De mythische status van *oid*

Regel 2: De gegarandeerde toegang-regel

Frido van Orden

De tweede regel is feitelijk een herformulering van de fundamentele eis van primaire sleutels. De regel zegt dat iedere individuele scalaire waarde in de database logisch adresseerbaar moet zijn door specificatie van de naam van de tabel, de naam van de kolom en de primaire sleutelwaarde(n) van de regel die de waarde bevat.

Wat hebben we inmiddels geleerd over het relationele model? Bij de bespreking van 'regel 0' zijn het relationele model en het netwerkmodel tegen elkaar gelegd. De conclusie daarbij was dat het netwerkmodel beschouwd kan worden als het relationele model met toevoeging van het concept van 'Codasyl sets'. Bij de bespreking van de informatieregel, in het vorige artikel, is onder meer het concept van de *pointer* behandeld (tegenwoordig vooral vertegenwoordigd in de wereld van object-oriëntatie).
In beide gevallen was de conclusie dat het extra concept niets toevoegt aan de uitdrukkingskracht van het model. Sterker nog, de relationele join-operatie, gebaseerd op het vergelijken van waarden, kan alles aan elkaar relateren zolang de waarden die vergeleken worden maar van hetzelfde type zijn. Natuurlijk zal die relatie vaak een verwijzende sleutel relatie zijn (lees: Codasyl set, pointer), maar het hoeft niet!

Object id

Het was al beloofd aan het eind van het vorige artikel: deze keer gaan we het onder meer hebben over een van de heilige huisjes van de object-oriëntatie, de *object id* of kortweg *oid*. Waar de pointer de object-georiënteerde versie van de verwijzende sleutel is, is de oid het equivalent van de primaire sleutel. Toch wordt in de OO-literatuur de oid welhaast een mythische status toegekend en betiteld als iets geheel anders dan de primaire sleutel.

Er worden daarbij verschillende argumentaties gehanteerd, die we een voor een zullen bespreken:

- *Primaire sleutels kunnen uit meer dan 1 kolom bestaan, terwijl de oid een enkelvoudige waarde is.* Dit is natuurlijk geen wezenlijk onderscheid, een primaire sleutel bestaande uit 1 kolom is gewoon een bijzonder geval van de algemene regel dat de verzameling primaire sleutelkolommen een deelverzameling is van alle kolommen in de tabel;
- *Primaire sleutelwaarden zijn uniek binnen een tabel, de oid is uniek over de gehele database.* Dit is feitelijk een extra uniciteits-eis die aan oid-waarden wordt gesteld. Een veelvoorkomende constructie in het relationele model is het hanteren van zogenaamde 'surrogate keys', primaire sleutels bestaande uit een kolom die automatisch wordt gevuld met volgnummer, afkomstig van een volgnummergenerator (bijvoorbeeld het autonumber feature van SQL Server of een sequence in Oracle). Het toekennen van een unieke oid binnen de database kan door voor alle tabellen dezelfde volgnummergenerator te gebruiken;
- *De waarden van primaire sleutelkolommen zijn zichtbaar, net als niet-sleutelkolommen, de oid is een onzichtbare en betekenisloze eigenschap van een object die alleen dient om objecten van elkaar te kunnen onderscheiden.* Hier raken we de informatieregel weer! Het relationele model kent geen onzichtbare eigenschappen, immers, alles informatie wordt gerepresenteerd door middel van een waarde op een kolompositie. Indien de oid daadwerkelijk betekenisloos is, kan het natuurlijk ook geen kwaad om de waarde van een oid gewoon te kunnen zien;
- *Primaire sleutelkolommen zijn wijzigbaar, de oid is onveranderlijk.* Hier zijn de meningen binnen de object-oriëntatie al evenzeer verdeeld als in de relationele wereld. In de object-oriëntatie, waar de oid betekenisloos dient te zijn, is de noodzaak tot het kunnen wijzigen van de oid natuurlijk feitelijk afwezig. Er is echter geen enkele fundamentele reden waarom de oid van een object niet gewijzigd zou kunnen worden, mits natuurlijk de nieuwe oid wederom uniek is en de verwijzingen naar de te wijzigen oid navenant worden aangepast (cascade update).

