Storagemigratie zonder downtime

Slechts een uur niet in de lucht tijdens migratie

KPN beschikt over een aantal zeer grote databases. Een aantal van deze VLDB databases worden door Atos Origin beheerd. Voor één van deze databases moest de storage uitgebreid worden en tevens diende de storage oplossing vervangen te worden van leverancier A naar leverancier B. Uiteraard met de eis van functioneel beheer om de ruim 9 Tb aan data en nog eens 12 Tb aan archives en backup met zo min mogelijk downtime te migreren.

Vanuit regelgeving is KPN verplicht om alle call detail records (CDR) 183 dagen te bewaren. Een call detail record bevat gesprekssspecificaties. Vanuit het perspectief van de dba geredeneerd levert dat direct een uitdaging op, omdat een dergelijke richtlijn leidt tot een zeer forse database. Op dit moment is de database 9 Tb en vanwege een aanstaande verlenging van de bewaartermijn naar 366 dagen gaat de database nog fors groeien. De database dient 7 x 24 uur beschikbaar te zijn voor online queries. Tevens worden er dagelijks met behulp van SQL*Loader via frequente jobs heel veel nieuwe records in de database geladen. Hierbij dient de database nog steeds optimaal te presteren en ook de maintenance windows voor de back-up en andere zaken moeten zo klein mogelijk te zijn. Downtijd is bijna niet mogelijk, want als de database niet of minder beschikbaar is hopen de CDR's die geladen moeten worden zich pijlsnel op.

Om aan deze eisen te kunnen voldoen heeft Atos Origin gebruik gemaakt van een Oracle10g database (10.2.0.4.0) en diverse Oracle functionaliteiten ingezet: onder andere partitionering, tabel compressie, materialized views, block change tracking (om de doorlooptijd van de backup te versnellen) en ASM. Tevens zijn met behulp van PL/SQL diverse procedures geïmplementeerd die zorgen voor een zo efficiënt mogelijke afhandeling van de CDR's, waarbij de CDR's elke dag in dagpartities worden geladen via SQL*Loader en ze na zestig dagen worden gestript. Uiteindelijk worden de CDR's na afloop van de bewaartermijn weer verwijderd uit de database.

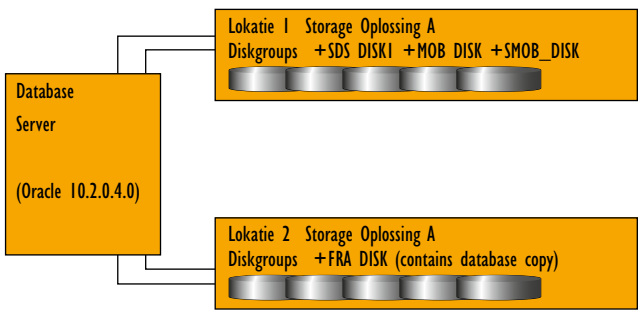
ASM

In verband met beheergemak en performance is ASM (automatic storage management) ingezet bij de database. ASM is door Oracle in versie 10g geïntroduceerd en borduurt voort op de al eerder door Oracle in gang gezette Oracle Managed Files. ASM is feitelijk een geïntegreerd file systeem en volume manager van Oracle en biedt diverse voordelen voor de DBA. Een groot voordeel in dit landschap is onder andere het large file support van ASM en tevens de mogelijkheid om online diverse beheershandelingen uit te voeren met de storage. Voor de DBA is het niet minder belangrijk dat hij verder zelf alles in eigen hand heeft. Het enige dat van de systeembeheer en storage afdeling gevraagd wordt is dat men SAN storage (een aantal LUN's) aanlevert voor de database. En vanaf dat punt neemt de DBA het over.

De storagemigratie

Stap 1: de uitgangssituatie

In onderstaande figuur is de storagesituatie op het moment voorafgaand aan de migratie afgebeeld:



Figuur 1: De uitgangssituatie.

