Methode kan centraal datawarehouse modelleren

Relationeel versus Data Vault

Karien Verhagen en Bart Vrijkorte

De Data Vault methode is een relatief nieuwe manier om een centraal datawarehouse te modelleren. De methode wordt nog niet zoveel in praktijk toegepast, maar de geclaimde voordelen zoals sterk verbeterde flexibiliteit en schaalbaarheid zijn interessant genoeg om de aanpak nader te bekijken.

In DB/M3 2007 schreef Pieter Rambags een success story over een Data Vault project. Hij sprak over een generiek model dat probleemloos uitbreidbaar is. Een lofrede die nieuwsgierig maakt. Want hoe zit een Data Vault model nu precies in elkaar? Is het inderdaad flexibeler dan een traditioneel model en moeten we nu elk project volgens deze nieuwe methode uitvoeren?
Een case study is een geschikt middel om daar achter te komen; de voor- en nadelen van de Data Vault methode komen dan boven water. Door zowel de relationele methode als de Data Vault methode los te laten op een casus zullen er twee modellen ontstaan. De verschillen tussen die modellen geven inzicht in de voor- en nadelen van elke methode. Verdere analyse leidt tot richtlijnen die helpen om tot een keuze voor een van beide methoden te komen.

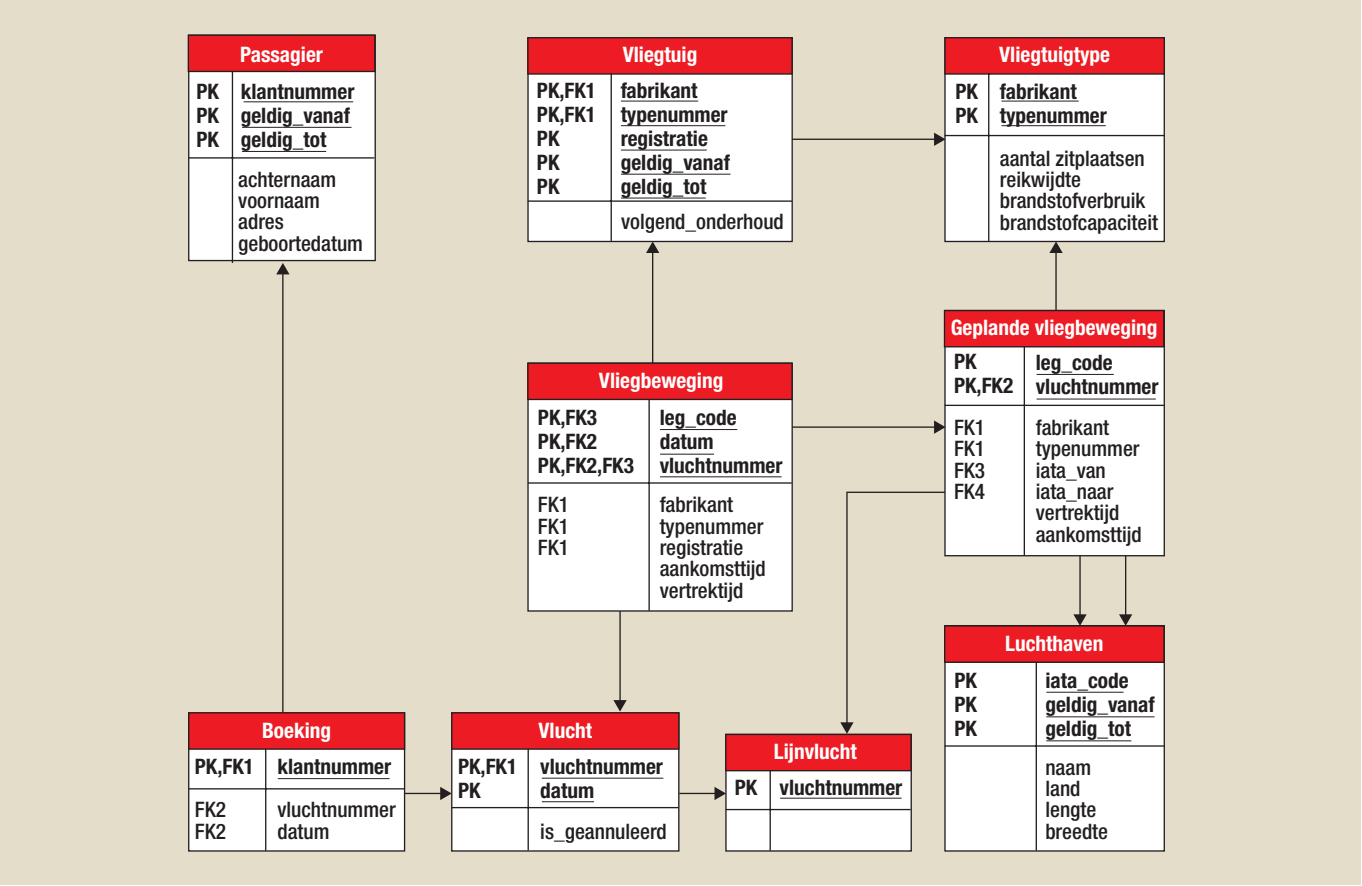
Case

De casus in dit artikel brengt een deel van het datamodel in beeld van een luchtvaartmaatschappij. De centrale entiteit is vliegbeweging, een combinatie van een arrival op een bepaalde luchthaven en een departure vanaf een andere luchthaven. Een vliegbeweging wordt uitgevoerd door een vliegtuig. Een vlucht is in dit model een instantie van een lijnvlucht en wordt geïdentificeerd door een vluchtnummer en de datum. Een fysieke vlucht kan uit meerdere vliegbewegingen of *legs* bestaan.
