'alter table '||owner||'.'||table_name||'
disable constraint '||constraint_name
     from dba_constraints where constraint_type='R' and owner in
('SCOTT','TEST');
    BEGIN
    OPEN c_trigger;
    LOOP
    FETCH c_trigger INTO v_sql;
    EXIT WHEN c_trigger%NOTFOUND;
    execute immediate v_sql;
    end loop;
    close c_trigger;
    end;
    /


    ----------disable references---------
    declare
    v_sql varchar2(2000);
    CURSOR c_ref IS SELECT 'alter table '||owner||'.'||table_name||' disable
constraint '||constraint_name from dba_constraints where constraint_type='R'
and owner in ('scott','test');
    BEGIN
    OPEN c_trigger;
    LOOP
    FETCH c_ref INTO v_sql;
    EXIT WHEN c_ref%NOTFOUND;
    execute immediate v_sql;
    end loop;
    close c_ref;
    end;
    /
```

示例 4-6

l) 启动容灾端的 Replicat 进程，使得在 copy 数据库过程中发生的交易变化在容灾端再重放一遍。

```
cd /GoldenGate
./ggsci

GGSCI (target) 2> start repya
Sending START request to MANAGER ...
REPLICAT REPYA starting.
```

示例 4-7

m) 使用命令 info replicat repya 查看 Replicat 进程的时间信息一直到容灾端数据追到 copy 完成的时间点。表明在数据初始化过程中的数据变化已经应用到到容灾端。

```
GGSCI (target) 2> info repya

REPLICAT   REPYA     Last Started 2011-01-15 13:11   Status RUNNING
Checkpoint Lag       00:00:00 (updated 00:00:15 ago)
Log Read Checkpoint  File ./dirdat/ya000002
                     2011-01-16 21:23:56.938715  RBA 1160096 --查看这个时间
```

示例 4-8

n) 由于在 copy 数据库过程中的数据变化已经应用到容灾端，copy 完成这个时间的点容灾端和生产端的数据时一致的，在这个时间点以后只是纯数据的变化，所以不需要冲突处理。使用命令 SEND REPLICAT <Replicat group name>, NOHANDLECOLLISIONS 禁用冲突处理。

```
GGSCI (target) 2> SEND REPLICAT repya, NOHANDLECOLLISIONS

Sending NOHANDLECOLLISIONS request to REPLICAT REPYA ...
REPYA No tables found matching GGS.* to set NOHANDLECOLLISIONS
```

示例 4-10

o) 去掉容灾端的中的 HANDLECOLLISIONS 参数。

```
GGSCI (target) 2> edit params repya

REPLICAT repya
USERID ggs, PASSWORD ggs
--SETENV (NLS_LANG = "American_America.ZHS16GBK")
--REPORT AT 01:59
REPORTCOUNT EVERY 30 MINUTES, RATE
REPERROR DEFAULT, ABEND
numfiles 50000
DBOPTIONS ALLOWUNUSEDCOLUMN
MAXTRANSOPS   500000
GROUPTRANSOPS 10000
CHECKPOINTSECS 40
--HANDLECOLLISIONS                        使用--注释掉参数
assumetargetdefs
DISCARDFILE ./dirrpt/repya.dsc, APPEND, MEGABYTES 1000
```

```
GETTRUNCATES
ALLOWNOOPUPDATES
DDL INCLUDE MAPPED
DDLOPTIONS REPORT
map scott.* , target scott.* ;
```

示例 4-10

p) 在初始化完成以后，Extract 进程和 Replicat 进程持续运行来保证生产端和容灾端的数据一致。

1.3 Oracle 的 RMAN 在线初始化

RMAN 在线初始化是使用 RMAN 工具备份生产端的数据库，然后在容灾端恢复数据库到一个 SCN，从这个 SCN 以后启动 Replicat 进程，所以这种方法不需要冲突处理。在备份和恢复的过程中需要启动 Extract 进程组来获取在 backup 和 recover 过程中的数据变化。容灾库恢复完成后启动 Replicat 进程，此后，Extract 和 Replicat 持续运行，保证生产端数据库和容灾端数据库一致。

q) 启动生产端和容灾端的管理进程：

```
cd /goldengate
./ggsci

GGSCI (target) 1> start mgr
Manager started.
```

示例 4-11

r) 启动生产端的抽取进程，从现在开始获取数据库中变化的事物，然后写到 trail 文件中，在这过程中可以把数据传送到容灾端，但千万不要启动容灾端的 Replicat 进程：

