Model Klausel - Der Excel-Killer von Oracle?

Andrea Kennel
Trivadis AG
Glattbrugg, Schweiz

Schlüsselworte:

Model Klausel, SQL, Data Warehousing, OLAP

Zusammenfassung

Ein Data Mart kann als ein Würfel mit mehreren Dimensionen betrachtet werden. Dies ist auch möglich, wenn die Daten relational in einem Star Schema abgelegt sind. Was bei der Würfelbetrachtung aber interessant ist. Es können neue, berechnete Würfelzellen oder ganze Würfelscheiben zugefügt werden. So können nicht oder noch nicht vorhandene Daten simuliert werden. In Oracle ist dies mit der SQL Model Klausel möglich. Dieser Artikel ziegt, was mit dieser Model Klausel möglich ist und welche Denkart dahinter steckt.

Einleitung und Grundbegriffe

Die folgenden Beispiele bauen auf den Tabellen des SH-Schemas auf. Wir betrachten die Verkaufszahlen der Länder 'Italy' und 'Japan' über die Produkte 'Bounce' und 'Y Box' über alle vorhandenen Jahre. Dazu wird folgende View benötigt:

```
CREATE VIEW sales_view AS
SELECT
  country_name country,
  prod_name product,
  calendar_year year,
  SUM(amount_sold) sales
FROM sh.sales, sh.times, sh.customers, sh.countries, sh.products
WHERE sales.time_id = times.time_id
  AND sales.prod_id = products.prod_id
  AND sales.cust_id = customers.cust_id
  AND customers.country_id=countries.country_id
GROUP BY country_name, prod_name, calendar_year;
```

Betrachten wir die Daten als Würfel, so erhalten wir zwei zweidimensionale Würfel, für jedes Land (country) einen. Die Würfel haben eine X-Achse Zeit (year) und eine Y-Achse Produkt (product). Dies ergibt folgendes Bild:

Italy	Sales		
	1999	**2000**	**2001**
Bounce	2474.78	4333.69	4846.3
Y Box			

Japan	Sales		
	1999	**2000**	**2001**
Bounce	2961.3	5133.53	6303.6
Y Box	22161.91	45690.66	89634.83

In einem ersten Beispiel wollen wir die Daten für das Jahr 2002, die noch nicht vorhanden sind, berechnen. Weiter soll ein Total über die beiden Produkte berechnet werden
In den Würfeln sieht dies folgendermassen aus:

Italy	Sales			
	1999	**2000**	**2001**	**2002**
Bounce	2474.78	4333.69	4846.3	9179.99
Y Box	15215.16	29322.89	81207.55	110530.44
All Prod				119710.43

Japan	Sales			
	1999	**2000**	**2001**	**2002**
Bounce	2961.3	5133.53	6303.6	11437.13
Y Box	22161.91	45690.66	89634.83	135325.49
All Prod				146762.62

In SQL werden die Werte folgendermasse berechnet.

```
SELECT SUBSTR(country, 1, 20) country,
      SUBSTR(product, 1, 15) product, year, sales
FROM sales_view
WHERE country IN ('Italy', 'Japan')
  AND product IN ('Bounce','Y Box')
MODEL
  PARTITION BY (country) DIMENSION BY (product, year)
  MEASURES (sales sales)
  RULES
  (sales['Bounce', 2002] = sales['Bounce', 2001] + sales['Bounce', 2000],
   sales['Y Box', 2002] = sales['Y Box', 2001] + sales['Y Box', 2000],
   sales['All_Products', 2002] = sales['Bounce', 2002] + sales['Y Box',
2002])
```

Die PARTITION legt fest, auf welchen Datenausschnitt sich das Modell bezieht. Die Partition im Modell hat dieselbe Bedeutung wie die Partition bei den analytischen Funktionen.
Mit DIMENSION geben wir die Achsen des Würfels an, mit MEASURES legen wir fest, welche Werte gezeigt werden sollen. Der wichtigste Schritt geschieht unter RULES. Hier wird definiert, wie die neuen Zellen berechnet werden. In unserem Beispiel legen wir fest, dass für die beiden Produkte im Jahr 2002 soviel verkauft werden soll, wie in den Jahren 2001 und 2000 zusammen. Dazu definieren wir, wie die entsprechende Zelle berechnet werden soll, indem wir nach dem Measure in eckiger Klammer die Dimensionswerte angeben.
Als Resultat erhalten wir dann:

```
COUNTRY              PRODUCT              YEAR      SALES
-------------------- -------------- ---------- ----------
Italy                All_Products           2002  119710.43
Italy                Bounce                 1999    2474.78
Italy                Bounce                 2000    4333.69
Italy                Bounce                 2001    4846.30
Italy                Bounce                 2002    9179.99
Italy                Y Box                  1999   15215.16
Italy                Y Box                  2000   29322.89
Italy                Y Box                  2001   81207.55
Italy                Y Box                  2002  110530.44
Japan                All_Products           2002  146762.62
Japan                Bounce                 1999    2961.30
Japan                Bounce                 2000    5133.53
Japan                Bounce                 2001    6303.60
Japan                Bounce                 2002   11437.13
Japan                Y Box                  1999   22161.91
Japan                Y Box                  2000   45690.66
Japan                Y Box                  2001   89634.83
Japan                Y Box                  2002  135325.49
```

