Klaas Brant over IBM's DB2 Universal Database

Fris en Stevig

Laatst in de supermarkt stond bij de aardbeien een bord met daarop: "Fris en Stevig". In een flits schoot door mij heen: "Zuur en Hard". Ik moest een beetje lachen om het prachtig eufemisme "Fris en Stevig". Ik ben in het verleden verantwoordelijk geweest voor de pre sales-activiteiten van een software-leverancier en heb toen altijd mijn medewerkers voorgehouden om zorgvuldig met woorden om te gaan. Mijn filosofie was altijd: "Je hoeft niet liegen, maar je hoeft ook niet onmiddellijk de botte waarheid op tafel te gooien". Pas als de klant gaat doorgraven kun je ook over de minder mooie kanten van je product spreken.

Stel je toch eens voor dat een sushi restaurant op de menukaart zou zetten: dode rauwe vis (pas op, indien niet vers kun je er erg ziek van worden). Geen bezoeker zou nog sushi willen eten. Ieder software-product heeft zijn slechte kanten, maar daar spreekt men maar liever niet over. Het is aan de marketing-mensen om via mooie slogans en eufemismen te zorgen dat niemand gaat nadenken over die slechte kanten.

Vervelend wordt het, als mensen gek gemaakt worden met kreten die niet waar zijn of men door onwetendheid gewoon denkt dat een product perfect is. Ik zal een voorbeeld geven en aangezien ik het toch heb over zuur en hard, blijf ik in dezelfde bewoordingen: ACID. Dit is een afkorting van Atomicity, Consistency, Isolation, Durability. Deze vier basisbegrippen zijn lang geleden door Chris Date op papier gezet als zijnde de basis waaraan ieder relationeel DBMS zou moeten voldoen.

Volgens de guru zou een leverancier zijn product geen database mogen noemen als het niet aan de ACID-regels voldoet. Toch weten veel mensen niet van het bestaan van de regels, maar dat is vaak het geval met basisregels. Het is vervelend als er tegen de basisregels gezondigd wordt en daardoor het eindresultaat een inconsistente database of, erger nog, een kapotte database is. Laten we de basisregels daarom eens langslopen.

1. Atomicity (ondeelbaarheid). Dit wil zeggen dat een database alles of niets doet. Dit kan op twee niveaus: een SQL statement gaat helemaal goed of er gebeurt helemaal niets. Dus als er halverwege een deling door nul gebeurt in het updaten van meerdere rijen (set update) dan moeten eerdere updates ongedaan gemaakt worden. Dit vereist een technologie van logging of versioning. De producten die zoiets niet kennen zijn dus om deze reden GEEN database.
2. Consistency (consistentie). Als er integriteit-regels zijn zoals Primairy Key → Foreign Key relaties of uniekheid, dan mag het nooit en te nimmer zo zijn dat men daar tegen kan zondigen of dat men ooit data in de database kan vinden die daar (tijdelijk) tegen zondigen. Dat kan soms een groot probleem zijn. Diverse databases verbieden bepaalde SQL-constructies, omdat er anders tijdens de uitvoering hiervan problemen met deze regel zou ontstaan (denk aan het updaten van een primairy key bij een self-referencing query).
3. Isolation (isolatie). Het simultaan benaderen van de database door meerdere applicaties heeft geen invloed, het resultaat is altijd correct. Hierbij bestaan grote verschillen tussen databases (bijvoorbeeld Oracle heeft *versioning* en DB2 *locking*). Maar vaak kan men ook aangeven hoe men met isolation wil omgaan. Veel mensen snappen de isolation levels niet goed, maar weten vaak wel dat uncomitted read een slechte optie is omdat men dan isolation compleet uitschakelt. Vreemd is dus dat MySQL met een MyISAM die geen isolation kent (jawel uncomitted read!) zo populair is en een database genoemd wordt. Deze populaire combinatie die bij meer dan 80 procent van de web-implementaties in gebruik is, zondigt tegen alle ACID-regels en is volgens de regels van de kunst GEEN database. Als een website een virtuele winkel heeft die gebaseerd is op MySQL/MyISAM, dan moet men niet klagen als een korte stroomonderbreking de 'database' kapot achterlaat.
4. Durability (duurzaamheid). Dit wil zeggen dat als een update *gecommit* is, deze ook 100 procent zo blijft. Het kan en mag niet zo zijn dat een crash toch gevolg heeft dat de data nog in de buffers waren en daarmee verloren zijn gegaan. Simpele implementaties schrijven hun updates tijdens commit, want een crash recovery-algoritme is niet altijd even simpel.

Zo, de regels staan weer en we kunnen dus 80 procent van wat zich database wil noemen naar de categorie file system sturen. Onwetendheid kan dus betekenen dat u verkeerde keuzes maakt waar u later veel spijt van heeft. Ik onderstreep hiermee alleen maar de conclusie van Carel-Jan Engel in zijn column. Want voor u het weet neemt u per ongeluk aardbeien mee die Fris en Stevig zijn, om vervolgens thuis tot de conclusie te komen dat ze Zuur en Hard zijn.

Klaas Brant (kbrant@kbce.nl) is DB2-specialist en directeur van KBCE b.v. Meer informatie over DB2 is te vinden op www.kbce.nl en www.db2-times.com