DBA Daily Checks

Table of Contents

[1. Database Backups 2](#_Toc477183851)

[1.1. Using the Log Files 2](#_Toc477183852)

[1.2. Using SQL\*Plus 2](#_Toc477183853)

[2. Data Guard Time Lag 4](#_Toc477183854)

[2.1. Using SQL\*Plus 4](#_Toc477183855)

[2.2. Using DGMGRL 5](#_Toc477183856)

The following is a list of the daily checks that must be carried out in Azure environments. This is required until such time as we get full access to OEM (Oracle Enterprise Manager) and can set up proper monitoring.

# Database Backups

## Using the Log Files

These are your top priority. The database backups logs must be checked on the servers as soon as possible each day, especially Mondays as there is a weekend's worth of backups to check.

The primary CFG database is backed up at 03:00 daily, with a level 0 (aka full) backup on Sundays. Other days perform a level 1 incremental backup.

Logs are found at \\Backman01\Rmanbackup\backups\lohs\<database\_name> and should be checked to ensure that:

* The database backup completed successfully;
* The archived logs backup completed successfully;
* The controlfile auto-backup completed successfully.

## Using SQL\*Plus

On the database being backed up, which is normally the primary, but need not be - RMAN can backup the standby if necessary - run the following query to determine the state of the backups over the last few days - this covers the weekends:

set lines 3000 trimspool on pages 2000  
  
select parent\_recid, start\_time, end\_time, object\_type, status, operation   
from v$rman\_status  
where row\_type in ('COMMAND', 'RECURSIVE OPERATION')  
and (  
 -- BACKUP of archived log and database...  
 (object\_type is not null and operation = 'BACKUP')   
 or   
 -- AUTOBACKUP of controlfile and spfile  
 ( object\_type is null and operation like 'CONTROL FILE%')  
 )  
-- Everything from Monday, Sunday, Saturday and Friday, maximum.  
and start\_time >= trunc(sysdate) -3  
order by start\_time desc;

The resulting output can be seen as follows:

PARENT\_ START\_TIME END\_TIME OBJECT\_TYPE STATUS OPERATION  
------- ------------------- ------------------- ----------- --------- ----------------  
 104 2017/03/13 08:55:11 DB INCR FAILED BACKUP  
 102 2017/03/13 03:03:39 2017/03/13 03:03:46 COMPLETED CONTROL FILE AND SPFILE AUTOBACK  
 99 2017/03/13 03:02:08 2017/03/13 03:03:48 ARCHIVELOG COMPLETED BACKUP  
 100 2017/03/13 03:01:10 2017/03/13 03:01:16 COMPLETED CONTROL FILE AND SPFILE AUTOBACK  
 99 2017/03/13 03:00:19 2017/03/13 03:01:19 DB INCR COMPLETED BACKUP  
 97 2017/03/11 03:03:30 2017/03/11 03:03:36 COMPLETED CONTROL FILE AND SPFILE AUTOBACK  
 94 2017/03/11 03:01:32 2017/03/11 03:03:39 ARCHIVELOG COMPLETED BACKUP  
 95 2017/03/11 03:00:41 2017/03/11 03:00:48 COMPLETED CONTROL FILE AND SPFILE AUTOBACK  
 94 2017/03/11 03:00:07 2017/03/11 03:00:51 DB INCR COMPLETED BACKUP  
 92 2017/03/10 17:05:42 2017/03/10 17:05:49 COMPLETED CONTROL FILE AND SPFILE AUTOBACK  
 89 2017/03/10 17:03:11 2017/03/10 17:05:51 ARCHIVELOG COMPLETED BACKUP  
 90 2017/03/10 17:02:22 2017/03/10 17:02:28 COMPLETED CONTROL FILE AND SPFILE AUTOBACK  
 89 2017/03/10 17:01:09 2017/03/10 17:02:31 DB INCR COMPLETED BACKUP  
 87 2017/03/10 03:04:15 2017/03/10 03:04:21 COMPLETED CONTROL FILE AND SPFILE AUTOBACK  
 84 2017/03/10 03:01:50 2017/03/10 03:04:24 ARCHIVELOG COMPLETED BACKUP  
 85 2017/03/10 03:01:07 2017/03/10 03:01:13 COMPLETED CONTROL FILE AND SPFILE AUTOBACK  
 84 2017/03/10 03:00:11 2017/03/10 03:01:16 DB INCR COMPLETED BACKUP

In the event of a problem, for example, the first line of output above, use the PARENT\_RECID to query VRMAN\_OUTPUT, as follows:

select output  
from v$rman\_output  
-- Use the parent\_id for the failed session...  
where session\_recid=104  
and output <> ' '  
order by recid asc;

The output will show what would have been found in the RMAN log for that particular backup. If no errors are listed, then you can be certain that a DBA abandoned the backup - as any other errors from RMAN will be shown.

