RAC für Single Instance DBAs

Sinan Petrus Toma
Finanz Informatik GmbH & Co. KG
Hannover

Schlüsselwörter:

RAC, Real Application Cluster, Administration, Einsteiger, Einführung

Einleitung

Der Vortrag richtet sich vor allem an Datenbank-Administratoren, die in die spannende Welt des RACs einsteigen oder dabei immer noch etwas Kribbeln im Bauch haben.
Es werden die RAC-Basis-Konzepte und Differenzen zu Single Instance erläutert. Dazu immer die Best Practices und die passenden (SQL-)Statements, die Sie beim täglichen Arbeiten benötigen.
Die häufige Frage „muss ich das auf einem oder auf allen Knoten ausführen???“ werden Sie in Zukunft ganz sicher selber beantworten können.

RAC-Architektur

Der Hauptunterschied zur Single-Instance-Datenbank besteht darin, dass die RAC-Datenbank im Shared-Storage gespeichert ist und über mehrere Instanzen gleichzeitig zugänglich ist. Um die Datenintegrität zu gewährleisten, werden globale Locks eingesetzt und die Datenblöcke zwischen den Instanzen über das Interconnect ausgetauscht (Cache Fusion).

SPFILE

Jede Instanz kann ihr eigenes SPFILE nutzen. Die empfohlene Konfiguration ist jedoch, dass alle Instanzen ein gemeinsames SPFILE nutzen, das im Shared-Storage gespeichert ist.
Die Parameter können sowohl instanz-spezifisch (sid.parameter=value) oder instanz-übergreifend (*.parameter=value) definiert werden. Die instanz-spezifischen Parameter haben Vorrang.

```
SQL> alter session set open_cursors = 100 scope=both sid='*';
SQL> alter session set open_cursors = 200 scope=both sid='inst1';
```

Es gibt Parameter, die in allen Instanzen den gleichen Wert haben müssen (db_files) und Parameter, die in jeder Instanz einen unterschiedlichen Wert haben sollen (undo_retention) oder sogar müssen (instance_name).

```
SQL> select name, isinstance_modifiable from v$parameter;
```

Global Views

Für fast jede v$-View gibt es eine korrespondierende gv$-View mit einer zusätzlichen INST_ID Spalte, die Informationen über alle Instanzen anzeigt.

```
SQL> select inst_id, name, value from gv$parameter where name='sga_target';
```

Online Redo Logs

Jede Instanz nutzt ihr eigenes Thread zum Schreiben von Redo Logs. Jedes Thread kann aus einer unterschiedlichen Anzahl oder Größe von Redo Log Gruppen bestehen. Im Falle eines Instanz-Recoverys, kann jede Instanz die Redo Logs der anderen Instanzen lesen, da diese im Shared-Storage gespeichert sind.

```
SQL> alter database add logfile thread 1 group 5 ('+DATA', '+FRA') size 1G;
```

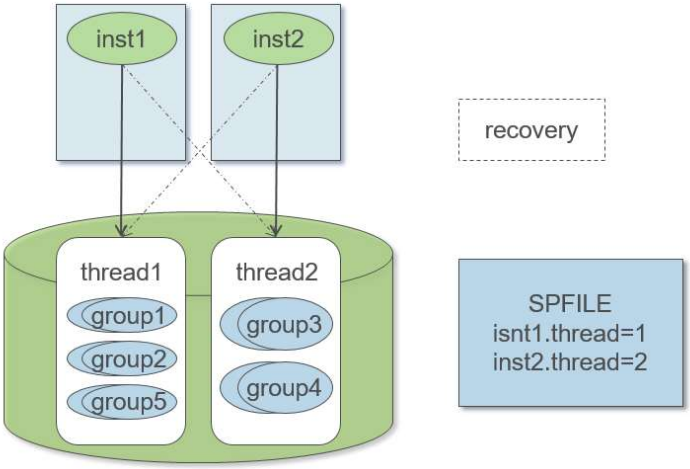

Abb. 1: Online Redo Logs

Undo Tablespace

Jede Instanz nutzt ihr eigenes Undo-Tablespace zum Schreiben. Alle Instanzen müssen im gleichen Undo-Mode operieren. Um lese-konsistenz zu gewährleisen, haben alle Instanzen Lese-Zugriff auf allen Undo-Tablespaces.

```
SQL> alter system set undo_tablespace=undotbs3 sid='inst1';
```

