# 【DG】物理 DG 中主库的 LNSn、NSS、NSA 进程的比较

## 1.1 BLOG 文档结构图

## 1.2 前言部分

### 1.2.1 导读和注意事项

各位技术爱好者，看完本文后，你可以掌握如下的技能，也可以学到一些其它你所不知道的知识，~O(∩_∩)O~：

① 检查物理 DG 是否正常的常用 SQL

② 日志传输进程 LNSn、NSS、NSA 的区别

③ 日志传输的 2 种方式：lgwr 和 arch，10g 和 11g 有了变化

④ dg 的架构图

**Tips：**

① 本文在 ITpub（http://blog.itpub.net/26736162）和博客园（http://www.cnblogs.com/lhrbest）有同步更新

② 文章中用到的所有代码，相关软件，相关资料请前往小麦苗的云盘下载（http://blog.itpub.net/26736162/viewspace-1624453/）

③ 若文章代码格式有错乱，推荐使用搜狗、360 或 QQ 浏览器，也可以下载 pdf 格式的文档来查看，pdf 文档下载地址：
http://blog.itpub.net/26736162/viewspace-1624453/

④ 本篇 BLOG 中命令的输出部分需要特别关注的地方我都用灰色背景和粉红色字体来表示，比如下边的例子中，thread 1 的最大归档日志号为 33，thread 2 的最

大归档日志号为 43 是需要特别关注的地方；而命令一般使用黄色背景和红色字体标注；对代码或代码输出部分的注释一般采用蓝色字体表示。

```
  List of Archived Logs in backup set 11
  Thrd Seq     Low SCN    Low Time            Next SCN   Next Time
  ---- ------- ---------- ------------------- ---------- ---------
  1    32      1621589    2015-05-29 11:09:52 1625242    2015-05-29 11:15:48
  1    33      1625242    2015-05-29 11:15:48 1625293    2015-05-29 11:15:58
  2    42      1613951    2015-05-29 10:41:18 1625245    2015-05-29 11:15:49
  2    43      1625245    2015-05-29 11:15:49 1625253    2015-05-29 11:15:53



[ZHLHRDB1:root]:/>lsvg -o
T_XDESK_APP1_vg
rootvg
[ZHLHRDB1:root]:/>
00:27:22 SQL> alter tablespace idxtbs read write;


====》2097152*512/1024/1024/1024=1G
```

**本文如有错误或不完善的地方请大家多多指正，ITPUB 留言或 QQ 皆可，您的批评指正是我写作的最大动力。**

## 1.2.2 相关参考文章链接

关于 dg 的安装和一些高级应用参考：

| 【DATAGUARD】 DG 系列 | |
|---|---|
| 【推荐】 【故障处理】DG 归档丢失的恢复 | http://blog.itpub.net/26736162/viewspace-2087473/ |
| 【DG】主 rac + 备 rac dg 部署 | http://blog.itpub.net/26736162/viewspace-1991449/ |
| 【推荐】【DATAGUARD】物理 dg 配置客户端无缝切换 (八.4)--ora-16652 和 ora-16603 错误 | http://blog.itpub.net/26736162/viewspace-1811947/ |

| | |
|---|---|
| 【推荐】【DATAGUARD】物理 dg 配置客户端无缝切换 (八.3)--客户端 TAF 配置 | http://blog.itpub.net/26736162/viewspace-1811944/ |
| 【推荐】 【DATAGUARD】物理 dg 配置客户端无缝切换 (八.2)--Fast-Start Failover 的配置 | http://blog.itpub.net/26736162/viewspace-1811936/ |
| 【推荐】【DATAGUARD】物理 dg 配置客户端无缝切换 (八.1)--Data Guard Broker 的配置 | http://blog.itpub.net/26736162/viewspace-1811839/ |
| 【推荐】【DATAGUARD】物理 dg 在主库丢失归档文件的情况下的恢复(七) | http://blog.itpub.net/26736162/viewspace-1780863/ |
| 【DATAGUARD】物理 dg 的 failover 切换(六) | http://blog.itpub.net/26736162/viewspace-1753130/ |
| 【DATAGUARD】物理 dg 的 switchover 切换（五） | http://blog.itpub.net/26736162/viewspace-1753111/ |
| 【推荐】 【DATAGUARD】 将 11g 物理备库转换为 Snapshot Standby | http://blog.itpub.net/26736162/viewspace-1525548/ |
| 【推荐】【DATAGUARD】基于同一个主机建立物理备库和逻辑备库 (四)--添加一个物理 dg 节点 | http://blog.itpub.net/26736162/viewspace-1484878/ |
| 【推荐】【DATAGUARD】 基于同一个主机建立物理备库和逻辑备库 (三) | http://blog.itpub.net/26736162/viewspace-1481972/ |
| 【推荐】【DATAGUARD】 基于同一个主机建立物理备库和逻辑备库（二） | http://blog.itpub.net/26736162/viewspace-1448207/ |
| 【推荐】【DATAGUARD】 基于同一个主机建立物理备库和逻辑备库（一） | http://blog.itpub.net/26736162/viewspace-1448197/ |

### 1.2.3 本文简介

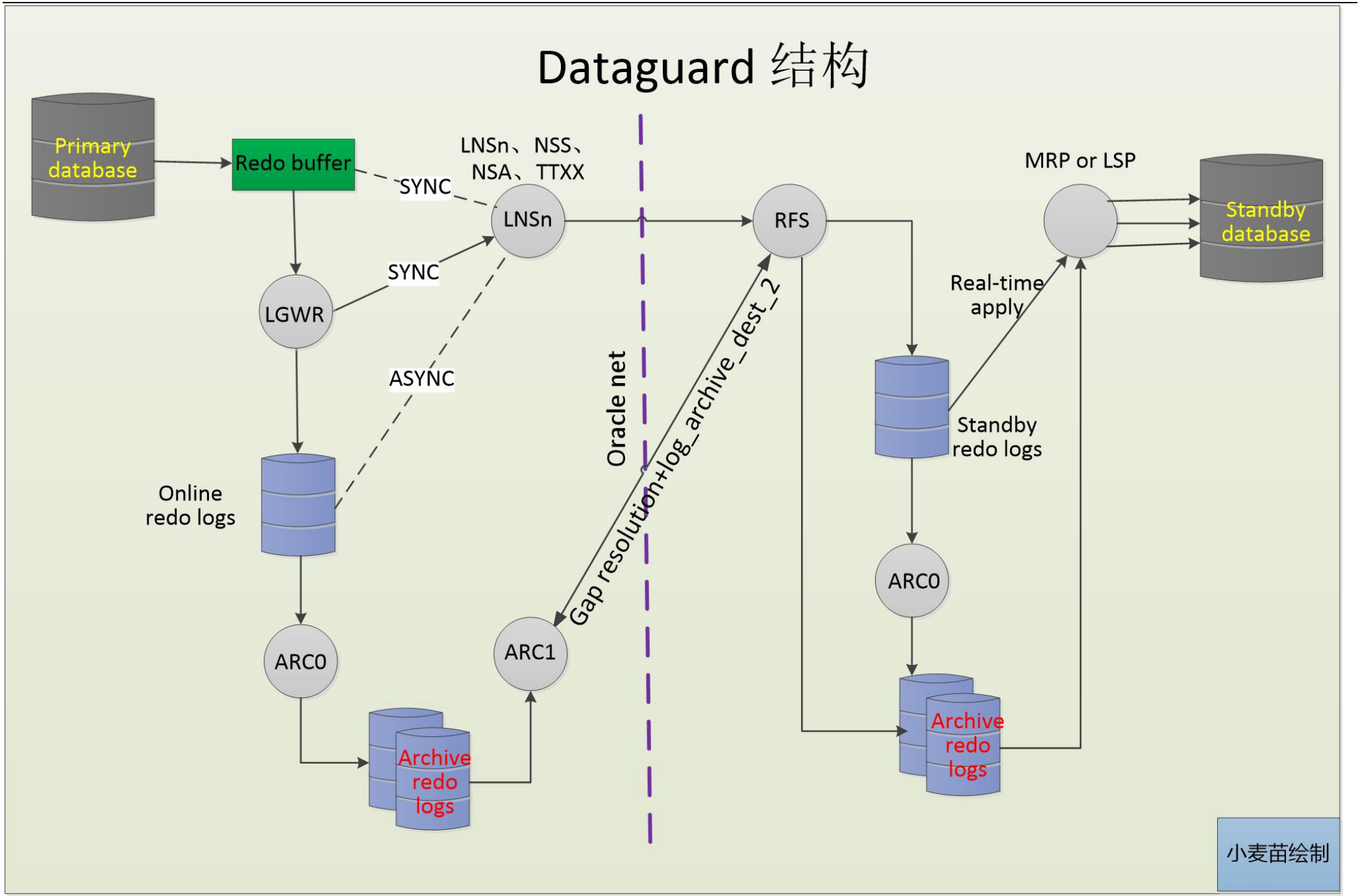
同事说 dg 不能同步，让我帮忙看看，我用自己写的 2 个视图查看了下，首先发现主库没有常见的 LNSn 进程，下意识的认为主库这个进程没有启动，需要切换日志唤醒 LNSn 进程，事实上也这样做了，（alter system set log_archive_dest_state_2='defer'; alter system switch logfile; alter system set log_archive_dest_state_2='enable'; alter system switch logfile;），切换后发现日志可以正常传输了，但是主库还是看不到 LNSn 这个进程，于是找找资料，深入的研究了一下这个问题。

