André Wijnholds is consultant bij XCESS expertise center bv

Performance tuning van SQL Server met standaard tools

SOFTWAREMATIGE IDENTIFICATIE EN AANPAK VAN PERFORMANCEPROBLEMEN

Performance, snelheid... vraag Formule 1-coureur Jos Verstappen wat snel is en je krijgt een ander antwoord dan van zwemmer Pieter van den Hoogenband. Toch hebben ze het allebei over snelheid en behoorden ze ooit tot de snelsten in 'hun' wereld. Ook in 'onze' wereld zie we verschillen, maar dan door extreme ontwikkelingen. Nog maar een paar jaar geleden was je "de man" als je thuis beschikte over een 56K6 modem. En nu? Veelvoudige snelheden worden bereikt met ADSL alsof het de normaalste zaak van de wereld is. Snelheid? 't Is maar wie je het vraagt en wanneer je het vraagt. En hoe zit dat met performance bij SQL Server? Ook hier blijkt dat het voor ons ontwikkelaars niet snel genoeg kan zijn.

Het liefst besteden we uren om onze query te tunen, indexen te verleggen en veldtypes te wijzigen. En dat voor het fraaiste resultaat. Om de parallel met het zwemmen opnieuw te leggen: het liefst schitteren we door onze directe tegenstanders iedere keer te verslaan. Maar uiteindelijk gaat het erom dat we ons doel bereiken. Wanneer je bij het zwemmen het Olympisch Goud haalt is het minder interessant wanneer je op het EK tweede wordt. Uiteindelijk gaat het om je prestatie op een Olympisch toernooi. En bij het verbeteren van de performance van SQL Server? Wat is dan ons doel? En hoe bereiken we dat doel? Welke hulpmiddelen staan ons daarbij ter beschikking? Zijn die hulpmiddelen altijd softwarematig? In dit artikel blijft de configuratie van de hardwareomgeving en de ontwerpkeuzes buiten beschouwing.

Doel performanceverbetering

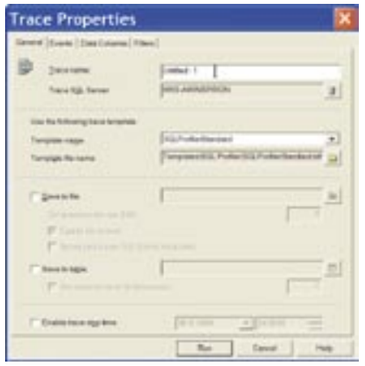
Het doel om de performance binnen SQL Server te verbeteren verschilt per situatie en per organisatie. Er is een preventieve reden: klachten voorkomen door aan normen te blijven voldoen. Performanceverbetering is ook vaak een gevolg van klachten doordat

zich onverwachte situaties voordoen. Performanceverbetering kun je op twee manieren aanpakken: hardwarematig of softwarematig. Welke je kiest is afhankelijk van drie factoren:
- de mogelijkheid van de organisatie tot nieuwe investeringen;
- de invloed van de organisatie op de infrastructuur;
- de tijdsduur waarmee de performanceverbetering voor de organisatie effectief moet zijn.

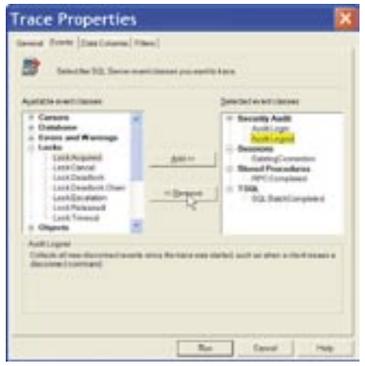
Als de organisatie geen investeringen in de hardwareomgeving kan doen, dan moet je een (dreigende) performance bottleneck via de software aanpakken. Daar staat tegenover dat als de organisatie de hardware wel kan uitbreiden, je serieus moet afwegen of een softwarematige aanpak prijstechnisch opweegt tegen een hardwarematige ingreep. Overigens is lang niet altijd de beschikbare financiële ruimte de reden waardoor de keuze valt op softwaretuning. De invloed van de beheersafdeling op de infrastructuur is in veel gevallen beperkt doordat bijvoorbeeld de organisatie het serverpark extern huurt. Wij hebben dit zelf meegemaakt bij webapplicaties die extern gehost werden. Hardwarematige ingrepen kunnen dan kostentechnisch gezien wel de beste optie zijn, maar zijn dan niet of lastig te realiseren. Daarbij komt dat nieuwe hardware zelden direct voor handen is. Als vandaag het systeem omvalt, dan is er geen tijd om een investeringsaanvraag te doen, te wachten op de levering en het moment waarop systeembeheer tijd inruimt om het systeem uit te breiden. Het verbeteren van de performance moet dan softwarematig gebeuren. Namelijk nu, vandaag!

