Slim gebruik van de datumdimensie in een datawarehouse-omgeving

Doe slimme dingen met uw tijd!

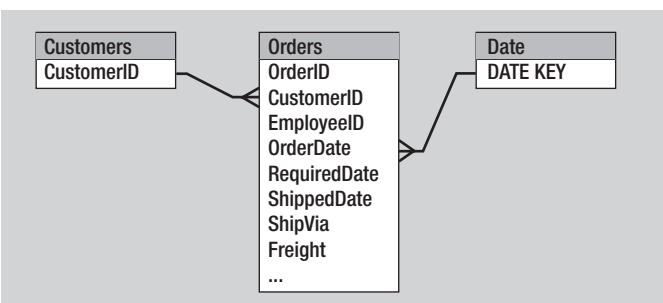
Michiel Brunt

Organisaties implementeren datawarehouses om op eenvoudige wijze te kunnen rapporteren en te analyseren met gegevens over legio onderwerpen. Als er één aspect is dat terugkomt in al deze analyses, dan is het wel het aspect 'tijd'. Heeft men een dimensioneel gemodelleerd datawarehouse, dan kan het dus geen kwaad stil te staan bij de inrichting van de dimensie 'datum'. Dit artikel geeft inzicht in hoe de datumdimensie op eenvoudige wijze de analysemogelijkheden van het relationele datawarehouse kan vereenvoudigen én vergroten.

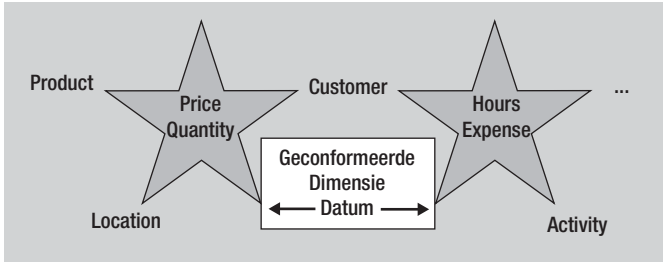
In de voorbeelden wordt uitgegaan van een gebruiker die, zonder kennis van SQL, zelfstandig rapportages samenstelt op basis van een dimensioneel gemodelleerd datawarehouse. De gebruiker heeft hiervoor een query- en rapportagehulpmiddel ter beschikking. Dit hulpmiddel genereert SQL gebaseerd op de geselecteerde objecten uit de semantische laag (de business-representatie van het datawarehouse).
De voorbeelden in dit artikel zijn gebaseerd op enkele tabellen van de Microsoft Northwind database. In afbeelding 1 staan ze aangegeven. De tabel Date is voor dit artikel toegevoegd aan de database.

De datumdimensie

Een dimensioneel model bevat veelal meerdere feitentabellen, bijvoorbeeld orderfeiten die met een orderdatum worden geïdentificeerd en inkoopfeiten die met een factuurdatum worden geïdentificeerd. In rapportages maakt men vergelijkingen in de tijd, waarbij voor de orderfeiten de orderdatum wordt gebruikt, terwijl



Afbeelding 1: Tabellen.



Afbeelding 2: Koppeling datumdimensie.

voor de uitgavenfeiten de factuurdatum wordt gebruikt. De datum is in de wereld van het dimensioneel modelleren een geconformeerde dimensie. Dit is te zien in afbeelding 2: de datumdimensie is gekoppeld aan zowel de orderdatum als aan de factuurdatum.

Om de datum als geconformeerde dimensie te gebruiken, is het nodig om een aparte datumdimensietabel te maken waarin minimaal het attribuut datum staat dat wordt *gejoined* met de orderdatum, als de orderfeiten worden opgevraagd en met de factuurdatum als de uitgavenfeiten worden opgevraagd.
Veel rapportages zijn echter niet zo gedetailleerd dat de datum op een rapportage wordt gebruikt. Vaak worden de gegevens geaggregeerd. Zo worden orders met een orderdatum bijvoorbeeld gesommeerd per maand en worden uitgaven met een factuurdatum gesommeerd per week. Een andere keer is het weer nodig om gegevens per financiële periode te bekijken of misschien alleen voor de weekenden.
Het is dus nodig om in de datumdimensie naast de datum ook de maand van de datum op te nemen en na te denken over andere uitbreidingen. Kimball[1] heeft al eens een aantal standaard-attributen voor de datumdimensie opgesomd. Enkele voor de hand liggende datumattributen zijn Dagnummer binnen de maand, Dagnummer binnen de week, Naam van de dag, Datum, Weeknummer, Indicatie werkdag, Begindatum van de maand en Opgemaakte maand. Door deze attributen op te nemen in de datumdimensie zijn aggregaties eenvoudig te maken door het attribuut datum te vervangen door bijvoorbeeld de maand en de meetwaarden te aggregeren.

