Minimal Downtime 12c DB-Upgrade und -Split

Olaf Nowatzki
interface systems GmbH
Dresden

Schlüsselworte
Upgrade 12c Minimal Downtime DB-Split TDE Total Recall RAC

Einleitung / Ausgangslage und Anforderungen
In einem Kundenprojekt war eine RAC DB (stretched) Version 11.2.0.4 mit einem Volumen von einigen hundert GB im Einsatz. Von mehreren darin konsolidierten Applikationsschemas verlangten einige ein Upgrade nach 12cR2 und die restlichen einen Verbleib in Version 11.2.0.4. Außerdem sind Flashback Data Archives (= Total Recall, 10 Jahre Retention) und TDE Tablespace Encryption im Einsatz (letztere wegen des Betriebes in einem externen Rechenzentrum mit Wallets OHNE Auto-Login). Die Downtime durfte 15 min nicht überschreiten und ein Hardwarewechsel war nicht vorgesehen.
Der Vortrag soll die Suche nach einer geeigneten Migrations- und Upgrade-Strategie und deren Umsetzung beschreiben.

Variantendiskussion
Die folgenden „gebräuchlichen" Upgrade-Wege wurden bezüglich ihrer (in Auszügen) Vor- und Nachteile sowie Anwendbarkeit in diesem Szenario diskutiert. Wichtig dabei waren die Prämissen:

- „Es muss funktionieren"
- „Es darf nur möglichst wenig Aufwand erfordern (finanziell und zeitlich)"

a) Migration der Daten(schemas) mittels Data Pump

Vorteile: nur Daten für 12c „wandern" nach 12c, sehr universell einsetzbar und mit impliziter Segmentreorganisation, Fallback ist einfach vor Inbetriebnahme der 12c
Nachteile: Dauer stark abhängig vom Datenvolumen, ohne read-only Setzen der Quelle droht Datenzustandsdivergenz, die nur mit höherem Aufwand beseitigt werden kann, FBDAs gehen verloren
Ausschlusskriterium: Verlust der FBDAs, Downtime-Limit nicht einhaltbar

b) Migration der Daten(tablespaces) mittels Data Pump TTS

Vorteile: breit einsetzbar und relativ schnell unabhängig vom Datenvolumen (dies nur, wenn entweder die zu migrierenden TS aus der 11g-DB entfernt werden oder zuvor synchronisierte Datafilecopies erstellt werde, z. B. in einer physical Standby DB)
Nachteile: FBDAs gehen verloren, Fallback schwierig, wenn ohne Kopieren der DB-Files gearbeitet wird
Ausschlusskriterium: Verlust der FBDAs

c) Upgrade einer (blockweisen) DB-Kopie

c1) Upgrade einer (blockweisen) DB-Kopie - offline

Vorteile: FBDAs bleiben voll funktionstüchtig, Fallback ist einfach vor Inbetriebnahme der 12c
Nachteile: temporär reichlich doppelter Platzverbrauch, Länge der Downtime abhängig von der Hardware, vom DB-Volumen (beim Kopieren) und nur begrenzt minimierbar (Dictionary-Upgrade)
Ausschlusskriterium: maximal zugebilligte Downtime nicht erreichbar

c2) Upgrade einer (blockweisen) DB-Kopie – mit physical DG

Vorteile: FBDAs bleiben voll funktionstüchtig, Länge der Downtime unabhängig vom DB-Volumen (beim Kopieren), Fallback ist einfach vor Inbetriebnahme der 12c
Nachteile: temporär reichlich doppelter Platzverbrauch, Länge der Downtime abhängig von der Hardware und nur begrenzt minimierbar (Dictionary-Upgrade)
Ausschlusskriterium: maximal zugebilligte Downtime nicht erreichbar

