Goed gereedschap is het halve werk

Gebruiksimpressie Oracle DBA-tool (10)

Rondom het Oracle RDBMS leveren diverse leveranciers hulpmiddelen die het leven, maar vooral het werk van een DBA helpen te veraangenamen. Een eerste blik in de IT Vendor Guide 2004 leert dat er tientallen tools worden vermeld die op één of andere manier onder deze definitie vallen. Daarnaast zijn er ongetwijfeld nog tools die een bespreking waard zijn, maar die niet in Rubriek 13 (database-beheer), al is het alleen maar omdat blijkt dat DBA's soms voor ontwikkelaars bedoelde producten graag 'misbruiken' voor het beheer.

Deze aflevering bespreek ik de Hotsos Profiler. Dit is in de handen van de professionele performance-analist een zeer krachtig stuk gereedschap. De Hotsos Profiler is een analyser voor de op het 10046-event gebaseerde trace-files. In 1991 begon Oracle met het inbouwen van *extended SQL trace*, waarmee analyse van performanceproblemen sterk verbeterde. Helaas is het tracen zelf en de analyse van de vastgelegde informatie niet echt simpel. Hotsos heeft hiervoor de Hotsos Profiler ontwikkeld. De Profiler is te zien als een krachtiger tegenhanger van het bij de Oracle database meegeleverde tkprof. Sparky is een GUI die wordt gebruikt voor aansturen van het tracen zelf.

Historie

Cary Millsap richtte samen met Gary Goodman in oktober 1999 Hotsos op. De naam wordt uitgesproken als het Engelse 'hot sauce', en is een naam met een knipoog. Tijdens zijn carrière bij Oracle was Cary Millsap in 1995 initiatiefnemer van de System Performance Group. Veel van zijn teamleden legden een bijzondere belangstelling voor zo heet (gekruid) mogelijke sauzen aan de dag. Alles wat u en ik voor 'heet' vinden doorgaan is naar hun smaak maar flauw. Met Hotsos bedoelen ze dan ook echt 'hete saus'.

De bouw van de Hotsos Profiler werd in 2001 gestart. Daarbij werd gekozen voor Perl als programmeertaal. De reden hiervan was dat hiermee verwacht werd sneller te kunnen ontwikkelen. Wellicht zou het resulterende programma wat langzamer draaien dan met een andere taal mogelijk was, maar de mogelijkheid om snel te kunnen ontwikkelen en verbeterde versies te kunnen uitbrengen gaf de doorslag. De eerste versie leek in zijn uitvoer nog erg op tkprof, en de uitvoer werd gesorteerd op 'elapsed time'. Er waren nog geen andere sorteermogelijkheden. Begin 2002 was versie 1.1 beschikbaar. In deze versie werd het voor het eerst mogelijk de hiërarchie van het *recursive SQL* te volgen in de uitvoer.

In juli 2002 volgde de sprong van versie 1.33 naar versie 3.65. Vanaf dat moment werd het buildnummer als versienummer gebruikt, waardoor versie 2 werd overgeslagen. Versie 4.0 kwam uit in februari 2003. De huidige versie is versie 4.0.10. Voor versie 5 staan meerdere wensen op het programma: op XML gebaseerde uitvoer, trend analyse, opslag van de resultaten in de database, XSLT-transformatie voor de HTML-uitvoer, context gevoelige *highlighting*.

Productopbouw

Sparky en de Hotsos Profiler vormen een twee-éénheid. Dat is dan ook de reden ze hier samen te bespreken. Ze zijn echter ook zonder meer los van elkaar te gebruiken. Als tracefiles met

Verantwoording

Deze serie tool-besprekingen is gebaseerd op mijn persoonlijke ervaringen met de betreffende tools. Ieder besproken tool wordt geïnstalleerd in de omgeving waarvoor het is bedoeld en de software wordt met de nodige testcases uitgeprobeerd. Het artikel kan worden gezien als een 'gebruiksimpressie'. De weergave van de resultaten is daarom per definitie subjectief en voor discussie vatbaar. Ik stel me graag open voor discussie. Opmerkingen, vragen en suggesties (ook voor eventueel in volgende afleveringen te bespreken tools) zijn welkom op mijn email-adres. Op uw reacties kom ik graag terug.

een ander product worden geanalyseerd, zoals het eerder genoemde tkprof of de tracefile repository van Miracle, dan is Sparky nog steeds erg handig om de juiste sessie te vinden, het tracen te starten en de tracefile vanaf de server op te halen. Andersom is de Profiler ook prima te gebruiken voor het analyseren van tracefiles die los van Sparky zijn aangemaakt.

