Data Guard 12.2 New Features

By: Charles Kim, Viscosity North America

Oracle ACE Director, January 2017

Platinum Partner

Oracle Database 12c Release 2, packs a multitude of new features for Data Guard with high availability, data protection, and disaster recovery. Through the new functionality shared in this paper, DBAs can provide better protection for mission critical production databases from human errors, data corruptions, failures, and disasters. With the new features in Oracle 12.2, DBAs can deliver a robust reporting environment while addressing corporate disaster recovery goals.

Create a Standby Database with DBCA

Starting with this release, we can leverage the Database Configuration Assistant (DBCA) command-line interface, to instantiate the Data Guard standby database from the primary database. Leveraging the DBCA command-line interface, will eliminate many of the manual steps needed to create a standby database. With the DBCA command-line interface, we can also embed post instantiation scripts to be executed with the –customScripts option, as a post standby database instantiation process. For multiple scripts, we can supply a list of script names followed by a comma. The scripts will be executed in the order of the supplied list.

As of this release, several restrictions apply. First, only non-multitenant, primary databases can be instantiated as standby databases. Also, RAC physical standby databases cannot be created. To convert the single instance database to RAC, DBAs can leverage OEM Cloud Control.

DBCA offers a new -createDuplicateDB parameter, to duplicate a database from the primary database. The –createAsStandby parameter, is available to duplicate a standby database from the primary database. Additional options for dbca with the –createDuplicateDB parameter are:

```
dbca -createDuplicateDB
    -gdbName global_database_name
    -primaryDBConnectionString easy_connect_string_to_primary
    -sid database_system_identifier
    [-createAsStandby
        [-dbUniqueName db_unique_name_for_standby]]
    [-customScripts scripts_list]
```

For the –primaryDBConnectionString, we can provide the easy connect string in the form of `"host[:port][/service_name][:server][/instance_name]"`.

This eliminates the need to create a TNSNAMES.ORA entry.

Support for Diagnostic Pack with ADG

When running in read-only mode, with Active Data Guard (ADG), DBAs can fully leverage the diagnostic pack to capture performance metrics data to the AWR. This also allows the DBAs to take advantage of ADDM, from the AWR repository.

AWR snapshots, on the ADG standby database are called remote snapshots. A database node is referred to as a destination.

A destination, is where we store snapshots that are collected from remote, ADG standby database nodes. Remote ADG standby database nodes, are referred to as sources. Each source must have two database links, one for destination to source database link and another one for source to destination database link.

We can take remote snapshots manually or automatically at a scheduled interval, and are started by a destination node.
In summary:

- The destination creates the snapshot.
- The sources push snapshot data to the destination via DB Links.
- DBAs can pull AWR reports on snapshot data on the destination.
- DBAs can leverage ADDM to access AWR data for performance metrics.

Support for SQL Tuning Advisor

In this release, DBAs can initiate the SQL Tuning Advisor on the primary database and execute the SQL statements on a remote database. DBAs can offload the tuning workload of the primary database, to the Active Data Guard standby database. The SQL tuning process is initiated on the primary database, but the SQL tuning process workload is executed on the Active Data Guard standby database; while the database is open for read-only operations. The results of the SQL tuning efforts are written back to the primary database over database links. SQL profile recommendations, on the primary database, are applied to the Active Data Guard standby database, using the standard redo apply mechanism.

NoLogging Changes for Data Guard

Nologging activities to tables and indexes, on the primary database, have always caused havoc on Data Guard. Developers and even DBAs continue to execute SQL commands with the nologging option, believing that the operation is not logged in the online redo stream. Now, nologging blocks are recorded in the control file on the physical standby database. On the physical standby database, we can execute the new RMAN RECOVER DATABASE NONLOGGED BLOCK, to recover nonlogged blocks.

Before we issue the RMAN RECOVER DATABASE NONLOGGED BLOCK, we must first stop managed recovery process. In the event we encounter unrecoverable blocks after a switchover, the old primary database must be in a mounted state.

Multiple Observers

Oracle added the functionality, for up to three observers, to monitor and support a single Oracle Data Guard Broker configuration. Each observer must be assigned a name, and the name assigned to the observer must be unique in the configuration. The name of the observer is also case sensitive. When we start the observers with DGMGRL, the START OBSERVER command is enhanced to accept the name of the observer.

In a three observer configuration, we have a concept of the master observer and two backup observers. When fast-start failover (FSFO) is initiated, the primary and standby database will randomly choose from the list of registered observers and designate a master observer. If there is no observer registered, then the first observer that is started becomes the master observer. The subsequent observers that join the FSFO configuration, will become the backup observers.