Sortering

Een andere handige uitbreiding van de datumdimensie heeft betrekking op de maand. Vaak wordt de maand door gebruikers

als de naam van de maand gepresenteerd. Helaas is dan de sortering lastig, immers april komt voor januari bij een alfa-numerieke sortering. Dit probleem lost men op door de begin-datum van de maand van het datatype Date, als extra attribuut in de dimensie op te nemen. Dit attribuut kan gewoon bij de query worden opgevraagd zonder dat het aantal rijen van het resultaat verandert, het attribuut levert immers voor iedere maand slechts één begindatum. De resultaten kunnen nu eenvoudig worden gesorteerd op deze datum. Een ander voordeel is, dat veel rapportagehulpmiddelen dit datumattribuut ook nog kunnen presenteren met bijvoorbeeld een maandnaam, maar waarbij de sortering altijd met de datum (dus in de goede volgorde) zal plaatsvinden.
Er zijn database management-systemen (DBMS) die functies bieden voor het maken van berekeningen met datums. Hiermee kan vergelijkbare functionaliteit worden geboden, maar dit kent echter een drietal nadelen. Ten eerste is deze oplossing niet generiek, maar afhankelijk van de mogelijkheden die de database biedt. Bij eventuele migraties naar een ander DBMS is het dan ook niet zeker of deze oplossing nog steeds werkt. Daarnaast neemt de complexiteit van query of de semantische laag toe, wat gevolgen voor de performance kan hebben. Een derde nadeel is dat deze specialistische functionaliteit moet worden doorgevoerd in iedere applicatie die gebruik maakt van het datawarehouse.

Met deze simpele uitbreidingen van de datumdimensie wordt de rapportage-omgeving al veel eenvoudiger in het gebruik. De kracht van het concept van geconformeerde dimensies is dat de datumdimensie eenmalig ontworpen en gerealiseerd hoeft te worden, om er vervolgens bij elke nieuwe feitentabel (lees: onderwerp) gebruik van te kunnen maken. De genoemde uitbreidingen zijn echter nog weinig opzienbarend, we gaan nu kijken naar enkele meer geavanceerde analyses met de datumdimensie.

Flexibele rapportages

Organisaties maken vaak gebruik van een groot aantal standaard-rapportages die periodiek worden ververst. Het is vanzelfsprekend erg voordelig als het beheer van deze rapportages minimaal is doordat ze zonder handmatige tussenstappen ververst kunnen worden. Dit is mogelijk door hier bij het ontwerpen van het datawarehouse rekening te houden. Met een voorbeeld wordt dit toegelicht.
Een standaardrapport dat maandelijks inzicht geeft in de omzet van de afgelopen drie maanden kan worden gemaakt door in de query een WHERE-statement op te nemen op het attribuut maand. Om bijvoorbeeld in februari de afgelopen drie maanden weer te geven zou het WHERE-statement kunnen zijn: MONTH in ('2004-01', '2003-12', '2003-11'). Een dergelijke oplossing is echter niet gebruikersvriendelijk en ook niet flexibel. Immers, volgende maand zal de WHERE-clause moeten worden aangepast. Zeker als meerdere maanden worden geselecteerd, zal dit als gevolg van jaarovergangen al snel niet meer te beheren zijn.
Door het opnemen van een extra attribuut in de datumdimensie, kan men een hoop extra gebruiksgemak leveren. In dat attribuut wordt het aantal maanden vanaf de huidige maand opgeslagen. Het resultaat staat in afbeelding 3 waarbij de huidige maand februari 2004 is.