Beschikbare hulpmiddelen

Alle overwegingen zijn geweest; hardwarematig de performance verbeteren is geen optie, dus komt softwarematige performance tuning in beeld. Welke hulpmiddelen staan ons ter beschikking? Waarmee kunnen we onze performance in kaart brengen én verbeteren? Als we kijken naar de beschikbare hulpmiddelen, dan kunnen wij die onderverdelen in twee groepen. Aan de ene kant hebben we hulpmiddelen die Microsoft meelevert met SQL Server. Aan de andere kant hebben we tools van derden, zoals FMS INC of Veritas. De tools van derden blijven in dit artikel buiten beschouwing omdat de tools die Microsoft zelf meelevert in veel gevallen toereikend zijn. Het is onze ervaring dat deze hulpmiddelen vaak onvoldoende benut worden. Veel ontwikkelaars zijn zich niet



Afbeelding 1. Trace Properties: configuratie



Afbeelding 2. Trace Properties: Wijziging events

Afbeelding 3. Aanleggen filters

bewust van de mogelijkheden die deze 'standaard' hulpmiddelen bieden, laat staan dat ze deze gebruiken. Daarom hier een toelichting op drie bruikbare gereedschappen.

Bij SQL Server krijgen we out-of-the box drie belangrijke hulpmiddelen:
- Query Analyzer;
- SQL Profiler;
- Index Tuning Wizard.

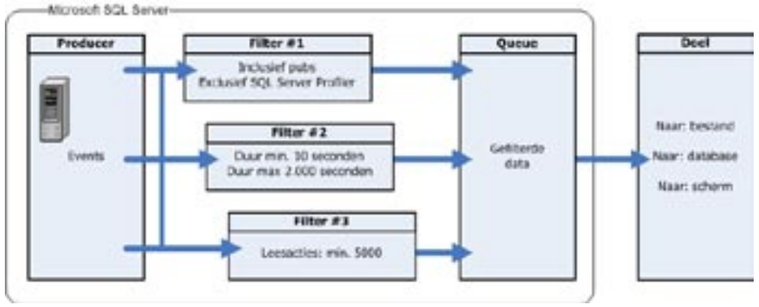
Samen met Performance Monitor, onderdeel van het OS, hebben we dan heel wat munitie om de performance van onze SQL Serveromgeving in korte tijd te verbeteren. Maar hoe passen we nou die hulpmiddelen toe? Laten we kijken naar een aantal voorbeelden uit de praktijk, ons afvragen welke hulpmiddelen inzetbaar zijn en wat daarvan de resultaten zijn.

Praktijkcase

De casus gaat als volgt. Gebruikers melden dat de nieuwe .NET-webapplicatie, waarbij SQL Server als backend fungeert, sinds maandag traag loopt. Pas na 9 seconden komen zoekresultaten op het scherm, terwijl dat voor het weekend nog in 3 seconden gebeurde. Nu is er in het weekend inderdaad wat gebeurd. Door een fusie van de organisatie is het aantal gebruikers uitgebreid van ongeveer 850 gebruikers naar 2500 gebruikers en is historische data aan de applicatie toegevoegd. Hierdoor is het aantal records explosief gestegen. We verhelpen deze vertraging in drie stappen:
- allereerst traceren we het pijnpunt, identificeren we de probleemquery;
- vervolgens onderzoeken we waardoor het probleem ontstaan is;
- ten slotte bedenken we een oplossing en voeren haar door.

