RMAN in praktijk

Deel 2: Het maken van een backup

In het voorgaande artikel hebben we aangegeven hoe RMAN wordt geconfigureerd. In dit artikel bespreken we hoe de backup wordt gemaakt. Behandeld worden de volgende backups: Daily, Weekly, Monthly, Freeze en Incremental.

De eerste drie spreken voor zich. Een Freeze backup is een tussentijdse backup, die je voor langere tijd wilt bewaren. Dit doe je bijvoorbeeld voordat je een update wilt installeren. Een Incremental backup slaat alleen de laatste veranderingen ten opzichte van de voorgaande backup op. Dan bestaat ook nog de backup voor standby. Behalve dat die klein en compleet moet zijn is de inhoud ook iets anders. De current controlefile wordt gebackupt als een 'standby controlfile'. Het is met RMAN ook mogelijk om backups te maken op basis van image files.

Voor het gemak ga ik uit van scripts die de backup runnen. Daarbij is er een klein verschil tussen het runnen van een script vanaf je filesystem of vanuit de catalog.
Een run script vanaf het filesystem:

```
Command line:
cmdline > rman target / cmdfile d2dbackup.rmn log d2dbackup.log
alternatief vanuit RMAN:
RMAN > @d2dbackup.rmn
```

Een run script vanuit catalog:

```
Command line :
cmdline > rman target / script d2dbackup.rmn log d2dbackup.log
vanuit RMAN;
RMAN> run {execute script d2dbackup ; }
```

Hiermee hebben we de basis gelegd. Afhankelijk van de scheduler op je platform voer je dan een backup uit. Let op hoe je scheduler omgaat met foutmeldingen, de return code van RMAN wil alleen maar zeggen dat het script correct is beëindigd. Dit zegt dus niets over de inhoud, in de afloop van je script kunnen een of meer fouten zijn voorgekomen. Gebruik daarom een wrapper. Dit is een operating system script, dat de RMAN- meldingen uit je logfile filtert. Zo'n wrapper kan er bijvoorbeeld zo uitzien:

```
#! /usr/bin/ksh93
# file : backup1_full.sh
# date :           Jan 30 2007
# owner: rick van Ek
#
# Made dynamic script takes all database noted in /etc/oratab
# provided the start flag is active (Y)

# general settings, for location scripts etc.
. oravar.env

for SID in $SIDLIST_ALL
do
   #echo "SID = $SID "
   export ORACLE_SID=$SID
   . oraenv
  BACKUPTRUE=`grep "^$SID" /etc/oratab | cut -d : -f 3`
  # BACKUPTRUE="Y"
  if [[ $BACKUPTRUE == "Y" ]]
  then
rman target / catalog rman/recover@catrep cmdfile=$CMDFILE1 log=$LOG/
backup_$SID.log
uuencode $LOG/backup_$SID.log backup_$SID.log | mailx -s \
"Backup $SID Completed logfile included" $RECIEVER
grep "^RMAN-"  $LOG/backup_$SID.log > $LOG/backup_$SID.err
if [  -s $LOG/backup_$SID.err ]
then
    uuencode $LOG/backup_$SID.err backup_$SID.err | mailx -s \
  "Backup $SID FAILED errors included, check logfiles" $RECIEVER_EDC
fi
if [ ! -s $LOG/backup_$SID.log ]
then
mailx -s "Backup FAILED no logfile, check procedure" $RECIEVER_EDC
fi
mv $LOG/backup_$SID.log /disk06/oradata/$SID/backup/tsm
unset BACKUPTRUE
  fi
done
```

Onder Grid Control heb je dit effect ook en staat de output in de details. Het gevolg is dat je alles elke dag moet controleren, ongeacht de status. Bij het gebruik van een wrapper kun je de RMAN-XXXXX meldingen opvangen en eventueel naar je mailbox sturen. Hiermee zijn we klaar voor het echte werk. We beginnen met een standaard full backup. Als basis nemen we het volgende script:

```
Command line:
cmdline > rman target / cmdfile d2dbackup.rmn log d2dbackup.log
alternatief vanuit RMAN:
RMAN > @d2dbackup.rmn
</code>

<code>
run{
backup as compressed backupset
duration 02:00 minimize load database
archivelog all not backed up ;
crosscheck backup;
crosscheck archivelog all;
crosscheck backup of controlfile;
delete noprompt obsolete ;
}
```

Wat is er gebeurd met de standaard aanroep 'backup database plus archivelog all ;' ? Dit lijkt gecompliceerder dan het is. In een run block, maar ook op de command line, wordt een commandoregel afgesloten met een ; dus in dit geval zijn dat de eerste drie regels. Toelichting op de eerste commandoregel; 'as compressed backupset' betekent dat RMAN de ongebruikte blokken in de database niet backupt en ook niet controleert op corruption. De bezette blocks worden wel gebackupt en nog belangrijker: gecontroleerd op corruption, dit geldt voor iedere backup. Daarnaast maakt deze een op block niveau gecomprimeerde backupset. Hoe de doorlooptijd wordt is afhankelijk van het aantal parallelle processen en de CPU snelheid. De grootte na compressie hangt af van hoeveel blocks zijn gevuld.