```
GGSCI (target) 2> start extya
Sending START request to MANAGER ...
EXTRACT EXTYA starting.
```

示例 4-12

s) 查看数据库中所有事务的开始时间，直到其大于抽取进程的启动时间再开始备份数据库，因为 GoldenGate 的只获取在 Extract 启动以后的交易变化，在 Extract 启动之前开始而在 Extract 启动以后才完成的交易 GoldenGate 将会忽略这些交易，这些被忽略的交易数据就会丢失。所以需要等数据库中所以的交易都在 Extract 启动之后开始的才能开始备份数据库。通过 v$transaction 视图来查看数据库中的交易：

```
SQL>select min(start_time) from v$transaction;
START_TIME
--------------------
01/28/11 05:54:14
```

示例 4-13

t) 当所有在 Extract 启动之前的开始的交易都完成后，我们就可以使用 RMAN 备份生产端的数据库了。备份数据库的过程中一定要密切监控 Extract 进程的状态，保证其一直正常运行：

```
######## rman.sh  begin
```

```
export NLS_DATE_FORMAT='yyyymmdd hh24:mi:ss'
export ORACLE_SID=orcl

rman target / log=/goldnegate/rman.log <<EOF
crosscheck archivelog all;
run{
allocate channel ch1 type disk maxpiecesize 100M;
allocate channel ch2 type disk maxpiecesize 100M;

backup database tag 'full_epmln' format '/nas/backup/%d_full_%T_%U.bak';
sql 'alter system archive log current';
backup archivelog all tag 'arch_epmln' format
'/nas/backup/%d_arch_%T_%U.bak';

backup current controlfile tag 'ctl_epmln' format
'/nas/backup/%d_ctl_%T_%U.bak';
release channel ch1;
release channel ch2;
}
EOF
exit
######## rman.sh  end

--在后台执行：
nohup sh rman.sh &
```

示例 4-14

u) 将备份集传送到容灾端；

v) 在容灾端恢复数据库：

```
--恢复数据库
startup nomount

--恢复控制文件
run {
allocate channel d1 device type disk;
restore controlfile from 'controlfile_backuppiece_name';
release channel d1;
}

alter database mount;

--还原数据库
show all
CONFIGURE DEVICE TYPE DISK PARALLELISM 4 BACKUP TYPE TO BACKUPSET;

run {
allocate channel d1 device type disk;
restore database;
```

```
release channel d1;
}

--还原归档日志
run {
allocate channel d1 device type disk;
restore archivelog from logseq 34503;
release channel d1;
}

--恢复数据库
run {
allocate channel d1 device type disk;
recover database using backup controlfile until cancel;
release channel d1;
}

--查询并记录数据文件的 scn
SQL> select CHECKPOINT_CHANGE# ,file# from v$datafile_header;

CHECKPOINT_CHANGE# file#
------------------ ------------------
           2493327  1
           2493327  2
           2493327  3                ----一定要记住这个 SCN，启动的 Replicat
--打开数据库（resetlogs）
alter database open database resetlogs
```

示例 4-15

w) 修改容灾端数据库为非归档模式：

```
--修改参数文件
ALTER SYSTEM RESET log_archive_dest_1  SCOPE=SPFILE;

--关闭数据库
SHUTDOWN IMMEDIATE

--改为非归档模式
STARTUP MOUNT
ALTER DATABASE NOARCHIVELOG;
ALTER DATABASE OPEN;
```

示例 4-16

x) 禁用容灾端数据库的外键，trigger 和有 DML 操作的 JOB，禁用掉容灾端的外键，因为外键约束可能导致插入错误，而 trigger 和有 DML 操作的 JOB 可能导致冗余数据的产生，是容灾端的数据库出现重复数据。

```
------disable trigger------------------
declare
v_sql varchar2(2000);
```

```
    CURSOR c_trigger IS SELECT 'alter trigger '||owner||'.'||trigger_name||'
disable' from dba_triggers where owner in ('scott','test');
    BEGIN
    OPEN c_trigger;
    LOOP
    FETCH c_trigger INTO v_sql;
    EXIT WHEN c_trigger%NOTFOUND;
    execute immediate v_sql;
    end loop;
    close c_trigger;
    end;
    /

    ----------disable references---------
    declare
    v_sql varchar2(2000);
    CURSOR c_ref IS SELECT 'alter table '||owner||'.'||table_name||' disable
constraint '||constraint_name from dba_constraints where constraint_type='R'
and owner in ('scott','test');
    BEGIN
    OPEN c_trigger;
    LOOP
    FETCH c_ref INTO v_sql;
    EXIT WHEN c_ref%NOTFOUND;
    execute immediate v_sql;
    end loop;
    close c_ref;
    end;
    /

    ---------disable job--------------

    Sql> alter system set JOB_QUEUE_PROCESSES=0;

    --查询数据库运行 scheduler job
    col PROGRAM_OWNER for a20
    col PROGRAM_NAME for a40
    col JOB_NAME for a20
    set linesize 200 pagesize 100
    select job_name,owner,program_name,program_owner,state,enabled
    from dba_scheduler_jobs where owner not in ('SYS','SYSTEM');