Maand	Maanden geleden
2004-02	0
2004-01	1
2003-12	2
2003-11	3
2003-10	4
2003-09	5
2003-08	6
2003-07	7
2003-06	8
2003-05	9
2003-04	10
2003-03	11

Afbeelding 3: Extra attribuut in de datumdimensie.

Naast het 'Aantal maanden geleden' zijn natuurlijk ook attributen als 'Aantal weken geleden' en 'Aantal dagen geleden' handig om ter beschikking te hebben. Om de datumdimensie te voorzien van deze mogelijkheden, is het noodzakelijk dat de dimensie iedere dag wordt ververst. Dit is geen enkel probleem, omdat de dimensie zeer klein is en dit dus snel is gebeurd. Bij het verversen van de datumdimensie worden alle datums gerelateerd aan de systeemdatum. Op basis daarvan wordt het aantal dagen, weken, maanden geleden berekend.
Stel dat men nu een rapportage wil maken die iedere maand inzicht geeft in de resultaten van de afgelopen drie maanden. Met behulp van het nieuwe attribuut is dat te bereiken met de SQL in afbeelding 4.

In deze query is een filter op het attribuut te zien, die het aantal maanden vanaf de huidige maand aangeeft (MonthsAgo). Zoals eerder getoond is MonthsAgo gevuld met een 0 voor de huidige maand, een 1 voor de vorige maand en zo terug voor zover er maanden in de datumdimensie beschikbaar zijn. Hierdoor is men onafhankelijk van jaarovergangen en dergelijke. Iedere maand kan

```
SELECT
  Date.MONTH_FORMATTED,
  Sum( Orders.Freight )
FROM
  WTD_DATE  Date,
  Orders
WHERE
  ( Date.DATE_KEY=Orders.OrderDate  )
  AND  (  Date.MONTHS_AGO  IN  (1, 2, 3)  )
GROUP BY
  Date.MONTH_FORMATTED
```

Afbeelding 4: SQL voor rapportage.

Maand	Omschrijving	Vracht
2004-01	Last month	6.606,23
2003-12	Two months ago	4.970,70
2003-11	Three months ago	3.974,25

Afbeelding 5: Tabel met beschrijvende attributen.

het rapport worden ververst zonder de query aan te passen en worden automatisch de juiste gegevens getoond. Dezelfde oplossing is natuurlijk op bijvoorbeeld weken en kwartalen toe te passen. Door het toevoegen van beschrijvende attributen zijn de resultaten nog iets mooier weer te geven. Ook dit is als standaardfunctionaliteit beschikbaar in de dimensie, zonder dat de gebruiker er moeite voor hoeft te doen, zie afbeelding 5.

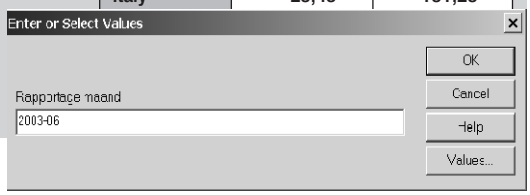
Vergelijking met vorig jaar

Het gebruik van dit attribuut MonthsAgo geeft de gebruiker ook de mogelijkheid van het maken van een vergelijking van de vorige maand met dezelfde maand in het vorig jaar. Dit kan door de conditie op MonthsAgo te zetten op 1 en 13. Hiermee worden de gegevens van de laatste maand en van die maand in het vorige jaar opgevraagd. Uiteraard is het grote voordeel hierbij weer dat deze conditie iedere maand de juiste gegevens toont. Ook bij jaarovergangen blijft het attribuut altijd de juiste waarden tonen. Kenmerkend bij al deze voorbeelden is dat door simpele uitbreidingen van de datumdimensie het datawarehouse voor de gebruiker een stuk eenvoudiger wordt. Met de wetenschap dat eenvoud een van de belangrijkste succesfactoren voor een datawarehouse is, zijn deze uitbreidingen zeer waardevol. In de volgende voorbeelden zijn andere toepassingen te zien van de attributen die gerelateerd zijn aan de systeemdatum.