Het domeinmodel is eenvoudig te realiseren in een relationele database. Elke entiteit komt terug als tabel met als primaire sleutel de business key. Voor passagier is in dit voorbeeld het klantnummer de key en voor lijnvlucht is dit het vluchtnummer, bijvoorbeeld de KL123. De unieke sleutel van de vlucht is de combinatie van de datum en het vluchtnummer.
Zoals bekend zijn *many-to-many* verbanden in een relationeel model niet direct implementeerbaar. Om dit op te lossen vervangen twee afzonderlijke *one-to-many* verbindingen via een tussenschakel het many-to-many verband. In het voorbeeld is Boeking de aangemaakte tussenentiteit die het many-to-many verband tussen passagier en vlucht opheft.
In een datawarehouse is het meestal nodig om historische attribuutwaarden op te slaan. In dit model is dat het geval voor de entiteiten vliegtuig, passagier en luchthaven. Deze tabellen krijgen de extra attributen *geldig vanaf* en *geldig tot*. De *geldig vanaf* datum wordt daarbij toegevoegd aan de primaire sleutel van de tabel. Het is ook mogelijk de historie van verwijzingen bij te houden voorzover die geen deel uitmaken van de primaire sleutel. Historie op vliegbewegingen zou inzicht kunnen geven in de inzet van verschillende vliegtuigtypen op hetzelfde vluchtnummer. Het aldus ontworpen relationeel model is afgebeeld in afbeelding 1.

De ontwikkeling van een Data Vault model begint net als die van een relationeel model met het identificeren van de relevante entiteiten. Voor elke gevonden entiteit in het applicatiedomein wordt een zogeheten hubtabel aangemaakt. Deze hubtabel bevat de business key. Aan de hubtabel wordt daarnaast een gegenereerde surrogaatsleutel toegevoegd. De surrogaatsleutel functioneert als de primaire sleutel voor de hubtabel.

