Praktijkervaringen met Data Guard

Gestructureerde configuratie heeft aandacht nodig

In het vorige nummer schreef ik over de mogelijkheden van Data Guard, de verbeterde technieken rondom standby databases. Als u dit leest is de site die met gebruikmaking van deze technieken is gebouwd inmiddels 'in de lucht'. Bijgaand een resumé van praktijkervaring die werd opgedaan bij het opzetten van de database voor de site. En passant worden een aantal tips gegeven om een meer transparante installatie op te zetten dan in de handleiding wordt voorgesteld.

Het gebruik van Oracle9i met Data Guard had bij aanvang van het project absoluut niet mijn voorkeur. Ik vond – en vind – dat een bedrijfskritische omgeving als in dit project niet geschikt is om te gaan experimenteren met een major release van een software-product. Krachten die buiten mijn invloedssfeer lagen bepaalden echter dat 9i het product was maarmee de wedstrijd gespeeld moest worden. Als speler in het team besloot ik mij niet met de wedstrijdleiding te bemoeien (dat kostte enige moeite….) en de aangeboden basisplaats in een team met vernieuwende iedeeën met beide handen aan te grijpen. Ik heb er geen spijt van. Zoals bij alle nieuwe ontwikkelingen waren de planningen optimistisch, werden de benodigde human resources onderschat en de bleven externe invloeden onbeheersbaar. We hebben een mooi vak, je moest alleen geen (eind)-gebruikers hebben….. Slagen in zo'n omgeving kan alleen als er een bevlogen ploeg mensen staat. Met deze ploeg is een mooi stukje werk neergezet, en ondanks mijn scepsis vooraf ben blij en trots aan het project meegewerkt te mogen hebben.

> ***Reken maar op een paar weken 'spelen' voordat de concepten en werkwijzen beheerst worden***

Er vielen de nodige hindernissen te overwinnen bij het opzetten van de database. Dat begint met de documentatie. Oracle levert een gids met concept-beschrijvingen en voorbeelden mee. Deze gids is zeker nog geen volwaardige manual. Reken zeker op een paar dagen tot een paar weken 'spelen' met de omgeving voordat de concepten en werkwijzen beheerst worden. Data Guard 'even' in een bestaande omgeving inzetten is vragen om moeilijkheden.

De gids is niet volledig. Zo bevat het hoofdstuk 'Troubleshooting, Problems During Standby Database Preparation' drie hele problemen. Ik kan u verzekeren: er zijn er veel meer! De huidige release van de software bevat nog wel wat fouten. Uitgaande van het project loop ik een aantal van de problemen langs.

Unieke naamgeving

Toen we met het project begonnen was de definitieve omgeving nog niet beschikbaar. In de uiteindelijke situatie bestaat de configuratie naast twee webservers en twee applicatieservers uit twee databaseservers. Bij de aanvang was alleen de uiteindelijke testomgeving beschikbaar, die van ieder type server (Web, Applicatie en Databaseserver) één exemplaar bevatte. Dit had toch wel belangrijke gevolgen voor het realiteitsgehalte van de proeftuin. Doordat er maar één databaseserver was moest de Data Guard omgeving op deze ene server worden uitgetest. Dat betekent extra complexiteit doordat er met twee verschillende SID's gewerkt moet worden. Ook kan de directory-structuur waarin de databasefiles worden opgeslagen niet identiek zijn voor beide databases. Tenslotte moet er gewerkt worden met de zgn. LOCK_NAME_SPACE parameter in de init<SID>.ora file. Hiermee wordt geregeld dat de dynamic lock manager voor iedere database over een unieke naamgeving voor het instellen van locks kan beschikken. In de omgeving waarin ik werkte (Sun met Solaris 8) bleek dit deels te werken. Het is van belang dat de standby database met gebruik van deze parameter worden opgestart. In eerste instantie probeerde ik het met de waarde 'STBY'. Dit werkte totdat een

switchover werd uitgevoerd waarbij de waarde van de parameter mee-wisselde: het lukte niet meer te staren. Door de waarde te wijzigen in 'STBY1' lukte de start wel. Kennelijk werd een éénmaal gezette waarde niet meer vrijgegeven.

Ik vind dat de handleiding behoorlijk rommelig is opgesteld. De belangrijkste informatie kan worden gevonden in de hoofdstukken 2, 5 en 6. Integraal lezen van deze hoofdstukken is aan te bevelen alvorens te beginnen met het opzetten van de omgeving. Hoofdstuk 6 bevat een aantal scenario's voor het opzetten van een standby database (twee keer op dezelfde server, op twee verschillende servers), onderhoud van de omgeving inclusief het uitbreiden van datafiles en/of tablespaces en het omschakelen van de primary database naar de standby database (failover en switchover). Ook tijdens het langslopen van de scenario's is regelmatig heen en weer bladeren tussen de verschillende hoofdstukken noodzakelijk, nergens wordt het trajact echt compleet weergegeven.