    --禁用 scheduler job
    exec dbms_scheduler.disable('EPSA_LN.EPSA_LOG_JOB')
```

示例 4-17

y) 启动容灾端的入库进程，一定要根据上面查到的 SCN 启动数据库，因为数据文件恢复到了这个 SCN，由于 RMAN 保证了这个 SCN 之前生产库和容灾库的数据一致性，所以只需要让 GG 来部署这个 SCN 以后出现的数据变化。

```
GGSCI 1> start repya , aftercsn xxxxxxx
```

示例 4-18

1.4 GoldenGate initial load 直接传输初始化

GoldenGate 直接传输初始化的原理图：

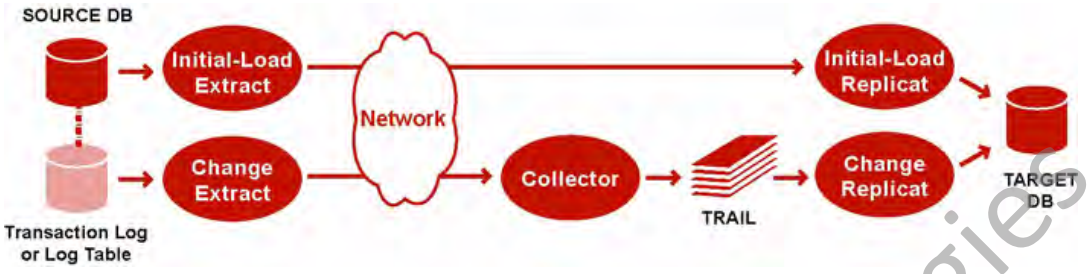


图 4-2

GoldenGate initial load 直接传输初始化是 GoldenGate initial load Extract 进程获取生产端的记录直接传送给容灾的 initial-load Replicat 进程，这个过程由 MGR 进程动态的启动不需要使用 Collect 进程和文件。

在 initial-load 过程中，需要启动一致 Extract 和 Replicat 进程来获取 initial-load 过程中的数据变化，然后部署到容灾端最终使初始化的结果一致。这种方式不支持 LOB 和 LONG 类型的数据。GoldenGate 会根据 MGR 进程中 DYNAMICPORTLIST 配置的端口列表来为 Replicat 进程动态的分配端口。

在生产端和容灾 GoldenGate 安装目录下，执行 ./ggsci，然后 start mgr：

```
cd /goldengate
./ggsci

GGSCI (target) 1> start mgr
Manager started.
```

示例 4-19

1.4.1 源端批量抽取的配置

```
---------编辑批量抽取参数文件--------------
edit params extinit

EXTRACT extinit
SOURCEDB orcl ,userid ggs , password ggs
RMTHOST ip_addr , MGRPORT 7839, compress
RMTTASK replicat,GROUP repinit                ----目标端 Replicat
TABLE scott.* ;

---------添加批量 Extract 进程---------------
ADD EXTRACT extinit, SOURCEISTABLE
```

示例 4-20

和添加抽取变化进程的方式不同，添加批量抽取进程组的命令为：

```
ADD EXTRACT <initial-load Extract name>, SOURCEISTABLE
```

示例 4-21

其中，

<initial-load Extract name>：批量抽取进程组的名字，最多八个字符

SOURCEISTABLE：标识这是一个 initial-load 抽取进程，直接从数据库的表中读取数据

1.4.2 目标端批量复制的配置

```
---------编辑批量复制参数文件-------------
edit params repinit

REPLICAT repinit
TARGETDB orcl,USERID ggs, PASSWORD ggs
ASSUMETARGETDEFS
MAP scott.* , target scott.* ;

---------添加批量 Replicat 进程---------------
ADD REPLICAT repinit, SPECIALRUN
```

示例 4-22

和添加抽取变化的命令不同，添加批量复制进程的命令为：

```
ADD REPLICAT <initial-load Replicat name>, SPECIALRUN
```

示例 4-23

其中：

<initial-load Replicat name>：批量复制进程的名字

SPECIALRUN：标识批量复制进程组只在一段时间运行，不是永久的 running

1.4.3 启动批量更新同步

z）启动生产端的抽取进程 start extya：

```
GGSCI (target) 2> start extya
Sending START request to MANAGER ...
EXTRACT EXTYA starting.
```

示例 4-24

aa） 启动生产端的批量抽取进程 start extinit，不需要启动 repinit 进程，MGR 会自动启动它，等同步结束，他会自动关闭：

```
GGSCI (target) 3> start extinit
Sending START request to MANAGER ...
EXTRACT EXTINIT starting.
```

示例 4-25

ab） 在源端 view report extinti 直到 load 结束，然后做下一步；

ac） 在容灾端启动投递进程 start repya：