In een groot aantal praktijkgevallen is de compressed backup set tussen de 10 en 20 procent van de fysieke grootte van de datafiles op disk. Bij 'duration 02:00 minimize load database' gebeuren dingen die je niet zomaar kunt overnemen. Met duration geef ik aan hoe lang de backup mag duren, ongeacht of deze klaar is of niet. Het is dus zaak om eerst te kijken hoe lang de backup er over doet. Hier wordt het gebruikt om te zorgen dat RMAN niet alle disk resources van de database afpakt. Dit script wordt gedurende een batch window gedraaid. Normaal doet de backup er een uur over, door er de rem (minimize load) op te zetten willen we 50 procent van de resources vrij laten voor de batch jobs. In het logfile kom je dan ook de melding throttle tegen, 'channel ORA_DISK_1: throttle time: 1:95:12'.

Als je dit voor alle backups wilt laten gelden dan kan je er voor kiezen de bandbreedte te beperken via de rate parameter bij het configureren van het channel. Als laatste deel van de backup worden de archivelog files gedaan die nog niet gebackupt waren. Dit is een keuze die je moet maken. In dit geval gaat men uit van drie archivelog files gedurende retention periode, namelijk een op de schijf in de 'archivelog_dest', een in de backup en een op de standby server. Hieruit blijkt al dat de keuze afhankelijk is van je configuration en de eisen die aan de infrastructuur zijn opgelegd. Dit is eenvoudig aan te passen met 'archivelog not backed up X times to disk'.
Nu kan men overwegen om de archivelog files in eigen backup files op te slaan. Daar moet men dan een aparte commandoregel voor maken. Indien men later de mogelijkheid wil hebben om de retention aan te passen zodat een backup langer of korter bewaard wordt dan is dit noodzakelijk. De 'retention time' van een backup file kan niet worden aangepast als hierin ook archivelog files zitten. Aangezien de database met een spfile was opgestart en in de 'configuration controlfile autobackup' was aangezet worden beide automatische aan het einde van de backup weggeschreven.

```
List of Backup Sets
===================

BS Key  Type LV Size       Device Type Elapsed Time Completion Time
------- ---- -- ---------- ----------- ------------ ---------------
2       Full    6.98M      DISK        00:00:06     18-DEC-08
        BP Key: 2   Status: AVAILABLE  Compressed: NO  Tag:
TAG20081218T142253
        Piece Name: D:\ORACLE\ORADATA\BACKUP\C-697544247-20081218-00
  Control File Included: Ckp SCN: 699813       Ckp time: 18-DEC-08
  SPFILE Included: Modification time: 18-DEC-08
```

Na het backuppen komen regels met het crosscheck commando. Hier wordt de backup van de database-, archivelog- en controlfiles gecontroleerd, deze controle houdt in dat er fysiek wordt gecontroleerd of de backup files bestaan en in de database catalogus staan. De status in de database catalogus wordt eventueel bijgewerkt.

Als allerlaatste wordt er een opschoning gedaan van de backup- en archivelogfiles. Op basis van de "retention time" worden de files die niet meer nodig zijn verwijderd. Dit houdt in dat wanneer een tussentijdse backup is mislukt, het mogelijk is dat een backup set wordt behouden om aan de gestelde retention time te voldoen. Bij een backup naar disk kan dit consequenties hebben voor de diskruimte. Voor een backup naar tape kan dat ook gevolgen hebben, afhankelijk van de omloop van de tapes.

```
list archivelog all ; # files in archivelog_dest
```

De archivelogs in de backup en in de archivelog_dest kunnen worden opgevraagd:

```
list archivelog all backed up 1 times to disk ; # files in backupset
```

Uitgaande van een retention time van 2 dagen kun je ook beslissen om de backup files niet zomaar weg te gooien maar eerst op tape te zetten en daarna pas te verwijderen. Het voordeel hiervan is dat de backup files niet weggegooid worden als de tape backup mislukt. De voorkeur gaat altijd uit naar

methodes om de backups compleet te houden zonder handmatig ingrijpen.

```
BACKUP
  DEVICE TYPE sbt
  BACKUPSET
    COMPLETED BEFORE 'SYSDATE-2'
  DELETE INPUT;
```