Variabele periodeselectie

Een flexibel rapport kan worden gemaakt met de variabele MonthsAgo. Hierdoor wordt het mogelijk om flexibel gegevens te selecteren uit verschillende maanden, bijvoorbeeld 1 en 2 maanden geleden ten opzichte van dezelfde maanden in het vorige jaar. Een situatie die in de praktijk nogal eens voorkomt is die waarbij men zo'n rapport niet wil maken op basis van de huidige maand

	2002-12	2003-6
Argentina		1,10
Austria	26,06	585,81
Brazil	139,37	401,38
Canada	196,14	4,33
Denmark		322,32
Finland	7,28	73,16
France	147,54	503,58
Germany	462,26	273,07
Ireland	33,97	
Italy	23,43	131,23



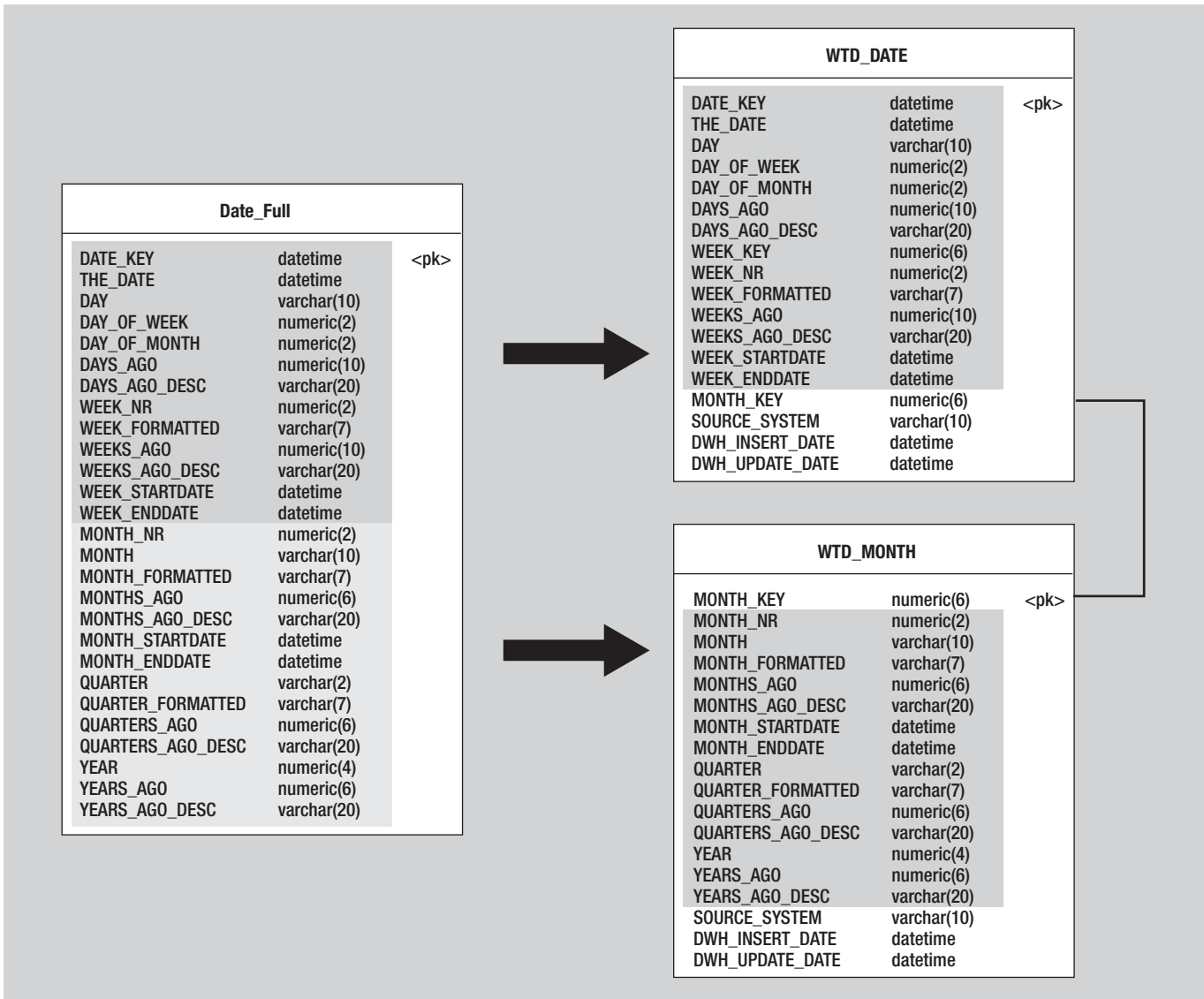
Afbeelding 6: Prompt voor invoer rapportagemaand.