De database heeft een aantal diskgroepen met data op lokatie 1 en tevens een diskgroep waar een kopie van de database bevindt op lokatie 2. Deze kopie is tevens de Oracle RMAN backup van de database en wordt elke dag bijgewerkt met incremental backups van de database zelf. (Met een tijdsverschil van 2 dagen).

Via een aantal queries op de ASM instance wordt informatie getoond over de diskgroepen en 'disken'. (Hoewel ASM het ziet als een disk is het feitelijk een LUN op de SAN storage omgeving)

Query 1: overzicht ASM diskgroepen.

```
select group_number GR, name, round (total_mb/1024) total_gb, round
(free_mb/1024) free_gb, round(free_mb/total_mb*100) "free %"
from v$asm_diskgroup order by diskgroup;

GR DISKGROUP                       TOTAL_GB    FREE_GB     free %
-------------------------------- ---------- ---------- ----------
1  FRA_DISK                         12485        3312          27
2  MOB_DISK                          6160        2266          37
3  SDS_DISK1                         1440         581          40
4  SMOB_DISK                         4295         528          12
```

Query 2: overzicht ASM disken van diskgroep SDS_DISK1.

```
select disk_number DN, name, state, total_mb, path
from v$asm_disk where group_number = 3 order by disk_number;

DN  NAME                 STATE    TOTAL_MB    PATH
-----------------------------------------------------------------
1   SDS_DISK1_0001       MEMBER   220160
/dev/rdsk/c9t6006016057851800F657C8DD33E7DA11d0s0

2   SDS_DISK1_0002       MEMBER   220160
/dev/rdsk/c9t60060160578518009E34A0DC32E7DA11d0s0

3   SDS_DISK1_0003       MEMBER   220160
/dev/rdsk/c9t6006016057851800421EC59134E7DA11d0s0

4   SDS_DISK1_0004       MEMBER   220160
/dev/rdsk/c9t6006016057851800 66BF693734E7DA11d0s0
```

Stap 2: Toevoegen van nieuwe storage product

In deze stap wordt het nieuwe storageproduct aangesloten op de databaseserver. Twee van de vier hba-kaarten worden daarvoor ingezet. Voor het herconfigureren van de hba-kaarten dient de databaseserver gestopt en gestart te worden. Dit is tevens de reden dat we van een bijna-zero downtime moeten spreken en niet van een zero downtime. Nadat de hba-kaarten zijn geherconfigureerd is de database server inclusief het hele applicatie landschap weer gestart en zijn de nieuwe storage LUNs beschikbaar voor de Oracle database.

De migratie van de storage kan nu beginnen. De migratie wordt stapsgewijs uitgevoerd, diskgroep voor diskgroep. In dit artikel wordt de diskgroep SDS_DISK1 als uitgangspunt genomen. Maar de handelingen zijn verder voor alle diskgroepen identiek.

Nadat de nieuwe LUNs voor de diskgroep SDS_DISK1 beschikbaar zijn is dit met onderstaande query geverifieerd:

```
Select path, total_mb  from v$asm_disk where header_status='CANDIDATE';

    PATH                                               TOTAL_MB
-------------------------------------------------- ----------
/dev/rdsk/c9t6001438002A5436B0000A00000B30000d0s0      1023990
/dev/rdsk/c9t6001438002A5436B0000A00000BF0000d0s0      1023990
/dev/rdsk/c9t6001438002A5436B0000A00000AD0000d0s0      1023990
```

Uit de query blijkt dat er 3 LUNs van 1 Tb elk beschikbaar zijn voor de diskgroep SID_DISK1. Via onderstaand statement worden de LUNs toegevoegd aan de diskgroep. Dit statement kan online worden uitgevoerd.

```
ALTER DISKGROUP SDS_DISK1 ADD DISK '/dev/rdsk/c9t6001438002A5436
B0000A00000B30000d0s0' name SDS_DISK1_0005,
'/dev/rdsk/c9t6001438002A5436B0000A00000AD0000d0s0' name SDS_DISK1_0006,
'/dev/rdsk/c9t6001438002A5436B0000A00000BF0000d0s0' name SDS_DISK1_0007
/
```

