Asymmetrische links in de Data Vault (slot)

Terug naar de praktijk

Harm van der Lek

Harm van der Lek heeft in twee artikelen (in DB/M 1 en 2) uiteengezet waarom hij van mening is dat asymmetrische links een goede en theoretisch correcte uitbreiding zijn op de Data Vault methodiek. Ronald Kunenborg stelde in een reactie (DB/M 3) dat dit strijdig is met de uitgangspunten van de Data Vault architectuur en op termijn kan leiden tot problemen in de opslaglaag. Harm van der Lek reageert op zijn opmerkingen.

Beste Ronald, dank voor je reactie op mijn artikelen over asymmetrische links. Dit geeft mij de gelegenheid tot verduidelijking. Ik vind het een beetje overbodig dat je begint met het herhalen van de Data Vault uitgangspunten. Als je zegt: "Vanuit die uitgangspunten zal het nu al wel duidelijk zijn waar mijn problemen met de asymmetrische links vandaan komen", dan suggereer je dat ik het met die uitgangspunten niet eens ben c.q. er tegen zondig in mijn artikel en dat is niet zo. Laat ik, voor ik je verder van repliek dien, nog eens duidelijk definiëren waar we over praten:

Definitie 1: Een linktabel is een tabel die één of meer verwijzingen naar HUB- en/of andere linktabellen bevat.

Definitie 2: Een asymmetrische link is een linktabel waarbij één van de verwijzingen naar HUB- (of andere link-) tabellen een bijzondere rol speelt. Er geldt dan een bedrijfsregel, in het bronsysteem geïmplementeerd als een verwijzende sleutel, dat de relatie, op één moment in tijd gezien, een één-op-veel relatie is.

Let wel: met definitie 2 is nog helemaal niet gezegd dat dit tot verschil in implementatie zou moeten leiden, met andere woorden: aan een gegeven Data Vault ontwerp kan men wellicht helemaal niet zien welke van de linktabellen er nu asymmetrisch zijn. En als ik jouw goed begrijp moet dat zo blijven.
Wie het onderscheid helemaal niet wil maken of zien (dus zelfs niet op metaniveau), bewijst daarmee nog nooit data aan een Data Vault database te hebben onttrokken (bijvoorbeeld in de vorm van ETL naar een datamart), want dan is het wel degelijk van belang en niet alleen maar 'makkelijk' of zo. Zo zal bijvoorbeeld een feitentabel vaak gevoed worden vanuit een symmetrische linktabel, terwijl asymmetrische linktabellen een rol zullen spelen bij de voeding van (eventueel gedenormaliseerde) dimensietabellen. En als je dan, zoals ik, ook nog dit soort zaken wilt automatiseren (via een datamart generator) dan zul je onderscheid tenminste op metaniveau moeten vastleggen.

Ik neem aan dat jij niet tot deze categorie van 'totaalontkenners' behoort, dus gaat de discussie nog steeds alleen maar over de vraag of je het onderscheid mag zien aan de structuur van de Data Vault database. Zoals ik in mijn artikel aangeef zijn er eigenlijk twee punten waarop het verschil in de fysieke implementatie tot uitdrukking zou kunnen komen:
1. Andere of aanvullende alternatieve sleutels;
2. Het toevoegen van een einddatum.

Het is een beetje jammer dat je in je reactie vrij veel aandacht besteedt aan de problemen die voort zouden komen aan het implementeren van een zekere unieke sleutel op de tabel. Als je het artikel namelijk goed leest zie je dat ik aan punt 1 niet zo hecht. Wat mij betreft zet je geen unieke indexen op de tabel, maar volsta je eventueel met het achteraf checken of ze (nog) uniek zijn. Dit laatste in verband met het feit dat het bij de ETL naar datamarts (zoals boven aangegeven) van belang is dat het wel in orde is. Over corrigeren van gegevens heb ik het al helemaal niet gehad. Tot zover dus over de unieke sleutels, maar er moet me nog wel één opmerking van het hart: je zegt, en ik citeer: "Het argument dat je zonder deze unieke sleutels geen structuur in je database hebt is weinig steekhoudend: de HUB's, links en satellieten bieden juist een hele degelijke, schaalbare en consistente structuur, voorzien van metadata, over de ingeladen gegevens heen. En het argument is ook weinig consistent: waarom wel deze bedrijfsregel, maar geen andere?" Het recht om het 'weinig consistent zijn'-verwijt te maken komt toch echt mij toe: waarom doe jij zo moeilijk over bepaalde alternatieve sleutels (waar ik dus niet eens zo aan hecht) terwijl je, naar ik mag aannemen, toch ook wel primaire sleutels zal zetten op je tabellen.
In het bijzonder zal je op een satelliet een primaire sleutel zetten over de velden XXX_SQN (verwijzing naar zijn HUB of Link) en LOAD_DTS. En deze laatste sleutel bewaakt toch echt de regel dat er op één moment in de tijd maar één burgerlijke staat bij die klant hoorde (gesteld dat burgerlijke staat een attribuut is in de satelliet), althans in het bronsysteem.