Relational Rules (4)

De vorig jaar overleden dr. E.F. Codd werd wereldberoemd met zijn serie publicaties over een gegevens(meta)model dat later bekend zou worden onder de naam 'Relationeel Model'. De eerste uit die serie publicaties was een intern IBM Research Report dat uitkwam in 1969. 2004 zal dus gelden als een lustrum – 35 jaar Relationeel Model.
Frido van Orden schrijft in Database Magazine een serie artikelen over de betekenis en de erfenis van het gedachtengoed van Codd. Aflevering vier behandelt de tweede regel.

Er blijkt dus geen enkele unieke eigenschap van het gebruik van oid's die niet kan worden bereikt met primaire sleutels, behalve dan het onzichtbaar maken van de oid. Hebben de object-georiënteerden hier een punt?

Het verschil tussen dit en dat

In de object-oriëntatie wordt onderscheid gemaakt tussen *identiciteit* (twee objecten zijn daadwerkelijk hetzelfde object) en *gelijkheid* (twee objecten hebben dezelfde eigenschappen, maar zijn toch verschillend). Het relationele model verwerpt dit onderscheid nadrukkelijk: indien twee objecten verschillend zijn moet dit aan de hand van het verschil in waarde van tenminste een eigenschap te onderkennen zijn (met andere woorden uit gelijkheid volgt identiciteit).

Laten we dit illustreren aan de hand van een voorbeeld. We nemen een pak à 500 vel A4 kopieerpapier. In dit pak bevinden zich 500 objecten. Al deze objecten hebben natuurlijk precies dezelfde eigenschappen (ze zijn gelijk), maar toch zijn ze verschillend (niet identiek). Zie hier, de noodzaak voor identiciteit is aangetoond. Of toch niet helemaal?

Laten we eens kijken wat we precies met het voorbeeld beschrijven. Hebben we het over een pak, bevattende A4 kopieerpapier, en wel 500 vel, of hebben we het over een pak, waarin een stapel papier, bestaande uit het eerste, tweede, enzovoort t/m het vijfhonderdste vel.

Pak

Artikel	Aantal
A4 kopieerpapier	500

Pak

Positie	Artikel
1	A4 kopieerpapier
2	A4 kopieerpapier
...	A4 kopieerpapier
500	A4 kopieerpapier

In het eerste geval zijn we helemaal niet geïnteresseerd in het onderscheid tussen de individuele vellen, in het tweede geval nadrukkelijk wel. Wat gebeurt er indien we twee vellen uit het pak halen?

In het eerste geval wordt het aantal vellen in het pak met twee verlaagd. Indien we willen weten waar de twee vellen gebleven zijn, zal er ergens anders in de database de mogelijkheid moeten bestaan om het aanwezig zijn van de twee vellen te registeren. Indien het puur om het registreren van de voorraad in termen van vellen in pakken gaat zijn de twee vellen als het ware 'verdwenen'.

In het tweede geval is de situatie anders en moeten we meer weten over de informatiebehoefte. Is het bijvoorbeeld relevant om te weten waar de twee vellen oorspronkelijk in de stapel zaten? Zo nee, dan bevat het pak na verwijdering van de twee vellen altijd de posities 1 t/m 498. Zo ja, dan bevat het pak de posities 3 t/m 500 als we de vellen van bovenaf pakten, de posities 1 t/m 498 als we de vellen van onderaf pakten en weer iets anders als we de vellen ergens anders uit de stapel hebben gepakt.

Daarnaast is de vraag of we willen weten waar de twee vellen gebleven zijn ook hier weer relevant. Indien het antwoord op de vraag 'ja' is, dan moet tenminste worden gewaarborgd dat er geen vellen uit de database 'verdwijnen'. Wellicht is er ergens een tabel met artikelen, waarin we registeren dat er in totaal, zeg, 1000 vellen A4 kopieerpapier zijn. Een constraint dwingt dan af dat overal waar we de locatie van deze vellen registreren (bijvoorbeeld bij de pakken) het totaal altijd 1000 vel is. Dit levert dan onderstaand gegevensmodel op.