(dus bijvoorbeeld over de afgelopen maand), maar op basis van een zelf op te geven rapportagemaand. De gebruiker kan dan dus in de maand november een standaardrapport opvragen met omzetgegevens van een op te geven maand ten opzichte van zes maanden daarvoor. Het rapport toont dan dus bijvoorbeeld gegevens van juni 2003 en december 2002. Dit type rapportage is lastiger te maken met een rapportagehulpmiddel. Het vraagt namelijk tegelijkertijd om de integratie van een prompt aan de gebruiker (voor het opgeven van de periode) en de mogelijkheid om met het resultaat van de prompt verder te rekenen.
Toch is het mogelijk om met enkele eenvoudige aanpassingen deze gegevens op te vragen. Afbeelding 6 toont eerst een prompt waarin de rapportagemaand wordt gevraagd. Op basis daarvan worden de juiste gegevens uit de database opgevraagd voor de rapportagemaand en een half jaar daarvoor.

Hoe is een dergelijk rapport nu te maken? In deze situatie is sprake van twee verschillende maanden: de maand van de orderdatum en de ingevoerde selectiemaand. Afbeelding 7 toont het gebruik van de verschillende maanden. Links staat de

	Month_Selection			Month_Of_Date		
	Month	Months_Ago		Month	Months_Ago	Calculated Months_Ago
	2004-02	0		2004-02	0	
	2004-01	1		2004-01	1	
	2003-12	2	Join	2003-12	2	
User Selection →	2003-11	3	Month <= Month	2003-11	3	0
	2003-10	4		2003-10	4	1
	2003-09	5		2003-09	5	2
	2003-08	6		2003-08	6	3
	2003-07	7		2003-07	7	4
	2003-06	8		2003-06	8	5
	2003-05	9		2003-05	9	6

Afbeelding 7: Selectiemaand en ordermaand.



Afbeelding 8: Bestaande datumdimensie en opgesplitste situatie.

selectiemaand, daarnaast staat de maand gebaseerd op de orderdatum. De gebruiker selecteert één rij uit de selectiemaand en door de relatie met de ordermaand worden alle maanden geselecteerd die kleiner of gelijk zijn aan de gekozen maand. Nu is het aantal maanden geleden opnieuw te berekenen voor de ordermaanden. Dit kan door het aantal maanden geleden in de selectiemaand af te trekken van het aantal maanden geleden in de ordermaand.

Om deze selecties door middel van SQL te kunnen maken is het nodig om de bestaande datumdimensie, waarin zowel de datum als de maand en het jaar staan, op te splitsen. De bestaande datumdimensie bevat namelijk net zoveel rijen per maand als er dagen zijn, terwijl we een dimensie nodig hebben die iedere maand slechts 1 rij bevat. In afbeelding 8 staat links de bestaande datumdimensie en rechts de opgesplitste situatie. Alle attributen uit de oorspronkelijke dimensie van het niveau maand en daarboven (kwartaal, jaar) worden verplaatst naar de nieuwe maand dimensie.

Het datamodel is nu klaar voor het maken van de gewenste query. Zoals te zien is in afbeelding 7, is twee keer de maanddimensie nodig. Dit kan door een alias van de dimensie te maken. Een alias wordt voor de maandselectie gebruikt, de andere voor de maand van de order. Deze twee aliassen worden gejoined door een kleinere/gelijke join. De maand (inclusief jaar) uit de selectieperiode-alias moet kleiner of gelijk zijn aan de maand in de ordermaand-alias. De gebruiker kiest een maand uit de maandselectie-alias en selecteert daarmee meerdere maanden uit de ordermaand-alias. De SQL die nodig is voor deze functionaliteit staat in afbeelding 9.