Nadat het statement is uitgevoerd, wordt er voor ASM een rebalance operatie opgestart. Dit houdt in dat alle data van de diskgroep evenredig wordt verdeeld over alle aanwezige disken. De ASM rebalance operatie vindt ook gewoon online plaats. De impact van de rebalance operatie op de databaseperformance kan worden gestuurd met behulp van het statement 'alter diskgroup SDS_DISK1 rebalance power n', waarbij n kan variëren van 0 tot 11. Hoe hoger n wordt gezet, hoe meer impact op de performance. (Op de achtergrond worden namelijk n rebalance processen gestart.) Het is tevens mogelijk om overdag (als er ook gebruikers werken in de database) een lagere waarde in te stellen dan 's avonds. Maar omdat deze database zowel overdag als 's nachts zwaar wordt belast is ervoor gekozen om de waarde continu op 6 te zetten. De waarde 6 zorgde voor een zo snel mogelijke doorlooptijd van de rebalance operatie met een acceptabele performance. De gebruikers hebben geen verminderde performance ervaren tijdens de rebalance operatie. Tijdens de rebalance operatie gaan de diverse batchjobs, het laden van de CDR's en de backup ook gewoon onverhinderd door. Wel is bijvoorbeeld tijdens de backup te zien dat de rebalance operatie langzamer gaat lopen. Met onderstaande query op de +ASM instance is de voortgang van de rebalance operatie te volgen:

```
select operation, sofar, est_work, est_rate, est_minutes
from v$asm_operation;

OPERA       SOFAR    EST_WORK    EST_RATE EST_MINUTES
----- ---------- ---------- ---------- -----------
REBAL     1746871    3292478         546         2830

Toelichting kolommen view V$ASM_OPERATION
SOFAR               aantal allocation units dat al verplaatst is
EST_WORK  geschatte hoeveelheid werk. (in allocation units)
EST_RATE  geschatte hoeveelheid allocation untis per minuut
EST_MINUTES         geschatte hoeveelheid resterende tijd in minuten

(In Oracle10g zijn de allocation units gelijk aan 1 Mb. Vanaf Oracle11g
kunnen voor de allocation units ook grotere waardes worden gekozen.)
```

Naast deze view wordt de rebalance operatie ook weggeschreven naar de alertfile van de +ASM instance. Hierin wordt bijvoorbeeld de start en het einde van de operatie gemeld en tevens het opstarten van de ARBn rebalance achtergrond processen.

Uiteindelijk zijn alle diskgroepen op deze manier uitgebreid met de nieuwe LUNs. In onderstaande tabel staan de doorlooptijden vermeld. De nieuwe LUNs zijn groter gesized omdat er rekening is gehouden met verwachte groei van de database. In onderstaande tabel staan de doorlooptijden vermeld:
De doorlooptijden kunnen redelijk verschillen, omdat de snelheid van de rebalance operatie rechtstreeks samenhangt met de

Diskgroep	Omvang bestaande LUN's (Tb)	Omvang nieuwe LUN's (Tb)	Doorlooptijd rebalance operatie
SDS_DISK1	1,4	3	7 uur
MOB_DISK	6	7	48 uur
SMOB_DISK	4,2	12	54 uur
FRA_DISK	12,2	22	71 uur

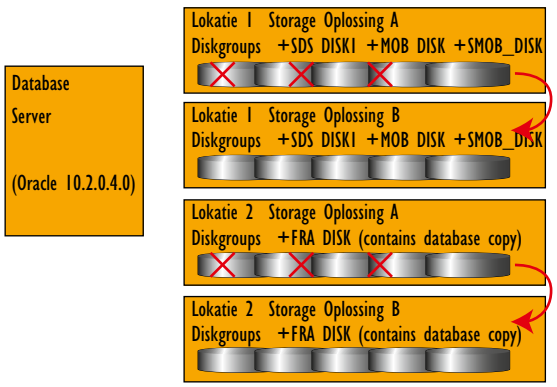
drukte op het systeem. Zo was duidelijk merkbaar dat de snelheid gedurende het weekeinde toenam en gedurende de werkweek weer afnam. Met name tijdens zware gebruikersactiviteiten.