Satelliettabellen

De Data Vault architectuur maakt het ook mogelijk om meerdere zogeheten satelliettabellen per entiteit op te nemen. In de Data Vault methode zorgen satelliettabellen voor de opslag van alle attribuutwaarden. Elke hub krijgt één of meerdere satelliettabellen. Alle inhoudelijke eigenschappen, de laaddatum en de identificatie van het bronsysteem van een entiteit komen terecht in een satelliettabel. Daarnaast is er een extra kolom aan de satelliettabel toegevoegd die verwijst naar de surrogaatsleutel van de hubtabel. In de Data Vault architectuur slaan satelliettabellen de geschiedenis van de attributen op.
Het opbouwen van de historie in een satelliettabel gaat volgens het type 2-principe van Kimball. De Data Vault architectuur maakt het mogelijk om meerdere satelliettabellen per entiteit op te nemen. Dit is nuttig als een entiteit veel attributen heeft waarvan slechts een klein aantal vaak van waarde verandert. Deze snelveranderende attributen kunnen dan in een afzonderlijke satelliet geplaatst worden, waardoor de historie van de andere satelliettabellen niet steeds hoeft te worden bijgewerkt.
Een ander handig gebruik van satelliettabellen doet zich voor



Afbeelding 1: Relationeel model.

bij de integratie van systemen. Als één setje attributen afkomstig is uit het ene bronsysteem, en het andere uit een ander systeem, kan het handig zijn om ze in verschillende satelliettabellen onder te brengen. De laadprocedure voor de entiteit laat zich dan opsplitsen in twee eenvoudigere modules die parallel kunnen draaien.

Linktabel

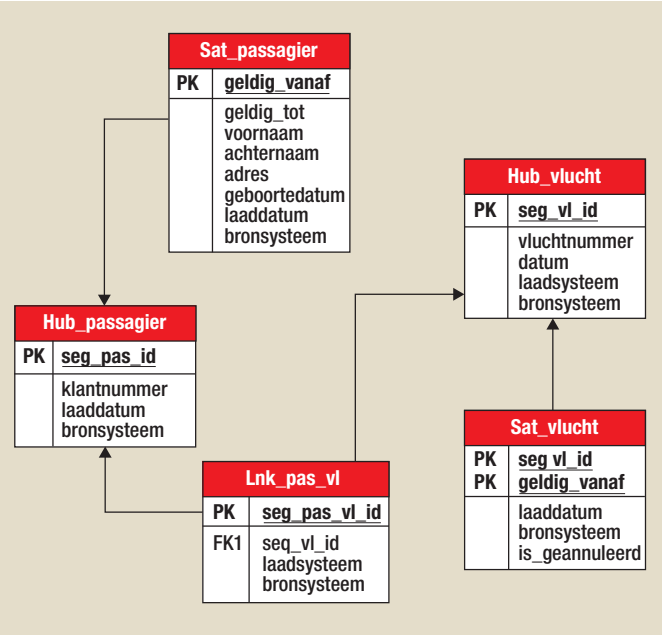
De laatste stap en de meest typerende voor de Data Vault methodiek is het definiëren van linktabellen. Elk verband tussen twee entiteiten resulteert in een aparte linktabel. De linktabel bevat als attributen de surrogaatsleutels van de entiteiten die met elkaar verbonden zijn. De linktabel heeft zelf ook een surrogaatsleutel. Door in een linktabel *geldig vanaf* en *geldig tot* attributen op te nemen wordt het mogelijk om de historie van een verband tussen entiteiten vast te leggen. Soms komt het voor dat het verband attributen heeft (zoals bijvoorbeeld de boekingsdatum in het verband tussen vlucht en passagier). In dat geval krijgt de linktabel een of meerdere satelliettabellen. De relatie tussen passagier en vlucht ziet er nu uit als in afbeelding 2.
Als we het hele proces doorlopen hebben en de case consequent hebben doorgewerkt zijn we de gelukkige eigenaar van een model met vijfentwintig tabellen en daarmee is tevens het grootste nadeel van de Data Vault methode geïdentificeerd. Bij een vergelijking tussen het relationele model en het Data Vault model valt onmiddellijk op dat laatstgenoemde veel meer tabellen en foreign key constraints bevat. De oorzaak is ook duidelijk: elke entiteit levert ten minste twee tabellen op in het Data Vault model tegenover slechts één tabel in het relationele model. Daarnaast is er in de Data Vault methode een extra linktabel nodig voor elk verband tussen twee entiteiten. Direct gevolg is dat het model groter wordt en daardoor minder toegankelijk dan het relationele model.
Is deze complexiteit te verdedigen? Deels wel, want het Data Vault model is inderdaad flexibel. Stel bijvoorbeeld dat het nodig is om van elk vliegtuigtype de vrachtcapaciteit vast te leggen en dat het model daarop moet worden aangepast. In het relationeel model moeten we daarvoor de vliegtuigtype tabel aanpassen en uitbreiden met het extra attribuut. Daarnaast moet de laadprocedure bijgewerkt worden om te zorgen voor vulling en eventuele opbouw van historie. Het Data Vault model is eenvoudiger uit te breiden: het toevoegen van een extra satelliettabel is voldoende.