c3) Upgrade einer (blockweisen) DB-Kopie – mit Golden Gate

Vorteile: breit einsetzbar und relativ schnell unabhängig vom Datenvolumen, einfaches Fallback, FBDAs können erhalten werden, wenn blockweise DB Kopie als Basis des OGG-Replikates verwendet wird
Nachteile: Aufwand, temporär reichlich doppelter Platzverbrauch, es entsteht ein „Streckungs-/Stauchungsintervall“ für die Zeitstempel in den FBDAs während der Laufzeit des Upgrade-Vorganges
Ausschlusskriterium: Aufwand (Lizenz und Implementierung OGG)

c4) Upgrade einer (blockweisen) DB-Kopie – mit physical/logical DG

Vorteile: FBDAs bleiben funktionstüchtig, Länge der Downtime unabhängig vom DB-Volumen (beim Kopieren), Länge der Downtime unabhängig von der Hardware (falls Transaktionsrate erreicht wird, Fallback ist einfach vor Inbetriebnahme der 12c
Nachteile: temporär reichlich doppelter Platzverbrauch, es entsteht ein „Streckungs-/Stauchungsintervall“ für die Zeitstempel in den FBDAs während der Laufzeit des Upgrade-Vorganges, Limitierungen hinsichtlich unterstützter Datentypen
Ausschlusskriterium: keines

Umsetzung

Voraussetzungsprüfungen
Startpunkt war eine Prüfung der Upgradefähigkeit aller Applikationen (Herstellerfreigaben, Testumgebungen, Clientversionen… - ein separates Thema)
Die Ausgangsdatenbank muss grundsätzlich die Voraussetzungen zur Erstellung/Betrieb einer Logical Standby erfüllen, recht schnell überprüfbar z. B. wie in Note 738643.1 zu finden

```
SQL> SELECT OWNER, TABLE_NAME FROM DBA_LOGSTDBY_NOT_UNIQUE WHERE
(OWNER, TABLE_NAME) NOT IN (SELECT DISTINCT OWNER, TABLE_NAME FROM
DBA_LOGSTDBY_UNSUPPORTED) AND BAD_COLUMN = 'Y';
```

Tests

Um den kompletten Ablauf möglichst realitätsnah validieren zu können, wurde das komplette Setup der 11.2.0.4 Produktionsdatenbank (leer) auf einer Testumgebung nachgebaut, sämtliche Arbeitsschritte durchgeführt und dokumentiert und damit ein Drehbuch + Scripts mit Zeitplanung erstellt.
Außerdem wurden in der Produktionsumgebung (nach sowieso erforderlicher Platzerweiterung im ASM) eine RMAN-Duplikation der Original-DB + anschließendes 12c-Upgrade durchgeführt, um Laufzeiten mit Original-Dictionary und -Hardware zu messen und eventuelle Probleme festzustellen.

Vorbereitungen / Upgrade Grid Infrastructure

Das Upgrade der Grid Infrastructure Version von 11.2.0.4 auf Version 12.2.0.1 wurde gemäß der üblichen Best Practices durchgeführt, also mit den Schritten:

1) Schaffung von ausreichend Platzreserven für das GI Management Repository
2) Software-only Installation der GI 12.2.0.1 auf beiden Clusterknoten
3) OPatch-Update und Installation des aktuellen GI-RU
4) Rolling GI Upgrade
5) Entfernung des GI-Homes 11.2.0.4

Außerdem wurden natürlich ein RDBMS-Home 12.2.0.1 mit aktuellem RU installiert + separate Local Listener für die „neue“ 12c DB erstellt und zugehörige Firewall Regeln implementiert.