```
GGSCI (target) 1> start repya
```

```
Sending START request to MANAGER ...
REPLICAT repya starting.
```

示例 4-26

ad)　　查看容灾端投递进程的状态 info repya,直到它大于 load 结束的时间

```
GGSCI (target) 2> info repya

REPLICAT   REPYA     Last Started 2011-01-15 13:11   Status RUNNING
Checkpoint Lag       00:00:00 (updated 00:00:15 ago)
Log Read Checkpoint  File ./dirdat/ya000002
2011-01-16 21:23:56.938715  RBA 1160096     --查看这个时间
```

示例 4-27

ae)　　关掉冲突检查选项 SEND REPLICAT repya, NOHANDLECOLLISIONS

```
GGSCI (target) 3> SEND REPLICAT repya, NOHANDLECOLLISIONS

Sending NOHANDLECOLLISIONS request to REPLICAT REPYA ...
REPYA No tables found matching GGS.* to set NOHANDLECOLLISIONS
```

示例 4-28

af)　　去掉 repya 文件中的 HANDLECOLLISIONS 参数

```
GGSCI (target) 3> edit params repya
--HANDLECOLLISIONS
```

示例 4-29

1.5　GoldenGate initial load 使用文件传输初始化

使用文件传输初始化的原理图:

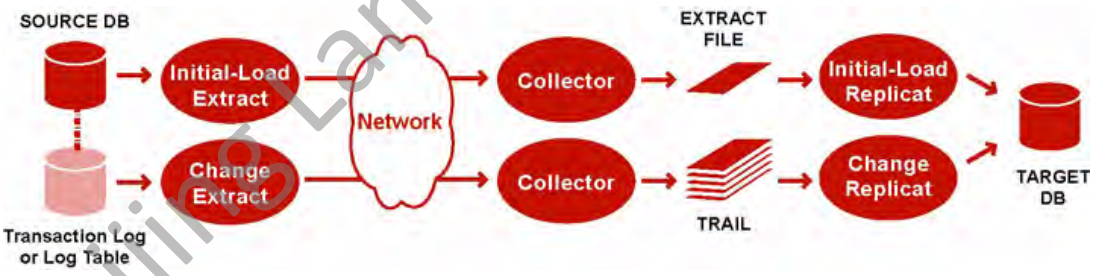


图 4-3

GoldenGate Initial-load 使用文件传输初始化的方式是 initial-load Extract 进程直接从源端数据库表中读取数据然后把他写到一个通用格式的文件中，initial-load Replicat 进程通过数据库接口把文件中的数据复制到容灾端数据库中。

在 initial-load 的过程中 GoldenGate 启动一组 Extract 和 Replicat 进程来获取数据的变化，来保证容灾端数据库和生产端数据库的数据一致。

在生产端和容灾 GoldenGate 安装目录下，执行 ./ggsci，然后 start mgr:

```
cd /goldengate
./ggsci
```

```
GGSCI (target) 1> start mgr
Manager started.
```

示例 4-30

1.5.1 配置 initial load extract 进程组

```
---------编辑批量抽取参数文件--------------
edit params extinit

SOURCEISTABLE
SOURCEDB orcl ,userid ggs , password ggs
RMTHOST ip_addr , MGRPORT 7839, compress
RMTFILE /GoldenGate/rmtfile,MAXFILES 5000, MEGABYTES 500
TABLE scott.* ;
```

示例 4-31

这种初始化的方式不需要添加进程组，只需要运行所编辑的文件就可以了。

1.5.2 执行 initial load 捕获进程

ag) 在生产端启动抽取变化进程组 start extya;

ah) 启动 initial 进程 ./GoldenGate/dirdat/extinit.prm reportfile /GoldenGate/dirrpt/extinit.rpt;

ai) 等待直到 initial 完成。

1.5.3 配置 initial load replicat 进程组

```
---------编辑批量复制参数文件--------------
edit params repinit

SPECIALRUN
END RUNTIME
EXTFILE /GoldenGate/rmtfile
ASSUMETARGETDEFS
MAP scott.* , target scott.* ;
```

示例 4-32

不需要添加 initial load replicat 进程，只需要启动文件就可以。

1.5.4 执行 initial load 复制进程

aj) 在容灾端启动 initial load replicat 进程 ./GoldenGate/dirdat/repinit.prm reportfile /GoldenGate/dirdat/repinit.rpt;

ak) 等待到 Replicat 完成;

al) 启动 Replicat 进程组 start repya;

am) 查看 Replicat 进程状态，直到其追到 Replicat 完成的时间 info repya;

an) 关闭容灾端的冲突检查操作 SEND REPLICAT <Replicat group name>, NOHANDLECOLLISIONS;

ao) 注释掉 Replicat 进程中的 HANDLECOLLISIONS 参数。