在读完整个文章后，大家就会了解我这里碰到的问题，说明配置的时候不是采用的异步方式，而小麦苗后来也的确去查看了下，采用的是 LGWR SYNC 的方式，在读完这篇文章后大家对这个问题就非常明朗了。

## 1.3 相关知识点扫盲(摘自网络+个人总结)

### 1.3.1 DG 架构图

下图是小麦苗绘制的 dg 结构图，对于里边的 redo buffer 到底如何传递到 LNSn，众说纷纭，10g 和 11g 也有不同，但这个不是我们今天讨论的内容，详细点的资料可以参考：http://www.itpub.net/thread-1841337-1-1.html，我们讨论并实验 LNSn、NSS、NSA 进程在 10g 和 11g 的中表现形式。


Dataguard 结构
Primary database
Redo buffer
LNSn、NSS、
NSA、TTXX
SYNC
LNSn
RFS
MRP or LSP
Standby database
SYNC
LGWR
Real-time apply
ASYNC
Oracle net
Gap resolution+log_archive_dest_2
Online redo logs
Standby redo logs
ARC0
ARC1
ARC0
Archive redo logs
Archive redo logs
小麦苗绘制

## 1.3.2 DG 日志传输

DG 架构可以按照功能分成 3 个部分：

1） 日志发送（Redo Send）

2） 日志接收（Redo Receive）

3） 日志应用（Redo Apply）

我们今天着重来讲讲这里的日志发送的部分。

Primary Database 运行过程中，会源源不断地产生 Redo 日志，这些日志需要发送到 Standy Database。 这个发送动作可以由 Primary Database 的 LGWR 或者 ARCH 进程完成， 不同的归档目的地可以使用不同的方法，但是对于一个目的地，只能选用一种方法。 选择哪个进程对数据保护能力和系统可用性有很大区别。

如果你配置一个目的地来使用 LGWR 进程， 但是由于某些原因 LGWR 进程变得无法归到目的地了，则重做传输将会回复到使用 ARCn 进程来完成归档操作。

```
alter system set log_archive_dest_2='SERVICE=tns_mydgwl LGWR ASYNC db_unique_name=mydgwl
valid_for=(ONLINE_LOGFILES,PRIMARY_ROLE)' sid='*';
alter system set log_archive_dest_2='SERVICE=tns_mydgwl db_unique_name=mydgwl
valid_for=(ONLINE_LOGFILES,PRIMARY_ROLE)' sid='*';
```

**若不写传输进程和模式的话，11g 下默认为 LGWR ASYNC 方式，10g 为 ARCH SYNC 模式。**

### 1.3.2.1 使用 ARCH 进程

1）Primary Database 不断产生 Redo Log，这些日志被 LGWR 进程写到联机日志。

2）当一组联机日志被写满后，会发生日志切换（Log Switch），并且会触发本地归档，本地归档位置是采用 LOG_ARCHIVE_DEST_1='LOCATION=/path' 这种格式定义的。如：alter system set log_archive_dest_1 = 'LOCATION=/u01/arch' scope=both;

3）完成本地归档后，联机日志就可以被覆盖重用。

4）ARCH 进程通过 Net 把归档日志发送给 Standby Database 的 RFS（Remote File Server） 进程。

5）Standby Database 端的 RFS 进程把接收的日志写入到归档路径中。

6）Standby Database 端的 MRP(Managed Recovery Process)进程（Redo Apply）或者 LSP 进程（SQL Apply）在 Standby Database 上应用这些日志，进而同步数据。

说明：

逻辑 Standby 接收后将其转换成 SQL 语句，在 Standby 数据库上 LSP 进程执行 SQL 语句实现同步，这种方式叫 SQL Apply。

物理 Standby 接收完 Primary 数据库生成的 REDO 数据后，MRP 进程以介质恢复的方式实现同步，这种方式也叫 Redo Apply。

#### 1.3.2.2 使用 LGWR 进程的 SYNC 方式

1）Primary Database 产生的 Redo 日志要同时写到日志文件和网络。也就是说 LGWR 进程把日志写到本地日志文件的同时还要发送给本地的 LNSn 进程（Network Server Process），再由 LNSn（LGWR Network Server process）进程把日志通过网络发送给远程的目的地，每个远程目的地对应一个 LNS 进程，多个 LNS 进程能够并行工作。

2）LGWR 必须等待写入本地日志文件操作和通过 LNSn 进程的网络传送都成功，Primary Database 上的事务才能提交，这也是 SYNC 的含义所在。

3）Standby Database 的 RFS 进程把接收到的日志写入到 Standby Redo Log 日志中。

4） Primary Database 的日志切换也会触发 Standby Database 上的日志切换，即 Standby Database 对 Standby Redo Log 的归档，然后触发 Standby Database 的 MRP 或者 LSP 进程恢复归档日志。

#### 1.3.2.3 使用 LGWR 进程的 ASYNC 方式

使用 LGWR SYNC 方法的可能问题在于，如果日志发送给 Standby Database 过程失败，LGWR 进程就会报错。也就是说 Primary Database 的 LGWR 进程依赖于网络状况，有时这种要求可能过于苛刻，这时就可以使用 LGWR ASYNC 方式。 它的工作机制如下：

1）Primary Database 一旦产生 Redo 日志后，LGWR 把日志同时提交给日志文件和本地 LNS 进程，但是 LGWR 进程只需成功写入日志文件就可以，不必等待 LNSn 进程的网络传送成功。

2） LNSn 进程异步地把日志内容发送到 Standby Database。多个 LNSn 进程可以并发发送。

3） Primary Database 的 Online Redo Log 写满后发生 Log Switch，触发归档操作，也触发 Standby Database 对 Standby Redo Log 的归档；然后触发 MRP 或者 LSP 进程恢复归档日志。

### 1.3.3 进程 LNSn:LGWR Network Server process

On the primary database, the LGWR process submits the redo data to one or more network server (LNSn) processes, which then initiate the network I/O in parallel to multiple remote destinations.

```
[root@rhel6_lhr lhr]# ps -ef|grep ora_ln
```

```
oracle    8090     1  0 03:57 ?        00:01:40 ora_lns1_oradg10g
oracle   11862     1  0 05:06 ?        00:01:18 ora_lns2_oradg10g
root     26450 25545  0 09:35 pts/4    00:00:00 grep ora_ln
[root@rhel6_lhr lhr]#
```

DG 可以使用 ARCH，LGWR 来传送日志，但他们都是把日志发送给本地的 LNS（如果有多个目标备库，那会启动相应数量的 LNS 进程，同时发送数据）进程，然后备库那边的 RFS 进程接收数据，接收到的数据可以存储在备库的备用重做日志文件中或备库的归档日志中，然后再应用到备库中。

主库切换(alter system switch logfile;)可以启动 LNS 进程， V$MANAGED_STANDBY 视图可以查看 LNS 进程的具体情况：

```
col group# format a5
set line 9999 pagesize 9999
SELECT a.PROCESS,a.PID,a.STATUS,a.GROUP#,a.SEQUENCE#, a.DELAY_MINS, a.RESETLOG_ID FROM V$MANAGED_STANDBY a;
```

#### 1.3.3.1 如何启动 LNS 进程？

有 3 种方法：

① alter system switch logfile;