Stap 1: Identificatie pijnpunt

Om na te gaan waar het probleem zich voordoet is SQL Profiler een uitstekend hulpmiddel en in mijn optiek nog steeds onvoldoende ontdekt. SQL Profiler is vanuit verschillende plekken op te starten. Je vindt de tool in ieder geval in het startmenu van Windows, onder menuoptie Microsoft SQL Server. Als je de tool opstart krijg je een leeg scherm te zien. Wat nu? Je moet een zogeheten trace definiëren. Met een trace geef je aan wat je wilt monitoren op SQL Server. Dat 'wat' kan erg uitgebreid zijn. Allerlei events die zich op SQL Server voordoen kun je meten, zoals het aanloggen van een gebruiker, het uitvoeren van een SQL statement en het ontstaan van errors. Om je meetgegevens te beperken kun je filters op de events plaatsen, zodat je alleen die data van de events meet waar je ook wat mee wilt doen. Doe je dat niet, dan kun je de performance van je systeem echt omver blazen. Want SQL Profiler kan dan een mooie tool zijn, het meten van alle acties op SQL Server heeft wel zijn prijs; het registreren van de events in je tracewindow of een tracetabel vergt namelijk resources van het systeem. Beperk je daar dus in, zeker in de productieomgeving. Ga naar het menu File | New | Trace en connect met de SQL Server waarop je events wilt monitoren. Je komt dan in een scherm waarin je de trace configureert; zie afbeelding 1. Ik doe dit voor de pubs database, zodat je de voorbeelden ook in je eigen omgeving kunt uitvoeren.
Op het eerste tabblad specificeer je de naam van de trace, de SQL Server waarop je events gaat monitoren. Je kunt zelf bepalen wat je gaat meten. Het is ook mogelijk gebruik te maken van één van de acht default templates. Mijn ervaring is dat je bij een eerste kennismaking met SQL Profiler met de template SQLProfiler-Standard goed uit de voeten kunt, omdat deze template veelgebruikte basis-events meet en bruikbare kolommen toont. Erg handig dus om mee te starten en later na believen uit te breiden. Je kiest je template in de uitklaplijst op dit scherm of start blanco met de gelijknamige optie uit de uitklaplijst. Vervolgens kun je de output van de trace op twee manieren opslaan: als bestand of als tabel. De eerste optie is handig als je de output van de trace op een andere pc wilt analyseren dan de server waarop je gemeten hebt. De tweede optie is aan te bevelen als dat niet nodig is, omdat je dan snel en eenvoudig op je tracer-resultaten kunt queriën vanuit Query Analyzer.
Bij een productiemeting is het essentieel om het juiste meetmoment te kiezen. Meten is weten, maar je moet wel het juiste meten! Zeker bij performance tuning van SQL Server is dit doorslaggevend. Indexen zijn bijvoorbeeld afhankelijk van het gebruik. Zijn er vooral schrijfacties of wordt de data uit de tabel met name gelezen? De performanceverbetering die je in beide gevallen doorvoert is volledig tegengesteld. In plaats van een korte meetperiode kun je overwegen om de trace een week lang te laten lopen. Echter, dan heb je zeker een performanceprobleem! De meting van de events eist ook haar tol. Je moet dus je meetpriode zoveel mogelijk beperken. Daarom is het handig om gebruik te maken van de optie 'Enable tracestop time'. Daarin kun je aangeven wanneer de trace eindigt, zodat je het systeem niet langer belast. Het tweede tabblad - zie afbeelding 2 - biedt de mogelijkheid om de events van de template te wijzigen door nieuwe events toe te voegen of bestaande events te verwijderen. Om ons probleem te identificeren hebben we genoeg aan de stored procedures en T-SQL statements die de default template selecteert. De events laten we dus ongemoeid.
Op het derde tabblad kunnen we datakolommen toevoegen of verwijderen, waarmee we dieper inzicht krijgen in de events. Bijvoorbeeld welk SQL Statement is uitgevoerd in de datakolom Text, of hoeveel milliseconden het lezen van de data heeft geduurd in de kolom Reads. Ook hier is de default trace toereikend. Het laatste tabblad biedt ons de mogelijkheid om filters aan te leggen; afbeelding 3. Dat heeft twee voordelen: we vermijden daarmee onnodige serverbelasting en we voorkomen dat we allerlei data moeten doorworstelen waar we toch niets mee doen. Als ik de performance van de pubs database wil verbeteren, hoef ik niet te weten wat er op de andere databases gebeurt. Dat leidt



Afbeelding 4. Proces van filtering

alleen maar af. Zo kunnen we ook de gebruikers beperken of alleen de trage queries tonen. In afbeelding 4 is schematisch weergegeven hoe het proces van filtering in haar werk gaat. De server produceert events, die vervolgens het filter passeren en leiden tot een output. Deze output gaat op basis van de keuzes uit het eerste scherm van naar bestand, tabel en/of scherm.