Wat hier wellicht ook een rol speelt is wat vaagheid en verwarring met betrekking tot het begrip 'bedrijfsregel'. Eigenlijk hebben we het, in dit kader, alleen maar over regels die zijn geïmplementeerd in het bronsysteem en daarvan dan ook alleen nog maar een speciaal type: de één-op-veel regels. Het voorbeeld hierboven (maar één burgerlijke staat op één moment) is er zo eentje en die nemen we over in Data Vault, zij het dat we het daar historiseren in een satelliet. Een verwijzende sleutel in het bronsysteem is er ook zo een, en die zouden we dan niet mogen meenemen? Wat ik bedoel te zeggen is dit: in Data Vault geldt het principe dat we de data uit het bronsysteem overnemen. Zoals ze zijn, zonder transformaties gebaseerd op bedrijfsregels, the good, the bad en the ugly. We gebruiken echter wel degelijk enkele in het bronsysteem gebruikte regels, die mede de structuur van het bronsysteem bepalen, om structuur te geven aan ons Data Vault databasemodel. Wie dat ontkent is mij in ieder geval volledig kwijt.

> In Data Vault geldt het principe dat we de data uit het bronsysteem overnemen

Komen we tenslotte tot punt 2: mag je een einddatum toevoegen aan de link? Met andere woorden: mag je historie gaan bijhouden in een link? Hier lijkt zich dus de wezenlijke discussie af te spelen. Zoals ik heb duidelijk gemaakt kan dit nauwelijks problemen opleveren voor de re-engineering van de structuur van de database op het moment dat de onderliggende één-op-veel regel verandert (hetgeen dus ook alleen maar kan als ook het onderliggende bronsysteem wordt aangepast, of er een ander bronsysteem bijkomt). Immers deze kolom bevat alleen maar afgeleide waarden en kan dus gemakkelijk later verwijderd, dan wel genegeerd worden.

Als we kijken naar de 'foundations', dan heb ik toegegeven dat end-dating een link op het eerste gezicht een afschuwelijke verminking van de Data Vault modelleringprincipes lijkt.
Maar ik heb aangekondigd dat je hier wat anders tegenaan kunt kijken en dat ik daarover nog uitvoerig zal publiceren. Ik ben er vrij zeker van dat ik een heel goede formele mathematische fundering in mijn hoofd heb en als mensen dat later de 'HarmVanDerLek methode' gaan noemen dan zal ik mij vereerd voelen. Het goede nieuws is, dat de Data Vault modelleringprincipes, zoals door Dan gedocumenteerd, precies uit mijn 'theorie' volgen. Op één uitzondering na: van de linktabellen krijgen we twee soorten ...
Om te voorkomen dat we lezers kwijtraken wil ik even terug naar de praktijk. Laten we de 'Yardstick van Dan ' gaan toepassen. Hiermee bedoel ik de suggestie van Dan Linstedt om met elkaar empirische feiten te verzamelen over problemen die zich met de ene, dan wel de andere aanpak, in de praktijk voordoen. De praktijk dient immers uiteindelijk het laatste woord te hebben in een discussie over de theorie. In mijn praktijk heb ik een heel recent concreet voorbeeld waarbij het volgen van de officiële weg (zoals jij bepleit) tot een zeer gemeen probleem heeft geleid. Laat mij het proberen kort, maar toch concreet te vertellen. Wij werken momenteel met Quipu. Deze tool ondersteunt de end-dating van links niet en is wat dat betreft strak in de leer en dat zal jouw goedkeuring dus kunnen wegdragen. Ik vond dat geen punt, want ik begreep wel hoe ik toch de historie van de B'tjes kon volgen voor een gegeven A, ondanks het feit dat in de satelliet onder de link alleen maar de historie van de combinatie A-B wordt vastgelegd, door middel van één attribuut dat het al of niet bestaan daarvan aangeeft (in Quipu heet dat attribuut 'voided').

Dat gaat dan volgens de lijnen die ik in het artikel heb geschetst en die jou ongetwijfeld bekend zijn. Wel wat complex, maar à la. In eerste instantie merkte ik dat soms de tijdlijn die hierbij ontstaat (dus van de historie van de B'tjes ten opzichte van één A) wat inconsistenties vertoonde in de zin dat de einddatum niet helemaal netjes aansloot bij de begindatum van de volgende rij. Secondewerk, maar toch. Maar nog onlangs hadden we een groter probleem: er bleken dubbele rijen te zitten in de zogenaamde actual view die Quipu genereert. De tijdlijnen in de satelliet (die dus de historie van het A-B object bevatte) waren op zich correct kop-staart liggend. Maar wat na lang zoeken bleek is dat Quipu was 'vergeten' rijen toe te voegen met 'voided = true' (wat dus zoiets betekent als: de relatie A-B is niet langer geldig. Nog meer speurwerk leerde ons de uiteindelijke oorzaak: uit een metatabel waarmee Quipu dit gedrag per entiteit stuurt was een rij verdwenen. Dit heeft al met al twee dagen vertraging in ons project opgeleverd. Nu kun je natuurlijk zeggen: dat had Quipu beter moeten implementeren, maar het is volkomen duidelijk dat beide problemen helemaal niet waren opgetreden, als er 'gewoon' met een tijdlijn in de link was gewerkt. In mijn praktijk is het dus 2-0 in het voordeel van 'end-dating links'.

Waarschijnlijk kun jij dit in 2-3 veranderen, want als ik je goed begrijp wacht ontwerpers die de zonde van 'end-dating links' begaan hel en verdoemenis in de praktijk. Ik denk dat het in ieder geval een goede manier van discussiëren is om, als je dit soort rampspoed voorspelt, daarvan concrete voorbeelden te geven. Ik wil je dus vragen drie concrete gevallen te beschrijven, waarbij het mis ging in de praktijk, toen men een einddatum in de link opnam en wat er dan precies fout liep. Als dat niet lukt dan graag minstens twee voorbeelden. Als dat niet lukt dan toch maar ... (vrij naar Wouter Bos).

Harm van der Lek (vdlek@vdlek.nl) is BI Architect bij BinckBank en zelfstandig Docent.