Kostenefficiënt presteren met een uitdijende data-moloch

De Polis Papers (6): Oei, Ik Groei ...

René Veldwijk en Maarten van Nijnanten

Als databases echt groot worden dan verandert gegevensmanagement fundamenteel van karakter. In dit artikel schetsen we wat er bij de Polisadministratie echt anders is dan normaal en hoe je een enorme database die massaal wordt geraadpleegd en gemuteerd in de lucht krijgt en houdt. Meer dan gemiddeld is dat bij de Polisadministratie niet alleen een gevecht met Terabytes, maar ook met Mega-euro's; een gevecht dat pas net is begonnen.

In het eerste artikel in deze serie hebben we geschetst dat de Polisadministratie (PA) in alle opzichten omvangrijk is: veel gegevens, veel mutaties en veel raadplegingen. De kerntabel van de PA, de tabel met inkomstenopgaven, groeit maandelijks met ruim 20 miljoen records en gaat eind dit jaar door de miljard records heen. Het aantal mutaties en de groei van de database in absolute termen zijn beide ruwweg constant en geen knelpunt. Simpel gezegd geldt dat voor elke baan die verdwijnt er een andere baan of een uitkering terugkomt. Voor de PA maakt dat niets uit. De PA is per werkdag gemiddeld een uur of zes bezig met het verwerken van nieuwe loonaangiften die via de Belastingdienst worden aangeboden. De werkelijke aantallen dagelijks te verwerken loonaangiften verschillen enorm, maar ook dat is nog geen probleem want het proces van gegevensverwerking is niet extreem tijdkritisch. Daarbij komt ook dat de meeste tijd die verstrijkt tussen de loonaangifte door de Werkgever en opname in de PA zich afspeelt bij de Belastingdienst. Een dagje langer wachten met het verwerken van loonaangiften in de PA is, kortom, geen ramp.

Als er loonaangiften worden verwerkt dan kunnen er geen bulkraadplegingen op de PA plaatsvinden. De *read consistency* die het Oracle RDBMS garandeert leidt dan onherroepelijk tot een te groot beroep op de *rollback* segmenten waarna de query klapt. Ook dit is geen probleem: bulk-laadprocessen en leverprocessen worden ouderwets ingepland en sequentieel uitgevoerd en dan blijft er doorgaans nog een flink aantal uren over waarin de PA database alleen bezig is met het afhandelen van de atomaire *requests* die 7x24 binnenkomen en direct worden afgehandeld. Het gaat daarbij overwegend om webservice aanroepen met betrekking tot dienstverbanden en inkomensgegevens van één persoon. Geen vuiltje aan de lucht, zo lijkt het. Helaas, niets is minder waar.

Exit Index, Welkom Full Table Scan!

Efficient werken met een relationeel DBMS komt voor een groot deel neer op het programmeren van *joins* en *subselects*: hoe meer er in een query wordt gestopt, des te efficiënter kan de RDBMS optimizer de gegevens ophalen, zeker als de indexen netjes zijn gelegd. Dat zien we ook bij de PA: de belangrijkste webservice haalt via één SQL statement voor één persoon diens hele hebben en houden uit de database: persoonsgegevens, inkomstenverhoudingen, werkgevers en alle maandelijkse opgaven sinds 2006. Op drukke momenten gaat het om meerdere opvragingen per seconde en de database doet dit met twee vingers in de neus, ook als er tegelijk zware mutatie- of raadpleegprocessen lopen. Omgekeerd zijn er enkele webservices die de PA database in ping-pong modus aanroepen: “1: geef persoonsgegevens persoon X”, “2: geef diens inkomstenverhoudingen”, “3: geef voor inkomstenverhouding Y de bijbehorende werkgevergegevens”, “4: geef voor inkomstenverhouding Y de inkomstenopgaven”. Bij een uitzendkracht die sinds 2006 twintig baantjes heeft gehad heb je dan 61 database aanroepen. Het gevolg is een relatief beroerde performance maar, wat erger is, een veel grotere belasting van de PA database. Gelukkig zijn de webservices die op deze manier werken kwantitatief nog onbelangrijk, maar zodra dat dreigt te veranderen zullen ze moeten worden herschreven. Dat deze ping-pong webservices *überhaupt* bestaan is een erfenis van de het oude PA systeem: bij de vervanging door het PA systeem dat in deze serie wordt beschreven zijn deze reeds bestaande webservices één-op-één overgezet. Het is vervelend maar niet problematisch, temeer daar deze webservices *sowieso* moeten worden aangepast om te kunnen omgaan met het ‘corruptie-op-voorraad’ principe waarop de PA is gebaseerd (zie DBM 2009/2 of www.dbm.nl). Er is hier dus wel een vlekje dat moet worden weggewerkt, maar mits tijdig aangepakt is er geen probleem.

