Überwachung von RMAN Datenbanksicherungen

Thorsten Grebe
twg-it
Berlin

Schlüsselworte
RMAN Backup-Monitoring, Recovery Katalog.

Wie sollte man in mittleren bis großen Umgebungen das Monitoring der RMAN-Sicherungen einrichten? Welche Prüfungen sollten durchgeführt werden, um eine möglichst lückenlose Kontrolle über die Zuverlässigkeit seiner Datenbankbackups zu erreichen?

Das Überwachungsproblem
Wer eine einzelne Oracle Datenbank mit RMAN sichert, entwickelt hierfür in der Regel ein Skript, das die Sicherung ausführt und im Anschluss auf Fehler prüft. Im Fehlerfall wird eine Email versendet, eventuell auch im Erfolgsfall. Kommt eine zweite Datenbank hinzu, dann gibt es ein zweites Skript und eine weitere Email. Dieser Ansatz kann nur für eine sehr kleine Anzahl von Datenbanken funktionieren. Bei 30, 50 oder 500 Datenbanken muss der Skriptansatz zur Kontrolle der Sicherungen scheitern. Niemand hat im Blick, ob statt 50 nur 49 Kontrollmails eine erfolgreiche Sicherung gesendet haben, oder ob der Scheduler auf einem von 50 Datenbankservern deaktiviert wurde.

Größere Umgebungen mit geteilten Zuständigkeiten vertrauen daher auch nicht auf das Zählen von eingegangenen Erfolgsmails, sondern verwenden professionelle Monitoringlösungen wie Nagios, Grid/Cloud Control oder diverse Enterprise Scheduler. Hier garantiert der Monitor oder der globale Scheduler, dass keine eingerichtete Überwachung unbemerkt ausfällt. Ein Blick ins Dashboard oder in die Schedulerübersicht entlarvt sofort eine säumige Überwachung.

Doch was bedeutet es wirklich, wenn eine Erfolgsmail oder Nagios oder ein Enterprise Scheduler ein „OK“ ausgibt? Was sagt es über die Zuverlässigkeit der RMAN-Sicherungen aus? Strenggenommen gar nichts. Denn das „OK“ in der Mail, im Dashboard von Nagios oder in der Jobübersicht eines Schedulers besagt nur, dass das Kontrollskript den Rückgabewert „0“ erhalten hat. Es gibt jedoch viele Gründe, warum eine RMAN-Sicherung nichts taugen kann und das Wrapperskript trotzdem eine „0“ zurückgibt. Es ist daher in einer großen Umgebung, in der sich mehrere DBAs ihrer Leidenschaft der Skriptoptimierung hingeben, durchaus denkbar, dass RMAN Sicherungsskripte versehentlich Defekte aufweisen, die unbemerkt bleiben.

RMAN-Überwachungen scheitern im Stillen
Warum ist das so gefährlich? Es gibt einen grundsätzlichen Unterschied zwischen der Überwachung von RMAN-Sicherungen und der Überwachung anderer Objekte. Versagt beispielsweise die Überwachung eines Services, wird sehr schnell ein Anwender anrufen und es ist klar, dass die Überwachung, die hätte anschlagen sollen, offenbar einen Mangel hat. Der Endanwender ist der ultimative Monitor für Services. Verursacht der Ausfall eines Services keinen Endanwender-Aufschrei, dann darf der Sinn dieses Services vermutlich infrage gestellt werden.

RMAN-Überwachungen sind hier grundsätzlich anders. Wenn sie scheitern, dann wird es niemand bemerken. Die Überwachungen von RMAN-Sicherungen versagen im Stillen, sie werden niemals

einen Endanwender-Aufschrei auslösen. Der Ausfall wird im günstigsten Fall niemals, im schlimmsten Fall an dem Tag bemerkt, an dem eine Produktionsdatenbank zerstört wird.