Zoals bij de uitgangspunten is aangegeven wordt gebruik gemaakt van een query tool. Dat zorgt ervoor dat men zich om deze SQL geen zorgen hoeft te maken. De selectie voor de gebruiker blijft eenvoudig, het query tool zorgt voor het genereren van de SQL-statements.
In deze toepassing is te zien hoe het gebruik van een extra alias

```
SELECT
  Month.MONTH_FORMATTED,
  Sum( Orders.Freight )
FROM
  Orders,
  wtd_date   Date,
  wtd_month  Month,
  wtd_month  Month_Selection
WHERE
  ( Date.DATE_KEY=Orders.OrderDate  )
  AND  ( Month_Selection.MONTH_FORMATTED =
                                    '2003-06'  )
  AND  (
   Month.MONTH_KEY<=Month_Selection.MONTH_KEY  )
  AND  ( Date.MONTH_KEY=Month.MONTH_KEY  )
  AND  ( Month.MONTHS_AGO -
      Month_Selection.MONTHS_AGO  IN  (0, 6)  )
GROUP BY
  Month.MONTH_FORMATTED
```

Afbeelding 9: SQL-code.

van de maanddimensie zorgt voor een eenvoudige en generieke manier van het selecteren van gegevens. Ondanks de beperkingen die er zijn als men alleen met behulp van SQL gegevens wil opvragen, is het toch mogelijk om meer geavanceerde selectiecriteria toe te passen.

Year to Date-berekeningen

Een laatste veelgevraagde functionaliteit is die van Year to Date (YTD) cijfers. Omzetgegevens worden vaak gedurende het jaar opgebouwd. Men kan dan rapportages maken op een orderfeitentabel die de omzet per maand aangeeft, maar een standaardoplossing om per maand de opgebouwde omzet tot aan die maand vanaf het begin van het jaar op te vragen, is niet zomaar voorhanden. Afbeelding 10 toont de data die nodig zijn voor de YTD-cijfers van bijvoorbeeld maand 2 en voor maand 3. Zoals te zien is, moeten de cijfers van maand 1 en 2 zowel voorkomen in de YTD maand 2 als voor de YTD maand 3.

De YTD-omzet voor maand 2 bevat dus de maandomzet van maand 1 en maand 2. De YTD-omzet voor maand 3 bevat de maandomzet van maand 1 tot en met 3. Wat we graag uit de database zouden willen krijgen zijn de YTD-omzetten voor de eerste drie maanden van het jaar, zoals in afbeelding 11.
Om gegevens op deze manier beschikbaar te krijgen, moet het bedrag voor maand 1 drie keer worden meegeteld, namelijk voor iedere maand een keer. Het bedrag voor maand 2 zal twee keer moeten worden meegerekend. Dit resultaat is te verkrijgen door een soortgelijke oplossing toe te passen zoals hiervoor voor de maandselectie is gebruikt. De oplossing maakt weer gebruik van twee verschillende aliassen van de maanddimensie, de gewone maand-alias op basis van de orderdatum en de maand die zorgt voor de vermenigvuldiging van de gegevens tot en met de gekozen maand. Afbeelding 12 toont het gebruik van de twee maand-aliassen.
De twee aliassen worden voor YTD-cijfers *gejoined* door een samengestelde join, waarbij het jaar uit beide aliassen aan elkaar gelijk moet zijn en waarbij de maand uit de YTD-alias kleiner of gelijk is aan de maand in de ordermaand-alias. Hierdoor worden per maand uit de YTD-alias alle feiten van de voorgaande maanden binnen het jaar bij elkaar opgeteld. De twee maand-aliassen die worden gejoined zijn zeer klein, waardoor een goede performance kan worden bereikt. De goede performance wordt tevens bereikt door het feit dat alle berekeningen in de database worden uitgevoerd.