② 推荐： 备库启动实时应用后，主库 alter system set log_archive_dest_state_2='defer'; alter system switch logfile; alter system set log_archive_dest_state_2='enable'; alter system switch logfile;

③ 重启备库、主库

#### 1.3.3.2 LNS 进程的后台表现形式

经过小麦苗的研究，日志传输若采用 LGWR 进程来传输，则在 10g dg 中是 lns 的形式，到了 11g 变为了 nsa 和 nss 的形式了，具体可以参考本文实验部分的总结，不管 10 还是 11g 我们都可以用命令 ps -ef|grep -v grep|grep -E "ora_lns|ora_nsa|ora_nss"来查询后台进程。

```
[oracle@rhel6_lhr oradgphy]$ ps -ef|grep ora_nsa
```

```
oracle   60229     1  0 16:23 ?        00:00:01 ora_nsa2_oradg11g
oracle   60231     1  0 16:23 ?        00:00:01 ora_nsa3_oradg11g
oracle   62421     1  0 16:47 ?        00:00:00 ora_nsa2_oradglg
oracle   62423     1  0 16:47 ?        00:00:00 ora_nsa3_oradglg
oracle   64923 59476  0 17:32 pts/3    00:00:00 grep ora_nsa

oracle@ZT4FLMSDB1:/oracle$ ps -ef|grep -v grep|grep -E "ora_lns|ora_nsa|ora_nss"
  oracle 35258592       1   0 08:07:58      -  0:00 ora_nss2_oraFLMS1
```

| | | | |
|---|---|---|---|
| NSAn | Redo Transport NSA1 Process | Ships redo from current online redo logs to remote standby destinations configured for ASYNC transport | NSAn can run as multiple processes, where n is 1-9 or A-V.<br>**See Also: Oracle Data Guard Concepts and Administration** |
| NSSn | Redo Transport NSS1 Process | Acts as a slave for LGWR when SYNC transport is configured for a remote standby destination | NSSn can run as multiple processes, where n is 1-9 or A-V.<br>**See Also: Oracle Data Guard Concepts and Administration** |
| NSVn | Data Guard Broker NetSlave Process | Performs broker network communications between databases in a Data Guard environment | NSVn is created when a Data Guard broker configuration is enabled. There can be as many NSVn processes (where n is 0- 9 and A-U) created as there are databases in the Data Guard broker configuration. |

---

# 第 2 章 实验部分

## 2.1 实验目标

若采用 lgwr 进程传输日志的话，分别找到 10g 和 11g 中，后台进程的表现形式、切换日志时后台的告警日志及 V$MANAGED_STANDBY 视图展现的内容有何不同。

## 2.2 实验过程

**以下所有试验过程均在主库操作。**

### 2.2.1 10g

实验环境如下：

| 项目 | primary db |
|---|---|
| db 类型 | 单机 |
| db version | 10.2.0.5.0 |
| db 存储 | ASM |
| platform_name | AIX-Based Systems (64-bit) |

#### 2.2.1.1 LGWR SYNC(lgwr 同步)

-----LGWR SYNC

```
   alter system set log_archive_dest_2='SERVICE=tns_mydgwl LGWR SYNC db_unique_name=mydgwl
valid_for=(ONLINE_LOGFILES,PRIMARY_ROLE)';
```

```
SQL> set line 9999
SQL> col  DEST_NAME format a20
SQL> col DESTINATION format a15
SQL> col GAP_STATUS format a10
SQL> col DB_UNIQUE_NAME format a15
SQL> col error format a10
col APPLIED_SCN for 999999999999999
SQL> SQL> SELECT al.thread#,
  2         ads.dest_id,
  3         ads.DEST_NAME,
  4         (SELECT ads.TYPE || ' ' || ad.TARGET
  5            FROM v$archive_dest AD
  6           WHERE AD.DEST_ID = ADS.DEST_ID) TARGET,
  7         ADS.DATABASE_MODE,
  8         ads.STATUS,
  9         ads.error,
 10         ads.RECOVERY_MODE,
 11         ads.DB_UNIQUE_NAME,
 12         ads.DESTINATION,
 13         (SELECT MAX(sequence#) FROM v$log na WHERE na.thread# = al.thread#) Current_Seq#,
 14         MAX(sequence#) Last_Archived,
 15         max(CASE
 16               WHEN al.APPLIED = 'YES' AND ads.TYPE <> 'LOCAL' THEN
 17                al.sequence#
 18             end) APPLIED_SEQ#
 19    FROM (SELECT *
 20            FROM v$archived_log V
 21           WHERE V.resetlogs_change# =
 22                 (SELECT d.RESETLOGS_CHANGE# FROM v$database d)) al,
 23         v$archive_dest_status ads
 24   WHERE al.dest_id(+) = ads.dest_id
 25     AND ads.STATUS != 'INACTIVE'
 26   GROUP BY al.thread#,
 27            ads.dest_id,
 28            ads.DEST_NAME,
 29            ads.STATUS,
 30            ads.error,
 31            ads.TYPE,
 32            ADS.DATABASE_MODE,
 33            ads.RECOVERY_MODE,
 34            ads.DB_UNIQUE_NAME,
 35            ads.DESTINATION
 36   ORDER BY al.thread#, ads.dest_id;
```

```
   THREAD#    DEST_ID DEST_NAME           TARGET               DATABASE_MODE   STATUS    ERROR     RECOVERY_MODE          DB_UNIQUE_NAME  DESTINATION     CURRENT_SEQ#
LAST_ARCHIVED APPLIED_SEQ#
---------- ---------- -------------------- -------------------- -------------- --------- ---------- ---------------------- --------------- --------------- ------------
------------ ------------
         1          1 LOG_ARCHIVE_DEST_1   LOCAL PRIMARY        OPEN           VALID                IDLE                   mydg                                    39
38
         1          2 LOG_ARCHIVE_DEST_2   PHYSICAL STANDBY     MOUNTED-STANDBY VALID                MANAGED REAL TIME APPLY mydgwl          tns_mydgwl              39
38           38
```

====》说明备库处于 mount 状态且是实时应用的

```
SQL> alter system set log_archive_dest_2='SERVICE=tns_mydgwl LGWR SYNC db_unique_name=mydgwl valid_for=(ONLINE_LOGFILES,PRIMARY_ROLE)';

System altered.
SQL> startup force
ORACLE instance started.

Total System Global Area 1342177280 bytes
Fixed Size                  2096224 bytes
Variable Size             335545248 bytes
Database Buffers          989855744 bytes
Redo Buffers               14680064 bytes
Database mounted.
Database opened.
SQL>
SQL> alter system switch logfile;

System altered.

SQL> alter system switch logfile;

System altered.

SQL>  alter system switch logfile;

System altered.

SQL> col group_# format a5
SQL> col PROCESS format a8
SQL> col CLIENT_PID format a8
SQL> set line 9999 pagesize 9999
SQL> SELECT a.INST_ID,
  2         a.PROCESS,
  3         a.client_process,
  4         a.client_pid,
  5         a.STATUS,
```

```
  6         a.GROUP#          group_#,
  7         a.thread#,
  8         a.SEQUENCE#,
  9         a.DELAY_MINS,
 10         a.RESETLOG_ID,
 11         c.SID,
 12         c.SERIAL#,
 13         a.PID             spid
 14    FROM gV$MANAGED_STANDBY a, gv$process b, gv$session c
 15   WHERE a.PID = b.SPID
 16     and b.ADDR = c.PADDR
 17     and a.INST_ID = b.INST_ID
 18     and b.INST_ID = c.INST_ID
 19  order by a.INST_ID;

   INST_ID PROCESS  CLIENT_P CLIENT_P STATUS       GROUP    THREAD#  SEQUENCE# DELAY_MINS RESETLOG_ID        SID    SERIAL#        SPID
---------- -------- -------- -------- ------------ ----- ---------- ---------- ---------- ----------- ---------- ---------- ----------
         1 LGWR     LGWR     1712328  WRITING      2             1         17          0  1043078511        166          1    1712328
         1 ARCH     ARCH     1163504  CONNECTED    N/A           0          0          0           0        152          3    1163504
         1 ARCH     ARCH     909430   CLOSING      1             1         16          0  1043078511        155          3     909430
         1 ARCH     ARCH     311344   CLOSING      3             1         15          0  1043078511        153          3     311344
         1 ARCH     ARCH     1933380  CLOSING      2             1         14          0  1043078511        150          3    1933380
SQL> exit
Disconnected from Oracle Database 10g Enterprise Edition Release 10.2.0.5.0 - 64bit Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options
[oracle@ZT2CDS1:/cds/oradata]$ ps -ef|grep -v grep|grep -E "ora_lns|ora_nsa|ora_nss"
  oracle  712726      1   0 18:03:24      -  0:00 ora_lnsb_mydg
```