Rückgabewert „0“

Wer gerne Erfolgsemails im Posteingang zählt oder sich an den grünen Zeilen in einem Dashboard erfreut, der soll dies auch weiterhin tun, sinnlos ist das keineswegs. Es muss aber klar sein, dass eine Erfolgsmail oder eine grüne Zeile nur sagen, dass ein Job erfolgreich gestartet und mit Fehlercode 0 beendet wurde. Nicht mehr, nicht weniger. Er sagt nicht, dass RMAN eine ausreichende Sicherung der Datenbank, der Archivelogs, des Controlfiles und des Spfiles durchgeführt hat. Wunderbar lassen sich in einem Backupskript Fehlerausgaben von RMAN über *exclude*-Klauseln unterdrücken. Als Beispiel mag das folgende RMAN-Skript dienen:

```
RMAN> configure exclude for tablespace users  ;
RMAN> configure exclude for tablespace sysaux ;

RMAN> run{
            set maxcorrupt for datafile 1,2,3,4,5 to 1000 ;
            backup duration 00:10 partial
            nochecksum
            database
            skip inaccessible
            skip offline ;
         }
```

Abb. 1:RMAN-Sicherung, die nichts taugt, aber mit Rückgabewert 0 abschließt.

Hier wird RMAN über zwei *configure*-Kommandos dauerhaft so konfiguriert, dass die Tablespaces `users` und `sysaux` nie gesichert werden sollen. Im folgenden RUN-Block wird die Toleranz für korrupte Datenblöcke zunächst hochgesetzt (`set maxcorrupt...`). Dann wird die Dauer auf ein zeitliches Limit von 10 Minuten begrenzt (`backup duration 00:10`) – RMAN soll nach Ablauf von 10 Minuten sofort seine Sicherungstätigkeit beenden, egal wie weit er mit seiner Arbeit gediehen ist. Die Überprüfung der Blockintegrität soll beim Schreiben der Sicherung deaktiviert werden (`nochecksum`). Außerdem sollen alle Datendateien, die defekt (`skip inaccessible`) oder offline (`skip offline`) gesetzt sind, gar nicht erst gesichert werden. Von der Archivelogsicherung ist hier gar nicht erst die Rede. Es ist offensichtlich, dass diese RMAN-Ausführung eine Sicherung schreiben wird, die unbrauchbar ist, um daraus eine Datenbank verlustfrei wieder herstellen zu können. Und dennoch wird die Ausführung dieses RMAN-Kommandos den Rückgabewert „0“ ausgeben! Die Überwachungsmail und Nagios werden ein „OK“ liefern. In der Übersicht des Enterprise Schedulers wird alles grün aussehen.

Hieraus sollte folgen, dass es nicht genügt, sich bei der Überwachung von RMAN-Sicherungen auf die Rückgabewerte der Backupskripte zu verlassen. Dies gilt besonders für mittlere und große Umgebungen mit mehrköpfigen DBA- und Administratoren-Teams.

Neue Datenbanken

Hinzukommt ein weiteres Problem: Was ist mit neu hinzukommenden Datenbanken? Vielleicht denkt der DBA noch daran, die Datenbank im Katalog zu registrieren oder ins Grid/Cloud Control einzuhängen, weil dies so in der Todo-Liste beschrieben oder vom Setup automatisch eingerichtet wird. Was aber, wenn er schlicht vergisst, die Nagiosüberwachung für die RMAN-Sicherung zu aktivieren, oder die Beantragung der Überwachung nachzuverfolgen. Oder wenn er vergisst, im

Jobscheduler den RMAN-Job einzutragen? Eine Überwachung, die schlicht vergessen wurde, kann noch nicht einmal scheitern.

Die Lösung für diese ständig drohende, potentielle Überwachungslücke ist der Einsatz einer Katalogdatenbank in Kombination mit einer verbindlichen Referenztabelle.