Als dan weloverwogen de trace is geconfigureerd, klikken we op Run om de meting te starten. Op dat moment krijgen we in een overzicht van de door ons vastgestelde kolommen te zien wat er allemaal op SQL Server gebeurt. In afbeelding 5 zie je hiervan een voorbeeld, waarbij in de pubs database een select statement is uitgevoerd. Door een regel te selecteren zie je onderin je scherm detailinformatie uit de kolom Textdata, in dit geval het Select statement. Doordat ik meerder statements zie, kan ik ook in één oogopslag zien welk statement de meeste tijd in beslag neemt.

Dat is te zien aan de kolom Duration. In het voorbeeld van afbeelding 4 duurt de geselecteerde rij het langst. Overigens kom je er ongetwijfeld snel achter waarom het zinvol is om de data naar een tabel weg te schrijven. In een productieomgeving heb je in enkele seconden duizenden regels in de tracewindow. Die ene oogopslag is er dan niet meer bij. Het voordeel van de tabel is dat je snel de top 10 van de langzaamste queries kunt tonen, door vanuit Query Analyzer op de tabel te queriën.

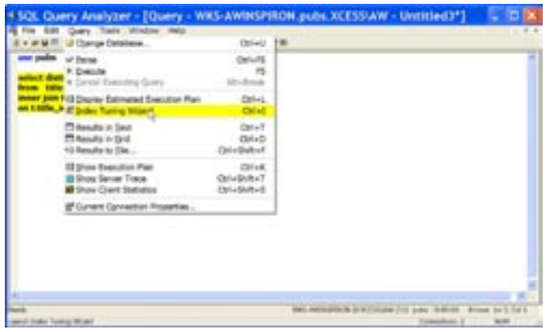
Stap 2: Oorzaak probleem

Zoals je ziet is het identificeren van het pijnpunt niet moeilijk. SQL Profiler openen, trace definiëren en resultaat beschouwen uit de kolom Duration. In deze voorbeeldsituatie is het natuurlijk altijd eenvoudig, maar ook de praktijk wijkt niet veel van dit voorbeeld af. De data is dan ongetwijfeld omvangrijker, het kan even puzzelen zijn om de juiste events te monitoren en de juiste filters te leggen, maar dan heb je de uitdagingen wel gehad.

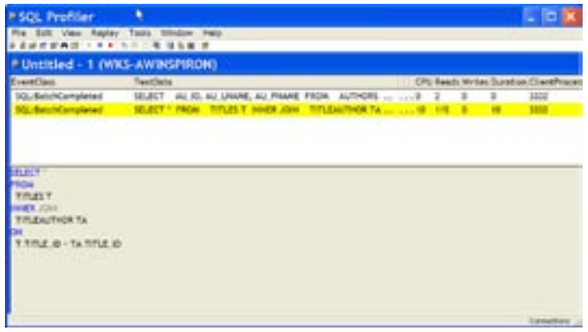
De tweede stap is complexer; het achterhalen van de oorzaak van het probleem. Daarvoor zijn twee manieren: of je maakt gebruik van de tools, of je gebruikt je eigen kennis om het probleem te doorgronden. Om met het eerste te beginnen wijs ik je op het nut van de Index Tuning Wizard. Veel problemen komen namelijk voort uit het verkeerd definiëren van indexen; teveel indexen, onlogische indexen, te weinig indexen of verkeerd samengestelde indexen. Op dit vlak kun je heel wat fouten maken. Fouten bijvoorbeeld die te vermijden zijn door gebruik te maken van de Index Tuning Wizard. Laten we eens in ons concrete geval kijken naar hoe we deze tool inzetten en wat daarvan het resultaat is. Wij starten de Index Tuning Wizard op vanuit Query Analyzer; zie afbeelding 6. Dat heeft namelijk als voordeel dat we alleen voor ons query statement een indexadvies krijgen als we ons query statement in het scherm vooraf selecteren.