说明 LGWR 同步方式，后台进程表现为 lns，且视图 V$MANAGED_STANDBY 中表现为 LGWR。

告警日志：

```
Sun Jun 14 18:45:58 BEIST 2020
Thread 1 cannot allocate new log, sequence 31
Checkpoint not complete
  Current log# 3 seq# 30 mem# 0: /cds/oradata/mydg/redo03.log
Sun Jun 14 18:46:03 BEIST 2020
Thread 1 advanced to log sequence 31 (LGWR switch)
  Current log# 1 seq# 31 mem# 0: /cds/oradata/mydg/redo01.log
```

### 2.2.1.2 LGWR ASYNC(lgwr 异步)

```
SQL> alter system set log_archive_dest_2='SERVICE=tns_mydgwl LGWR ASYNC db_unique_name=mydgwl valid_for=(ONLINE_LOGFILES,PRIMARY_ROLE)';

System altered.

SQL> startup force
ORACLE instance started.

Total System Global Area 1342177280 bytes
Fixed Size                  2096224 bytes
Variable Size             335545248 bytes
Database Buffers          989855744 bytes
Redo Buffers               14680064 bytes
Database mounted.
Database opened.
SQL> alter system switch logfile;

System altered.

SQL>  alter system switch logfile;

System altered.

SQL> col group_# format a5
SQL> col PROCESS format a8
SQL> col CLIENT_PID format a8
SQL> set line 9999 pagesize 9999
SQL> SELECT a.INST_ID,
  2         a.PROCESS,
  3         a.client_process,
  4         a.client_pid,
  5         a.STATUS,
  6         a.GROUP#          group_#,
  7         a.thread#,
  8         a.SEQUENCE#,
  9         a.DELAY_MINS,
 10         a.RESETLOG_ID,
 11         c.SID,
 12         c.SERIAL#,
 13         a.PID             spid
 14    FROM gV$MANAGED_STANDBY a, gv$process b, gv$session c
 15   WHERE a.PID = b.SPID
 16     and b.ADDR = c.PADDR
 17     and a.INST_ID = b.INST_ID
 18     and b.INST_ID = c.INST_ID
```

```
 19  order by a.INST_ID;

   INST_ID PROCESS  CLIENT_P CLIENT_P STATUS       GROUP     THREAD#  SEQUENCE# DELAY_MINS RESETLOG_ID        SID    SERIAL#        SPID
---------- -------- -------- -------- ------------ ----- ---------- ---------- ---------- ----------- ---------- ---------- ----------
         1 ARCH     ARCH     1802334  CONNECTED    N/A           0          0          0           0        155          3    1802334
         1 ARCH     ARCH     1683676  CLOSING      N/A           1         27          0  1043078511        150          3    1683676
         1 LNS      LNS      1454138  WRITING      3             1         30          0  1043078511        154          3    1454138
         1 ARCH     ARCH     909490   CLOSING      2             1         29          0  1043078511        152          3     909490
         1 ARCH     ARCH     1364102  CLOSING      1             1         28          0  1043078511        153          3    1364102

SQL>

[oracle@ZT2CDS1:/cds/oradata]$ ps -ef|grep -v grep|grep -E "ora_lns|ora_nsa|ora_nss"
  oracle 1454138      1   2 18:44:21      -  0:00 ora_lns1_mydg
```

说明 LGWR 异步方式，后台进程表现为 lns，且视图 V$MANAGED_STANDBY 中表现为 LNS。

告警日志：

```
Sun Jun 14 18:45:58 BEIST 2020
Thread 1 cannot allocate new log, sequence 31
Checkpoint not complete
  Current log# 3 seq# 30 mem# 0: /cds/oradata/mydg/redo03.log
Sun Jun 14 18:46:03 BEIST 2020
Thread 1 advanced to log sequence 31 (LGWR switch)
  Current log# 1 seq# 31 mem# 0: /cds/oradata/mydg/redo01.log
```

#### 2.2.1.3 ARCH（归档传输）

```
SQL> alter system set log_archive_dest_2='SERVICE=tns_mydgwl ARCH ASYNC db_unique_name=mydgwl valid_for=(ONLINE_LOGFILES,PRIMARY_ROLE)';

System altered.

SQL> startup force
```

```
ORACLE instance started.

Total System Global Area 1342177280 bytes
Fixed Size                  2096224 bytes
Variable Size             335545248 bytes
Database Buffers          989855744 bytes
Redo Buffers               14680064 bytes
Database mounted.
Database opened.
SQL> alter system switch logfile;

System altered.

SQL> alter system switch logfile;

System altered.

SQL> col group_# format a5
SQL> col PROCESS format a8
SQL> col CLIENT_PID format a8
SQL> set line 9999 pagesize 9999
SQL> SELECT a.INST_ID,
  2         a.PROCESS,
  3         a.client_process,
  4         a.client_pid,
  5         a.STATUS,
  6         a.GROUP#         group_#,
  7         a.thread#,
  8         a.SEQUENCE#,
  9         a.DELAY_MINS,
 10         a.RESETLOG_ID,
 11         c.SID,
 12         c.SERIAL#,
 13         a.PID            spid
 14    FROM gV$MANAGED_STANDBY a, gv$process b, gv$session c
 15   WHERE a.PID = b.SPID
 16     and b.ADDR = c.PADDR
 17     and a.INST_ID = b.INST_ID
 18     and b.INST_ID = c.INST_ID
 19  order by a.INST_ID;

   INST_ID PROCESS  CLIENT_P CLIENT_P STATUS       GROUP    THREAD#  SEQUENCE# DELAY_MINS RESETLOG_ID        SID    SERIAL#        SPID
---------- -------- -------- -------- ------------ ----- ---------- ---------- ---------- ----------- ---------- ---------- ----------
         1 ARCH     ARCH     1933434  CLOSING      3             1         21          0  1043078511        155          3    1933434
         1 ARCH     ARCH     1290450  CLOSING      1             1         22          0  1043078511        154          3    1290450
         1 ARCH     ARCH     487436   CLOSING      N/A           1         21          0  1043078511        152          3     487436
         1 ARCH     ARCH     311350   CLOSING      N/A           1         22          0  1043078511        150          3     311350
```

```
[oracle@ZT2CDS1:/cds/oradata]$ ps -ef|grep -v grep|grep -E "ora_lns|ora_nsa|ora_arc"
  oracle  311350       1   0 18:15:20      -  0:00 ora_arc1_mydg
  oracle  487436       1   0 18:15:20      -  0:00 ora_arc2_mydg
  oracle 1290450       1   0 18:15:20      -  0:00 ora_arc3_mydg
  oracle 1933434       1   0 18:15:20      -  0:00 ora_arc0_mydg
```

说明 arch 进程传输日志，后台进程表现为 arc，且视图 V$MANAGED_STANDBY 中表现为 ARCH。

告警日志输出：

```
Sun Jun 14 18:15:44 BEIST 2020
Thread 1 advanced to log sequence 23 (LGWR switch)
  Current log# 2 seq# 23 mem# 0: /cds/oradata/mydg/redo02.log
Sun Jun 14 18:15:44 BEIST 2020
Shutting down archive processes
Sun Jun 14 18:15:49 BEIST 2020
ARCH shutting down
ARC4: Archival stopped
```