Die Katalogdatenbank
Entgegen der weit verbreiteten Ansicht, der Katalog sei zur Wiederherstellung von Datenbanken notwendig, liegt der größte Alltagsnutzen des Katalogs in seinem verborgenen, brach liegenden Monitorpotential. Der Katalog kennt nämlich alle Datenbanksicherungen bis ins kleinste Detail. Er weiß, ob eine inkrementelle Sicherung gelaufen ist und ob auch Archivelogs und Controlfile mitgesichert wurden. Er kann über Tablespaces und Datendateien Auskunft geben, die wegen *exclude*-Klauseln nicht mitgesichert wurden. Er kennt alle Korruptionen, die in Datenbank oder Backupsets bemerkt wurden. Er kennt die komplette Logausgabe aller RMAN-Jobs und bekommt mit, wenn eine Sicherung aus Platzgründen oder wegen einer *duration*-Klausel vorzeitig abbricht. Kurz: der Katalog weiß alles. Die folgende Abfrage zeigt beispielsweise alle inkrementellen RMAN-Sicherungen an, die in den vergangenen 24 Stunden registriert worden sind:

```
select dbid,
       name
  from rc_backup_set
  join rc_database using ( db_key )
  where start_time > sysdate - 1
   and (  ( backup_type = 'I' and incremental_level = 1 )
       or ( backup_type = 'D' and incremental_level = 0 )
       )  ;
```

Abb. 2: Abfrage gegen Katalog-Views.

Der Katalogview `rc_backup_set` registriert alle Backupdateien aller Datenbanken, die gegen den Katalog gesichert werden. Eins kennt diese View allerdings nicht, nämlich den Datenbanknamen. Einige `rc_*` Views führen die Spalte `db_name`, aber nicht alle. Deshalb wird hier die View `rc_backup_set` mit `rc_database` verknüpft. Die View `rc_database` enthält einen Eintrag für jede im Katalog verwaltete Datenbank mit Datenbanknamen, `dbid` und `db_key`. Noch besser ist es, wenn man eine eigene Inventartabelle einrichtet, die hier stattdessen verknüpft wird (s.u.). Ähnliche `rc_*` Views gibt es für Controlfiles (`rc_backup_controlfile`), Spfiles (`rc_backup_spfile`) und Archivelogs (`rc_backup_redolog`). Ist der Nutzen des Katalogs erkannt, dann ist das Zusammenstellen der Abfragen nur noch Fleißarbeit.

Deshalb sollte sich der DBA nicht auf eine Welle von Erfolgsmails verlassen, die jeden Morgen mehr oder weniger vollständig in seine Inbox schwappt und ihn in trügerischer Sicherheit wiegt, sondern er sollte nur eine einzige Email erwarten, die wichtigste des Tages, die ihm das Resultat einer Serie von Abfragen gegen den Recovery Katalog so aufbereitet, dass er mit einem Blick erkennt, ob alle Sicherungen der vergangenen Nacht erfolgreich gelaufen sind oder ob es irgendwo vermisste Objekte, Korruptionen oder Fehlermeldungen gab.

Die Referenztabelle
Wer sich jetzt jedoch allein auf die Katalog-Views verlässt, riskiert eine andere Überwachungslücke. Es besteht das Risiko, Datenbanken zu übersehen, für die eine Registrierung in der Katalog-Datenbank verpasst wurde (`RMAN> register database;`). Und wie fängt man Datenbanken wieder ein, die durch ein fälschliches Austragen (`RMAN> unregister database;`) aus dem Katalog entfernt wurden?

Es fehlt daher noch ein Baustein für eine lückenlose und möglichst wasserdichte RMAN Backup-überwachung: eine selbst erstellte, für das Unternehmen verbindliche Inventartabelle, die als Referenz alle Datenbanken mit ihren aktuellen DBIDs und DB_KEYs enthält. Hier sollten alle Datenbanken gefunden werden können, solche die mit RMAN gesichert werden, aber auch solche, die nicht mit RMAN gesichert werden sollen. Neu hinzugekommene Datenbanken müssen in diese Tabelle eingetragen werden. Nach Möglichkeit sollten automatische Abgleiche mit anderen Quellen eingerichtet werden. Dazu zählen die Targettabellen eines existierenden Grid/Cloud Control (`sysman.mgmt_targets`), und die Katalogview `rc_database`, in der ebenfalls nach neu registrierten Datenbanken gesucht werden muss. So kann dann im täglichen RMAN-Monitorbericht auf diese Neuzugänge hingewiesen werden. Einige Felder sollten in dieser Tabelle nicht fehlen, wie die folgende `ihre_referenz_tabelle` genannte Beispieltabelle verdeutlicht.