De SQL die nodig is voor de YTD-cijfers, is met dit mechanisme prima te maken. Er is echter nog een overweging die meespeelt bij het maken van een toepassing voor gebruikers van het data-warehouse. Het is namelijk belangrijk dat gebruikers een eenduidige manier van gegevensselectie wordt geboden en dat we met goed geconformeerde dimensies werken. In de uitleg van dit voorbeeld was er sprake van de YTD-maand en de OrderMaand. De OrderMaand is het attribuut dat de gebruiker al tot zijn beschikking had om de maand van de order op te vragen. Voor deze berekening heeft de gebruiker echter de YTD-maand nodig als eindresultaat, omdat dat het attribuut is dat zorgt voor de gecumuleerde YTD-gegevens. Het is alleen voor een gebruiker niet eenduidig om hem de beschikking te geven over deze twee verschillende attributen die beide een maand aangeven. Het is voor de eindgebruiker duidelijker als hij de beschikking heeft over de dimensie Maand en de feiten Vracht en YTD_Vracht.
De dimensie Maand is hierdoor eenduidig gedefinieerd, en de gebruiker kan eenvoudig de twee meetwaarden (Vracht en YTD_Vracht) naast elkaar selecteren (met dank aan Nico Vis, Agis Zorgverzekeringen).
Om dit te bereiken worden twee aliassen gemaakt van de orderfeitentabel. Op basis van de eerste alias wordt de normale vracht berekend, op basis van de tweede wordt de YTD-vracht berekend. Afbeelding 13 toont de eerste alias die wordt gebruikt met de normale datum- en maanddimensie.

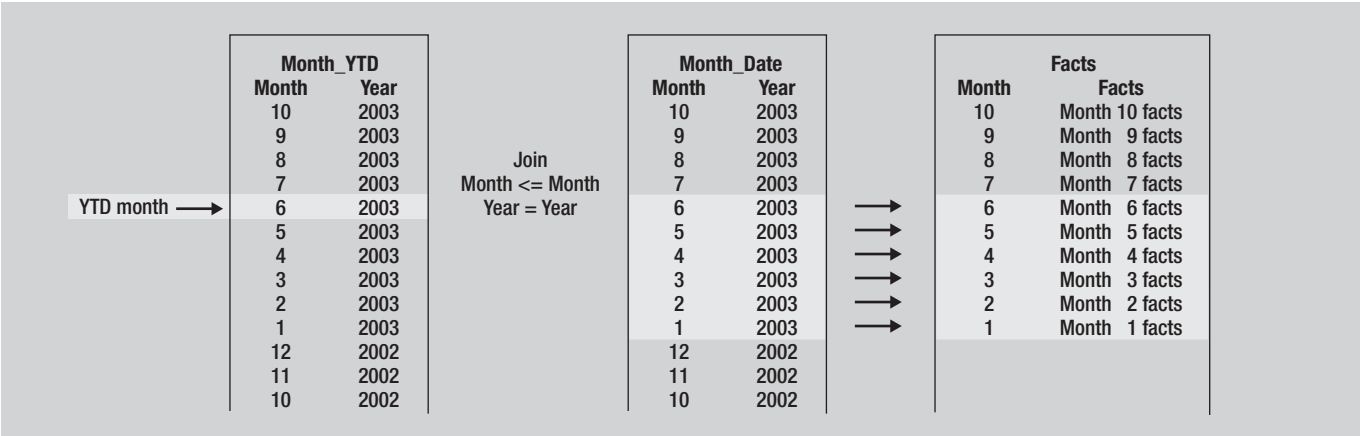
Maand	Vracht
2003-01	2.504,59
2003-02	2.937,56
YTD Maand 2	**5.442,15**

Maand	Vracht
2003-01	2.504,59
2003-02	2.937,56
2003-03	4.793,47
YTD Maand 3	**10.235,62**

Afbeelding 10: YTD-cijfers.