#### 2.2.1.4 默认传输模式测试

```
SQL> SELECT a.VALUE FROM v$parameter a WHERE a.NAME='log_archive_dest_2';

VALUE
--------------------------------------------------------------------------------
SERVICE=tns_mydgwl LGWR ASYNC db_unique_name=mydgwl valid_for=(ONLINE_LOGFILES,PRIMARY_ROLE)

SQL> SELECT a.PROCESS,a.TRANSMIT_MODE FROM V$ARCHIVE_DEST a WHERE a.DEST_NAME='LOG_ARCHIVE_DEST_2';

PROCESS   TRANSMIT_MOD
--------- ------------
LGWR      ASYNCHRONOUS

SQL> alter system set log_archive_dest_2='SERVICE=tns_mydgwl db_unique_name=mydgwl valid_for=(ONLINE_LOGFILES,PRIMARY_ROLE)' sid='*';
```

```
System altered.

SQL> SELECT a.VALUE FROM v$parameter a WHERE a.NAME='log_archive_dest_2';

VALUE
--------------------------------------------------------------------------------
SERVICE=tns_mydgwl db_unique_name=mydgwl valid_for=(ONLINE_LOGFILES,PRIMARY_ROLE)

SQL>  SELECT a.PROCESS,a.TRANSMIT_MODE FROM V$ARCHIVE_DEST a WHERE a.DEST_NAME='LOG_ARCHIVE_DEST_2';

PROCESS   TRANSMIT_MOD
--------- ------------
ARCH      SYNCHRONOUS

SQL>
```

可以看出 10g 中，默认情况下为 ARCH 的同步模式。

## 2.2.2 11g

实验环境如下：

| 项目 | primary db |
|---|---|
| db 类型 | rac |
| db version | 11.2.0.4.0 |
| db 存储 | ASM |
| platform_name | AIX-Based Systems (64-bit) |

### 2.2.2.1 LGWR SYNC(lgwr 同步)

```
SYS@oraDESDB1> set line 9999
SYS@oraDESDB1> col  DEST_NAME format a20
SYS@oraDESDB1> col DESTINATION format a15
```

```
SYS@oraDESDB1> col GAP_STATUS format a10
SYS@oraDESDB1> col DB_UNIQUE_NAME format a15
SYS@oraDESDB1> col error format a10
SYS@oraDESDB1> col APPLIED_SCN for 999999999999999
SYS@oraDESDB1> SELECT al.thread#,
  2         ads.dest_id,
  3         ads.DEST_NAME,
  4         (SELECT ads.TYPE || ' ' || ad.TARGET
  5            FROM v$archive_dest AD
  6           WHERE AD.DEST_ID = ADS.DEST_ID) TARGET,
  7         ADS.DATABASE_MODE,
  8         ads.STATUS,
  9         ads.error,
 10         ads.RECOVERY_MODE,
 11         ads.DB_UNIQUE_NAME,
 12         ads.DESTINATION,
 13         ads.GAP_STATUS,
 14         (SELECT MAX(sequence#) FROM v$log na WHERE na.thread# = al.thread#) Current_Seq#,
 15         MAX(sequence#) Last_Archived,
 16         max(CASE
 17               WHEN al.APPLIED = 'YES' AND ads.TYPE <> 'LOCAL' THEN
 18                al.sequence#
 19            end) APPLIED_SEQ#,
 20         (SELECT ad.applied_scn
 21            FROM v$archive_dest AD
 22           WHERE AD.DEST_ID = ADS.DEST_ID) applied_scn
 23    FROM (SELECT *
 24            FROM v$archived_log V
 25           WHERE V.resetlogs_change# =
 26                 (SELECT d.RESETLOGS_CHANGE# FROM v$database d)) al,
 27         v$archive_dest_status ads
 28   WHERE al.dest_id(+) = ads.dest_id
 29     AND ads.STATUS != 'INACTIVE'
 30   GROUP BY al.thread#,
 31            ads.dest_id,
 32            ads.DEST_NAME,
 33            ads.STATUS,
 34            ads.error,
 35            ads.TYPE,
 36            ADS.DATABASE_MODE,
 37            ads.RECOVERY_MODE,
 38            ads.DB_UNIQUE_NAME,
 39            ads.DESTINATION,
 40            ads.GAP_STATUS
 41   ORDER BY al.thread#, ads.dest_id;

   THREAD#    DEST_ID DEST_NAME           TARGET                DATABASE_MODE    STATUS    ERROR      RECOVERY_MODE          DB_UNIQUE_NAME  DESTINATION     GAP_STATUS
CURRENT_SEQ# LAST_ARCHIVED APPLIED_SEQ#      APPLIED_SCN
```

```
---------- ---------- -------------------- ---------------------- -------------- --------- ---------- ---------------------- --------------- --------------- ----------
------------ ------------- ------------ ----------------
         1          1 LOG_ARCHIVE_DEST_1   LOCAL PRIMARY          OPEN           VALID               IDLE                   oraDESDB        /arch
117            116                            0
         1          2 LOG_ARCHIVE_DEST_2   PHYSICAL STANDBY       OPEN_READ-ONLY VALID               MANAGED REAL TIME APPLY oraESKDB        oraESKDB        NO GAP
117            116            115           5673508
         2          1 LOG_ARCHIVE_DEST_1   LOCAL PRIMARY          OPEN           VALID               IDLE                   oraDESDB        /arch
93             92                             0
         2          2 LOG_ARCHIVE_DEST_2   PHYSICAL STANDBY       OPEN_READ-ONLY VALID               MANAGED REAL TIME APPLY oraESKDB        oraESKDB        NO GAP
93             92             92            5673508
                   3 LOG_ARCHIVE_DEST_3   LOCAL PRIMARY          OPEN           VALID               IDLE                   oraDESDB        /arch/arch2
0
```

====》说明备库处于 open 状态且是实时应用的

```
SYS@oraDESDB1> alter system set log_archive_dest_2='SERVICE=oraESKDB LGWR SYNC VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE) DB_UNIQUE_NAME=oraESKDB'
sid='*';
System altered.

[ZFZHLHRDB3:oracle]:/oracle>srvctl stop db -d oradesdb -o ABORT
[ZFZHLHRDB3:oracle]:/oracle>srvctl start db -d oradesdb
[ZFZHLHRDB3:oracle]:/oracle>ps -ef|grep -v grep|grep -E "ora_lns|ora_nsa|ora_nss"
  oracle 19988486        1   0 11:15:48      -  0:00 ora_nss2_oraDESDB1          ====》表现为 nss
[ZFZHLHRDB3:oracle]:/oracle>sqlplus / as sysdba

SQL*Plus: Release 11.2.0.4.0 Production on Wed Jul 6 11:16:57 2016

Copyright (c) 1982, 2013, Oracle.  All rights reserved.


Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.4.0 - 64bit Production
With the Partitioning, Real Application Clusters, Automatic Storage Management, OLAP,
Data Mining and Real Application Testing options

SYS@oraDESDB1> col group_# format a5
SYS@oraDESDB1> col PROCESS format a8
SYS@oraDESDB1> col CLIENT_PID format a8
SYS@oraDESDB1> set line 9999 pagesize 9999
SYS@oraDESDB1> SELECT a.INST_ID,
  2         a.PROCESS,
  3         a.client_process,
  4         a.client_pid,
  5         a.STATUS,
  6         a.GROUP#         group_#,
  7         a.thread#,
  8         a.SEQUENCE#,
```

```
 9          a.DELAY_MINS,
10          a.RESETLOG_ID,
11          c.SID,
12          c.SERIAL#,
13          a.PID             spid,
14          b.PNAME
15     FROM gV$MANAGED_STANDBY a, gv$process b, gv$session c
16    WHERE a.PID = b.SPID
17      and b.ADDR = c.PADDR
18      and a.INST_ID = b.INST_ID
19      and b.INST_ID = c.INST_ID
20   order by a.INST_ID,b.PNAME;

   INST_ID PROCESS  CLIENT_P CLIENT_P STATUS       GROUP     THREAD#  SEQUENCE# DELAY_MINS RESETLOG_ID        SID    SERIAL#        SPID PNAME
---------- -------- -------- -------- ------------ ----- ---------- ---------- ---------- ----------- ---------- ---------- ---------- -----
         1 ARCH     ARCH     16843000 CLOSING      N/A            1         88          0   916055651        130          1   16843000 ARC0
         1 ARCH     ARCH     15925468 CONNECTED    N/A            0          0          0           0        170          1   15925468 ARC1
         1 ARCH     ARCH     14221370 CLOSING      3              1         89          0   916055651        253          1   14221370 ARC2
         1 ARCH     ARCH     12648656 CLOSING      N/A            1         89          0   916055651        294          1   12648656 ARC3
         1 LGWR     LGWR     22347896 WRITING      1              1         90          0   916055651        126          1   22347896 LGWR
         2 ARCH     ARCH     12058730 CONNECTED    N/A            0          0          0           0        169          1   12058730 ARC0
         2 ARCH     ARCH     11599932 CLOSING      5              2         75          0   916055651        211          1   11599932 ARC1
         2 ARCH     ARCH     11403334 CLOSING      N/A            2         75          0   916055651        252          1   11403334 ARC2
         2 ARCH     ARCH     10944720 CLOSING      N/A            2         73          0   916055651        293          1   10944720 ARC3
         2 LGWR     LGWR     15532088 WRITING      6              2         76          0   916055651        126          1   15532088 LGWR

10 rows selected.
```

