Pervasive.SQL v8: 'Btrieve getrouwe' gebruikers extra verwend

Snelheidsverbetering voor ingebedde database

Filip Leys

We schrijven februari 1982. De prille, maar al snel tot de facto marktstandaard uitgroeiende Personal Computer van IBM is amper zes maanden oud. Nancy en Doug Woodward ontwikkelen in hun garage één van de eerste voor de PC bestemde databaseproducten; Btrieve. Het vult probleemloos één van de talrijke gaten binnen het toen nog karige software-aanbod in. Vijf jaar later gaat SoftCraft, het omkaderend bedrijfje dat de Woodwards intussen hadden opgericht, op in Novell en wordt Btrieve een onderdeel van het populaire NetWare.

In 1994 scheuren ze zich daar opnieuw van los, en dopen hun bedrijf om tot Btrieve Technologies, opererend vanuit Austin, Texas. Omdat het intussen druk was geworden op de databasemarkt (met spelers als dBASE, Microsoft, Oracle, Sybase, Informix, IBM) wordt het roer drastisch omgegooid. Men kiest resoluut voor een niche: embedded software en de daarbij horende ultralichte databanken. Btrieve Technologies wordt Pervasive Software, en ook andere werelden dan die van de PC worden ontdekt. Zo bestaan er van de meest recente release van de serverversie, Pervasive.SQL v8, varianten voor Windows NT en 2000, maar ook voor NetWare (4, 5.1, 6 en hoger) en Linux (vanaf Kernel 2.2).

Migratiekoorts

Aan de client-kant mogen daar nog Windows 98, ME en XP aan toegevoegd worden én zelfs nog DOS 5.0 of hoger. Zowel DOS, Windows 16-bit- als 32-bit-applicaties, kunnen er gebruik van maken. Het is in ieder geval tekenend voor het vertrouwen dat kan en mag gesteld worden in de duurzaamheid van het product: ontwikkelaars die ervoor kiezen om Pervasive als onderdeel van hun applicatie in te bedden, hoeven niet te vrezen voor een jaarlijkse of in het beste geval tweejaarlijkse door Microsoft aangestuurde opstoot van migratiekoorts. Het is dan ook niet verbazingwekkend dat van de boekhoudpakketten onder Windows bijna 70 procent gebruik maakt van een Btrieve- of Pervasive-database! Ingebedde databases zijn kleine, weinig eisen stellende applicaties, bestemd om op een zo transparant mogelijke manier gebruikt te worden binnen grotere systemen. Ze mogen dus geen repetitieve administratieve ingrepen vergen: een DBA hoort overbodig te zijn. Bovendien gaat het meestal om overkoepelende toepassingen die op grote schaal geproduceerd en gebruikt worden, dus moet de TCO zo laag mogelijk uitvallen. Dit impliceert niet alleen een lage aanschafprijs, maar ook zo weinig mogelijk kosten voor ontwikkeling, ontplooiing, onderhoud, training, upgrades en ondersteuning. Pervasive.SQL v8 heeft voorzieningen voor elk van deze deelaspecten, maar voor deze upgrade ging het meeste werk naar het verbeteren van de performance, ongeacht de voor gegevenstoegang gebruikte API of model. In deze context horen features als de Dynamic Cache en de Turbo Write Accelerator (TWA). Toch kon de producent het niet nalaten om 'Btrieve getrouwe' gebruikers extra te verwennen met bijkomende realisaties, waaronder de Client Cache.

Client Cache

De performance van een applicatie die met een database samenwerkt, wordt bepaald op vier domeinen: schijf-I/O, het beheer van simultane toegangen, het netwerkverkeer en de wijze waarop processor en werkgeheugen aangesproken worden. Pervasive.SQL v8 concentreert zich op de drie eerste, want enig 'morswerk' met processorcycli is te verwaarlozen tegenover het nadeel dat overmatige schijf- en netwerktoegangen met zich meebrengen.
Het Client Cache-concept bevat, naast het traditioneel bedienen van record per record client requests, nu ook een paginaserver. De client onderhoudt een motor, die werkt met een lokale cache van (geheugen)pagina's. Alleen leesoperaties met grendels, schrijfbewerkingen en operaties binnen een gebruikerstransactie, worden doorgesluisd naar de server op basis van records in plaats van pagina's. De Client Cache en de paginaserver beheren de simultane toegang tot de cache. Op basis van de hit ratio en het systeemverkeer in het algemeen, beslissen ze en schakelen ze waar nodig over tussen de Client Cache en de record requester. Concreet komt een en ander erop neer dat gegevens die vaak nodig zijn, altijd bij de hand blijven.

Terwijl deze winst in de eerste plaats gezocht wordt in het maximaal reduceren van netwerkverkeer, gaat Dynamic Cache het eerder zoeken in het optimaal beheren van de grootte van de cache, waardoor dus het aantal dure schijftoegangen op de korrel genomen wordt. Een statische hoeveelheid geheugen aan een dergelijke functie toewijzen en instellen via een configureerbare parameter, is wellicht geen bezwaar als de database op een toegewezen servermachine draait. Maar op een willekeurig client

station, waar het geheugengebruik permanent verandert en fragmenteert tijdens het starten en stoppen van allerlei processen, is zoiets een illusie. Vandaar het idee om te dynamiseren.
Dit gebeurt door een soort 2-tier cache in te voeren, L1 en L2. De 'actieve' L1 is statisch en bevat alle pagina's waarop effectief gewerkt wordt. De 'inactieve' L2 is het eigenlijke dynamische gedeelte en pagina's worden tussen L1 en L2 uitgewisseld. Het beheer hiervan kan in de achtergrond gebeuren, onafhankelijk van enige kernactiviteit. Bovendien verzekert deze opgesplitste aanpak ook de continuïteit van bestaande, fijn afgestemde applicaties: hun optimaal geconfigureerde cache krijgt een L1 statuut en niets verandert of – als er daarnaast nog plaats is om wat L2 cache toe te voegen – neemt de performance zelfs verder toe.