De Data Vault methode scheidt ook op een nette semantische manier entiteiten van relaties en attributen. Entiteiten hebben een hubtabel, relaties worden vastgelegd in linktabellen en attributen worden in een satelliettabel ondergebracht. Alle entiteiten die nog geen relatie hebben kunnen dat in de toekomst eenvoudig krijgen door toevoeging van een linktabel, terwijl het relationeel

model relaties moet vastleggen door middel van bijvoorbeeld een foreign key verwijzing. Als Business Intelligence toepassing kan het ook heel nuttig zijn om expliciet de historie van een relatie tussen entiteiten vast te leggen. Denk bijvoorbeeld eens aan het verband tussen afdeling en medewerker. In een relationeel model worden de afdelingscode en de afdelingschef vaak als attribuut van medewerker en als recursief verband met de medewerkerstabel in dezelfde tabel opgeslagen. De Data Vault methode schrijft voor dat medewerker en afdeling ieder een eigen hub krijgen met eigen satelliettabellen. Het verband tussen afdeling en medewerker en eventueel de historie van dat verband wordt op een keurige semantisch juiste manier bewaard in een linktabel tussen afdeling en medewerker. Medewerker en afdeling hebben gedurende een bepaalde periode immers een relatie, een link. Dat geldt ook voor de link tussen afdeling en afdelingschef. Dit is een ander soort relatie met dus ook een andere linktabel. De vaak impliciete informatie van relaties tussen entiteiten wordt op die manier netjes als relatie expliciet gemaakt.

Kritiekpunt

Bij het toevoegen van verbanden tussen entiteiten is echter voorzichtigheid geboden. Stel dat de vliegtuigmaatschappij niet alle vliegtuigen in eigen bezit heeft, maar ook vliegtuigen huurt bij externe partijen. Dat brengt de noodzaak met zich mee om de entiteit eigenaar toe te voegen. In het Data Vault model hoeft de bestaande vliegtuigentiteit niet aangepast te worden. Het volstaat de eigenaar hub- en satelliettabellen toe te voegen en vervolgens een linktabel voor de relatie tussen vliegtuig en eigenaar. Dat is eenvoudig, maar realiseert u zich dat een linktabel van elk verband een many-to-many relatie maakt. Het converteren van een many-to-one verband naar many-to-many is niet altijd gewenst. Zo is het in het voorbeeld mogelijk dat een vliegtuig onbedoeld meerdere eigenaren tegelijk heeft. Voorstanders van de Data Vault methode zien geen kwaad hierin, omdat zij vinden dat de laadprocedure ervoor moet zorgen dat de multipliciteiten correct zijn. Maar hiermee verschuiven zij het probleem naar de applicatieprogrammatuur. De applicatiecode moet dan functionaliteit implementeren die eigenlijk standaard al in de database zit.



Afbeelding 2: Gebruik van linktabellen.