Waar wel een enorm probleem zit is in de bulkleverprocessen, zoals een maandelijks overzicht voor een middelgroot pensioenfonds, een wekelijks overzicht van alle mutaties voor het Centraal Bureau voor de Statistiek (CBS) of de jaarlijkse levering voor de vooringevulde belastingaangifte die dit jaar is ingevoerd.

Dergelijke leveringen raken niet tientallen of honderden records maar miljoenen tot honderden miljoenen. De vraag die dan opkomt is wat dan nog de beste strategie is voor de RDBMS optimizer. Ergens is er een omslagpunt waarbij het niet meer handig is om gegevens op te halen via indexen, maar waarbij het handiger is om *brute force* door de tabeldata heen te lopen. Bij een oudere 'kleine' database ligt dat omslagpunt vaak ergens in de orde grootte van 10 procent van de data en in dat geval zou ook bij de meeste bulkleveringen gebruik worden gemaakt van indexen. Bij een database met de omvang van de PA en bij de systeemconfiguratie waarmee wij werken blijkt dat omslagpunt echter in de buurt van de 1,5 procent van de data te liggen. Anders gezegd, zelfs bij een 'kleine' bulklevering besluit de Oracle optimizer al om de meeste indexen te negeren en in plaats daarvan de tabellen met een *full table scan* te doorlopen en de tussenresultaten door middel van *hash joins* tot een eindproduct te verwerken. En de optimizer doet zijn werk goed: als we Oracle instrueren om de indexen te gebruiken dan zien we slechtere responstijden. Het is ook wel logisch: bij een database met de omvang van de PA is het intern geheugen van 32 Gigabyte, waarvan 8 Gigabyte aan database cachegeheugen, veel te klein om veel te verwachten van *caching* en daarmee ligt de drempel voor een RDBMS om af te zien van het 'prikken' in indexen en tabellen veel lager. Daarbij hebben we de grens zelf ook verder verschoven door gegevens op te halen in I/O blokken van 1 Megabyte waardoor *table scans* sneller worden. Hoe dan ook zien we ons bij de PA geconfronteerd met een probleem: voor eenvoudige raadplegingen hebben we als vanouds indexen nodig, maar voor bulkleveringen hadden we haast net zo goed een implementatie zonder indexen kunnen hebben. Hooguit worden indexen soms gebruikt als surrogaat tabellen voor *index scans*. En anders dan met indexgedreven query's nemen de doorlooptijden van *full table scan* query' op een gestaag groeiende database steeds toe. Er moet dus een list worden verzonnen.