Artikel

Code	Omschrijving	Aantal
A01	A4 kopieerpapier	1000
A02	Whiteboard stift	10

Pak

Positie	Artikelcode
1	A01
2	A01
...	A01
500	A01

Misschien gaan we zelfs zo ver om de 1000 vellen ieder apart te registreren. In dat geval kunnen we zelfs onderkennen dat het ene vel het andere vel niet is, onafhankelijk van de locatie waar het vel zich op enig moment bevindt. Het gegevensmodel ziet er dan als volgt uit:

Artikel

Code	Omschrijving
A01	A4 kopieerpapier
A02	Whiteboard stift

Voorraad

Code	Artikelcode
O0001	A01
...	A01
O1000	A01
O1001	A02
...	A02
O1010	A02

Pak

Positie	Objectcode
1	O0001
2	O0198
...	...
500	O0975

Merk op dat we nu de individuele vellen van elkaar kunnen onderscheiden, zelfs als we de registratie van de locatie buiten beschouwing zouden laten. Blijkbaar kunnen we aan een vel *sec* zien dat het vel O0193 is en niet vel O0287. Dan is de werkelijkheid natuurlijk absurd. Maar als we accepteren dat we twee vellen niet van elkaar kunnen onderscheiden, dan kan het niet zo zijn dat er een informatiebehoefte bestaat naar de exacte locatie van een bepaald vel. Immers, zouden we twee vellen onderling verwisselen (of bijvoorbeeld de stapel omdraaien, of twee stapels verwisselen), dan zouden we nergens aan kunnen zien dat de wereld veranderd is. Dit leidt tot de onvermijdelijke conclusie dat we blijkbaar toch slechts geïnteresseerd zijn in hoeveel vellen er op een bepaalde locatie op enig moment aanwezig zijn, en niet in welke vellen dat dan precies zijn. Zouden we dat wel willen weten, dan zouden we de vellen herkenbaar moeten maken door er bijvoorbeeld met potlood een nummer op te schrijven. Maar daarmee wordt de identificatie van het individuele vel zichtbaar! Het lijkt er dus sterk op dat het concept 'identiciteit' als aanvulling op 'gelijkheid' een theoretisch verzinsel is dat geen parallel heeft in de werkelijkheid. Dat blijkt eigenlijk ook al uit de definitie van oid als onzichtbaar en betekenisloos attribuut. Als de oid daadwerkelijk betekenisloos zou zijn, hoe kan diezelfde oid dan gebruikt worden om iets fundamenteels als het kunnen onderscheiden van twee objecten te realiseren? Als we de objecten in de werkelijkheid van elkaar kunnen onderscheiden, dan zien we dat aan een eigenschap, en daarvoor voldoet het concept van gelijkheid volledig.

Meer sleutelfeiten

We gaan nog even door op het concept 'gelijkheid'. Gelijkheid stelt dat twee objecten aan elkaar gelijk zijn indien alle eigenschappen van beide objecten aan elkaar gelijk zijn. Twee objecten die volledig gelijke eigenschappen hebben zijn niet van elkaar te onderscheiden en dienen als hetzelfde object beschouwd te worden. Uit deze kennis volgt een fundamentele regel in het relationele model, namelijk dat duplicaten (identieke rijen in een relatie) niet kunnen bestaan. De SQL-constructie 'select all ...' (die in tegenstelling tot 'select distinct ...' geen duplicaten uitfiltert) is dan ook geen nette relationele constructie, maar eerder een kunstgreep waarbij alle rijen in het resultaat van een onzichtbaar volgnummer worden voorzien. U ziet, niet alleen in de object-oriëntatie maakt men zich schuldig aan dergelijke praktijken!

> Er is geen enkele fundamentele reden waarom de oid van een object niet gewijzigd zou kunnen worden

Uit de definitie van gelijkheid volgt ook dat elke relatie altijd tenminste 1 sleutel bezit, namelijk de sleutel bestaande uit alle attributen van de relatie. Het andere uiterste, een sleutel bestaande uit geen enkel attribuut, is eveneens zinvol. De consequentie van het definiëren van een dergelijke sleutel op een relatie is dat de relatie uit maximaal 1 rij kan bestaan (immers, iedere extra rij zou voor alle 0 sleutelattributen dezelfde waarde hebben).

Conclusie

Als u nog eens goed terugleest ziet u dat we het in dit artikel eigenlijk nauwelijks over regel 2 zelf gehad hebben. De essentie van de regel vinden we ook terug bij niet-relationele (meta-) gegevensmodellen en is even belangrijk als saai. De hele discussie over object id's gaat ook eigenlijk niet over sleutels maar over de informatieregel, waaraan we in het vorige artikel reeds ruimschoots aandacht hebben besteed. De volgende keer houden we ons weer wel netjes aan de agenda, en hoe! Het gaat dan over 'ontbrekende informatie' en de vlammende strijd tussen de relationele boegbeelden Codd en Date. Gaat dat lezen!

Frido van Orden (frido.van.orden@faapartners.com) is partner bij FAA Partners.