说明 LGWR 进程同步传输日志，后台进程表现为 nss，且视图 V$MANAGED_STANDBY 中表现为 LGWR，告警日志中表现为 LNS: Standby redo logfile selected for thread 1 sequence 98 for destination LOG_ARCHIVE_DEST_2 。

告警日志：

```
Wed Jul 06 13:47:59 2016
Thread 1 advanced to log sequence 98 (LGWR switch)
  Current log# 3 seq# 98 mem# 0: +DATA/oradesdb/onlinelog/group_3.387.916309939
Wed Jul 06 13:47:59 2016
LNS: Standby redo logfile selected for thread 1 sequence 98 for destination LOG_ARCHIVE_DEST_2
```

```
Wed Jul 06 13:47:59 2016
Archived Log entry 219 added for thread 1 sequence 97 ID 0xffffffff8589c5d0 dest 1:
Wed Jul 06 13:48:33 2016
```

### 2.2.2.2 LGWR ASYNC(lgwr 异步)

默认为异步。

```
alter system set log_archive_dest_2='SERVICE=oraESKDB LGWR ASYNC VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE) DB_UNIQUE_NAME=oraESKDB' sid='*';
alter system set log_archive_dest_2='SERVICE=oraESKDB VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE) DB_UNIQUE_NAME=oraESKDB' sid='*';
```

```
SYS@oraDESDB1> alter system set log_archive_dest_2='SERVICE=oraESKDB LGWR ASYNC VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE) DB_UNIQUE_NAME=oraESKDB' sid='*';
System altered.
SYS@oraDESDB1> exit
Disconnected from Oracle Database 11g Enterprise Edition Release 11.2.0.4.0 - 64bit Production
With the Partitioning, Real Application Clusters, Automatic Storage Management, OLAP,
Data Mining and Real Application Testing options
[ZFZHLHRDB3:oracle]:/oracle>ps -ef|grep -v grep|grep -E "ora_lns|ora_nsa|ora_nss"
  oracle 27066620        1   0 11:19:12      -  0:00 ora_nsa2_oraDESDB1
[ZFZHLHRDB3:oracle]:/oracle>srvctl stop db -d oradesdb -o abort
[ZFZHLHRDB3:oracle]:/oracle>srvctl start db -d oradesdb
[ZFZHLHRDB3:oracle]:/oracle>sqlplus / as sysdba

SQL*Plus: Release 11.2.0.4.0 Production on Wed Jul 6 11:22:02 2016

Copyright (c) 1982, 2013, Oracle.  All rights reserved.


Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.4.0 - 64bit Production
With the Partitioning, Real Application Clusters, Automatic Storage Management, OLAP,
Data Mining and Real Application Testing options
```

```
SYS@oraDESDB1> col group_# format a5
SYS@oraDESDB1> col PROCESS format a8
SYS@oraDESDB1> col CLIENT_PID format a8
SYS@oraDESDB1> set line 9999 pagesize 9999
SYS@oraDESDB1> SELECT a.INST_ID,
  2         a.PROCESS,
  3         a.client_process,
  4         a.client_pid,
  5         a.STATUS,
  6         a.GROUP#         group_#,
  7         a.thread#,
  8         a.SEQUENCE#,
  9         a.DELAY_MINS,
 10         a.RESETLOG_ID,
 11         c.SID,
 12         c.SERIAL#,
 13         a.PID            spid,
 14         b.PNAME
 15    FROM gV$MANAGED_STANDBY a, gv$process b, gv$session c
 16   WHERE a.PID = b.SPID
 17     and b.ADDR = c.PADDR
 18     and a.INST_ID = b.INST_ID
 19     and b.INST_ID = c.INST_ID
 20  order by a.INST_ID,b.PNAME;

   INST_ID PROCESS  CLIENT_P CLIENT_P STATUS       GROUP    THREAD#  SEQUENCE# DELAY_MINS RESETLOG_ID        SID    SERIAL#       SPID PNAME
---------- -------- -------- -------- ------------ ----- ---------- ---------- ---------- ----------- ---------- ---------- ---------- -----
         1 ARCH     ARCH     21430492 CLOSING      1             1         93          0   916055651         87          1   21430492 ARC0
         1 ARCH     ARCH     20512816 CONNECTED    N/A           0          0          0           0        129          1   20512816 ARC1
         1 ARCH     ARCH     20840650 CONNECTED    N/A           0          0          0           0        169          1   20840650 ARC2
         1 ARCH     ARCH     20250652 CLOSING      N/A           1         93          0   916055651        211          1   20250652 ARC3
         1 LNS      LNS      19792010 WRITING      2             1         94          0   916055651        252          1   19792010 NSA2
         2 ARCH     ARCH     12124262 CLOSING      4             2         77          0   916055651         47          3   12124262 ARC0
         2 ARCH     ARCH     10944728 CLOSING      N/A           2         78          0   916055651         88          1   10944728 ARC1
         2 ARCH     ARCH     10551392 CONNECTED    N/A           0          0          0           0        130          1   10551392 ARC2
         2 ARCH     ARCH     23527548 CLOSING      5             2         78          0   916055651        212          1   23527548 ARC3
         2 LNS      LNS      12517514 WRITING      6             2         79          0   916055651        293          3   12517514 NSA2

10 rows selected.
```

告警日志：

```
Wed Jul 06 19:24:34 2016
```

```
Thread 1 advanced to log sequence 117 (LGWR switch)
  Current log# 1 seq# 117 mem# 0: +DATA/oradesdb/onlinelog/group_1.258.916309939
Wed Jul 06 19:24:34 2016
Archived Log entry 280 added for thread 1 sequence 116 ID 0xffffffff8589c5d0 dest 1:
Wed Jul 06 19:24:34 2016
LNS: Standby redo logfile selected for thread 1 sequence 117 for destination LOG_ARCHIVE_DEST_2
```

说明 LGWR 进程异步传输日志，后台进程表现为 nsa，且视图 V$MANAGED_STANDBY 中表现为 LNS，告警日志中表现为 LNS: Standby redo logfile selected for thread 1 sequence 98 for destination LOG_ARCHIVE_DEST_2 。

### 2.2.2.3 ARCH（归档传输）

```
SYS@oraDESDB1> alter system set log_archive_dest_2='SERVICE=oraESKDB ARCH SYNC VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE) DB_UNIQUE_NAME=oraESKDB'
sid='*';

System altered.


[ZFZHLHRDB3:oracle]:/oracle>srvctl stop db -d oradesdb -o abort
[ZFZHLHRDB3:oracle]:/oracle>srvctl start db -d oradesdb
[ZFZHLHRDB3:oracle]:/oracle>
[ZFZHLHRDB3:oracle]:/oracle>ps -ef|grep -v grep|grep -E "ora_lns|ora_nsa|ora_nss"

[ZFZHLHRDB3:oracle]:/oracle>sqlplus / as sysdba

SQL*Plus: Release 11.2.0.4.0 Production on Wed Jul 6 14:06:00 2016

Copyright (c) 1982, 2013, Oracle.  All rights reserved.

Connected to:
Oracle Database 11g Enterprise Edition Release 11.2.0.4.0 - 64bit Production
With the Partitioning, Real Application Clusters, Automatic Storage Management, OLAP,
```

```
Data Mining and Real Application Testing options

SYS@oraDESDB1>
SYS@oraDESDB1> alter system switch logfile;

System altered.

SYS@oraDESDB1> col group_# format a5
SYS@oraDESDB1> col PROCESS format a8
SYS@oraDESDB1> col CLIENT_PID format a8
SYS@oraDESDB1> set line 9999 pagesize 9999
SYS@oraDESDB1> SELECT a.INST_ID,
  2         a.PROCESS,
  3         a.client_process,
  4         a.client_pid,
  5         a.STATUS,
  6         a.GROUP#         group_#,
  7         a.thread#,
  8         a.SEQUENCE#,
  9         a.DELAY_MINS,
 10         a.RESETLOG_ID,
 11         c.SID,
 12         c.SERIAL#,
 13         a.PID            spid,
 14         b.PNAME
 15    FROM gV$MANAGED_STANDBY a, gv$process b, gv$session c
 16   WHERE a.PID = b.SPID
 17     and b.ADDR = c.PADDR
 18     and a.INST_ID = b.INST_ID
 19     and b.INST_ID = c.INST_ID
 20  order by a.INST_ID,b.PNAME;

   INST_ID PROCESS  CLIENT_P CLIENT_P STATUS       GROUP    THREAD#  SEQUENCE# DELAY_MINS RESETLOG_ID        SID    SERIAL#       SPID PNAME
---------- -------- -------- -------- ------------ ----- ---------- ---------- ---------- ---------- ---------- ---------- ---------- -----
         1 ARCH     ARCH     19595330 CLOSING      3              1        101          0  916055651         47          3   19595330 ARC0
         1 ARCH     ARCH     18874390 CONNECTED    N/A            0          0          0          0         88          1   18874390 ARC1
         1 ARCH     ARCH     15859954 CONNECTED    N/A            0          0          0          0        170          1   15859954 ARC2
         1 ARCH     ARCH     11534542 CLOSING      N/A            1        101          0  916055651        253          1   11534542 ARC3
         2 ARCH     ARCH     14090284 CLOSING      N/A            2         83          0  916055651         87          1   14090284 ARC0
         2 ARCH     ARCH     14745734 CLOSING      4              2         83          0  916055651        129          1   14745734 ARC1
         2 ARCH     ARCH     13500518 CONNECTED    N/A            0          0          0          0        169          1   13500518 ARC2
         2 ARCH     ARCH     13303964 CONNECTED    N/A            0          0          0          0        211          1   13303964 ARC3

8 rows selected.
```