Turbo Write Accelerator

Toch kunnen in een database, die nu eenmaal per definitie vaak verandert en aangepast dient te worden, I/O-aanvragen nooit volledig geëlimineerd worden. De TWA probeert deze te optimaliseren door schrijfoperaties zo aaneengrenzend mogelijk te maken. Het is voor het mechanisme van een harde schijf immers stukken minder kostbaar om te blijven schrijven dan om te schrijven, te stoppen, de schrijfkop naar een ander spoor of sector te verplaatsen en daar dan verder te schrijven. Bovendien vergt dit allemaal nog extra interactie met het besturingsysteem. In v8 probeert de TWA gaten te vinden of te creëren in het fysische bestand, waarin meerdere pagina's achtereen kunnen weggeschreven worden in één enkele, sequentiële schrijfoperatie op systeemniveau. Helaas lukt dit alleen voor databases die in het nieuwe v8-formaat opgeslagen staan: bestanden van oudere versies (toch vanaf 5.x) kunnen nog wel gelezen en gewijzigd worden, maar zonder de snelheidsboost die de TWA-functionaliteit oplevert.
Verder dient een compromis gezocht te worden tussen performance en de grootte van het databasebestand. TWA optimaal laten werken houdt immers in dat men er alle belang bij heeft zoveel mogelijk beroep te doen op het uitbreiden ervan. Niet verwonderlijk dat we een nieuwe "File Growth Factor" parameter aantreffen onder de "Performance Tuning" optie van de server.

Control Center

Het instellen van deze en alle andere relevante parameters gebeurt met één van de applicaties die onderdeel vormen van dit product, het Pervasive Control Center. Dit is een geïntegreerd, hiërarchisch gestructureerd raamwerk waarbinnen databases verder ook nog gecreëerd, ondervraagd en gewijzigd kunnen worden. De meeste essentiële taken zijn daarin beschikbaar in de vorm van Wizards, terwijl dit voor een aantal andere (nog?) niet het geval is. Dergelijke "outsiders" worden dan opgestart vanuit een 'Tools' menuoptie, waar nog andere externe programma's aan toegevoegd kunnen worden. Standaard gaat het om de Function Executor (om Btrieve-bewerkingen één per één uit te voeren, zodat testen en debuggen vlotter verloopt), de System Analyzer (een diagnoseprogramma dat de diverse componenten van een Pervasive opstelling, inclusief de netwerkomgeving, nagaat, archieven beheert), de Gateway Locator (waarmee men de Workgroup Engine, die gebruikt wordt als gateway voor data bestanden in een bepaald directory, bepaalt of verandert), de Monitor (om de activiteiten van de server – denk maar aan gebruikers, IP-adressen, ODBC-ommunicaties – in de gaten te houden), de License Administrator, de Maintenance utility (voor het uitvoeren van allerlei bestands- en gegevensmanipulaties) en de Rebuild utility (om oudere Btrieve-gegevensbestanden om te zetten naar recentere versies, zoals het nieuwe v8-formaat).

Pervasive.SQL v8 wordt aangeboden als een drieledige productfamilie, bestaande uit de meest volwaardige Server, die start vanaf een tiental maar kan groeien tot honderden concurrente gebruikers, de meer bescheiden Workgroup-variant, die één tot vijf gebruikers toelaat, en de Software Development Kit, specifiek op de ontwikkelaar gericht. Dit laatste vertaalt zich in de voorziening van een native ODBC driver, een OLE-DB provider die toegang biedt tot zowel de relationele als de transactionele interface, een Java interface, een aantal low-level API's, een ActiveX interface, complete voorbeeldapplicaties in omgevingen als Visual Basic, Delphi, Java en C/C++, en het online Developer Center, een webstek waar de laatste componenten en codevoorbeelden van afgehaald kunnen worden. Eerdere versies hadden nog een bijkomend aanbod, speciaal met de 'eenzame' ontwikkelaar op het oog: de Workstation variant. Maar deze is vanaf v8 gelukkig geïntegreerd in het Workgroup product.
Evenzeer valt het veranderde licentieconcept toe te juichen. Waar vroeger licentiesleutels in de vorm van binaire bestanden dienden te worden toegeleverd op cd of floppy, is men nu overgestapt naar een model op basis van leesbare, alfanumerieke strings. Elke licentie geeft toegang tot een bepaald aantal computers om gezamenlijk te verbinden met de database. Alle applicaties die toegang moeten krijgen tot de database en op dezelfde machine actief zijn, tellen samen als één gebruiker. Maar dit geldt – in tegenstelling tot vroeger – nu ook voor toepassingen op hetzelfde station als de server! Voor de volgende versie van de SDK is een verzameling API-calls aangekondigd, die zullen toelaten dit licentiemodel programmatorisch te benaderen en erop in te grijpen.

Conclusie

Pervasive.SQL slaat een totaal andere weg in dan de Btrieve-voorganger en aan deze weg timmert de v8 rustig verder. Er valt in deze tijden méér garen te spinnen met databases voor kleine, mobiele systemen dan indien men wil blijven opboksen tegen groten als Access of Oracle.
Het verder optimaliseren van de performance, de lagere TCO en de flexibiliteit als het op inbedden, ontplooien en connecteren via één van de talrijke industriestandaarden aankomt, moet de op zich al vrij grote bestaande gebruikersbasis en wellicht een nieuwe, overtuigen de upgradestap te zetten en mee te groeien.

Filip Leys

Ir. Filip Leys (filip.leys@sd.be) is freelance journalist.