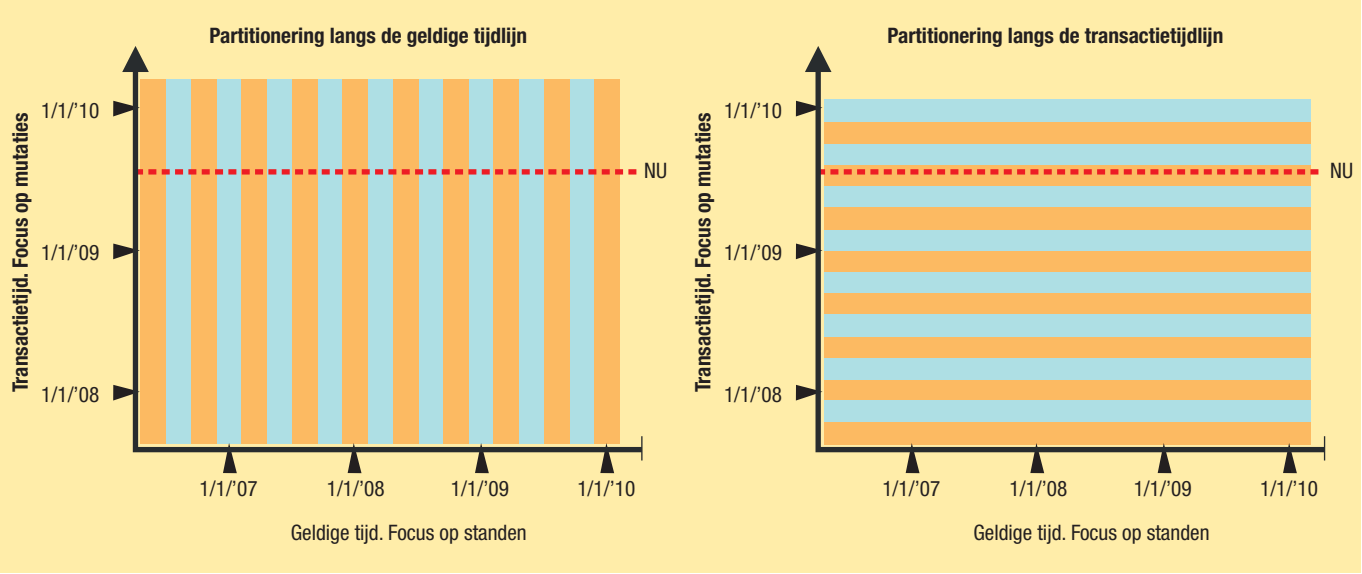
Waarschuwing: database optimalisaties zijn dodelijk!

Als het nutteloos worden van indexen voor grote query's op extreem grote databases de normale situatie is – en wij menen dat – dan heeft dit grote gevolgen voor het ontwerp van dergelijke databases. Juist bij Tera-databases bestaat de neiging om een gegevensbankontwerp te optimaliseren naar de belangrijkste of meest kritische functionaliteit. Dat nu blijkt levensgevaarlijk. 'Optimaliseren' van een database komt immers meestal neer op het toevoegen van redundantie aan een gegevensbank en meer redundantie betekent meer data en meer langdurige *table scans*. Dat dit niet zomaar een mening is zien we ook om ons heen. Zoals aangegeven in het inleidende artikel in deze serie hebben we in 2007 vóór de realisatie van de huidige PA eerst een noodvoorziening gebouwd in de vorm van een database met één-op-één naar Oracle tabellen omgezette XML-structuren. Deze XML-database was door alle redundantie circa zeven keer groter dan de huidige PA en de performance van bulkleveringen was significant slechter dan die van de huidige PA, waarin natuurlijk veel meer loonaangiften zitten. Helaas is het ding medio 2008 uitgeschakeld en zijn we niet meer in de gelegenheid om vergelijkende metingen te doen. Nog extremer was de situatie met de oorspronkelijke PA. Mogelijk vanuit overwegingen van flexibiliteit had men daar besloten om de tabel met inkomstenopgaven te 'imploderen'. Waar de huidige PA per maandelijkse opgave één 'plat en lang' record opslaat met ruim 30 inkomensgegevens, sloeg de oude PA per opgave ruim dertig korte records met een lange sleutel op. Wij gaan zoals gezegd binnenkort door de één miljard inkomstenopgave records, maar de oude PA zat anderhalf jaar geleden al op bijna *tien* miljard records en in totaal meer dan 10 Terabyte. Ook hier hebben we nooit een vergelijkende meting kunnen doen, maar het zal niet verbazen dat er uit deze database nooit inkomensgegevens zijn geleverd. Van de doden niets dan goeds, maar héél voorzichtig vermoeden we op basis van eerdere ervaringen dat een bulklevering op een geïmplodeerde database tussen de 20 en 50 keer langzamer loopt: een query voor een extreem zware bulklevering die bij 'onze' PA één dag loopt zou dan bij de oude PA weken doorlooptijd vergen. Alleen al omdat je in die periode niet gegevens in bulk kunt muteren (*read consistency*!) ben je dan dood.