Ongetwijfeld gaat het creëren van een standby database met de Data Guard Broker in eerste aanleg een stuk eenvoudiger dan de weg die ik heb gevolgd. De omgeving waarin in het project wordt gewerkt vereist echter remote beheer (de server staat in Willemstad op Curaçao). In geval van nood moet dit beheer via een dial-up verbinding en een console interface kunnen plaatsvinden. Dat maakt gebruik van een grafische interface niet altijd mogelijk. Om de kennis op te doen die in geval van nood noodzakelijk is om de herstelwerkzaamheden via de prompt uit te voeren is gekozen de gehele setup op deze manier aan te pakken. Hoewel waarschijnlijk niet de kortste weg wil ik deze toch sterk aanbevelen. Het gebruik van de verschillende commando's vanaf de SQL-prompt verschaft veel inzicht in de opzet van Data Guard. Het is de beste manier om inzicht te verkrijgen in de architectuur en de werkwijze van het geheel.

Opletten

Oracle geeft in de beschrijving van de praktijksituaties uitgebreide adviezen over de naamgeving van de SID's, de namen in de setup van Oracle Net en de namen in de directory-structuur. Ik vind de geadviseerde naamgeving erg verwarrend: overal komen verwijzingen naar de rol van de betreffende database voor. Databasenamen als 'standby1' en 'primary1' lijken in eerste aanleg erg duidelijk. O wee echter als de databases van rol wisselen na een switch-over. De primary database heet dan standby1 en omgekeerd. De verwarring wordt dan wel erg groot en het maken van fouten is bijna gegarandeerd. Ik vind dit niet bijdragen aan de kwaliteit van de oplossing die we de gebruiker zo bieden: De fouten die een DBA maakt, straalt uiteindelijk ook weer af op het product dat wordt gebruikt. Voor een omgeving met een primary en standby database op één server ontkomen we echter niet aan verschillende namen voor de databases. Kies in dat geval liever voor <SID>1 en <SID>2. Het blijft nog steeds opletten, maar het is minder verwarrend dan het voorstel uit de handleiding.

Nadat de databases op de ene testserver succesvol waren geïnstalleerd en switch-over en switch-back waren uitgeprobeerd kwamen de productieservers beschikbaar. Daarop is in eerste instantie na het vaststellen van de sizing en indeling van de filesystems de productiedatabase aangemaakt. Voor het opzetten van de standby-omgeving kwam vervolgens de inrichting van de parameterfiles om de hoek. De handleiding geeft in de praktijkvoorbeelden aan voor de standby database een kopie van de parameterfile van de primary database te maken. Deze wordt dan vervolgens aangepast om de standby database te kunnen starten. Dat heeft het bezwaar dat allerlei parameters die normaal gesproken voor beide rollen van een database dezelfde waarde hebben nu in twee parameterfiles voorkomen.

We hebben een mooi vak, je moest alleen geen eindgebruikers hebben

Als de gebruikte machines identiek zijn en beide zowel de rol van standby server als de rol van primary server kunnen vervullen dan staan deze parameters zelfs in 4 bestanden. Iedere versie moet immers op beide machines aanwezig zijn. Deze methode vind ik sterk af te raden. Het is een kwestie van wachten totdat er een parameter verandert. In de haast wordt ooit vergeten een wijziging consistent in alle files mee te nemen. Simpelweg kopiëren gaat niet, de inhoud van de files is verschillend.

Oracle Net

Een vergelijkbaar probleem treedt op bij het inrichten van Oracle Net. Ook hiervoor worden 'rolbevestigende' aliases voorgesteld voor de connectstrings in tnsnames. Mijn uitgangspunt is dat zaken geregeld moeten worden op het niveau waarop ze betrekking hebben. Oracle Net is slechts de transportlaag, het aanbrengen van afhankelijkheden en aanduidingen die betrekking hebben op Data Guard horen daar niet thuis. Onderstaand een deel uit tnsnames zoals ik deze gebruik:

In deze opzet worden voor iedere database drie connectstrings gedefinieerd. De eerste met dezelfde naam als de SID van de database. Hiermee kan op iedere database-server een verbinding naar de lokale database worden gelegd. Dit wordt bereikt door

als host 'localhost' te specificeren. De andere twee verwijzen direct naar een database met dezelfde SID op een specifieke host. Tnsnames kan zo op ieder machine gelijk zijn en met een distributiemechanisme als 'rdist' worden verspreid.