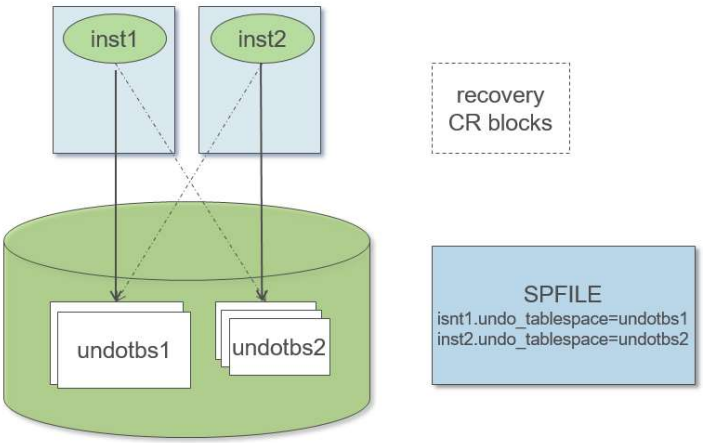

Abb. 2: Undo Tablespace

Temp Tablespace

Temp-Tablespaces werden gemeinsam von allen Instanzen genutzt. Jede Instanz versucht eine gleiche Anzahl von Extents aus jedem Temp-File zu allokieren, um Last gleichmäßig zu verteilen.

```
SQL> select inst_id, tablespace_name, file_id, extents_used
     from gv$temp_extent_pool;
```

Datafiles

Alle Datafiles müssen im Shared-Storage liegen. Ist dies nicht der Fall, kommt es zu einer ORA-Fehlermeldung, da jede Instanz den Zugriff auf allen Datafiles überprüft. Zum Hinzufügen von Databases im ASM kann nur der Diskgroup-Name mit einem + Zeichen verwendet werden. Zum Ändern kann sowohl die File-ID als auch der Komplette File-Pfad verwendet werden.

```
SQL> alter tablespace tbs1 add datafile '+DATA' size 1G;
SQL> alter database datafile '+DATA/…/tbs1.298.7211088' autoextend on
     maxsize 4G;
SQL> alter database datafile 42 autoextend on maxsize 4G;
```

Services

Datenbank-Services sollten nicht mit dem Package DBMS_SERVICE, sondern mit der Utility srvctl angelegt werden, damit diese als Cluster-Resource verwaltet werden. Diese können dann auf allen oder nur auf bestimmten Instanzen verfügbar sein. Zahlreiche Service-Attribute können modifiziert werden, wie z.B. für Connection Failover oder Load Balancing. Des Weiteren können diese Services gestoppt oder von einer Instanz zur anderen verlagert werden.

```
srvctl add service -db DB -service SRV -preferred inst1 -available inst2
```

Sessions

Um eine Session zu beenden, die nicht mit der gleichen Instanz verbunden ist, an dem der Administrator angemeldet ist, kann die Instanz-Nummer verwendet werden.

```
SQL> select sid, serial#, inst_id from gv$session;
SQL> alter system kill session '80,193,@1';
```

Virtual IP

Zusätzlich zu jeder Server-IP besitzt jeder Knoten eine weitere IP, die als Cluster-Resource verwaltet wird, die sog. Virtual-IP (VIP).

Local Listener

Der Local Listener auf jedem Knoten wird über die Server-IP oder über die entsprechende Virtual-IP erreicht. Fällt ein Knoten aus, so schwenkt die Virtual-IP auf einen anderen verfügbaren Knoten. Eine Verbindung über diese Virtual-IP würde dann mit einer sofortigen TNS-Fehlermeldung fehlschlagen, ohne auf das TCP-Timeout zu warten, wie im Falle der Server-IP.

Der Parameter LOCAL_LISTENER verweist auf diese IPs und hat somit in jeder Instanz einen unterschiedlichen Wert:
isnt1.local_listener=
(ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)(HOST = 5.240.186.**46**)(PORT = 1521)))
inst2.local_listener=
(ADDRESS_LIST = (ADDRESS = (PROTOCOL = TCP)(HOST = 5.240.186.**47**)(PORT = 1521)))

SCAN Listener

Zusätzlich zu den Local Listeners können pro Cluster, unabhängig von der Anzahl der Knoten, ein bis drei SCAN-Listener erstellt werden. Zu jedem SCAN-Listener gehört eine SCAN-Virtual-IP, die ebenso als Cluster-Resource verwaltet wird.
Der Parameter REMOTE_LISTENER verweist auf diese IPs und hat in allen Instanzen den gleichen Wert:
*.remote_listener=(ADDRESS_LIST =
(ADDRESS=(PROTOCOL=TCPS)(HOST=5.240.186.**34**)(PORT=1523))
(ADDRESS=(PROTOCOL=TCPS)(HOST=5.240.186.**35**)(PORT=1523))
(ADDRESS=(PROTOCOL=TCPS)(HOST=5.240.186.**36**)(PORT=1523)))

Jede Instanz registriert sich einerseits beim Local Listerner, der auf dem gleichen Knoten ausgeführt wird, und anderseits bei allen SCAN Listeners, die über alle Knoten des Clusters verteilt sein können.

