Oracle en Security (3)

Features voor beschikbaarheid informatie

Oracle is vanaf release 7 bezig geweest met het uitbrengen van nieuwe hulpmiddelen om de beschikbaarheid van informatie opgeslagen in een Oracle database tot zeer hoog te kunnen opvoeren. Het merendeel van de hier behandelde 'nieuwe' features begint echter pas goed tot zijn recht te komen vanaf Oracle9i, terwijl er ook in Oracle10g nog nieuwe functionaliteit aan is toegevoegd. Alle features op het gebied van beschikbaarheid tezamen vormen nu in feite een behoorlijk kompleet gamma van hulpmiddelen waarmee voldaan kan worden aan (bijna) alle eisen.

In dit artikel wil ik een aantal van deze hulpmiddelen doornemen en deze beoordelen op wat zij kunnen doen op het gebied van beschikbaarheid.
De eerste uitbreiding op het gebied van beschikbaarheid was 'Parallel Server', sinds Oracle 9 bekend onder de naam 'Real Application Cluster' ofwel RAC. RAC biedt de mogelijkheid om twee (of meer) instanties van één database op meerdere machines te hebben draaien. Het voordeel op het gebied van beschikbaarheid is, dat er bij uitval van één van de twee machines gewoon doorgedraaid kan worden op de andere machine. Tot en met Oracle8i moest hier echter een hoge prijs voor worden betaald op het gebied van performance. Indien een gebruiker een record wilde wijzigen, dat zojuist gewijzigd was op de andere machine moest dit record vanuit het geheugen van de andere machine via schijf naar het geheugen van 'zijn' machine worden getransporteerd. Twee dure IO's. De consequentie hiervan was of een relatief slechte performance, of ervoor zorgen dat een record (bijna) altijd op slechts één van de machine gewijzigd zal worden (datapartitionering).
Een alternatief was om slechts één van beide machines daadwerkelijk te gebruiken en de andere klaar te hebben staan om het werk over te kunnen nemen. Door parallel server kon deze machine bij uitval van de andere automatisch in zeer korte tijd de taken van de uitgevallen machine overnemen. Deze configuratie wordt meestal een HA-cluster (High Availability-Cluster) genoemd.

Met de komst van Oracle9i is dit alles niet meer nodig. Indien een record op de andere machine is gewijzigd wordt deze indien nodig naar de 'eigen' machine gehaald van geheugen naar geheugen, dus zonder de twee dure IO's.

Stand-by

Release 7 zag ook de geboorte van de 'Standby Database', een feature welke in Oracle9i de naam 'Data Guard' heeft gekregen. In eerste instantie was de standby database niets anders dan een kopie van de productie database, die continu in een staat van recovery was. Handmatig moest ervoor gezorgd worden, dat wijzigingen op de productie database (opgeslagen in redolog files) gekopieerd werden naar de machine waar de standby database op stond en dan vervolgens met het recovery mechanisme hierop aangebracht werden. Het doel van een standby database was om in geval van uitval van een database machine, de tijd die het kost om een andere machine zijn taak te laten overnemen, drastisch te verminderen. Dit lijkt functioneel gezien erg op de hierboven genoemde HA-cluster.

Oracle8 komt met de feature logminer. Deze feature is oorspronkelijk ontwikkeld door derden. Het biedt de mogelijkheid

> ***Een techniek om de beschikbaarheid te vergroten is de 'transportable tablespace'***

om in de redolog file een transactie te isoleren en deze vervolgens te gebruiken om die ene transactie terug te draaien. Het is hiermee bijvoorbeeld mogelijk om een tabel die per ongeluk is weggegooid weer terug te halen.
Dit is aan de ene kant een zeer krachtig hulpmiddel. De dingen die ermee kunnen zijn nauwelijks mogelijk op een andere manier. Aan de andere kant is het een zeer gevaarlijk hulpmid-

del. Het is mogelijk hiermee gegevens in de database te krijgen die er op een normale manier nooit in zouden kunnen komen. In een database wordt via constraints in de gaten gehouden of gegevens die bij elkaar horen compleet zijn. Van gegevens die in de tijd gezien achtereenvolgens opgevoerd moeten zijn kan achteraf niet meer gecontroleerd worden of dit wel gebeurd is. Veelal is dit op het moment van opvoeren wel gecontroleerd via de applicatie. De wijzigingen die met logminer achteraf aan te brengen zijn hoeven hier niet aan te voldoen.

Flashback query

Logminer is in latere releases de basis geweest van twee nieuwe features op het gebied van beschikbaarheid:

- Logische standby database
- Flashback query

De in een vorige paragraaf genoemde standby database waarbij wijzigingen op de productie database rechtstreeks worden getransporteerd via de redolog files heet vanaf nu een fysieke standby database. Deze database is blok voor blok identiek aan de productie database waar hij een kopie van is. Met behulp van een op logminer gebaseerde techniek is het echter ook

Door gebruik te maken van de logminer techniek kunnen veel menselijke fouten worden hersteld

mogelijk om de SQL-statements te reconstrueren welke tot de wijzigingen hebben geleid op de productie database. Deze nu kunnen vervolgens worden toegepast op de standby database. Deze hoeft nu niet meer blok voor blok gelijk te zijn aan de productie database, maar is dit wel inhoudelijk. De voordelen van de fysieke standby database zijn:

- Eenvoudige en vooral zeer beproefde techniek
- Performance

De techniek van de fysieke standby database is niets anders dan de techniek waarmee al meer dan een decennium lang een hot back-up van een Oracle database wordt gerecovered. Dit betekent dat, alhoewel de standby database pas in een van de latere versies van release 7 is geïntroduceerd, de basistechniek al heel oud en beproefd is.
Aangezien de logische standby database wat extra stappen kent ten opzichte van de fysieke zal de performance altijd achterblijven. De vraag is echter of dit belangrijk is. Het systeem waarop de standby database staat moet in staat zijn om de werkzaamheden van de productiemachine over te nemen. Dit eist impliciet een betere performance dan nodig is voor het bijwerken van de standby database. Het grote voordeel van een logische standby database is dat het mogelijk is om de statements op inhoudelijke kenmerken te groeperen, zodat het zeker in komende releases steeds meer mogelijk zal zijn om te kiezen wat er wel en wat er niet mee gaat naar de standby database. De standby database is dan niet meer een kopie van de productie database maar wel een inhoudelijk consistente subset hiervan.

Tablespaces

Een laatste techniek om de beschikbaarheid te vergroten is de 'transportable tablespace'. Deze feature, afkomstig uit Oracle8, maakt het mogelijk om een deel van de database te verplaatsen of te kopiëren naar een andere database. Dit deel moet dan wel bestaan uit één of meerdere complete tablespaces.
Bovendien moet de verzameling van tablespaces een complete integere set data bevatten. Dit uit zich in eisen als:

- Er mogen geen referentiële relaties zijn tussen tabellen in de set van tablespaces en tabellen daarbuiten;
- Als een tabel zich in een van de tablespaces bevindt moeten ook alle indexen op deze tabel zich in deze tablespaces bevinden;
- Als een partitie van een tabel zich in een van de tablespaces bevindt, moeten ook alle andere partities van deze tabel zich in deze tablespaces bevinden.

Op het gebied van beschikbaarheid kan deze techniek worden ingezet voor het kopiëren van data van de productiedatabase naar een database die gebruikt wordt voor management query's. Hiermee kan een in een productieomgeving meestal moeilijk beheersbaar deel van de load van de database worden afgesplitst. Als gevolg hiervan is het beter mogelijk om structureel te voldoen aan de performance-eisen die gebruikers minimaal noodzakelijk achten. Indien niet aan deze minimale eisen wordt voldaan, is er sprake van een vorm van onbeschikbaarheid.

Beschikbaarheid

Er kunnen verschillende redenen zijn waarom informatie opgeslagen in een Oracle database niet altijd beschikbaar is. Om de beschikbaarheid onder controle te kunnen houden is het gewenst om een zo volledig mogelijke inventarisatie te maken van deze redenen en per geval te beschrijven of dit acceptabel is of welke maatregelen er genomen moeten worden om de beschikbaarheid wel te kunnen garanderen. In het kader van dit artikel beperken we ons tot de database. Elke andere component van een informatiesysteem zal op dezelfde manier bekeken moeten worden.

Onbeschikbaarheid kan gepland en ongepland zijn. Redenen voor geplande onbeschikbaarheid kunnen zijn:

- In gebruik nemen nieuwe hardware
- Upgrade van systeem software (OS, DBMS etc.)

- Upgrade van de applicatie software
- Back-up
- Batch window

Ongeplande onbeschikbaarheid wordt altijd veroorzaakt door iets wat fout gaat:

- Hardware
  - Database server
  - Schijven
- Software
  - Systeem software (OS, DBMS, etc)
  - Applicatie
- Menselijke fouten

Met de eerder genoemde nieuwe features plus een aantal al lang bestaande is het mogelijk om een dusdanige architectuur voor een database systeem te ontwerpen dat de database beschikbaar blijft als één van bovengenoemde fouten optreedt. Ook kan de noodzaak voor geplande downtime zo minimaal mogelijk worden gehouden. Een aantal voorbeelden hiervan:

- Bij gebruik van een RAC is het mogelijk om een nieuwe database server toe te voegen, zonder dat de database hiervoor uit de lucht hoeft.
- In Oracle10g is het bij een RAC mogelijk om een upgrade te doen van de Oracle software eerst op de ene node en dan op de andere, zonder dat hier de database voor uit de lucht hoeft.
- Menselijke fouten kunnen worden teruggedraaid door gebruik te maken een logminer techniek
- Door een standby database zo in te richten dat de verwerking van wijzigingen met een bepaalde vertraging worden aangebracht, heb je de mogelijkheid om terug te gaan naar een tijdstip waarop de menselijke fout nog niet gemaakt is.

Conclusie

Omdat er zoveel verschillende mogelijkheden met elk hun eigen implicaties bestaan, is het zinvol om beschikbaarheid als een apart issue te beschouwen bij het ontwerpen van een database applicatie. Oracle heeft hier een whitepaper over geschreven waarin een voorbeeld wordt uitgewerkt. Hierin wordt op verschillende manieren gebruik gemaakt van standby databases in combinatie met RAC's. Tevens worden procedures aangegeven die in combinatie met de voorgestelde architectuur tot een zo hoog mogelijke beschikbaarheid moeten leiden.

Gerard Uiterwaal is Oracle- en security expert en werkzaam bij Motiv IT Masters. Indien u in een van de volgende uitgaven van Optimize graag een specifiek onderdeel belicht zou willen zien dan kunt u dit aangeven via e-mail: gerard.uiterwaal@motiv.nl. De auteur streeft ernaar zoveel mogelijk vragen te beantwoorden.

Adv. Array