Only the master observer has the privilege of coordinating the FSFO with the Data Guard broker. Other registered observers serve the role of backup observers, until the master observer is not available.

Observer placement continues to be a critical component to the Data Guard topology. Oracle has always recommended to place the observer in another data center. With multiple observers in play, the same recommendation still holds true. Now, with additional observers, there are other factors to consider. We should never place the observers on the same server or the same virtual server. We should also consider placing one or more of the observers in separate data centers.

The v$database view introduces two additional columns FS_FAILOVER_OBSERVER_HOST and FS_FAILOVER_OBSERVER_PRESENT. The FS_FAILOVER_OBSERVER_HOST displays the name of the host where the master observer is running from. The FS_FAILOVER_OBSERVER_PRESENT column, designates if the master observer is associated with the local database and displays a value of either YES or NO.

The view V$FS_FAILOVER_OBSERVERS, displays all the observers in the FSFO configuration.

```
SQL> desc V$FS_FAILOVER_OBSERVERS
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 NAME                                               VARCHAR2(513)
 REGISTERED                                         VARCHAR2(4)
 HOST                                               VARCHAR2(513)
 ISMASTER                                           VARCHAR2(4)
 TIME_SELECTED                                      TIMESTAMP(9)
 PINGING_PRIMARY                                    VARCHAR2(4)
 PINGING_TARGET                                     VARCHAR2(4)
 CON_ID                                             NUMBER
```

This view also displays the following information:

- Observer name
- The host that it resides on,
- If the observer is the master observer
- When the master observer became the master observer
- If the master observers is connected to the primary and/or physical standby database

For additional information, please visit the following URL:
http://docs.oracle.com/database/122/DGBKR/using-data-guard-broker-to-manage-switchovers-failovers.htm#DGBKR394

Simplified Observer Management

With a single DGMGRL broker session, we can now monitor and manage multiple Observers from fast-start failover configurations. This increases the operational efficiencies, thus reducing the cost of managing multiple Data Guard databases that have fast-start failover configurations.

Multiple Instance Redo Apply (MIRA) in RAC

Starting in Oracle 12.2, we can run Redo Apply all or on some standby instances. With this concept of multiple instance redo apply (MIRA), Redo Apply performance can scale as wide as the target RAC configuration allows. This feature is crucial for Exadata and RAC customers with demanding high workloads on the primary database. For Active Data Guard customers, they can have real-time access to the data being churned on the primary database.

The ALTER DATABASE RECOVER MANAGED STANDBY DATABASE command, now accepts new INSTANCES [ALL | integer] clause to start Redo Apply on multiple instances. The ALL option, starts redo apply on all the RAC standby instances that are in open or mount mode. All the instances must be in the same mounted or open mode; one instance cannot be in open mode (Active Data Guard or read-only mode) while others are in mounted mode. The integer option, specifies the number of RAC standby instances that will perform redo apply. We cannot specify which RAC instance(s) will perform the redo apply.

Starting redo apply on multiple instances has the following restrictions:

- In-Memory column store is not supported.
- Block Change tracking (BCT) is not supported.

The multi-thread redo stream from RAC primary, is shipped to the multiple-node RAC standby instances. With multi-node MRP, redo apply performance is now directly correlated to network bandwidth and latency between the primary and standby database environments.

With the DGMGRL command line interface, we can configure which RAC instances, in the physical standby environment, apply processes should be executed to use the new Oracle Active Data Guard multiple instance Redo Apply feature.

Subset Standbys

When the Multi-tenant option was introduced in Oracle Database 12c Release 1 (12.1), the physical standby was at the container level and all pluggable databases (PDBs) had to participate in the physical standby configuration. As of Oracle Database 12c Release 2, Oracle added a new feature that allows for the number of PDBs to be replicated to the physical standby, CDB database. Prior to this release, the ENABLED_PDBS_ON_STANDBY initialization parameter only supported two values: all PDBs or none. Now, we can provide a list of PDBs to the ENABLED_PDBS_ON_STANDBY initialization parameter.

The ENABLED_PDBS_ON_STANDBY parameter, is only applicable on the physical standby database. If this parameter is set on the primary databases, it is ignored during the startup of the database. If you use this initialization parameter, we recommend that you also set this parameter on the primary database; in case you perform a switchover or failover and your primary one day becomes your standby database. In the absence of this parameter, all PDBs in the CDB are created on the standby database.

