VAN DER LANS

ACID is niet heilig meer

Al tijdens mijn studie, een lange, lange tijd geleden, werd mij geleerd wat een *transactie* was. In het dikke boek van, wie anders, Chris J. Date getiteld 'Introduction to Database Systems' werd uitgelegd dat een transactie uit een aantal database-operaties bestond, zoals insert, update en delete, die allemaal al dan niet uitgevoerd moesten worden. En hier mocht niet mee gesjoemeld worden. En een transactie was pas een transactie als het aan enkele eigenschappen voldeed. Deze vier eigenschappen werden en worden met de afkorting *ACID* aangeduid. Later meer daar over. Elke zichzelf respecterende database server hoorde dit te ondersteunen. Punt.
De laatste jaren verschijnen er echter steeds meer database servers die wel transacties ondersteunen, maar die niet van ACID uitgaan. Tijdens mijn studie zou dit gelijk hebben gestaan aan vloeken in de kerk. Toch hebben de leveranciers hier een reden voor. Maar laten we bij het begin beginnen.

Waar staat de afkorting ACID ook al weer voor? A(tomicity) betekent dat als een transactie uit enkele database-operaties bestaat én als de database server aangeeft dat de transactie correct verwerkt is, dat alle of geen database-operaties verwerkt zijn. Met C(onsistency) wordt bedoeld dat na het uitvoeren van een transactie de database zich in een correcte toestand moet bevinden. I(solated) wil zeggen dat transacties geïsoleerd uitgevoerd moeten worden. Het mag niet zo zijn dat applicaties elkaars tussenresultaten kunnen zien. Tenslotte, D(urability) heeft te maken met dat het effect van een eenmaal afgeronde transactie permanent moet blijven; ook al gebeuren er rampen zoals het down gaan van de database server, de database-operaties moeten na herstel nog steeds zichtbaar zijn.
Bekende SQL database servers, zoals die van IBM, Microsoft en Oracle, ondersteunen allemaal ACID-transacties. Maar nieuwe database servers als CloudDB en MongoDB niet. Dit heeft alles te maken met *schaalbaarheid*.

Volgens de Amerikaanse professor Eric A. Brewer willen we allemaal van een database server een zo hoog mogelijk niveau van consistentie, schaalbaarheid en partitietolerantie. Perfecte partititietolerantie wil zeggen dat het systeem pas down is als alle netwerkpartities down zijn. Volgens Brewer bestaan er echter afhankelijkheden tussen deze drie eigenschappen. We kunnen maar twee van de drie eigenschappen verhogen. Dit wordt door hem het CAP (Consistency Availabality Partition Tolerance) principe genoemd. Bijvoorbeeld als we de consistentie van een database server verhogen, verlagen we daarmee waarschijnlijk de schaalbaarheid. Kortom, verhoog je er een, dan verlaag je een ander.

Het zijn vooral consistentie en schaalbaarheid die elkaar in de weg zitten. Bijvoorbeeld, als een database server ACID-transacties ondersteunt, kunnen applicaties elkaar blokkeren en dat kan ertoe leiden dat ze op elkaar gaan wachten. Tevens leidt het tot veel interne administratie voor een database server. Beide kunnen ertoe leiden dat een database server niet de schaalbaarheid haalt die een organisatie nodig heeft. Een oplossing zou dan kunnen zijn dat zo'n organisatie voor een database server kiest die wel de benodigde schaalbaarheid biedt, maar qua transacties wat water bij de wijn doet.

MongoDB van 10gen doet inderdaad water bij de wijn. Hier heeft de ontwikkelaar de keuze wat het consistentieniveau van transacties moet zijn. Er kan bijvoorbeeld aangegeven worden dat de mutaties alleen in het geheugen verwerkt moeten worden en pas later ook op disk. Als er in die tussentijd iets fout gaat, kan die transactie wel verloren gaan. Maar als dat niet het geval is, kunnen er zeer veel transacties per tijdseenheid verwerkt worden. Dus door het consistentieniveau iets te verlagen, kan de schaalbaarheid verhoogd worden. Het blijven dus wel transacties, alleen zijn het geen ACID-transacties meer. Het consistentieniveau is dus verlaagd.

Laten we eerlijk zijn, het is geen ramp als we in bepaalde omgevingen één of enkele transacties verliezen. Neem bijvoorbeeld een applicatie die al het verkeer op een website naar een tabel in een database schrijft. Stel dat er gemiddeld per dag ongeveer tien van deze schrijfoperaties verloren gaan. Is dat echt een ramp? Ik denk het niet.

Sommige omgevingen, zoals bijvoorbeeld de financiële wereld, hebben absoluut ACID-transacties nodig. Maar dit geldt niet voor alle omgevingen. Als we de schaalbaarheid naar een gewenst niveau kunnen verhogen door de consistentie te verlagen, dan is dit wellicht een prima oplossing. Kortom, als ontwerpers moeten we ACID dus niet meer als een gegeven zien, maar als een ontwerpbeslissing. Het wordt tijd dat Chris Date zijn boek hierop gaat aanpassen.

Rick van der Lans is zelfstandig IT-consultant.