Nog wel in leven is een derde database met dezelfde informatieinhoud als de PA. Het betreft hier een datawarehouse dat de Belastingdienst gebruikt om brieven met foutmeldingen te genereren ten behoeve van werkgevers en salarispakketleveranciers. Hier heeft men gekozen voor een moderne vorm van warehousemodellering, met als gevolg een database die circa vijf keer groter is dan de PA. Omdat er slechts sprake is van één soort bulklevering met een relatief lage frequentie is er hier geen probleem, maar of een dergelijk ontwerp levensvatbaar is voor een basisregistratie die honderden (soorten) afnemers bedient is de vraag. Hoe dan ook zou het interessant zijn als hier eens een keer naar zou worden gekeken. Een interessante vraag is bijvoorbeeld of het veel grotere geheugenbeslag inherent is aan warehousemodellering. Het belang voor het databasevakgebied van zo'n vergelijking is groot. Ideeën over database optimalisaties zijn wijd verbreid en de huidige situatie waarbij er kan worden vergeleken tussen terabyte databases met nagenoeg precies dezelfde informatie inhoud is hoogst zeldzaam.

Partitionering als verlossertechnologie?

Terug naar onze netjes genormaliseerde, platte en daardoor relatief kleine PA. Hoe lossen we het probleem op dat we de ruimte in onze leverkalender langzaam zien verdwijnen doordat bulkleveringen elke maand langer lopen en het aantal bulkleveringen toeneemt? De oplossing ligt grotendeels in een DBMS faciliteit die dateert van ruim na de database index: horizontale partitionering. Moderne databases als Oracle ondersteunen de mogelijkheid om grote tabellen fysiek in afzonderlijke brokken (partities) op te slaan via een eenvoudig expressiemechanisme. Bij de PA kunnen we de tabel met inkomstenopgaven bijvoorbeeld verdelen in partities per aangiftetijdvak. Omdat de Oracle



Afbeelding 1: Verschillende partitioneringsbehoeften.

optimizer deze partitioneringsregels gebruikt bij de bepaling van de toegangstrategie, worden nu die geheugenbrokken overgeslagen waarin gegarandeerd geen relevante gegevens te vinden zijn. Een bulklevering van gegevens over juni 2009 benadert bij een partitionering per tijdvak gegarandeerd alleen die partities die gegevens over juni 2009 bevatten. En daarmee lijken we gered: zelfs als de PA straks 25 jaar aan loonopgaven bevat blijft de performance van bulkleveringen ruwweg gelijk. Het is zelfs nog beter: omdat we nu al een database hebben met 3,5 jaar aan ongepartitioneerde data zal de performance van bulkleveringen na invoering van databasepartitionering initieel flink verbeteren. Ruimte genoeg voor nieuwe afnemers, zo lijkt het.

Helaas is de wereld complexer en is één enkele manier van partitionering waarschijnlijk niet genoeg om alle bulkleveringen te versnellen en te stabiliseren. De meeste afnemers van bulkgegevens zijn geïnteresseerd in *standen*. Zij krijgen uit de PA gegevens gebaseerd op tijdvakken, oftwel de geldige tijdlijn van artikel 1 in deze serie (zie www.dbm.nl). Maar er is ook een afnemer, het hierboven genoemde CBS, die juist *mutaties* wil zien: de transactietijdlijn van artikel 1. Afbeelding 1 geeft weer hoezeer de eisen dwars op elkaar staan.

Als we de PA straks per tijdvak (geldige tijd, zie artikel 1) hebben gepartitioneerd dan blijft de belangrijke bulklevering aan het CBS dus gestaag trager worden. En we kunnen ons er niet van afmaken met de uitvlucht dat de CBS-levering uitzonderlijk is. Het is precies omgekeerd: CBS is de enige bulkafnemer van PA-gegevens die gegarandeerd alle relevante gegevens uit de PA krijgt. De afnemers van standgegevens zijn haast per definitie incompleet: aangiften over 2006 en 2007 die nu nog binnenkomen, komen via de reguliere bulkleveringen alleen bij CBS terecht. Natuurlijk kan dat niet zo blijven en is het op termijn haast onvermijdelijk dat alle bulkafnemers overgaan naar mutatieleveringen vanuit de PA. Helaas gaat dat vermoedelijk nog vele jaren duren en ergens in die periode krijgt de PA mogelijk te maken met een vollopende leveragenda door conflicterende eisen van afnemers. De klok tikt.