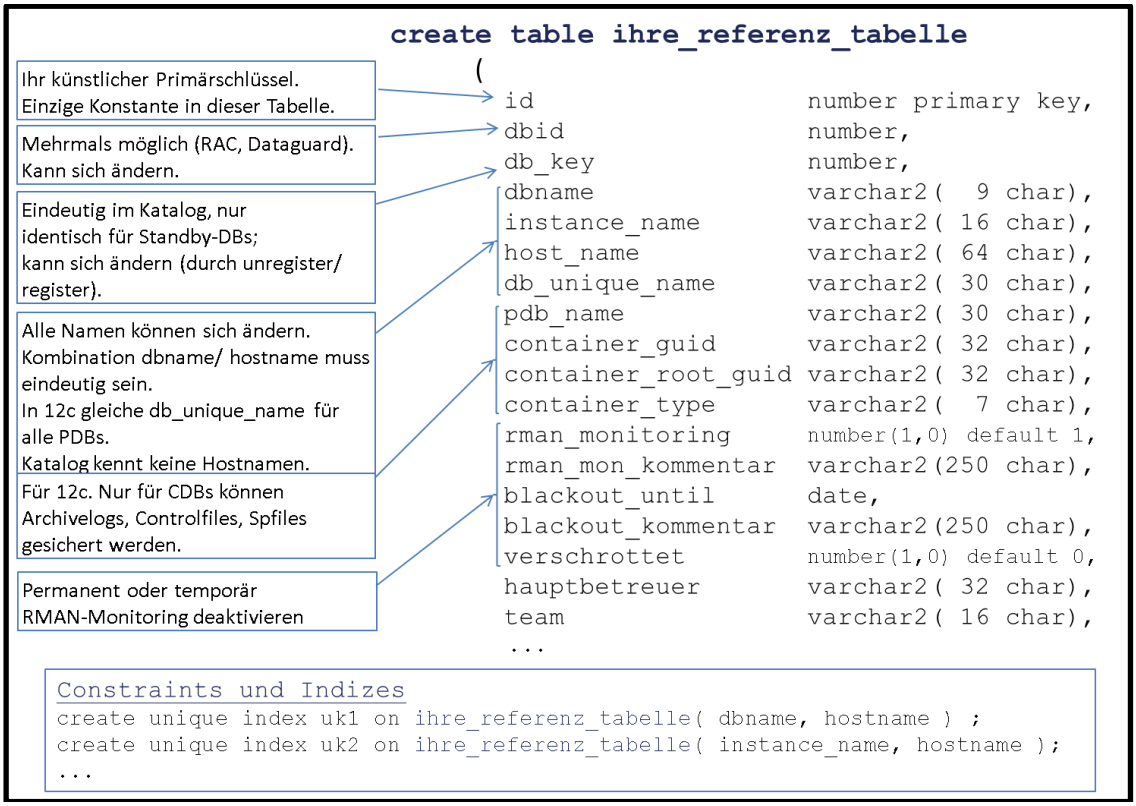

Abb. 2: Eine verbindliche Inventartabelle muss existieren.