Een bezwaar doet zich voor bij entiteiten waarbij (een deel van) de primaire sleutel een verwijzing bevat naar een andere entiteit. Dit is bijvoorbeeld het geval voor de entiteit vlucht (de KL 123 die op woensdag 3 april vertrekt). Het vluchtnummer, de KL 123 is namelijk een verwijzing naar de entiteit lijnvlucht. De Data Vault architectuur schrijft voor dat er een linktabel moet zijn tussen vlucht en lijnvlucht. De informatie in deze linktabel lijkt toch wel erg weinig toegevoegde waarde te hebben en wanneer er geen strikte integrity rules worden gebouwd, bestaat bovendien het gevaar dat de linktabel-informatie in strijd is met de directe verwijzing tussen vlucht en lijnvlucht. Het lijkt het beste om in een dergelijk geval maar af te zien van de linktabel en gewoon foreign key constraints te gebruiken.

> De methode scheidt op een nette semantische manier entiteiten van relaties en attributen

Een belangrijk kritiekpunt op de Data Vault is dat de modellen lastig bevraagbaar zijn. Dit komt met name doordat er veel joins nodig zijn, zeker als er gewerkt wordt met meerdere satelliettabellen per hub. Daarnaast is voor elke bevraagde satelliettabel een filter nodig op de velden *geldig vanaf* en *geldig tot*.
Stel dat de luchtvaartmaatschappij wil weten wat de gemiddelde bezettingsgraad is van elk vliegtuigtype. In het relationele model is deze vraag niet zo moeilijk te beantwoorden: neem de vliegtuigtype tabel en join die met de vliegbeweging tabel (via fabrikant en typenummer) en op boeking met behulp van het vluchtnummer. In de Data Vault methode is het nodig om zeven (!) tabellen te joinen, om uiteindelijk diezelfde link te kunnen leggen tussen een boeking en het aantal zitplaatsen van een vliegtuigtype.
Ook in een relationeel model is overigens een keuze te maken voor surrogaatsleutels. De complicaties bij het bevragen die daaruit volgen zijn dus niet noodzakelijkerwijs een exclusief nadeel van de Data Vault methode. Wanneer een vraag met de hand geprogrammeerd moet worden, zijn vergissingen veel waarschijnlijker in een query op het Data Vault model dan in een query op een relationeel model. Een speciale laag, gebouwd bovenop het centrale datawarehouse, die de query's transparant maakt en de joins in de achtergrond regelt is onmisbaar.

Datawarehouse-generator

Ontwerp en bouw van de ETL zullen bij gebruik van een Data Vault model meer tijd kosten dan bij gebruik van een relationeel model. De oorzaak hiervan is dat een Data Vault model meer tabellen bevat en dat het slowly changing dimension principe op meer tabellen moet worden toegepast.
Wanneer besloten wordt tot opslag in een Data Vault model is daarom een extra investering voor een datawarehouse-generator sterk te overwegen. BIReady, het product uit de Van der Lek-fabriek implementeert desgewenst een volledig authentiek Data Vault model. Van een dergelijk product mag verwacht worden dat het de integriteitsregels in de ETL-fase impliciet afdwingt. Ook vereenvoudigt zo'n product de toegang voor de ontwikkelaars van eindgebruikersapplicaties door een view die op de achtergrond de joins voor zijn rekening neemt. Andere argumenten voor de aanschaf van zo'n tool zijn de verwachte dynamiek in de BI-omgeving en de aanwezige kennis in het Business Intelligence Competence Center (BICC), ook na de ontwikkeling van het eerste increment. Hoe meer onderhoud en hoe minder parate kennis hoe eerder de tool zich terugbetaalt.