在备库查询：

```
SYS@oraESKDB1> set line 9999 pagesize 9999
SYS@oraESKDB1> col message format a85
SYS@oraESKDB1> SELECT inst_id, severity, FACILITY, TIMESTAMP, MESSAGE
  2    FROM (select d.inst_id,
  3                 FACILITY,
  4                 SEVERITY,
  5                 TIMESTAMP,
  6                 MESSAGE,
  7                 rank() over(partition by d.inst_id ORDER BY d.message_num desc) rank_order
  8            from gv$dataguard_status d)
  9   where rank_order <= 5
 10   order by inst_id, rank_order desc;

   INST_ID SEVERITY      FACILITY                 TIMESTAMP           MESSAGE
---------- ------------- ------------------------ ------------------- -------------------------------------------------------------------------------------
         1 Control       Log Transport Services   2016-07-06 14:59:09 ARC0: Beginning to archive thread 1 sequence 104 (5565428-5589436)
         1 Control       Log Transport Services   2016-07-06 14:59:09 ARC0: Completed archiving thread 1 sequence 104 (0-0)
         1 Informational Log Apply Services       2016-07-06 14:59:09 Media Recovery Log /arch/2_85_916055651.dbf
         1 Informational Log Apply Services       2016-07-06 14:59:09 Media Recovery Log /arch/1_104_916055651.dbf
         1 Warning       Log Apply Services       2016-07-06 14:59:10 Media Recovery Waiting for thread 1 sequence 105
         2 Informational Log Transport Services   2016-07-06 08:50:30 ARC2: Becoming the active heartbeat ARCH
         2 Error         Log Transport Services   2016-07-06 08:50:31 Error 12541 received logging on to the standby
         2 Warning       Log Transport Services   2016-07-06 08:50:31 Check whether the listener is up and running.
         2 Error         Log Transport Services   2016-07-06 08:50:31 FAL[client, ARC0]: Error 12541 connecting to oraESKDB for fetching gap sequence
         2 Informational Log Apply Services       2016-07-06 08:51:44 Managed Standby Recovery starting Real Time Apply

10 rows selected.
```

说明 LGWR 进程异步传输日志，后台进程表现为 nsa，且视图 V$MANAGED_STANDBY 中表现为 LNS，告警日志中表现为 LNS: Standby redo logfile selected for thread 1 sequence 98 for destination LOG_ARCHIVE_DEST_2 。

### 2.2.2.4 默认传输模式测试

```
SYS@oraDESDB1> SELECT a.VALUE FROM v$parameter a WHERE a.NAME='log_archive_dest_2';
```

```
VALUE
--------------------------------------------------------------------------------------------------------------------------------
SERVICE=oraESKDB ARCH SYNC VALID_FOR=(ONLINE_LOGFILES,PRIMARY_ROLE) DB_UNIQUE_NAME=oraESKDB

SYS@oraDESDB1> SELECT a.PROCESS,a.TRANSMIT_MODE FROM V$ARCHIVE_DEST a WHERE a.DEST_NAME='LOG_ARCHIVE_DEST_2';

PROCESS   TRANSMIT_MOD
--------- ------------
ARCH      SYNCHRONOUS

SYS@oraDESDB1> alter system set log_archive_dest_2='SERVICE=oraESKDB db_unique_name=oraESKDB valid_for=(ONLINE_LOGFILES,PRIMARY_ROLE)' sid='*';

System altered.

SYS@oraDESDB1> SELECT a.PROCESS,a.TRANSMIT_MODE FROM V$ARCHIVE_DEST a WHERE a.DEST_NAME='LOG_ARCHIVE_DEST_2';

PROCESS   TRANSMIT_MOD
--------- ------------
LGWR      ASYNCHRONOUS

SYS@oraDESDB1> SELECT a.VALUE FROM v$parameter a WHERE a.NAME='log_archive_dest_2';

VALUE
--------------------------------------------------------------------------------------------------------------------------------
SERVICE=oraESKDB db_unique_name=oraESKDB valid_for=(ONLINE_LOGFILES,PRIMARY_ROLE)
```

可以看出 11g 下，默认情况为 LGWR 的异步模式。

## 2.3 实验总结

| 版本 | 10g | | | 11g | | |
|---|---|---|---|---|---|---|
| | LGWR ASYNC(异步) | LGWR SYNC(同步) | ARCH | LGWR ASYNC(异步) | LGWR SYNC(同步) | ARCH |
| 后台进程表现(ps -ef\|grep -v grep\|grep -E "ora_lns\|ora_nsa\|ora_nss" | ora_lns1_mydg | ora_lnsb_mydg | ora_arc3_mydg | ora_nsa2_mydg | ora_nss2_mydg | ora_arc3_mydg |

| ) | | | | | | |
|---|---|---|---|---|---|---|
| 视图 GV$MANAGED_STANDBY | LNS | LGWR | ARCH | LNS | LGWR | ARCH |
| 切换日志的时候告警日志 | 出现过一次，LNS: Standby redo logfile selected for thread 1 sequence 13 for destination LOG_ARCHIVE_DEST_2，但再切换的时候就不出现了 | 无 | 无 | LNS: Standby redo logfile selected for thread 1 sequence 98 for destination LOG_ARCHIVE_DEST_2 | LNS: Standby redo logfile selected for thread 1 sequence 98 for destination LOG_ARCHIVE_DEST_2 | ARC0: Standby redo logfile selected for thread 1 sequence 102 for destination LOG_ARCHIVE_DEST_2 |
| 是否默认 | 否 | 否 | 是 默认归档同步 | 是 ,默认归档异步 | 否 | 否 |