Arrays von Daten ansprechen

Im ersten Beispiel haben wir gesehen, dass wir beim Measure die Dimensionswerte angeben können, um uns auf eine spezifische Zelle im Würfel zu beziehen. So haben wir für 2002 die Werte von 2001 und 2000 zusammen gezählt. Wollen wir als neuen Wert den Durchschnitt über mehrere vergangenen Jahre, so ist es möglich, einen ganzen Array von Zellen anzusprechen. Im folgenden Beispiel berechnet sich der Wert für 2002 aus dem Durchschnitt der Jahre 1999 bis 2002:

```
SELECT SUBSTR(country, 1, 20) country,
      SUBSTR(product, 1, 15) product, year, sales
FROM sales_view
WHERE country IN ('Italy', 'Japan')
  AND product IN ('Bounce','Y Box')
MODEL
  PARTITION BY (country) DIMENSION BY (product, year)
  MEASURES (sales sales)
  RULES
  (sales['Bounce', 2002] = AVG(sales)['Bounce',year BETWEEN 1999 AND 2001],
   sales['Y Box', 2002] = AVG(sales)['Y Box', year BETWEEN 1999 AND 2001],
   sales['All_Products', 2002] = sales['Bounce', 2002] + sales['Y Box',
2002])
ORDER BY country, product, year
```

Bestehende Werte übersteuern

Was wäre nun, wenn wir im Jahre 2001 für Bounce nur 1000 sales erreicht hätten?
Um hier eine Antwort zu erhalten, müsste man die effektiven Quelldaten ändern. Da diese aber für andere Auswertungen unverändert bleiben müssen, geht das sicher nicht. In der Model Klausel kann ich für meine Auswertung Werte temporär verändern. Dazu stehen die Befehl UPDATE und UPSERT zur verfügung. UPDATE betrifft nur Werte, die gesetzt sind. UPSERT Setzt den Wert auch wenn er vorher nicht existierte. Wird nichts angegeben, so wird ein UPSERT ausgeführt.

```
SELECT SUBSTR(country, 1, 20) country,
      SUBSTR(product, 1, 15) product, year, sales
FROM sales_view
WHERE country IN ('Italy', 'Japan')
  AND product IN ('Bounce','Y Box')
MODEL
  PARTITION BY (country) DIMENSION BY (product, year)
  MEASURES (sales sales)
  RULES
  (UPDATE sales['Bounce', 2001] =  1000,
   UPSERT sales['Bounce', 2002] = AVG(sales)['Bounce',year BETWEEN 1999 AND
2001],
   UPSERT sales['Y Box', 2002] = AVG(sales)['Y Box', year BETWEEN 1999 AND
2001],
   UPSERT sales['All_Products', 2002] = sales['Bounce', 2002] + sales['Y
Box', 2002])
ORDER BY country, product, year;
```

```
COUNTRY              PRODUCT               YEAR      SALES
-------------------- --------------- ---------- ----------
Italy                All_Products          2002   44518.02
Italy                Bounce                1999    2474.78
Italy                Bounce                2000    4333.69
Italy                Bounce                2001    1000.00
Italy                Bounce                2002    2602.82
. . .
```