Andere producten die door Hotsos worden gevoerd zijn Laredo en Interceptor. Laredo is een product dat het mogelijk maakt de gevolgen voor performance van wijzigingen in een applicatie te voorspellen. Dat geldt voor bijvoorbeeld de overgang van de Rule Based Optimizer naar de Cost Based Optimizer, veranderingen in de door de optimizer gevolgde strategie tussen twee Oracle versies of voor het toevoegen of verwijderen van een index. Met Laredo kunnen problemen in een applicatie worden ontdekt voordat deze in productie wordt uitgerold.

Veel van Millsap's teamleden legden een bijzondere belangstelling voor zo gekruid mogelijke sauzen aan de dag

Interceptor werd ontwikkeld als antwoord op de toenemende hoeveelheid applicaties die zeer inefficiënt SQL genereren. Een voorbeeld hiervan is een applicatie die na ieder afgevuurd SQL statement een *commit* geeft. Interceptor onderschept op SQL*Net niveau alle communicatie en brengt hierin een forse efficiencyverbetering aan door de onnodige commits na een query achterwege te laten.

Tenslotte levert Hotsos het tool Sparky. Dit is een GUI voor het Windows platform. Sparky maakt het mogelijk eenvoudig tracefiles aan te maken en klaar te zetten voor analyse met de Profiler. De eerste releases van de Profiler waren alleen op abonnementbasis via de website van Hotsos te gebruiken. De op die manier vergaarde duizenden tracefiles hebben Hotsos veel inzicht gegeven bij het doorontwikkelen van de Profiler. Met Sparky wordt het proces voor het verzamelen van tracefiles geautomatiseerd, waardoor deze op een eenduidige manier kunnen worden vervaardigd.

Installatie

Om efficiënt van de Profiler gebruik te maken is Sparky erg handig. Sparky is bovendien gratis, dus niets weerhoudt ons ervan om te beginnen met het installeren van Sparky. Sparky bestaat uit twee delen: het GUI-deel, dat beschikbaar is voor de Windows versies NT, 2000 en XP, en de daemon die op de databaseserver het ophalen en inpakken van de tracefiles verzorgt. Hotsos adviseert om één diagnosesysteem te configureren, waarvandaan alle databaseservers centraal kunnen worden benaderd. Sparky is (na registratie) te downloaden van www.hotsos.com. Het is even zoeken op de site, downloaden gebeurt niet vanuit het 'Products' menu met de beschrijving van Sparky, maar via de menukeuze 'Library'. Bij Sparky zit een uitgebreide handleiding, die stap voor stap aangeeft aan welke eisen de systemen moeten voldoen, hoe dit is te verifiëren en hoe vervolgens de installatie moet worden uitgevoerd. De belangrijkste eisen zijn: De Oracle client software (SQL*Net) moet op het diagnosesysteem aanwezig zijn en op de database server moeten minimaal Oracle 7.0.12 of later en Perl v.5.5.3 of later beschikbaar zijn. Onder Windows wordt na het uitpakken van de zip-file het programma setup.exe gedraaid, dat met Installshield de GUI installeert. Vanuit de hotsos programmadirectory kan daarna de Perl-gebaseerde daemon software naar de server worden gekopieerd. Voor de met veel beveiliging omgeven systemen is het tenslotte nog van belang ervoor te zorgen dat er een port wordt opengezet in de firewall, om de Sparky-GUI met de daemon te kunnen laten communiceren.

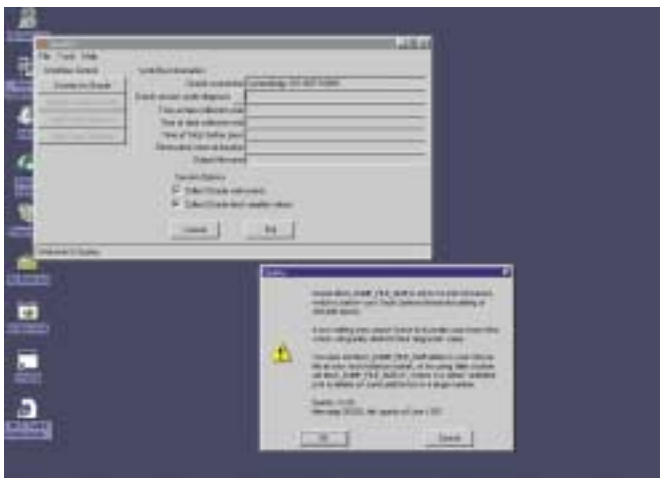
Installatie van de Profiler is net zo eenvoudig als Sparky. Ook de Profiler is alleen op de genoemde Windowsvarianten beschikbaar. Hier geen installatie van een serverdeel, wel registratie van de software in combinatie met de hardware waarop het geïnstalleerd is. Het registratieprogramma legt hiervoor verbinding met een registratieserver van Hotsos. Daarvoor is dus een verbinding met het internet noodzakelijk. Tijdens de registratie wordt een signature vastgelegd in een licentiebestand, samen met de begin- en einddatum van de verkregen licentie. Na het verlopen van de licentie moet deze worden vernieuwd met dezelfde procedure, nadat bij Hotsos een nieuwe licentie is aangeschaft.