# 第 3 章　实验中用到的 SQL 总结

## 3.1 10g

```
col group_# format a5
col PROCESS format a8
col CLIENT_PID format a8
set line 9999 pagesize 9999
SELECT a.INST_ID,
       a.PROCESS,
       a.client_process,
       a.client_pid,
       a.STATUS,
       a.GROUP#        group_#,
```

```
      a.thread#,
      a.SEQUENCE#,
      a.DELAY_MINS,
      a.RESETLOG_ID,
      c.SID,
      c.SERIAL#,
      a.PID          spid
  FROM gV$MANAGED_STANDBY a, gv$process b, gv$session c
 WHERE a.PID = b.SPID
   and b.ADDR = c.PADDR
   and a.INST_ID = b.INST_ID
   and b.INST_ID = c.INST_ID
order by a.INST_ID;


set line 9999
col  DEST_NAME format a20
col DESTINATION format a15
col GAP_STATUS format a10
col DB_UNIQUE_NAME format a15
col error format a10
col APPLIED_SCN for 999999999999999
SELECT al.thread#,
      ads.dest_id,
      ads.DEST_NAME,
      (SELECT ads.TYPE || ' ' || ad.TARGET
         FROM v$archive_dest AD
        WHERE AD.DEST_ID = ADS.DEST_ID) TARGET,
      ADS.DATABASE_MODE,
      ads.STATUS,
      ads.error,
      ads.RECOVERY_MODE,
      ads.DB_UNIQUE_NAME,
      ads.DESTINATION,
      (SELECT MAX(sequence#) FROM v$log na WHERE na.thread# = al.thread#) Current_Seq#,
      MAX(sequence#) Last_Archived,
      max(CASE
           WHEN al.APPLIED = 'YES' AND ads.TYPE <> 'LOCAL' THEN
            al.sequence#
          end) APPLIED_SEQ#
  FROM (SELECT *
         FROM v$archived_log V
        WHERE V.resetlogs_change# =
              (SELECT d.RESETLOGS_CHANGE# FROM v$database d)) al,
      v$archive_dest_status ads
 WHERE al.dest_id(+) = ads.dest_id
   AND ads.STATUS != 'INACTIVE'
 GROUP BY al.thread#,
```

```
        ads.dest_id,
        ads.DEST_NAME,
        ads.STATUS,
        ads.error,
        ads.TYPE,
        ADS.DATABASE_MODE,
        ads.RECOVERY_MODE,
        ads.DB_UNIQUE_NAME,
        ads.DESTINATION
 ORDER BY al.thread#, ads.dest_id;


col name for a100
set linesize 9999  pagesize 9999
col NEXT_CHANGE# for 9999999999999999
SELECT THREAD#,
      NAME,
      sequence#,
      archived,
      applied,
      a.NEXT_CHANGE#
FROM   v$archived_log a
WHERE  a.sequence# >= (select max(b.sequence#)-5 from v$log b where b.THREAD#=a.THREAD# )
AND    resetlogs_change# = (SELECT d.RESETLOGS_CHANGE# FROM v$database d)
ORDER  BY a.THREAD#,
        a.sequence#;


SELECT a.VALUE FROM v$parameter a WHERE a.NAME='log_archive_dest_2';
SELECT a.PROCESS,a.TRANSMIT_MODE FROM V$ARCHIVE_DEST a WHERE a.DEST_NAME='LOG_ARCHIVE_DEST_2';
alter system set log_archive_dest_2='SERVICE=tns_mydgwl db_unique_name=mydgwl valid_for=(ONLINE_LOGFILES,PRIMARY_ROLE)' sid='*';
SELECT a.VALUE FROM v$parameter a WHERE a.NAME='log_archive_dest_2';
SELECT a.PROCESS,a.TRANSMIT_MODE FROM V$ARCHIVE_DEST a WHERE a.DEST_NAME='LOG_ARCHIVE_DEST_2';
```

## 3.2 11g

```
col group_# format a5
col PROCESS format a8
col CLIENT_PID format a8
set line 9999 pagesize 9999
SELECT a.INST_ID,
      a.PROCESS,
      a.client_process,
```

```
      a.client_pid,
      a.STATUS,
      a.GROUP#        group_#,
      a.thread#,
      a.SEQUENCE#,
      a.DELAY_MINS,
      a.RESETLOG_ID,
      c.SID,
      c.SERIAL#,
      a.PID           spid,
      b.PNAME
  FROM gv$MANAGED_STANDBY a, gv$process b, gv$session c
 WHERE a.PID = b.SPID
   and b.ADDR = c.PADDR
   and a.INST_ID = b.INST_ID
   and b.INST_ID = c.INST_ID
order by a.INST_ID,b.PNAME;


set line 9999
col  DEST_NAME format a20
col DESTINATION format a15
col GAP_STATUS format a10
col DB_UNIQUE_NAME format a15
col error format a10
col APPLIED_SCN for 999999999999999
SELECT al.thread#,
      ads.dest_id,
      ads.DEST_NAME,
      (SELECT ads.TYPE || ' ' || ad.TARGET
         FROM v$archive_dest AD
        WHERE AD.DEST_ID = ADS.DEST_ID) TARGET,
      ADS.DATABASE_MODE,
      ads.STATUS,
      ads.error,
      ads.RECOVERY_MODE,
      ads.DB_UNIQUE_NAME,
      ads.DESTINATION,
      ads.GAP_STATUS,
      (SELECT MAX(sequence#) FROM v$log na WHERE na.thread# = al.thread#) Current_Seq#,
      MAX(sequence#) Last_Archived,
      max(CASE
           WHEN al.APPLIED = 'YES' AND ads.TYPE <> 'LOCAL' THEN
            al.sequence#
         end) APPLIED_SEQ#,
      (SELECT ad.applied_scn
         FROM v$archive_dest AD
        WHERE AD.DEST_ID = ADS.DEST_ID) applied_scn
```

```
 FROM (SELECT *
        FROM v$archived_log V
       WHERE V.resetlogs_change# =
             (SELECT d.RESETLOGS_CHANGE# FROM v$database d)) al,
     v$archive_dest_status ads
WHERE al.dest_id(+) = ads.dest_id
  AND ads.STATUS != 'INACTIVE'
GROUP BY al.thread#,
         ads.dest_id,
         ads.DEST_NAME,
         ads.STATUS,
         ads.error,
         ads.TYPE,
         ADS.DATABASE_MODE,
         ads.RECOVERY_MODE,
         ads.DB_UNIQUE_NAME,
         ads.DESTINATION,
         ads.GAP_STATUS
ORDER BY al.thread#, ads.dest_id;




------------物理 dg 日志应用情况(备库查询为准)
col name for a100
set linesize 9999  pagesize 9999
col NEXT_CHANGE# for 999999999999999
SELECT THREAD#,
      NAME,
      sequence#,
      archived,
      applied,
      a.NEXT_CHANGE#
FROM   v$archived_log a
WHERE  a.sequence# >= (select max(b.sequence#)-3 from v$log b where b.THREAD#=a.THREAD# )
AND    resetlogs_change# = (SELECT d.RESETLOGS_CHANGE# FROM v$database d)
ORDER  BY a.THREAD#,
         a.sequence#;

----物理备库应用进程日志
--select * from gv$dataguard_status d order by d.inst_id,d.timestamp,d.message_num;
set line 9999 pagesize 9999
col message format a85
SELECT inst_id, severity, FACILITY, TIMESTAMP, MESSAGE
  FROM (select d.inst_id,
             FACILITY,
             SEVERITY,
             TIMESTAMP,
```

```
            MESSAGE,
            rank() over(partition by d.inst_id ORDER BY d.message_num desc) rank_order
        from gv$dataguard_status d)
 where rank_order <= 5
 order by inst_id, rank_order desc;


---是否启用实时应用
ps -ef|grep ora_mrp
--通用：select RECOVERY_MODE from v$archive_dest_status;

alter database recover managed standby database using current logfile disconnect from session;
alter database recover managed standby database cancel;

select thread#,low_sequence#,high_sequence# from v$archive_gap;


SELECT a.VALUE FROM v$parameter a WHERE a.NAME='log_archive_dest_2';
SELECT a.PROCESS,a.TRANSMIT_MODE FROM V$ARCHIVE_DEST a WHERE a.DEST_NAME='LOG_ARCHIVE_DEST_2';
alter system set log_archive_dest_2='SERVICE=tns_mydgwl db_unique_name=mydgwl valid_for=(ONLINE_LOGFILES,PRIMARY_ROLE)' sid='*';
SELECT a.VALUE FROM v$parameter a WHERE a.NAME='log_archive_dest_2';
SELECT a.PROCESS,a.TRANSMIT_MODE FROM V$ARCHIVE_DEST a WHERE a.DEST_NAME='LOG_ARCHIVE_DEST_2';
```

---

## About Me

.....

本文作者：小麦苗，只专注于数据库的技术，更注重技术的运用

本文在 ITpub（http://blog.itpub.net/26736162）和博客园(http://www.cnblogs.com/lhrbest)有同步更新

本文地址：http://blog.itpub.net/26736162/viewspace-2121688/

本文 pdf 版：http://yunpan.cn/cdEQedhCs2kFz （提取码：ed9b）

小麦苗分享的其它资料：http://blog.itpub.net/26736162/viewspace-1624453/

联系我请加 QQ 好友(642808185)，注明添加缘由

于 2016-07-06 10:00~ 2016-07-07 19:00 在中行完成



.....