Index Tuning Wizard toont vervolgens een aantal schermen. In het eerste scherm (afbeelding 7) geven we aan of we bestaande indexen willen behouden, of indexed views wenselijk zijn (let op: alleen te gebruiken door de Enterprise Edition van SQL Server) en hoe diep de screening van de Index Tuning Wizard moet gaan. Let erop dat je nooit de tuning afbreekt, want dan heb je daarna zeker een performanceprobleem! De Wizard maakt namelijk testindexen aan als je de Index Tuning Wizard uitvoert. Op het moment dat die test wordt afgebroken, blijf je met een hoop extra, onnodige indexen achter. Het tweede scherm (afbeelding 8) biedt ons de mogelijkheid om de bron voor de tuning aan te geven: een workload file of een trace tabel, bijvoorbeeld als je een trace hebt uitgevoerd op een productiemachine en de trace-resultaten op een andere machine analyseert, maar ook een selectie in het scherm van Query Analyzer. Met de advanced options van dit scherm kun je vervolgens de Index Tuning Wizard zelf verder tunen, maar dat laten we nu voor wat het is. Het derde scherm (afbeelding 9) is van groot belang voor de tijdsduur van de tuning. Weet je van tevoren al welke tabellen een probleem geven? Beperk de tuning dan tot deze tabellen. Dat scheelt veel analysewerk door de server. Als je een sterke toename van gegevens verwacht in een aantal tabellen, geeft dat dan aan in de kolom Projected Rows. Index Tuning Wizard houdt dan rekening met die toekomstige tabelomvang in haar advies (zie afbeelding 10).

Ten slotte ben je nog de bekende ‘Next, Next, Finish’ verwijderd van het einde van de Wizard. Overigens gaat de analyse na dit scherm al starten. Dat is verraderlijk, omdat je dat onderdeel dus beter niet kunt afbreken. Na de analyse krijg je een advies van de Index Tuning Wizard over welke (clustered) indexen hij voorstelt en welke overbodig zijn. Mijn eigen ervaring is dat je hier niet tegenop kunt redeneren. Het advies van de Wizard is grondig en gedegen en levert mij geregeld performanceverbeteringen op van meer dan 200%. Maar niet voor eens en altijd! Als er data in de database verandert, toeneemt of afneemt, dan moet je geregeld de Wizard opnieuw uitvoeren. Dat komt doordat de indexstructuur



Afbeelding 6. Index Tuning Wizard vanuit Query Analyzer



Afbeelding 5. Voorbee ld van gebeurtenissen SQL Server

Afbeelding 7. Indexen

Afbeelding 8. Aangeven bron tuning

van twee variabelen afhankelijk is: de hoeveelheid gegevens en de manier waarop met die gegevens wordt omgegaan. Vanwege dit laatste aspect is het dan ook essentieel om de Index Tuning Wizard van een representatieve set brongegevens te voorzien! Doe je dat niet, dan los je wel je huidige probleem op, maar creeer je met dezelfde snelheid (en eenvoud) een nieuw probleem op een andere plek!

De wizard in mijn testomgeving leidt niet tot aanbevelingen. De oorzaak daarvan is voor de handliggend:

- de tabellen in de database Pubs zijn standaard van zinvolle indexen voorzien.
- de query die als bron dient voor de wizard is niet afwijkend van het oorspronkelijk verwachtte gedrag.
- de data in de tabel is ten opzichte van het origineel niet gewijzigd.

In je eigen productieomgeving kun je er zeker van zijn dat je hier gaat scoren, omdat je niet aan deze randvoorwaarden voldoet.

Neem alleen al het verschil tussen Clustered Indexen en Non-Clusterd Indexen. Hoe vaak ligt een Clustered Index op de primairy key terwijl er ontwerptechnisch gezien veel betere opties zijn? Ten slotte zijn Clustered Indexen bij uitstek geschikt als bij een veld op ranges wordt gezocht en zijn ze voor kleine tabellen overbodig.

Stap 3: Oplossing doorvoeren

De Index Tuning Wizard biedt ons het doorvoeren van de verbetering op een presenteerblaadje aan. Als de wizard verbeteringen adviseert krijgen we de mogelijkheid om een script op te slaan en op een later tijdstip uit te voeren. Doe dat en stap niet in de valkuil om direct de verbeteringen in je productieomgeving los te laten.

Ik heb in projecten meegemaakt dat het heroganiseren van een index meer dan een uur met zich meebrengt. Weliswaar praten we dan over miljoenen records, maar toch. Een clustered index op een ander veld brengt bijvoorbeeld met zich mee dat de tabel fysiek geheel anders wordt opgebouwd. En dan is het niet fijn als op dat moment een productieapplicatie data in de tabel gaat benaderen.