Datenbanken, die nicht mit RMAN gesichert werden sollen, müssen hier auch mit einem Hinweis aufgeführt werden, warum sie nicht gesichert werden sollen, denn nur so kann gewährleistet werden, dass zu jeder Instanz, die im Katalog, im Grid/Cloud Control oder anderen Übersichten des Unternehmens auftaucht, eine Aussage zu den RMAN-Sicherungen getroffen werden kann. Die Tabelle muss einen Surrogatschlüssel, also einen künstlichen Primärschlüssel führen, denn dies ist das einzige Feld, dass konstant bleibt, alle anderen Werte können sich ändern: Die DBID ist nicht zwangsläufig eindeutig (Standbyumgebungen, Kopieren von Datenbanken ohne Ändern der DBID) und darüber hinaus änderbar, wenn Datenbanken regelmäßig durch Klonen aus der Produktion entstehen, der DB_KEY im Katalog ändert sich nach einem *unregister/register*, der DBNAME kann vom Administrator geändert werden, der Instanzname ist änderbar. Der Hostname kann sich ändern und mehrere Hosts können Datenbanken mit demselben Namen beherbergen. Für 12c Container-Datenbanken müssen die globalen IDs (`container_guid`) und der Typ des Containers (Root oder

PDB) mitgeführt werden, damit Zuordnungen zwischen Controlfiles und Archivelogs zu PDBs zuverlässig gefunden werden können.

Konsistenzchecks
Die Katalogdatenbank unterscheidet alle registrierten Datenbanken über zwei Primärschlüssel: die DBID und den DB_KEY. Nur Standby-Datenbanken dürfen diese Schlüssel teilen. Für alle anderen muss sichergestellt sein, dass keine zwei Datenbanken mit derselben DBID gegen den Katalog gehen. Der Katalog fängt sonst an, die Backupsets durcheinander zu würfeln. Bei einem Restore ist es dann Glückssache, ob die gewünschte Datenbank wieder hergestellt wird, oder die nicht gemeinte. Aus diesem Grund muss eine penible Datenhygiene betrieben werden. Über die Referenztabelle ist dies einfach möglich:

a) Die Kombination aus Hostnamen und Datenbanknamen muss eindeutig sein.
b) Nur im RAC und bei Standby-Systemen darf dieselbe DBID verwendet werden.

Namenskonzept
Wer kann, der sollte in seiner Umgebung ein konsistentes Namenskonzept für alle Datenbanken erzwingen. Wenn am Datenbanknamen allein weder erkenntlich ist, ob es sich um ein Produktiv- oder Testsystem handelt oder auf welchem Host die Datenbank liegt, weil es mehrere Hosts mit identischem Datenbanknahmen gibt, dann wird bei Änderungen im Katalog regelmäßig Rechercheaufwand anfallen um identifizieren zu können, welche der beiden Datenbanken mit dem Namen ORCL gestern die DBID geändert hat. Dies ist vor allem deshalb so kniffelig und lässt sich unbefriedigend automatisch klären, weil der RMAN-Katalog nicht den Hostnamen der Datenbanken kennt. Datenbanken identifizieren sich bei RMAN einzig über ihre DBID. Innerhalb des Katalogs werden sie über einen eindeutigen DB_KEY unterschieden. Für Standby-Konfigurationen speichert der Katalog auch den DB_UNIQUE_NAME. Aber der Hostname bleibt stets anonym. Das ist schade und wäre sicherlich einen *Feature Request* bei Oracle wert. Aber weil der Hostname nicht für die Identifizierung von Katalogeinträgen herangezogen werden kann, ist es umso wichtiger ein unternehmensweites eindeutiges Namenskonzept durchzusetzen. Äußerst ungeschickt ist z.B. eine Situation, in der ein DBA durch die Räume laufen und sich erkundigen muss, wer die Datenbank namens ORCL kennt, die sich seit einigen Tagen beim Katalog meldet, deren Namen aber keine Rückschlüsse auf Herkunft oder Kritikalität zulässt.

In der Präsentation werden die für eine lückenlose Überwachung notwendigen Abfragen gegen den Recovery-Katalog vorgestellt.

Kontaktadresse:
Dr. Thorsten W. Grebe
twg-it
Geisenheimer Straße 6
D-14197 Berlin

Telefon: +49 (0) 176 31403337
E-Mail thorsten.grebe@twg-it.de
Internet: http://twg-it.de