The ENABLED_PDBS_ON_STANDBY parameter, can accept a list of PDB names or a glob pattern such as "VNAPDB?", "VNAPDB*a", or "VNAPDB2". Glob pattern rules, are similar to regular expression rules in common UNIX shells. The common expressions, such as asterisk (*) and question mark (?) wildcard characters are supported. The question mark (?) represents a single unknown character; whereas the asterisk (*) represents matches to any number of unknown characters. This parameter also accepts a minus sign (-), which can be used as the first character in a PDB name, to designate that the PDB should be excluded on the standby database. PDB names and rules associated with the PDBs, should be enclosed with double quotation marks. Oracle will remove the double quotation mark before processing the PDB list. Here are several examples of this parameter usage:

- ENABLED_PDBS_ON_STANDBY="*" means that all PDBs will be created on physical standby.
- ENABLED_PDBS_ON_STANDBY="VNAPDB1*" means that VNAPDB1A, VNAPDB1B, and VNAPDB1C will be created on the physical standby.
- ENABLED_PDBS_ON_STANDBY="VNAPDB*A" means that VNAPDB1A, VNAPDB2A, and VNAPDB3A will be created on physical standby.
- ENABLED_PDBS_ON_STANDBY="VNAPDB1*", "-VNAPDB*A" means that VNAPDB1B and VNAPDB1C will be created on physical standby but VNAPDB1A will be excluded

In the CREATE PLUGGABLE DATABASE … STANDBYS= statement, a new option is introduced in Oracle 12.2. We have an EXCEPT clause to designate which CDB we do not want this PDB to be excluded from. In Oracle 12.1, the CREATE PLUGGABLE DATABASE … STANDBYS= accepted:

- a list of CDBs that you want this PDB to replicate to: {('CDB1', 'CDB2', ...)
- NONE
- ALL

Now, the ALL option is enhanced to specify exclusion of CDBs: ALL [EXCEPT ('MYCDB', 'YOURCDB', ...)

Data Guard Database Compare

A new PL/SQL procedure, called DBMS_DBCOMP.DBCOMP, is introduced to detect lost writes and to identify inconsistencies between the primary and physical standby database. This supplied procedure compares the matching data blocks on the primary and physical standby databases. The coolest thing about this parameter is that it does not require the DB_LOST_WRITE_PROTECT parameter to be set. This procedure can be executed at any time, and you can monitor the progress by querying the V$SESSION_LONGOPS view. The dbverify utility cannot detect for lost write disk errors; instead the DBA would leverage this procedure to detect for silent corruptions introduced by the storage array at the physical standby database. Oracle already validates data being read or changed on both the primary or standby database. This stored procedure can provide comprehensive validation of the entire database including dormant data.

The DBCOMP procedure can be executed on the primary or on the physical standby with the database in MOUNT or OPEN mode. This procedure accepts three parameters:

```
DBMS_DBCOMP.DBCOMP (
  datafile IN varchar2,
  outputfile IN varchar2,
  block_dump IN boolean);
```

The datafile can be a number, name of the datafile or ALL for all the datafiles. The output file, is a prefix in the name of the output file. All output is stored in the $ORACLE_HOME/dbs directory and can be modified with either relative or absolute path. The block dump parameter is a Boolean parameter; by default, this parameter is FALSE. We can set this Boolean parameter to TRUE, if we want the content of the block to be dumped into the output file, when a pair of blocks between the primary and standby databases is not the same. Here's a sample code example of the DBCOMP procedure:

```
DECLARE
  DataFile VARCHAR2(1000);
  OutputFile VARCHAR2(1000);
BEGIN
  DataFile := 'all' ;
  OutputFile:='BlockCompareFULL_';
  SYS.DBMS_COMP.DBCOMP(DataFile, OutputFile, true);
END;
/
```

Broker Block Comparison Tool

The Data Guard Broker is enhanced to take advantage of this feature too. The new command VALIDATE DATABASE DATAFILE, provides the same feature as the DBCOMP procedure. Just like the DBCOMP procedure, we can validate the database at the datafile level or at the database level. Below are options for the VALIDATE DATABASE DATAFILE command with the Data Guard Broker:

```
VALIDATE DATABASE [database-name  |  ALL] DATAFILE [datafile-name  | datafile-
number | ALL] OUTPUT="output-file-name";
```

The output file is generated in the trace directory. Similar to the DBCOMP procedure, we can specify the datafile name, datafile number or the ALL option for all the database files. Below is an usage example:

```
DGMGRL> VALIDATE DATABASE prod DATAFILE ALL OUTPUT=BlockCompareFULL_prod.out;
```

Automatic Password File Synchronization

Effective Oracle 12.2, when the password for SYS, SYSDBA, SYSOPER and/or SYSDG are modified, the password file is updated on the primary database. The changes made on the password file, are replicated to the password files on all the ORACLE_HOME databases on the Data Guard configuration. The password file is updated on the physical standby database server when redo is applied. For far sync configurations, the changes to the password file must be manually copied. This is because a far sync server receives redo information, but does not apply redo. When the password file is updated on the far sync server, the password will automatically be propagated to the target, physical standby, database servers.