Joker

Wir haben in den beiden ersten Beispielen gesehen, wie man sich auf eine spezifische Zelle oder mit BETWEEN auf einen Array beziehen kann. Es gibt weitere Möglichkeiten, sich auf mehrere Werte zu beziehen. Einerseits kann mit IN eine Werteliste angegeben werden, oder mit ANY kann man sich auf alle möglichen Werte einer Dimension beziehen.
Will man sich nur auf den aktuellen Wert, den „Current Value“ einer Dimension beziehen, so kann mit dem Joker CV() gearbeitet werden. Dazu auch ein Beispiel:

```
SELECT SUBSTR(country, 1, 20) country,
      SUBSTR(product, 1, 15) product, year, sales
FROM sales_view
WHERE country IN ('Italy', 'Japan')
  AND product IN ('Bounce','Y Box')
MODEL
  PARTITION BY (country) DIMENSION BY (product, year)
  MEASURES (sales sales)
  RULES
  (sales['Bounce', NULL] = SUM(sales)['Bounce',ANY],
   sales['Y Box', NULL] = SUM(sales)['Y Box', ANY],
   sales['All_Products', NULL] = sales['Bounce', CV(year)] + sales['Y Box',
CV(year)])
ORDER BY country, product, year
```

Für die Berechnung von sales['Bounce', NULL] berechnen wir die Summe der Sales-Werte dieses Produktes für alle Jahre: SUM(sales)['Bounce',ANY].

Der Wert für alle Produkte ist sales['All_Products', NULL]. Zur Berechnung nehmen wir die Sales-Werte des Produktes Bounce des gleichen Jahres und des Produktes Y-Box des gleichen Jahres: sales['Bounce', CV(year)] + sales['Y Box', CV(year)].

Neue Kennzahlen definieren

Bisher haben wir als Measure, also als Kennzahl, nur sales gehabt. Aus dieser Kennzahl können wir eine neue Kennzahl sales_netto berechnen. Sales_netto ist für unser Beispiel 80% von sales.

Um eine neue Kennzahl zu erhalten, müssen wir diese zuerst als MEASURES definieren. Dazu verwenden wir die Cast-Funktion: `CAST(NULL AS NUMBER) sales_netto`.

Weiter müssen wir bei den RULES angebe, wie sich dieser Wert berechnet:

`sales_netto[ANY,ANY] = sales[CV(product),CV(year)]*0.8`

An diesem Beispiel sehen wir auch gut, wie die Jokern ANY und CV sinnvoll eingesetzt werden können.

```
SELECT SUBSTR(country, 1, 20) country,
      SUBSTR(product, 1, 15) product, year, sales, sales_netto
FROM sales_view
WHERE country IN ('Italy', 'Japan')
  AND product IN ('Bounce','Y Box')
MODEL
  PARTITION BY (country) DIMENSION BY (product, year)
  MEASURES (sales sales, CAST(NULL AS NUMBER) sales_netto)
  RULES
  (sales_netto[ANY,ANY] = sales[CV(product),CV(year)]*0.8)
ORDER BY country, product, year
```

Bezug auf andere Würfel

Der Nettowert wird wohl nicht jedes Jahr gleich berechnet werden. Gehen wir davon aus, dass die Reduktion zur Berechnung des Nettowertes in einem separaten Würfel (Tabelle) abgelegt ist.

```
select * from reduction;

      YEAR  REDUCTION
---------- ----------
      1999         .1
      2000         .5
      2001         .2
```

Diese Tabelle kann nun als Model referenziert werden. Referenziert Modelle können dann für Berechnungen im Main-Model eingesetzt werden:

```
SELECT SUBSTR(country, 1, 20) country,
      SUBSTR(product, 1, 15) product, year, sales, sales_netto
FROM sales_view
WHERE country IN ('Italy', 'Japan')
  AND product IN ('Bounce','Y Box')
MODEL
  REFERENCE reduction ON (SELECT year, reduction FROM reduction)
  DIMENSION BY (year) MEASURES (reduction)
MAIN sales_view
  PARTITION BY (country) DIMENSION BY (product, year)
  MEASURES (sales sales, CAST(NULL AS NUMBER) sales_netto)
  RULES
  (sales_netto[ANY,ANY] =
   sales[CV(product),CV(year)] * (1-reduction[CV(year)]))
ORDER BY country, product, year
```

COUNTRY	PRODUCT	YEAR	SALES	SALES_NETTO
Italy	Bounce	1999	2474.78	2227.30
Italy	Bounce	2000	4333.69	2166.85
Italy	Bounce	2001	4846.30	3877.04
Italy	Y Box	1999	15215.16	13693.64
Italy	Y Box	2000	29322.89	14661.45
Italy	Y Box	2001	81207.55	64966.04
Japan	Bounce	1999	2961.30	2665.17
Japan	Bounce	2000	5133.53	2566.77
Japan	Bounce	2001	6303.60	5042.88
Japan	Y Box	1999	22161.91	19945.72
Japan	Y Box	2000	45690.66	22845.33
Japan	Y Box	2001	89634.83	71707.86