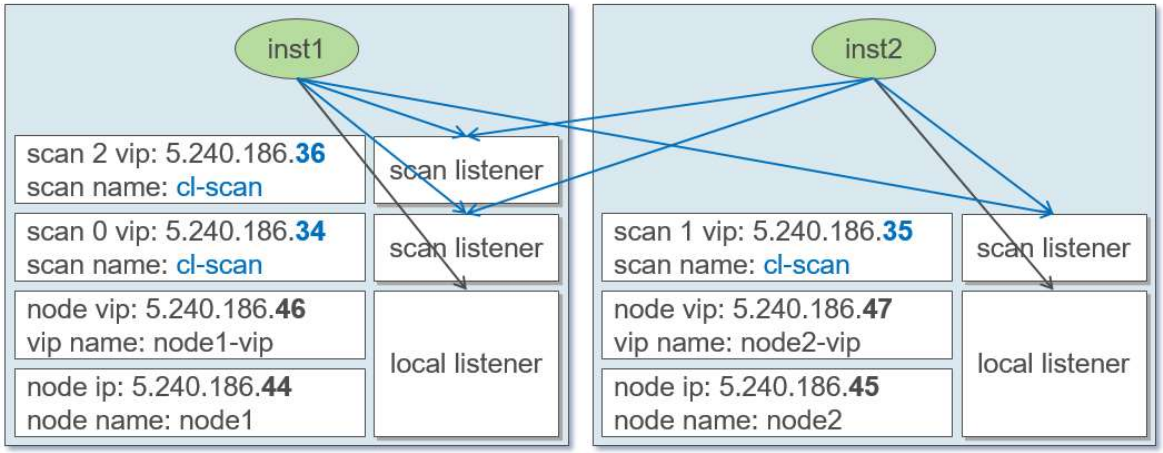

Abb. 3: Local und SCAN Listener

Erfolgt eine Verbindungsanfrage über einen SCAN-Listener, so ermittelt dieser den günstigsten (nach load balancing Informationen) Knoten und teilt dem Client die entsprechende Adresse des Local Listeners mit. Der Client sendet im zweiten Schritt eine Verbindungsanfrage zum Local Listener. D.h. der SCAN-Listener stellt keine Verbindungen her, sondern vermittelt nur die günstigste Instanz für den Client und antwortet mit deren Local Listener Adresse.

SCAN

SCAN steht für Single Client Access Name. Dieser Name kann im DNS auf bis zu drei IPs (SCAN-Virtual-IPs) aufgelöst werden. Dieser kann als Host-Name im Connection String des Clients verwendet werden, um unabhängig von den grade konfigurierten VIPs oder die Anzahl der SCAN-Listeners zu sein.
(ADDRESS=(PROTOCOL=TCP)(HOST=**cl-scan**)(PORT=1521))

Shutdown & Startup

Mit SQL-Plus kann nur die Instanz, mit der grade der Administrator verbunden ist, gestoppt oder gestartet werden. Um mehrere oder alle Instanzen zu stoppen und zu starten, ist die Utility srvctl zu verwenden.

```
srvctl stop instance -db DBNAME -instance inst1 -stopoption immediate
srvctl start instance -db DBNAME -instance inst1 -startoption open
```

Multitenant

Eine PDB kann in einer, mehreren oder allen CDB-Instanzen in einem bestimmten Status sein. Um die PDB in allen Instanzen zu öffnen, muss die Klausel INSTANCES verwendet werden.

```
SQL> alter pluggable database pdb1 open instances=all;
SQL> alter pluggable database pdb1 open instances=('inst1','inst3');
```

Die PDBs werden über Services als Cluster-Resource verwaltet.

```
srvctl add service -db CDBNAME -service PDB1_SVC -pdb PDB1
                   -preferred cdb1,cdb2 -available cdb3,cdb4
```

Wir der Services gestartet, so wird die PDBs in den *preferred* Instanzen geöffnet.

```
srvctl start service -db CDBNAME -service PDB1_SVC
```

AWR und ASH

AWR-Snapshots sammeln Daten über alle aktiven Instanzen. Die Daten werden jedoch nicht aggregiert, sondern pro Instanz gespeichert. Um einen AWR-Report für eine bestimmte Instanz zu erzeugen, kann das Skript awrrpti verwendet werden.

```
SQL> @?/rdbms/admin/awrrpti;
```

Entsprechend gibt es ein Skript für einen ASH-Report für eine bestimmte Instanz.

```
SQL> @?/rdbms/admin/ashrpti;
```

Kontaktadresse:

Sinan Petrus Toma
Finanz Informatik GmbH & Co. KG
Laatzener Str. 5
D-30539 Hannover

Telefon: +49 (0) 172 6183618
E-Mail sinan.petrus.toma@f-i.de
Internet: www.f-i.de