De economische insteek

Er is nog iets anders met de PA: er is op dit moment nog niet voorzien in uitwijk voor het geval er iets ernstig misgaat met de database server. In het geval van een ernstige storing zou er een *restore* moeten worden gedaan en zou de PA een paar dagen tot een week uit de lucht zijn. Vorig jaar zou dat heel vervelend zijn, volgend jaar ernstig en over een paar jaar rampzalig. Dus wordt de PA nog dit jaar voorzien van een *failover* voorziening waarin alles dubbel is uitgevoerd. Voor de 99,99 procent van de tijd dat alles werkt doet dat ding helemaal niets, behalve dan heel veel geld kosten. Die 0,01 procent is namelijk nagenoeg even duur als die andere 99,99 procent. Dat is heel erg als je je bedenkt dat voor de PA de rekencentrumkosten in dezelfde orde van grootte liggen als de directe kosten van exploitatie en ontwikkeling. Zelfs rekening houdend met de lage ontwikkelkosten van de PA is deze bijzondere kostenstructuur, die herinnert aan de jaren zeventig van de vorige eeuw, een aandachtspunt en natuurlijk willen we voorkomen dat de *failover* voorziening het beeld nog schever maakt.

Combineer de conflicterende eisen van bulkleveringen met de verwachte toename in aantallen en soorten (bulk)leveringen en met de onvermijdelijke overgang naar een kostbare *failover* voorziening en de oplossing dringt zich onontkoombaar op: gebruik die tweede machine ook in normale omstandigheden en partitioneer daarbij de primaire server langs de geldige tijdlijn en de secundaire langs de transactietijdlijn. Gaat alles goed dan worden beide servers gebruikt en is er voldoende capaciteit voor nog heel wat groei in de omvang van de database en de belasting door (bulk)leveringen. En begeeft de ene server het dan neemt de andere de urgente leveringen over. Zo managen we simultaan een technisch en een economisch probleem.

Piano ... Pianíssimo ...

Op dit punt aangeland is het passend om op drie zaken te wijzen. Allereerst gaat veel van het voorgaande over zaken waarover wordt nagedacht en die worden uitgetest, maar die nog niet zijn gerealiseerd. De DBA is bij de PA een belangrijk man en niet voor niets medeauteur van dit artikel, maar de reëel bestaande PA is *as we speak* een simpele, volgens klassieke principes ontworpen en geïmplementeerde Oracle 9 (sic) database zonder veel meer slimmigheden dan indexen op de sleutels en goed ingeregelde database parameterinstellingen. Ten tweede veronderstellen we van alles over toekomstige kwantitatieve en kwalitatieve ontwikkelingen, maar bestaat de kans natuurlijk dat we ons vergissen in de richting of de snelheid daarvan. We hebben beslist te maken met enkele verontrustende trends maar niet met acute problemen. En tenslotte richten we onze inventiviteit eerst op verbeteringen die transparant zijn voor de programmeur en de applicatiesoftware, vóórdat we slimmigheden gaan doorvoeren in de applicatielogica of in de databasestructuur.