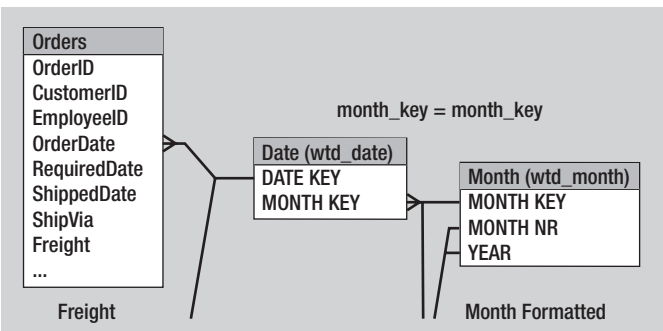
Maand	Vracht YTD
2003-01	2.504,59
2003-02	5.442,15
2003-03	10.235,62

Afbeelding 11: YTD-omzetten van de eerste drie maanden.

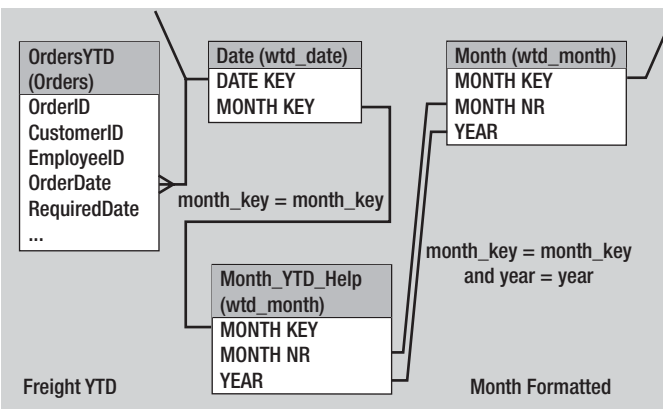


Afbeelding 12: Twee aliassen 'gejoined'.

In afbeelding 14 wordt de tweede order-alias gebruikt in combinatie met de bestaande datum- en maanddimensie. In dit geval is de datumdimensie echter niet direct verbonden met de maand-dimensie. De relatie loopt via de maand-alias Month_YTD_Help. De join tussen de datumdimensie en de YTD-help-alias is een normale join via de month_key. De join tussen de YTD_help en de Month zorgt ervoor dat voor iedere maand in de maanddimensie alle maanden in de YTD_help worden geselecteerd tot en met de maand in de maanddimensie. De gebruiker selecteert alleen de maand, de vracht en de YTD-vracht. Een goed query tool zorgt er voor dat er twee query's worden uitgevoerd. De eerste query haalt via de tabellen in de eerste order-alias de gewone vrachtcijfers per maand op.



Afbeelding 13: De eerste order-alias.



Afbeelding 14: De tweede order-alias.

Maand	Vracht YTD	Vracht
2003-01	2.504,59	2.504,59
2003-02	5.442,15	2.937,56
2003-03	10.235,62	4.793,47

Afbeelding 15: Het resultaat.

De tweede gebruikt de YTD_Help alias om de YTD-cijfers op te bouwen. Het resultaat is eenvoudig en mooi, zoals te zien is in afbeelding 15.

Door de datumdimensie op deze manier te gebruiken in de rapportage-omgeving is het mogelijk om een stuk meer functionaliteit beschikbaar te stellen aan de gebruikers van het datawarehouse. De getoonde oplossing is in alle opzichten generiek te noemen; het werkt voor alle DBMS'en, werkt zowel voor weken, maanden als voor kwartalen en is herbruikbaar voor iedere gegevensverzameling die aan het datawarehouse wordt toegevoegd.

Conclusie

De voordelen van een goede datumdimensie zijn aangetoond. Het selectieproces voor de eindgebruiker kan worden vereenvoudigd door de datumdimensie op een goede manier op te zetten. Omdat deze dimensie in alle datawarehouse-omgevingen wordt gebruikt, zijn de (minimale) investeringen van de uitbreidingen snel terugverdiend. Als de basis van een goede datumdimensie eenmaal is gelegd, wordt het ook mogelijk om met behulp van die dimensie meer complexe rapportages op te leveren, en dat op een database-onafhankelijke en generieke manier! Voldoende redenen dus om aan de slag te gaan met het motto: Doe slimme dingen met uw tijd!

Michiel Brunt (mbrunt@inergy.nl) is architect bij Inergy Analytical Solutions en consultant op het gebied van Datawarehousing en Business Intelligence.

Noot

1. Ralph Kimball, Data Warehouse Lifecycle Toolkit.