Erstellung Physical Standby 11.2.0.4

Die physical Standby wurde „klassisch“ per RMAN „duplicate for standby“ erzeugt mit der Besonderheit, dass KEIN Auto-Login Wallet für TDE erstellt wurde (auch nicht vorübergehend – im Interesse der Datensicherheit), sondern die Zeitpunkte zum notwendigen manuellen Öffnen des Wallets per Beobachtung des alert.log des Duplikates erkannt und abgepasst wurden. Danach wird eine DG Broker Konfiguration im Maximum Availability Modus erzeugt.
Synchronisierung-Check per

```
DGMGRL> show configuration
DGMGRL> show database '<DB>'
```

Zur Vorbereitung des nächsten Schrittes:

```
SQL>ALTER DATABASE ADD SUPPLEMENTAL LOG DATA;
```

Umwandlung zur Logical Standby / Syncing

Hierzu wurde zur Aufwandsminimierung das physru.sh Skript aus Note 949322.1 verwendet, welches eigentlich zum Komplett-Upgrade einer DB vorgesehen ist. Jedoch kann ersparte es Aufwand für eigenes Scripting, hier nur den 1. Lauf (von vorgesehenen 3) zu nutzen und (wir wollten ja splitten) eine Modifikation vorzunehmen: Entfernung des KEEP IDENTITY beim Recovery zur Logical Standby, was zur Vergabe einer neuen DB-ID führt und uns anschließend den DB_NAME ändern lässt.
Wichtig zur Vorbereitung:
Auf beiden DG Sites muss der Data Guard Broker gestoppt werden:

```
SQL> alter system set dg_broker_start=false scope=both;
```

Weiterhin muss der bei der Erstellung der DG Broker Konfiguration automatisch gesetzte Parameter LOG_ARCHIVE_DEST_2 um den NET_TIMEOUT reduziert werden.
Eventuell vorhandene in-doubt distributed Transactions müssen zuvor bereinigt und eigene Login-Trigger disabled werden. Dann erfokgt die Transformation zur Logical Standby:

```
bash> ./physru_v3_change_identity.sh sys <DB11g_TNS> <DB12c_TNS>
<DB11g_UNIQUE_NAME>  <DB12c_UNIQUE_NAME> 12.2.0.1.0
```
Durch die Modifikation scheitert das Script zwar mit „`ERROR: failed to verify database role of PZS1 is LOGICAL STANDBY`“, hat jedoch alle hier benötigten Schritte erledigt. Nun kann im SPFILE der 12c Datenbank der neue DB_NAME eingetragen und die DB wird (mit manueller Öffnung des Encryption Wallets und OPEN RESETLOGS) gestartet.
Recompile invalider Objekte und danach Aktivieren und Kontrolle des SQL_APPLY:
```
SQL> @?/rdbms/admin/utlrp
SQL> ALTER DATABASE START LOGICAL STANDBY APPLY IMMEDIATE;
SQL> select * from v$logstdby_progress;
```

12c-Upgrade der Logical Standby / Syncing

Hierbei waren beim Upgrade ohne DBUA diese Punkte besonders zu beachten:

Ausführung des 12c Pre-Upgrade-Tools
```
SQL>EXECUTE DBMS_STATS.GATHER_DICTIONARY_STATS;
SQL> exec dbms_java_dev.enable;
SQL> ALTER DATABASE STOP LOGICAL STANDBY APPLY;
SQL> shutdown immediate
```
Mit 12c-Environment:
```
SQL> startup nomount
SQL> alter system set encryption wallet open identified by ***;
SQL> alter database mount;
SQL> alter database open migrate;
bash> cd $ORACLE_HOME/bin/
bash> ./dbupgrade
```
…
Start und Kontrolle des SQL-APPLY
```
SQL> shutdown immediate;
SQL> startup nomount;
SQL> alter system set encryption wallet open identified by ***;
SQL> alter database mount;
SQL> alter database open;
SQL> ALTER DATABASE START LOGICAL STANDBY APPLY IMMEDIATE;
SQL> select * from v$logstdby_progress;
```

Problem: DML auf AUD$ bremst SQL-APPLY massiv und das Lag wird nicht kleiner => nach Rücksprache mit Kunden und gemäß Note 862173.1 :
```
SQL> exec dbms_logstdby.skip('DML','SYS','AUD$');
```