Stap 3:Verwijderen van de oude storage
Nadat op bovenstaande wijze de nieuwe LUNs zijn toegevoegd aan de diskgroepen kunnen de oude LUNs verwijderd worden. Ook het verwijderen van de oude LUNs is weer een online operatie. Nadat het statement voor het verwijderen van de oude LUNs is gegeven, vindt weer een ASM rebalance operatie plaats analoog aan hetgeen in de voorgaande stap is beschreven.

Statement voor het droppen van de oude LUNs:

```
SQL> alter diskgroup sds_disk1 drop disk SDS_DISK1_0001, SDS_DISK1_0002,
SDS_DISK1_0003, SDS_DISK1_0004
```

In onderstaande figuur is stap 3 toegelicht:



Figuur 2: Droppen van oude LUNs.

Nadat zo de oude LUNs zijn gedropt, worden tot slot de 2 hba-kaarten die op de oude storage zijn aangesloten overgezet naar de nieuwe storage. Hiervoor moet op de databaseserver weer een restart uitgevoerd worden. In totaal kost dit opnieuw dertig minuten, inclusief stoppen en starten van database en applicatie omgeving.

Backup van de CDR database

Het uitgangspunt is om de backup window zo klein mogelijk te houden. Bij de CDR database is daarom als backup strategie gekozen om eenmalig m.b.v. rman een level 0 copy van de database te maken. Aansluitend wordt dagelijks een incremental level 1 backup gemaakt. Hierbij is block change tracking aangezet. Bij block change tracking worden alle wijzigingen op de data bijgehouden in een apart bestand. Bij het maken van de incremental level 1 backup hoeft rman dan niet langer de complete datafiles te scannen op wijzigingen, maar volstaat het raadplegen van de block change tracking file. Dit levert een aanzienlijke versnelling op van de incremental backup. Als laatste stap van de backup strategie wordt de copy van de database bijgewerkt met de wijzigingen. Hierbij loopt de copy 2 dagen achter op de database zelf. Deze strategie heeft geleid tot een backup window van gemiddeld 2 uur voor de CDR database van 9 Tb. Tevens heeft deze strategie als voordeel dat feitelijk dagelijks tevens een recovery wordt uitgevoerd, namelijk op de copy. En hiermee wordt dus dagelijks de beschikbaarheid en recoverability van de backup aangetoond. Onderstaand het script dat de incremental level 1 backup maakt en de copy bijwerkt op basis van die backup:

```
run {
recover copy of database with tag 'PROD_FULL_COPY' until time
'SYSDATE - 2';
backup as compressed backupset incremental level 1 for recover of
copy with tag 'PROD_FULL_COPY' database plus archivelog delete
input;
delete noprompt obsolete;
delete noprompt expired archivelog all;
delete noprompt expired backup;
delete noprompt expired copy;
```

Conclusie

Dankzij +ASM is het mogelijk geweest om in totaal meer dan 20 Tb aan data met een minimum aan downtime te migreren van storage product A naar storage product B. De downtijd besloeg twee keer een half uur voor het herconfigureren van de hba-kaarten van de databaseserver. Zonder het gebruik van +ASM was dit niet mogelijk geweest en zou de migratie tot een veel langere downtijd hebben geleid. Door het gebruik van +ASM is de DBA in staat om de migratie bijna geheel zelfstandig uit te voeren en de diverse views bieden de mogelijkheid om de voortgang van de operatie nauwgezet te volgen.

Rob Lasonder is Oracle DBA bij Atos Origin.

Cor de Waaij is Oracle architect bij KPN.