De Index Tuning Wizard is dus een zeer gedegen hulpmiddel om snel resultaat te bereiken bij de strijd tegen performanceproblemen. Zeker in combinatie met SQL Profiler staan er fraaie hulpmiddelen tot iemands beschikking die niets extra’s kosten. Naast deze twee hulpmiddelen zit in SQL Query Analyzer ook nog een derde hulpmiddel ‘verborgen’, het Execution Plan.

Execution Plan

Waarom nog een hulpmiddel, als problemen door de Index Tuning Wizard prima worden aangepakt? Dan is dat toch niet nodig? Helaas wel. Niet aan alle problemen ligt een verkeerde indexstructuur ten grondslag. Soms is de indexstructuur prima, maar blijft de query slecht presteren. Dan bestaat er de optie om de query in stukjes te knippen, anders op te bouwen, nog eens kritisch te beschouwen. En dat is een beproefde methode. Maar doe dat niet zonder gebruik te maken van het Execution Plan, dat Query Analyzer je kan tonen. Dat Execution Plan zit boordevol informatie om je query te optimaliseren. Waar vinden we het Execution Plan?

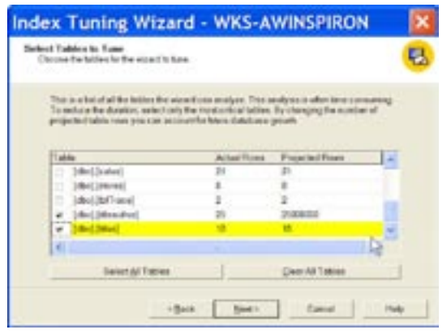
Als we Query Analyzer geopend hebben activeren we met CTRL-K het Execution Plan. We kunnen ook het zogeheten Estimated Execution Plan bekijken. De query wordt dan niet uitgevoerd, alleen geanalyseerd. Het Estimated Execution Plan krijg je te zien met CTRL-L. Beide opties vind je trouwens ook terug in het menu van Query Analyzer. Als je dan een query uitvoert, dan krijg je – na activering van het Execution Plan – een extra tabblad te zien in je outputwindow. In dat tabblad is grafisch weergegeven hoe de query binnen de server engine wordt uitgevoerd. Op het eerste oog lijken die grafische symbolen vooral leuk in plaats van zinvol. De grafische weergave heeft echter zeker haar nut. Ze stellen je in staat om snel de uitvoer van de query te interpreteren en te zien wat waar gebeurt. Kijk eens in de Books Online voor een uitleg van de symbolen; zie afbeelding 11.

Naast inzicht in het queryplan biedt het Execution Plan-scherm nog tal van andere features. In afbeelding 12 zie je bijvoorbeeld dat je met rechtermuisklik op een symbool allerlei menuopties tot je beschikking krijgt. Dit stelt je in staat om direct Indexen aan te maken (als je van mening bent dat de Index Tuning Wizard toch niet goed zijn werk heeft gedaan), of Statistieken bij te werken.

Wil je effectief te werk gaan, kijk dan naar het label ‘Cost’ in dit scherm. Daarin zie je welke actie het meeste tijd in beslag neemt. Dit is procentueel weergegeven ten opzichte van de totale query. Bij complexe queries kun je daardoor direct je focus leggen op de grootste bottleneck. Een fraaie eigenschap is ook de tooltip die je te zien krijgt als je over een symbool gaat met je muis. Dit is een bron van informatie over welke acties de Engine heeft gedaan.

Drie gulden regels

We hebben gezien dat performanceverbetering soms softwarematig moet vanwege tijd of budget. Ook hebben we gezien dat Microsoft een aantal hulpmiddelen levert waarmee we performanceproblemen kunnen identificeren (SQL Profiler) en oplossen (Index Tuning Wizard/Query Analyzer met Execution Plan). Vooral dit laatste vergt ervaring en inzicht. Daarom nog een drietal gulden regels, bij het analyseren van de query.