Conclusie

Een besluit tot een volledige Data Vault opslagmethode voor het centrale datawarehouse impliceert hoge initiële kosten voor de zelfbouw dan wel een extra investering voor een datawarehouse-generator. Een applicatie om de view op het datawarehouse-model te genereren zodat de toegang eenvoudiger en veiliger wordt, is daarbij geen overbodige luxe. Daarmee heeft men dan wel een toekomstvast en flexibel platform waarbij kostbare reconstructies niet nodig zijn, ook niet als de informatiebehoefte of het bronaanbod onverwacht verandert. Afhankelijk van de bedrijfssituatie kan dat een heel groot voordeel zijn. Wanneer dynamiek in de organisatie de vraag steeds doet veranderen en als ook het aanbod blijft wijzigen, is de Data Vault methode een semantisch correcte, inzichtelijke manier van bedrijfsdata vastleggen die kwaliteit en consequentie afdwingt, veel meer dan een relationeel model.
De initiële kosten voor de ontwikkeling van een Data Vault model zijn hoger, doordat ontwikkelaars de methode moeten leren en het model vele malen groter is dan het geval zou zijn bij relationeel modelleren. Wanneer geen datawarehouse-generator wordt aangeschaft, lijkt zelfbouw van een applicatie die geparametriseerd de integriteit op foreign keys afdwingt ook een noodzakelijke voorinvestering. Als die investeringen eenmaal gemaakt zijn kunnen de onderhoudskosten en vooral ook de kosten van functionele aanpassingen omlaag, door de flexibiliteit van het model gecombineerd met de strikte structurering die het model afdwingt.
Hybride keuzes zijn ook mogelijk maar hebben voornamelijk als nadeel dat ze noch vlees noch vis zijn. Generieke tools die aansluiten op de ene dan wel de andere methode moeten bij een hybride oplossing rekening houden met de uitzonderingen. Wanneer de Data Vault methode populair wordt zullen er ongetwijfeld meer tools komen die ontwikkelaar en beheerorganisatie ondersteuning geven bij implementatie en onderhoud. Dan gaan ook de kosten voor dat noodzakelijke voorwerk omlaag. Wie durft daarop te anticiperen?

Karien Verhagen (karien.verhagen@getronics.com) en **Bart Vrijkorte** (bart.vrijkorte@getronics.com) zijn BI consultant bij GetronicsPinkRoccade.
Met dank aan Theo Wesselink.

Update

Doe mee aan het Nationaal Datawarehouse Onderzoek 2008

Zusterbladen Database Magazine en Business Process Magazine organiseren samen met Atos Origin het Nationaal Datawarehouse Onderzoek 2008 (DWO 2008). Het onderzoek richt zich niet alleen op de IT-afdeling, maar ook op de business als gebruiker van het datawarehouse. De gebruikers kunnen immers inzicht verschaffen in de mate van acceptatie en de ondersteuning die de datawarehouse-omgeving hen biedt bij het nemen van beslissingen. Voor IT-afdelingen en business-afdelingen zijn aparte vragenlijsten opgesteld.

Deelnemers aan het DWO 2008 krijgen:
- een speciale PDF-editie van de gedetailleerde uitkomsten van het onderzoek, de conclusies en aanbevelingen;
- 75 euro korting (bovenop eventuele andere van toepassing zijnde kortingen) op de toegang tot het Datawarehousing & Business Intelligence congres op 2 oktober 2008 in Leiden;
- indien de deelnemer dat wenst, een kostenloze terugkoppeling van zijn/haar specifieke datawarehouse-omgeving. Hierbij worden in een advies verbeterpunten aangegeven.

De vragenlijsten worden anoniem geaggregeerd en verwerkt. In onze bladen worden de resultaten van het onderzoek toegelicht. Daarnaast verschijnt er een uitgebreide rapportage, met daarin de uitgangspunten, resultaten, conclusies en aanbevelingen van het onderzoek, die tijdens het jaarlijkse congres Datawarehousing & Business Intelligence zal worden gepresenteerd.

Doe mee aan de enquête!
Ga naar www.dbm.nl en klik op het logo DWO 2008. Het invullen van de vragenlijst duurt ongeveer 10 minuten.