De Profiler kan ook worden aangeschaft als service op de website van Hotsos. De tracefiles worden dan, nadat deze met Sparky zijn verzameld, ge-upload naar Hotsos en op de server van Hotsos geanalyseerd.

Performance-analyse

De performance-analyse met de Hotsos Profiler is gebaseerd op de methode zoals die het eerst door Anjo Kolk in zijn Yapp-paper werd beschreven. Simpel gezegd komt het erop neer dat de responstijd, die een gebruiker ervaart tussen het geven van een opdracht en het krijgen van het bijbehorende antwoord bestaat uit twee componenten: verwerkingstijd en wachttijd. Performance-tuning draait om het verminderen van de repons-

tijd. Dat kan alleen maar door één of beide samenstellende componenten te reduceren. De verwerkingstijd kan (soms) worden gereduceerd door meer hardware bij te plaatsen. Nog simpeler is het om minder te verwerken. Veel systemen geven slechte responstijden door inefficiënte verwerking van volstrekt overbodig SQL. Het stoppen met verwerken van overbodig SQL is de beste methode van performance-tuning. Niet alleen het betreffende proces zelf heeft er baat bij, voor andere processen blijft ook nog eens meer verwerkingscapaciteit over. De beste en tevens de goedkoopste manier om de responstijd te beïnvloeden, is doorgaans door de wachttijd te reduceren. De kernel van het Oracle RDBMS kent in versie 9i rond de 400 verschillende redenen om te wachten. Denk hierbij aan het wachten op latches en locks, op I/O of op CPU. Met *extended SQL tracing* kan deze wachttijd per event worden vastgelegd. De Hotsos Profiler is in staat om deze wachttijden te analyseren, en ook om de recursieve SQL-statements met de door de applicatie uitgevoerde statements in verband te brengen. Cary Millsap heeft over het verbeteren van Oracle Performance onlangs een boek uitgebracht, dat onder de titel *Optimizing Oracle Performance* is verschenen bij O'Reilly. In dit boek wordt performance-tuning volgens deze methode uitstekend uitgelegd.



Afbeelding 1: Sparky waarschuwt voor een te krappe instelling van de parameter max_dump_file_size

Sparky

Het aanzetten van tracing voor een Oracle sessie is niet echt eenvoudig. Eerst moeten de SID en het bijbehorende Serial# worden gevonden, en vervolgens moet het commando worden gegeven om de tracing te starten. Daarbij is het ook van belang dat de maximale grootte van een tracefile binnen de Oracle parameters ruim genoeg staat ingesteld. Een tracefile kan gemakkelijk vele megabytes groot worden. De minimaal benodigde grootte kan in de voorkeursinstellingen van Sparky worden vastgelegd. Sparky zal bij het verbinden met de database controleren of de grootte aan deze instelling voldoet, en zal dan ook waarschuwen als dit niet het geval is (zie afbeelding 1). Ook de controle op de parameter timed_statistics wordt voorafuitgevoerd. Zonder timed_statistics is de analyse vrij zinloos.