Bij een weekly/monthly backup werkt het anders, het voorbeeld hieronder maakt een 'cold backup'. Vanuit RMAN gezien is het een gewone backup maar omdat de business requirements iets anders vereisen. Kan men door de retention van een enkele backupset te veranderen aan die eisen voldoen.

```
connect target sys/*
spool log to 'Q:\bst\logs\WeeklyBackup.log' append ;
run{
shutdown immediate;
startup mount;
backup as compressed backupset
       database
       keep until time 'to_char( sysdate + 7)'  logs ;
crosscheck backup;
crosscheck archivelog all;
crosscheck backup of controlfile;
alter database open;
}
spool log off;
exit;
```

In dit voorbeeld wordt uitgegaan van een backup window in het weekeinde waarin de database down mag. Het is een cold backup omdat de database in mount status staat. Wat verder bijzonder is dat het keep commando wordt gebruikt om de retention parameter te negeren. Het venijn zit hier in het staartje, namelijk het keywoord 'logs'. Deze bepaalt dat alle archivelog files die nodig zijn voor deze backup bewaard blijven tot de keep time is afgelopen. Daarna geldt voor deze backup files en archivelog files dezelfde regels als wanneer de retention time is verlopen. Indien men de archivelog files niet nodig heeft kan men 'nologs' gebruiken en worden de archivelog files niet bewaard. Een monthly backup ziet er hetzelfde uit alleen wordt dan "(SYSDATE + 31)" gebruikt. Als men het zelfde script wil gebruiken maar dan met variabelen wil werken, moet men deze anders aanroepen.

```
>   rman  target sys/* using $keeptime
#          $keeptime is for weekly7 and for monthly 28,29, 30 or 31
spool log to 'Q:\bst\logs\Backup.log' append ;
run{
shutdown immediate;
startup mount;
backup as compressed backupset
       database  keep until time 'to_char( sysdate + &1)'  logs ;
crosscheck backup;
crosscheck archivelog all;
crosscheck backup of controlfile;
alter database open;
}
spool log off;
exit;
```

De keep option kan ook gebruikt worden voor een backup die al is uitgevoerd.

```
change backup tag 'TAG20080521T215936' keep until time 'sysdate+1' logs
```

Hierbij moet je er wel voor zorgen dat de backup files geen archivelog files bevatten.

```
RMAN> change backup tag 'TAG20090121T153236' keep until time '(sysdate
+ 1)' logs ;

using channel ORA_DISK_1
RMAN-00571: ===========================================================
RMAN-00569: =============== ERROR MESSAGE STACK FOLLOWS ===============
RMAN-00571: ===========================================================
RMAN-03002: failure of KEEP command at 01/21/2009 23:17:40
RMAN-06530: CHANGE ... KEEP not supported for backup set which contains
archive logs

RMAN>

RMAN>

RMAN> change backup tag 'TAG20090120T111110' keep until time '(sysdate
+ 1)' logs ;

using channel ORA_DISK_1
keep attributes for the backup are changed
backup will be obsolete on date 22-JAN-09
archived logs required to recover from this backup will expire when
this backup expires
backup set key=14 recid=14 stamp=676638676
```

Eigenlijk hebben we hiermee ook de 'freeze' backup aangesproken, hier onder versta ik een backup die tussentijds wordt gemaakt. De noodzaak doet zich voor wanneer men een release wijzigt of patches uitvoert. Dan kan men een backup maken om te bewaren of sneller te kunnen restoren. Aangezien men met 'keep until time' de retention kan 'overrullen' , kan men de backup langer of zelfs korter bewaren. De noodzaak komt meestal voort uit de techniek en deze wordt vaak z.s.m. weer opgeruimd. Een 'archive' backup komt overeen. Alleen wordt dat vanuit de bussiness gevraagd en is de bewaartijd meestal wel bekend, b.v. Een backup t.b.v. finance gegevens.

Incremental backup

Ik ben altijd voorzichtig met incremental backup (alleen de wijzigingen ten opzichte van voorgaande backup). Het kan voordelen hebben maar roept veel vragen op. Veel is afhankelijk van de grootte van de omgeving alsook de hoeveelheid transactions. Tevens beïnvloedt het de restore tijd en complexiteit, terwijl extra schijfruimte goedkoper wordt. Voor gemiddelde omgevingen is het alleen de moeite waard als er weinig transacties zijn. Denk daarbij aan test omgevingen waar lange perio-