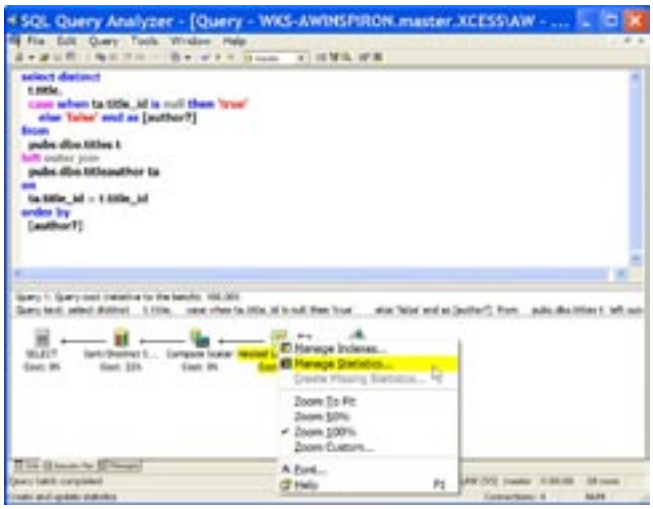
Afbeelding 9. Omvang testdata

Afbeelding 10. Index aanbevelingen

Afbeelding 11. Books Online

1. Zorg dat tabellen altijd over unieke sleutelvelden en relaties beschikken. Dat zorgt ervoor dat je optimaal gebruik maakt van de database engine van SQL Server.
2. Vermijd het gebruik van Text- en Ntext-velden. Zeker bij migraties vanuit MS Access komt dit type veld veel voor, doordat de Upsizing Wizard van MS Access standaard memo-velden omzet naar Text-velden. Bij de vele migraties vanuit MS Access naar SQL Server zijn wij op dit punt altijd zeer scherp. Maar waarom moet je dit vermijden? SQL Server neemt voor een Text-veld een verwijzing op in de rij. SQL Server slaat de gegevens dus niet in dezelfde rij op als de overige gegevens. Dat zorgt ervoor dat de Engine eerst de rij met de data benadert en daarna op een andere plek de daadwerkelijke gegevens. Dat zijn dus twee leesacties (of schrijfacties). Bij veel records verslechtert de performance merkbaar. Test dit maar eens door een veld in een tabel te hernoemen van Varchar naar Text en vervolgens te kijken hoe lang de query erover doet om de gegevens op te halen. Bij een tabel van meer dan 2000 records is het resultaat al merkbaar.
3. Beperk het aantal kolommen dat je in queries ophaalt. De reden daarvan is tweeërlei. Als je altijd alles (Select * From…) ophaalt, kun je problemen krijgen als later de tabel extra velden krijgt (bijvoorbeeld Text-velden). Problemen ontstaan ook als de data in de tabel toenemen. De query kan dan in eerste instantie goed functioneren, maar later eisen die extra 14 kolommen toch hun tol. En dan weet je niet zomaar welke kolommen weg kunnen. Mijn advies: denk daar bij voorbaat kritisch over na en val niet in de valkuil dat zoiets later wel komt…

Ik ben niet ingegaan op het onderhoudsplan dat je binnen SQL Server kunt inrichten. Zelf kom ik regelmatig bij organisaties die wel een SQL Server-omgeving in productie hebben, maar die hier geen gebruik van maken. Het onderhoudsplan kun je instellen vanuit Enterprise Manager (Tools | Database Maintenance Planner) en stelt je onder meer in staat om wekelijks (periodiek) de indexen te herorganiseren.



Afbeelding 12. Menuopties onder de rechtermuisknop

Ten slotte

SQL Server bestaat nu 11 jaar. De opvolger, SQL Server 2005 (Yukon), belooft weer veel nieuwe functionaliteiten. Tot die tijd moeten we onze systemen blijven onderhouden. Dat dit niet een eenmalige actie is moge duidelijk zijn. Dat we daarvoor al een aantal jaar hulpmiddelen 'out-of-the-box' tot onze beschikking hebben, is ook duidelijk. Daarbij ben ik me bewust dat optimaliseren complex kan zijn. Zo hanteren wij voor een performance-audit een checklist met meer dan 70 aandachtspunten op het terrein van configuratie, tabellen, queries, indexen, replicatie-inrichting en frontend. Daarom heb ik me nu beperkt tot de beschreven hulpmiddelen. Veel snelheid toegewenst!

Referenties

http://www.sql-server-performance.com

http://www.sqlteam.com

http://msdn.microsoft.com/SQL/sqlperf

SQL Server 2000 Optimization Guide / Jenny Lynne Fields

André Wijnholds is consultant bij XCESS expertise center bv uit Leusden (www.xcess.nl). De afgelopen jaren heeft hij zich toegelegd op het vertalen van klantwensen naar technisch haalbare oplossingen. Zijn technische ervaring spitst zich toe op .NET en SQL Server. André Wijnholds is bereikbaar via aw@xcess.nl