In-Memory Support for ADG

Starting in Oracle Database 12c Release 2, the In-Memory (IM) Column Store is supported on the standby database, if you are running Active Data Guard. The IM option can be configured on the primary database, on an ADG standby database, or on both the primary and the ADG standby databases.

On the Active Data Guard environment, the INMEMORY_ADG_ENABLED parameter needs to be enabled in addition to the in-memory cache size. By default, the INMEMORY_ADG_ENABLED parameter is set to true.

This parameter is only applicable on the standby databases. For RAC configuration, this parameter must be set to the same value across all the RAC instances. The parameter has no relevance on a primary database.

Minimize Impact to Primary Databases with Multiple Sync Standby Databases

Oracle introduced a new initialization parameter called DATA_GUARD_SYNC_LATENCY, which allows you to specify how long the primary database Log Writer (LGWR) should wait for a response, from multiple synchronous standby databases during redo transport. The default value of 0, specifies that the LGWR process will wait until the number of seconds specified by the NET_TIMEOUT attribute of the LOG_ARCHIVE_DEST_N parameter. The NET_TIMEOUT redo transport attribute, specifies the duration in seconds for how long the primary database needs to wait for a response, from each of the standby database in SYNC redo transport.

Prior to Oracle 12.2, with multiple synchronous standby databases, the primary database must wait for all synchronous standby databases to acknowledge receipt of the redo or exceed their individual NET_TIMEOUT period before continuing. This DATA_GUARD_SYNC_LATENCY parameter defines the number of seconds that the primary database must wait; once one of the synchronous standby databases acknowledges receipts of the redo. Other synchronous database destinations must return receipt within this threshold or become disconnected from the primary database.

Here's an example with four synchronous standby destinations and the DATA_GUARD_SYNC_LATENCY parameter is set to 2 (2 seconds). If the first standby database acknowledges redo receipt immediately, the remaining three standby databases have up to 2 seconds to respond with acknowledgement of redo receipt. The primary database will not wait more than the specified threshold specified in the DATA_GUARD_SYNC_LATENCY parameter. If one or more of the remaining synchronous, physical standby configurations fail to acknowledge redo receipt, the LGWR will disconnect from the standby database and put the destination in error state. The primary database still operates in zero data loss, Maximum Protection mode, since one of the synchronous standby databases has acknowledged receipt of redo. Once the duration of the REOPEN attribute seconds have elapsed, LGWR will reconnect to the failed synchronous standby databases.

You cannot set the value of DATA_GUARD_SYNC_LATENCY to be greater than the value of NET_TIMEOUT. LGWR will not wait longer than the value of NET_TIMEOUT attribute of the LOG_ARCHIVE_DEST_n parameter.

DUPLICATE Command Enhancements for DG

As of Oracle 12.1, we are restricted to creating a physical standby database when we are connected to a physical standby database, as the target database with the DUPLICATE command. In Oracle 12.2, Oracle extends the DUPLICATE command to create a database from the physical standby database. Oracle also enhanced the DUPLICATE command to create the far sync standby instance.

We can substitute the keyword STANDBY with FARSYNC on the command line (the DORECOVER option not allowed for far sync instances) to create a farsync instance. The following command will create a far sync instance from the active database:

```
DUPLICATE TARGET DATABASE FOR FARSYNC FROM ACTIVE DATABASE;
```

We can also create a Data Guard far sync instance from a backup-based duplication. Using the previous command above, we simply exchange FROM ACTIVE DATABASE to BACKUP LOCATION '+DATA/backup':

```
DUPLICATE TARGET DATABASE FOR FARSYNC BACKUP LOCATION '+DATA/BACKUP';
```

Connection Preservation During Role Changes

When a role transition happens where the Active Data Guard configuration becomes the new primary database, all connections established on the ADG are terminated; and connections must be re-established which causes state information to be lost. Starting with Oracle 12.2, connections already established on the ADG database will not be disconnected during a role transition to a primary database. As we incorporate a database service that is architected to run both primary and the standby databases, users will stay connected as the role transition occurs. If we use a database service that only connects to the standby database, user sessions will be terminated and forced to re-connect.