Afbeelding 2: Een woud van sessies in de databases

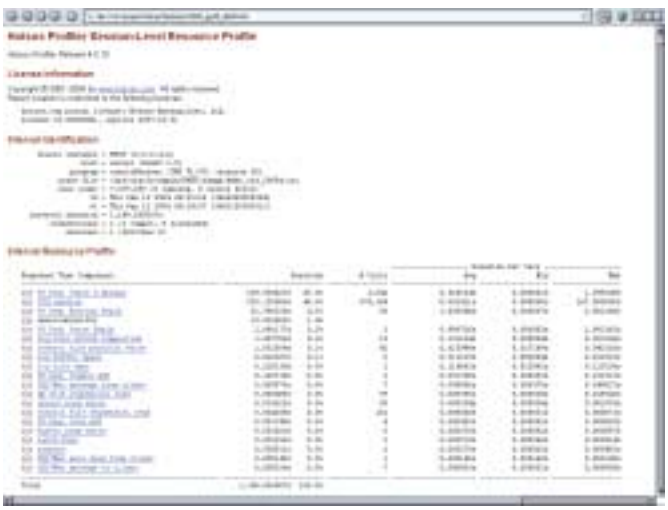
Het zoeken naar de juiste sessie is, zeker op een druk bezet systeem, als zoeken naar een speld in een hooiberg. Afbeelding 3 laat een aantal sessies zien binnen de database op mijn laptop. Zelfs op een klein systeem is dat al een fors aantal sessies. Maar, als we weten wat de naam van de ingelogde gebruiker is, zoals in afbeelding 2 is ingevuld, wordt het geheel al een stuk eenvoudiger. We zien dat *osuser* Administrator maar drie sessies heeft. Eén daarvan is Sparky zelf, de andere twee zijn sqlplus sessies. Nadat één van de sessies is geselecteerd is een druk op de knop genoeg om tracing van de betreffende sessie te starten. Voor complexe omgevingen als bijvoorbeeld de ERP systemen van JDEdwards, Siebel of Peoplesoft kunnen desgewenst eigen selectiemethoden worden toegevoegd aan de ingebouwde selectiemogelijkheden van Sparky. De handleiding beschrijft hoe dit kan worden uitgevoerd.

Afbeelding 3: De selectie op basis van de osuser van de sessie

Profiler

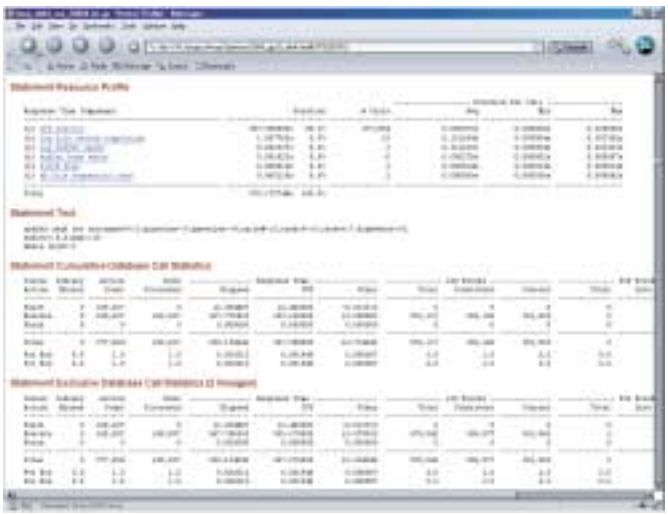
Zodra het tracen van een sessie met Sparky wordt beëindigd wordt de betreffende tracefile door de daemon op de server ingepakt in een zip-file, samen met een xml-bestandje met

informatie over de server waarop Oracle draait. Vervolgens wordt dat bestand door Sparky opgehaald en in een directory naar keuze opgeslagen. Vanaf dat moment kan de trace-file met de Profiler worden geanalyseerd. Over dat analyseren is niet zoveel te schrijven. In de Profiler wordt het te analyseren bestand geselecteerd, en direct daarna begint de analyse. Deze analyse vergt meestal minder dan een minuut. Zodra de analyse gereed is wordt het resultaat in de browser getoond.
Een paar weken geleden heb ik geassisteerd bij een vreemd performanceprobleem. Bij een proefrun voor het laden van een Datawarehouse werd één procent van de verwachte hoeveelheid data ingelezen. In de testomgeving was hiervoor ongeveer een minuut nodig. In de productieomgeving duurde dit veel langer, bijna twintig minuten. Beide omgevingen bevonden zich in dezelfde database. Het enige verschil was dat in de testomgeving de te vullen fact-tabel werd samengesteld uit vijf andere tabellen in het zelfde schema, terwijl in de productieomgeving deze vijf tabellen over twee schema's waren verdeeld. De omgevingen waren identiek, en men begon ernstig te twijfelen aan het vermogen van Oracle om een vijfvoudige join uit te voeren als de tabellen over meerdere schema's waren verdeeld. Het probleem leende zich prima voor analyse met de Profiler.



Afbeelding 4: De bovenste twee events zijn verantwoordelijk voor 93,5 procent van de verwerkingstijd.