Deze drie zaken hangen met elkaar samen. Omdat we verbeteringen zoveel mogelijk richten op de wereld onder de Oracle-motorkap, kunnen we deze snel en relatief risicoloos doorvoeren en hoeven we pas in te grijpen wanneer de situatie nijpend wordt. Wat elders vaak strategische beslissingen zijn is bij de PA eerder tactisch van aard, met dank aan relationele databasetechnologie en onze bewust ouderwetse ontwikkelstijl. Met alle ruimte in de verwerkingskalender en de praktijk bij de gemiddelde afnemer in gedachten, gaan we ons voorlopig alleen richten op partitionering langs de geldige tijdlijn en daarmee een forse performancebonus scoren. Dat gebeurt min of meer tegelijk met de realisatie van de onvermijdelijke *failover* voorziening die we voor de 99,99 procent van de tijd dat alles goed gaat gaan benutten voor vooral ad hoc gegevensleveringen, zodat we zowel meer zekerheid als extra ruimte voor groei creëren en de throughput van de reguliere gegevensleveringen kunnen garanderen. Als de soep niet zo heet wordt gegeten als die nu wordt opgediend dan kunnen we mogelijk nog jaren verder. Wie weet kunnen we dan elke bulk-afnemer of de architecten in overheidsland overtuigen van de superioriteit van het principe van mutatieleveringen, zodat we kunnen volstaan met partitionering langs de transactietijdlijn.
En zelfs als de soep wél heet wordt genuttigd dan pakken we pas in laatste instantie uit met verschillend gepartitioneerde databases. We zijn daarin terughoudend omdat we dan ofwel genoodzaakt zullen zijn om slimme software te ontwikkelen die bepaalt welke database een bepaalde query het beste kan uitvoeren, ofwel moeten vertrouwen op Oracle-faciliteiten ('*query rewrite*') die we niet zonder meer vertrouwen. Onnodig te zeggen dat we niet staan te springen om zo'n meta-optimizer te ontwikkelen op een RDBMS-platform dat volgens ons daarin zelf zou moeten voorzien, maar als het moet dan doen we het. Voordat het zover komt kunnen we ook nog tactische middelen inzetten, bijvoorbeeld:

- bij bulkleveringen standaard werken met DDL in plaats van DML: "Create Table ... As Select ..." bespaart veel transactionele overhead;
- bundelen van leveringen: één bulklevering voor alle 600 pensioenfondsen op basis van één bulklevering die naderhand wordt uitgesplitst is véél efficiënter dan 600 query's;
- optimalisatie van de server- en netwerkconfiguratie in samenwerking met de rekencentrum exploitant;
- gebruik maken van subpartitioneringsmechanisme (vanaf Oracle 11) om daarmee toch enigszins beide tijdlijnen in de partitionering van de database te verwerken.

Natuurlijk beschikken we als het nodig is ook over zwaardere (maar relatief kostbare) middelen, met name een overgang naar een gespecialiseerde versie (RAC) van het Oracle RDBMS of in het uiterste geval zelfs een ander, meer gespecialiseerd RDBMS. In geen geval willen we echter toe naar een situatie die wordt gekenmerkt door veel dataservers en de daarmee gepaard gaande extreme kosten van exploitatie en beheer.

Met dit palet aan mogelijke lichtere maatregelen denken we in staat te zijn om de PA gedurende een flink aantal jaren met (bijna) 100 procent beschikbaarheid te laten functioneren op twee serverplatforms. Zoals we hebben gezien geven we geen euro te vroeg uit aan andere dingen dan meten, extrapoleren en nadenken. En als we over een aantal jaren moeten vaststellen dat we het met onze simpele aanpak niet gaan trekken dan komt dat ofwel door onvermijdelijke verschillen in ICT-principes van onze afnemers, ofwel door een extreme groei in de afname van gegevens. Misschien loopt het kostenplaatje van de PA, dat nu in de miljoenen per jaar loopt weer op naar de tientallen miljoenen van voor 2008, maar de prijs voor de laagste kosten per geleverde overheids-Terabyte moet te winnen zijn, zeker als onze bulkdata afnemers eens goed kijken naar hoe het CBS het doet. En dat alles bereiken we door zoals altijd de dingen technisch zo simpel mogelijk te houden.
Het is jammer voor de redactie van DB/M en voor Oracle Corp, maar wie de zaken maximaal simpel houdt komt met de bestaande hardware en RDBMS-technologie heel ver.

Na zes artikelen over de opzet van de PA als zodanig kijken we in het komende artikel echt naar buiten. Dat moet een keer gebeuren want de PA is uiteindelijk slechts een schakel in een veel breder stelsel van onderling verbonden ICT bouwstenen – overheidsbasisregistraties – dat op termijn moet uitgroeien tot het skelet van de gegevenshuishouding van de BV Nederland. De PA kan op termijn alleen zijn volle potentieel bereiken wanneer dat skelet goed in elkaar zit. En dat blijkt nog niet het geval.

René Veldwijk is partner bij FAA Partners, onderdeel van de Ockham Groep. **Maarten van Nijnanten** is werkzaam bij SSB-ICT en als DBA verbonden aan het Polisadministratie programma.