Data Guard Broker

The Data Guard broker is a distributed management framework leveraged to create, manage, maintain, and monitor the Data Guard environment. With each release, Oracle enhances the DG broker. This section describes the new features and capabilities that were added to Oracle Data Guard broker in Oracle Database 12c Release 2:

Oracle Data Guard Broker Support for Executing DGMGRL Command Scripts

The Broker allows for scripts to be executed, like SQL*Plus, with the @ sign (i.e. @scriptname). Each line in the script must have a semi-colon (;) at the end. We cannot execute the command "Start Observer" in the script. All commands after the "Start Observer" command will be ignored. We can, however, use the command "Start Observer In Background;", and Commands subsequent to this command will be accepted. Just like SQL*Plus, we can use the word REM or -- (two dashes followed by a space), to comment a line. Similarly, OS commands such as host or !, can be executed as we do in SQL*Plus.

Broker Support for Redo Transport Destinations of Different Endianess with ZDLRA

With the ZDLRA in play, Oracle Data Guard broker can manage a remote, redo destination that has a different endianess than the primary database. For example, this allows the Oracle Data Guard broker to manage and configure Oracle Data Guard transport services on Linux (or on Exadata or even on Oracle Database Cloud), when the primary database resides on the AIX operating system.

Imagine the capabilities of cross platform migrations with the ZDLRA, and the ability to synchronize databases of heterogeneous configurations with different endianness. This feature will significantly improve the flexibility of migrating databases from one platform to another with ZDLRA.

Broker Support for Multiple Automatic Failover Targets

Starting in Oracle 12.2, we have the capability to configure multiple failover targets, in a fast-start failover configuration. We can designate one or more failover targets to achieve a higher level of success for automatic failover, when the need arises. It is also possible to set one or more DB_UNIQUE_NAME to the Data Guard Broker, FastStartFailoverTarget property. The list of target standby databases, are called candidate fast-start failover targets. These candidate, fast-start, failover targets cannot be a far sync instance, a snapshot standby database or a ZDLRA. The FastStartFailoverTarget property, can be set to the list of DB_UNIQUE_NAME, and the broker will attempt failover in the order they are listed. If the property is set to the keyword ANY, then the broker can select any of the candidate targets as the current, fast-start, failover target.

To change the FastStartFailoverTarget property for the candidate fast-start failover targets, we must disable fast-start failover, modify the FastStartFailoverTarget property, and re-enable fast-start failover.

Support for ADG Rolling Upgrades

Starting in Oracle 12.2, the broker does not have to be disabled during a rolling upgrade. In Oracle 12.1, Oracle introduced rolling upgrades for ADG, by simplifying the process of becoming a transient logical standby database by automating the manual steps with the DBMS_ROLLING package. In Oracle 12.2, the broker now supports rolling upgrades. During the rolling upgrade process, the "show configuration" command from DG Broker will report the status of "`ROLLING DATABASE MAINTENANCE IN PROGRESS`".

During the rolling upgrade process, FSF must be disabled. With the DG Broker managed rolling upgrades, DBAs will be able to deliver more reliable rolling upgrades with minimal downtime and risk.

Data Guard Broker PDB – Migrate PDB

Oracle has supplied a new command to the Data Guard Broker, MIGRATE PLUGGABLE DATABASE. This allows movement of a single PDB from one container to another or failover a PDB from the standby database, to a new production container database. Several use cases include:

1. Moving a PDB from one container to another container, on the same server;
2. Failover a PDB from a physical standby database, to a new production database on the same physical standby server.

In either of the scenarios, data files for the PDB being "migrated" must be presented and made available, to both the current container database and the target container database. When you migrate a PDB from one container database to another database, the database version must be equal or higher than the source container. If the PDB being migrated is at a lower version than the target container database, the PDB must be upgraded first prior to being used.

The source and target CDBs must participate in different Data Guard Broker configurations. If the source CDB happens to be a physical standby database, the source and target CDBs must be running on the same version and patches, and must have the same compatible initialization parameter. When the source CBD is a primary database, the target CDB cannot be on a lower version of Oracle. Likewise, the compatible initialization on the target CDB cannot be on a lower version.

Enhancement for Alternate Destinations (GROUP and PRIORITY)

As of Oracle 12.2, the GROUP and PRIORITY attributes of LOG_ARCHIVE_DEST_n parameter have replaced the ALTERNATE attribute for remote destinations. These attributes are used to configure alternate redo transport paths, to the standby database, when the far sync instance is not available.

Fast-Start Failover in Maximum Protection Mode

Now, Oracle Data Guard supports FSFO in Maximum protection mode for zero data loss protection, when multiple synchronous destinations exist.