Het probleem met de parameterfiles heb ik als volgt opgelost: er worden drie parameterfiles aangemaakt met de namen XYZ_generic.ora, XYZ_primary.ora en XYZ_standby.ora. Deze bevatten respectievelijk de algemene geldende parameters, de specifieke parameters voor de primary-rol en de specifieke parameters voor de standby-rol. De specifieke parameters zijn per server (meestal) verschillend. De algemene parameters voor bijv. tuning en SGA-sizing toon ik hier niet. Het eerste getoonde voorbeeld laat de parameters zien die voor de primary role nodig zijn. Ten eerste wordt de service waarnaar de redolog informatie moet worden geschreven vastgelegd. Daarrvoor wordt de hierboven gedefiniëerde connectstring XYZ2 gebruikt. Deze file staat op de dbsrv1. Op de dbsrv2 zal de service XYZ1 worden gespecificeerd.

Vervolgens wordt het file-management ingesteld. De getoonde waarde 'auto' verzorgt het automatisch creëren van datafiles op de standby server als er op de primary server datafiles worden toegevoegd.

Het volgende voorbeeld toont de standby parameters op de dbsrv1. De FAL_SERVER en FAL_CLIENT activeren de archive log gap detection. Het proces dat dit verzorgt detecteert automatisch of de standby database beschikt over alle noodzakelijke archived redologfiles. Mocht hierin een aantal files ontbreken dan worden deze automatisch naar de standby server gekopieerd. Let op dat op de andere server de waarden van de FAL-parameters moeten worden verwisseld. Verder zien we de STANDBY_FILE_MANAGEMENT parameter hier weer terugkomen. Waarom deze dan niet in de generieke file onderebracht? Omdat ik daarin niet graag Data Guard parameters opneem: zo kan de database met de generieke parameterfile ook nog worden opgestart zonder Data Guard opties. Als het er meer worden kan altijd worden overwogen een extra file met algemene DATA GUARD parameters aan te maken.

Vervolgens zijn er drie parameterfiles, initXYZ_s.ora, initXYZ_p en initXYZ.ora aangemaakt die worden gebruikt bij het opstarten van de database. Deze bevatten alleen IFILE-opdrachten om de juiste set parameters samen te stellen uit de verschillende files. Als voorbeeld de 'pfile' voor de primary database:

Om vergissingen te voorkomen zet ik geen pfile meer in de <ORACLE_HOME>/dbs directory. De parameterfiles komen alleen voor in de pfile-directory zoals die in een volgens OFA-standaarden geïnstalleerde omgeving bestaat. Hiermee wordt afgedwongen dat bij het starten van de database altijd de optie 'pfile=…..' aan het startup commando moet worden meegegeven. Daardoor kan automatisch een al dan niet verkeerde pfile worden gebruikt.

Uiteraard is het in het bestek van dit artikel niet mogelijk de handleiding die Oracle verstrekt even over te doen. Wel hoop ik een aantal punten onder de aandacht gebracht te hebben zodat het gestructureerd configureren van een Data Guard omgeving een zetje in de goede richting heeft gekregen. Ik hoor uw vragen en opmerkingen graag via de mail.

Prijsvraagje:

In het vorige nummer schreef ik een klein prijsvraagje uit: Wie kent er nog het 'Oracle-tool' UFI?

Het juiste antwoord was : 'User Friendly Interface', de benaming die Oracle bedacht had voor dat wat tegenwoordig SQL*Plus heet. Inmiddels denken we wel iets anders over gebruikersvriendelijk. Toen werd het tool nog geschikt geacht voor eindgebruikers….

Ik kreeg binnen de gestelde 2 weken ruim 30 antwoorden per email waarvan 18 de goede oplossing bevatten. De foute antwoorden vermeldden meestal dat UFI de voorloper zou zijn geweest van SQL*Forms. SQL*Forms bestond in die tijd echter uit de tools iac/iag/iap, in Forms 1.3 met het befaamde (beruchte) vraag/antwoordspel. De weergave daarvan zou nog jarenlang het source-format van Forms in de versies 2.0 en 2.3 bepalen.

Na loting zijn er drie prijswinnaars bepaald:
1e prijs: Bagage trolly Jan Broos van Expertview
2e prijs: RVS thermosfles Kees Winkelman van Transfer Solutions
3e prijs: Muismat Fred Willemse van CMG

Als het goed is zijn de adressen van de prijswinnaars inmiddels allemaal achterhaald (via de mail kan alleen een virtuele prijs worden gestuurd) en hebben ze hun prijs ontvangen.

Carel-Jan Engel
is Technisch Directeur van, en Senior Problem Solver bij Ease Automation BV. Hij is vanaf 1985 in verschillende rollen vrijwel ononderbroken bezig met ontwikkeling en beheer van systemen op basis van Oracle. Email: cjpengel@ease.nl