# Data Guard Time Lag

## Using SQL\*Plus

Run the following on the *primary database*:

select dest\_id, Dest\_name, destination, archived\_seq#, applied\_seq#,  
error, db\_unique\_name, gap\_status  
from v$archive\_dest\_status  
where status <> 'INACTIVE'  
and dest\_id in (2,3);

The results will resemble the following:

DEST\_ID DEST\_NAME DESTINATION ARCHIVED\_SEQ# APPLIED\_SEQ# ERROR DB\_UNIQUE\_NAME GAP\_STATUS  
------- ------------------ ----------- ------------- ------------ ----- -------------- ----------  
 2 LOG\_ARCHIVE\_DEST\_2 cfgsb 15975 15974 cfgsb NO GAP  
 3 LOG\_ARCHIVE\_DEST\_2 CFGDR 15975 15974 CFGDR NO GAP

"NO GAP" is what you are hoping to see in the GAP\_STATUS column. Other values that may appears here are:

|  |  |
| --- | --- |
| Status | Description |
| NO GAP | The desired result. There is not an apply gap. |
| LOG SWITCH GAP | The destination has not yet received all of the redo information from the most recently archived log file. |
| RESOLVAVBLE GAP | The destination has a redo gap but it can be resolved by fetching, automatically, the missing archived log files from *this* database. No action is required, unless FAL\_SERVER for this database does not point at the *this* database. |
| UNRESOLVAVBLE GAP | The destination has a redo gap which can not be resolved by fetching, automatically, the missing archived log files from *this* database, and there are no other destinations (standbys) where the missing information can be obtained. Action is required in this case as the standby database is not up to date, and cannot be brought up to date. (Someone deleted an archive log or two perhaps?) |
| LOCALLY UNRESOLVAVBLE GAP | The destination has a redo gap which can not be resolved by fetching, automatically, the missing archived log files from *this* database, however, other destinations (standbys) *may* be able to assist in resolving the missing data. Action is required in this case, but only to monitor that the standby doesn't get further and further out of date. |

In the above, where you see "*this* database", *this* refers to the database that the query was executed on - in our case, the *primary* database.

## Using DGMGRL

Run the following in dgmgrl on any of the servers, primary, standby or DR. You need to be logged in as the SYS user. The first command, show configuration, simply displays the names of the various databases configured:

show configuration  
  
Configuration - dgmgrl\_configuration  
  
 Protection Mode: MaxPerformance  
 Databases:  
 cfg - Primary database  
 cfgsb - Physical standby database  
 cfgdr - Physical standby database  
  
Fast-Start Failover: DISABLED  
  
Configuration Status:  
SUCCESS

The database names shown above are in lower case. This means that we can use them as-is. If they were in upper case, we would need to wrap them in double quotes, and in upper case too. Show database "CFGSB" for example.

The next commands display the two, in this case, standby databases:

DGMGRL> show database cfgsb  
  
Database - cfgsb  
  
 Role: PHYSICAL STANDBY  
 Intended State: APPLY-ON  
 Transport Lag: 0 seconds (computed 0 seconds ago)  
 Apply Lag: 0 seconds (computed 0 seconds ago)  
 Apply Rate: 230.00 KByte/s  
 Real Time Query: OFF  
 Instance(s):  
 cfgsb  
  
Database Status:  
SUCCESS

and:

DGMGRL> show database cfgdr  
  
Database - cfgdr  
  
 Role: PHYSICAL STANDBY  
 Intended State: APPLY-ON  
 Transport Lag: 0 seconds (computed 0 seconds ago)  
 Apply Lag: 1 second (computed 0 seconds ago)  
 Apply Rate: 211.00 KByte/s  
 Real Time Query: OFF  
 Instance(s):  
 cfgdr  
  
Database Status:  
SUCCESS

In either case we are looking to see that there isn't a (large) transport or apply lag and that the "computed" time is not excessive.

Various checks that can be applied when the status is not as desired are:

* Has the standby database just been started up after a while? If so, there will be potentially a large amount of un-applied redo to be obtained from the primary. Monitor the status and ensure that the lags do not simply keep increasing.
* Has the standby lost its managed apply? Check the alert log for details, stop and restart managed apply as per the document on building standby databases using RMAN.
* Has the network failed between the primary and standby? Try using tnsping standby to determine if this is the case. Once the network problems are alleviated, the standby should automatically start catching up.
* Has the primary database been run in NOACHIVELOG for a while? If so, the standby databases must be recreated.