Het kostte enige tijd om (zonder Sparky!) de juiste tracefiles te laten verzamelen, en ook het overbrengen van een trace-file van 37 MB veroorzaakte wat problemen, maar daarna was het probleem in korte tijd gevonden. In afbeelding 4 wordt de kop van de tracefile van de lange run getoond. Direct is te zien dat de tijd wordt besteed aan de bovenste twee events, *PX Deq: Table Q Normal* en *CPU service*. Ze nemen samen 93,5 procent van de totale verwerkingstijd voor hun rekening. Tunen van iets anders dan deze twee events is daardoor meteen zinloos, een significante verbetering kan niet worden bereikt door het tunen van vele SQL statements die samen maar 6,5 procent van de verwerkingstijd veroorzaken. Zelfs als deze met een ongelooflijke inspanning twee keer zo snel worden is er nog maar 3,25% performance verbetering bereikt! Dit illustreert de kracht van de Profiler: met enkele muisklikken door het trace-resultaat wordt het statement onthuld dat verantwoordelijk is voor de hoge wachttijd in de twee eerste events, zoals afbeelding 5 laat zien.



Afbeelding 5: De update van de sequence blijkt de veroorzaker van het probleem

Sessiecontext

De vergelijking tussen de tracefile van de langzame productieomgeving en de snelle testomgeving leerde dat het aantal calls van het *recursive SQL statement* dat de sequence ophoogt in de productieomgeving bijna 20 keer hoger was dan in de testomgeving. Dat gold ook voor het aantal *PX Deq* events. Dit event heeft te maken met de taakwisseling tussen verschillende processen die ieder een deel van een parallelle query uitvoeren.
De conclusie op basis van deze informatie was dat de sequence in de testomgeving moest zijn aangemaakt met de opties NOORDER CACHE, terwijl de productieomgeving een sequence moest hebben met de opties ORDER NOCACHE. Door de laatste optie moeten de parallelle query-processen om de beurt een nieuwe waarde ophalen uit de sequence, die daarbij bovendien doordat er niet wordt ge-cached iedere keer opnieuw moet worden bijgewerkt. Een met de optie CACHE aangemaakte sequence hoeft twintig keer minder te worden bijgewerkt, waarbij de optie NOORDER ervoor zorgt dat er minder taakwisselingen nodig zijn tussen de processen voor de parallelle query. Zonder de database zelf te hebben gezien kon deze conclusie uit de informatie uit de tracefile worden getrokken. Nadat de conclusie aan het ontwikkelteam was doorgegeven bleek inderdaad dat de sequences van de productieomgeving en de testomgeving verschillend waren geconfigureerd, met de genoemde verschillen in de parameters.
Natuurlijk had dit met een analyse van de verschillen tussen de

schema's ook kunnen worden ontdekt. Men was ervan overtuigd dat de omgevingen identiek waren, op het aantal schema's waarover de tabellen verdeeld waren na. De Profiler maakte in enkele ogenblikken duidelijk wat in ettelijke dagen zoeken door het ontwikkelteam niet was gevonden.
Bij omgevingen die gebruik maken van Multi Threaded Server of Connection Pooling zal een manier gezocht moeten worden om de betrokken sessie te isoleren. Voor MTS is dat soms mogelijk door een aparte listener te configureren zonder MTS, en de te analyseren sessie via die listener te laten lopen. In andere situaties is niet direct een oplossing voorhanden. Dat wordt niet veroorzaakt door de Hotsos producten: de Oracle database voorziet nog niet in mogelijkheden een bepaalde sessie-context onafhankelijk van de feitelijk betrokken sessie te tracen.

Samenvatting

Rekening houdend met een paar beperkingen is de Hotsos Profiler een prachtig hulpmiddel voor de serieuze performance-analist. Ondersteund door de uitgebreide achtergrondinformatie over alle events, direct vanuit de browser te benaderen door op de (i) naast de events te klikken, levert de Profiler informatie die elders niet of met veel meer moeite bijeen is te sprokkelen.

Productgegevens

Product
Profiler 4.0.10, Sparky 2.20

Prijs
Op aanvraag, afhankelijk van aantal en serveromvang

Leverancier
Hotsos Enterprises, Ltd.
P. O. Box 93081, Southlake, Texas 76092-1081, USA
+1.817.424.3443

Positief
Professioneel gereedschap op basis van gedegen theoretische uitgangspunten
Ongeëvenaarde performance analyse
Gebruikersgemak bij het aanmaken van tracefiles

Negatief
Alleen voor Windows, geen Xwindows variant beschikbaar
Niet geschikt voor Connection Pooling / Multi Threaded Server omgevingen

Carel-Jan Engel (e-mail: cjpengel.dbalert@xs4all.nl)