Kompatibilitätsparameter heraufsetzen und RAC-Betrieb ermöglichen:
```
SQL> alter system set compatible='12.2.0.1' scope=spfile;
SQL> alter system set cluster_database=TRUE scope=spfile;
SQL> shutdown immediate;
SQL> startup nomount;
SQL> alter system set encryption wallet open identified by ***;
SQL> alter database mount;
SQL> alter database open;
SQL> ALTER DATABASE START LOGICAL STANDBY APPLY IMMEDIATE;
SQL> select * from v$logstdby_progress;
```

Clusterressourcen für DB, Instanzen und Services 12c anlegen

Downtime mit Splitting

Unkritische Anwendungen + DB-Services stoppen
Read-only Services von kritischen Anwendungen bereits auf 12c starten + Funktionstest
Services von zu migrierenden Anwendungen in 11g stoppen und Sessions bereinigen
Verifizierung Sync-Status der Logical Standby
Aktivierung der Logical Standby
`SQL> alter database activate logical standby database finish apply;`
Accounts der migrierten Applikationen in 11g DB sperren
Accounts der nicht migrierten Applikationen in 12c DB sperren

Nacharbeiten
Diese Aufgaben waren als Nacharbeit zu erledigen:
1) Abschalten des Log-Shippings zur 12c-DB (`SQL> alter system set log_archive_dest_2 = '';`)
2) Deaktivierung von supplemental Logging (`SQL>ALTER DATABASE DROP SUPPLEMENTAL LOG DATA;`)
3) Restore Points löschen
4) Firewall Regeln bereinigen
5) Löschen von 11g Schemas/Tablespaces in der 12c DB und vice versa
6) AWR Update
7) Kontrolle/Anpassungen im Monitoring / Backupregime
8) Ggf. Rückgabe von ASM-Disks /LUNs

Fazit

Mit dem geschilderten Verfahren konnte erfolgreich und mit sehr geringer Verfügbarkeitsunterbrechung ein „Teil-Upgrade“ einer 11.2-Datenbank nach 12cR2 durchgeführt werden. Durch Verwendung von grundsätzlichen Funktionalitäten der Enterprise Edition und eines MOS-Skripts konnnte der Aufwand für Tests und produktive Migration begrenzt werden.

Lessons learned:
Tests sollten vollständig durchgeführt werden, in diesem Fall hätte dann die stark bremsende Wirkung der AUD$ auf das SQL-APPLY nicht für eine Überraschung am Umstellungstag gesorgt.

Dokumentationsreferenzen
Database Upgrade Guide 12.2
https://docs.oracle.com/en/database/oracle/oracle-database/12.2/upgrd/index.html
Data Guard Concepts and Administration 12.2
https://docs.oracle.com/en/database/oracle/oracle-database/12.2/sbydb/index.html
Using Transient Logical Rolling Upgrade for Database Migration (Doc ID 2350945.1)
Oracle FBDA/Total Recall Setup Upgrade And Migration (Doc ID 2252966.1)
Oracle11g Data Guard: Database Rolling Upgrade Shell Script (Doc ID 949322.1)
Step by Step Guide on How to Create Logical Standby (Doc ID 738643.1)

Slow Performance In Logical Standby Database Due To Lots Of Activity On Sys.Aud$ (Doc ID 862173.1)
Complete Checklist for Manual Upgrades to Non-CDB Oracle Database 12c Release 2 (12.2) (Doc ID 2173141.1)
Patches to apply before upgrading Oracle GI and DB to 12.2.0.1 (Doc ID 2180188.1)
Master Note for Troubleshooting Oracle Managed Distributed Transactions (Doc ID 100664.1)

Die Seitenzahl wird von uns eingefügt!
Bitte fügen Sie Ihre Kontaktadresse hinzu.

Kontaktadresse:
Olaf Nowatzki
interface systems GmbH
Zwinglistr. 11/13
01277 Dresden

Telefon: +49 (0) 351 31809-71
Fax: +49 (0) 351 31809-33
E-Mail Olaf.Nowatzki@interface-systems.de
Internet: www.interface-systems.de