des niets wijzigt. Voor hele grote omgevingen is het een andere zaak, daar gaat het om omvang, doorvoersnelheden en een beperkte backup window. Dan geldt 'Hoe minder men hoeft te doen des te beter'. Dit vraagt om een testcase voor de incremental backup wordt geïmplementeerd. Toch haal ik het even kort aan om het overzicht compleet te maken. De basis is een 'incremental level 0' backup, daar bovenop komt een 'incremental level 1' backup . De 'incremental level 1' backup komt in twee smaken, namelijk cumulatief en differentieel. Cumulatief is heel eenvoudig, iedere keer maakt deze een backup van alle veranderingen sinds de 'incremental level 0'. Diffirentieel maakt alleen de laatste wijzigingen ten opzichte van de vorige incremental backup. Als er geen 'cumulative' wordt opgegeven worden alle wijzigingen sinds de laatste backup gemaakt, ongeacht of dit een full backup was of niet. Als er geen 'archive level 0' aanwezig is wordt de 'archive level 1' automatisch een 'level 0' backup.

```
BACKUP
  INCREMENTAL LEVEL 1 CUMULATIVE
    DATABASE;
```

Standby backup

De backup voor standby databases is bedacht om eenvoudig een standby aan te maken terwijl de backup files zo klein mogelijk zijn. Uitgangspunt is dat de dataverbinding naar de standby database een beperkte bandbreedte heeft. In de backup worden geen archivelog files meegenomen, de upload van de archivelog files wordt al gestart voordat de backup op de primary wordt gemaakt. De compressed backup bevat alleen de datafiles en een control file for standby.

Een leuk detail is dat bij een test bleek dat alle blocks van de index tablespace anders waren op de standby dan op de primary. Toch waren de indexen correct, wat daar onder water gebeurt is niet duidelijk. De control file wordt apart gebackupt als een standby controlfile, zodra de database daarmee start is deze gelijk een standby database.

```
RMAN> backup current controlfile for standby ;

Starting backup at 22-JAN-09
using target database control file instead of recovery catalog
allocated channel: ORA_DISK_1
channel ORA_DISK_1: sid=119 devtype=DISK
channel ORA_DISK_1: starting full datafile backupset
channel ORA_DISK_1: specifying datafile(s) in backupset
including standby control file in backupset
channel ORA_DISK_1: starting piece 1 at 22-JAN-09
channel ORA_DISK_1: finished piece 1 at 22-JAN-09
piece handle=C:\ORACLE\PRODUCT\10.2.0\FLASH_RECOVERY_AREA\GC10_UN1\
BACKUPSET\2009_01_22\O1_MF_NCNNF_TAG20090122T113136_4QJLRBON_.BKP
tag=TAG20090122T113136 comment=NONE
channel ORA_DISK_1: backup set complete, elapsed time: 00:00:05
Finished backup at 22-JAN-09
```

Backup met image files.

Behalve het opslaan van de backup in backsets is het mogelijk om deze op te slaan in image files. Deze hebben dezelfde formaat als het originele file op het tijdstip van backup, dus ook lege blocks. Ook hier is het mogelijk om te werken met incremental backups. Block change tracking heeft een grote invloed op de performance van de incremental backups. De retention policy is hier ook van toepassing, maar slaat dan alleen op de image copies. Voordeel haalt men hier uit de snelle recovery bij lokaal opgeslagen files. Na een 'switch file' naar de image copy en het toevoegen van de archives is een recovery compleet. Het restoren van de datafile blijft dan achterwege, wat voordelen heeft bij grote files. In de praktijk kom je dit het meeste tegen bij het online verplaatsen van datafiles naar andere locaties op disk of naar ASM.

```
run {
backup as  copy  database ;
}
```

Samenvattend, voor de meest gemaakte backup bepaal je de retention time. Voor afwijkende backups geef je een 'keep time' aan in het backup script. Ook kan je een dagelijkse backup achteraf veranderen in een weekly door de retention time te veranderen. Als je de retention time wilt veranderen moet je dat doen op backup files die geen archivelogs bevatten. Als je de keep time verandert dan moet je aangeven of je de archivelog files wilt bewaren of niet. Dit heeft consequenties als je deze moet restoren, maar ook voor de opslag. Incremental backups zijn mogelijk, maar roepen overwegingen op m.b.t. complexiteit en restore time. Met de huidige technieken en hoeveelheden data en transacties is het een vraag of dit wel zinvol is.
Met RMAN kun je de backupset files die op disk liggen backuppen naar tape en van disk verwijderen. Hierdoor weet de catalog waar de backup files zijn. Dat kan handig zijn met een korte retention time en een tape archive, waarop men kan terugvallen. Backup as copy maakt zeer korte recovery tijden mogelijk, maar wordt het meest gebruikt voor het online verplaatsen van datafiles.

Rick van Ek is Oracle Senior DBA bij Van Ek IT-Consultancy B.V.