Diese Abfrage wäre natürlich auch mit einem ganz normalen Join möglich. Dabei sind beide Möglichkeiten etwa gleich schnell und haben fast denselben Ausführungsplan. Will man aber für unterschiedliche Jahre unterschiedliche Regeln anwenden, so ist dies mit der Model Klausel sicher etwas übersichtlicher.

```
SELECT SUBSTR(s.country, 1, 20) country,
      SUBSTR(s.product, 1, 15) product, s.year, s.sales,
      s.sales * (1 - r.reduction) sales_netto
FROM sales_view s inner join reduction r on (s.year = r.year)
WHERE country IN ('Italy', 'Japan')
  AND product IN ('Bounce','Y Box')
ORDER BY country, product, year
```

Bezug auf frühere Jahre

Der Bezug auf das Vorjahr ist sehr einfach, indem ich mich auf `CV(year) – 1` beziehe. Im folgenden Beispiel enthält sales_old die Daten des Vorjahres:

```
SELECT SUBSTR(country, 1, 20) country,
      SUBSTR(product, 1, 15) product, year, sales, sales_old
FROM sales_view
WHERE country IN ('Italy', 'Japan')
  AND product IN ('Bounce','Y Box')
MODEL
  PARTITION BY (country) DIMENSION BY (product, year)
  MEASURES (sales sales, 0 sales_old)
  RULES
  (sales_old[ANY,ANY] = sales[CV(product),CV(year) - 1])
ORDER BY country, product, year
```

```
COUNTRY              PRODUCT               YEAR      SALES  SALES_OLD
-------------------- --------------- ---------- ---------- ----------
Italy                Bounce                1999    2474.78
Italy                Bounce                2000    4333.69    2474.78
Italy                Bounce                2001    4846.30    4333.69
Italy                Y Box                 1999   15215.16
Italy                Y Box                 2000   29322.89   15215.16
Italy                Y Box                 2001   81207.55   29322.89
Japan                Bounce                1999    2961.30
Japan                Bounce                2000    5133.53     2961.3
Japan                Bounce                2001    6303.60    5133.53
Japan                Y Box                 1999   22161.91
Japan                Y Box                 2000   45690.66   22161.91
Japan                Y Box                 2001   89634.83   45690.66
```

Folgende analytische Funktion ergibt dasselbe Resultat mit der gleichen Laufzeit:

```
SELECT SUBSTR(country, 1, 20) country,
      SUBSTR(product, 1, 15) product, year, sales,
      LAG(sales) OVER (PARTITION BY country, product ORDER BY year)
sales_old
FROM sales_view
WHERE country IN ('Italy', 'Japan')
  AND product IN ('Bounce','Y Box')
```

Mehrere Zellen setzen mit FOR

Bisher haben wir in RULES immer nur eine spezifische Zelle berechnet. Dazu haben wir links der Gleichung eine Zelle angesprochen. Mit dem Befehl FOR können mehrere Zellen auf einmal gefüllt werden.

Ich möchte die Werte die sales-Werte für 2002 für beide Produkte in einem Schritt berechnen. Dazu brauche ich den Befehl FOR und eine Liste mit IN:

```
SELECT SUBSTR(country, 1, 20) country,
      SUBSTR(product, 1, 15) product, year, sales
FROM sales_view
WHERE country IN ('Italy', 'Japan')
  AND product IN ('Bounce','Y Box')
MODEL
  PARTITION BY (country) DIMENSION BY (product, year)
  MEASURES (sales sales)
  RULES
  (sales[FOR product IN ('Bounce','Y Box'), 2002] =
     sales[CV(product), 2001] + sales[CV(product
   sales['All_Products', 2002] =
     sales['Bounce', 2002] + sales['Y Box', 2002])
ORDER BY country, product, year
```

Im nächsten Beispiel gehen wir einen Schritt weiter. Wir berechnen für beide Produkte neue Werte für die Jahre 2002, 2003 und 2004. Für die Berechnung der Werte gehen wir von einem Wachstum von 5% aus.

```
SELECT SUBSTR(country, 1, 20) country,
      SUBSTR(product, 1, 15) product, year, sales
FROM sales_view
WHERE country IN ('Italy', 'Japan')
  AND product IN ('Bounce','Y Box')
MODEL
  PARTITION BY (country) DIMENSION BY (product, year)
  MEASURES (sales sales)
  RULES
  (sales[FOR product IN ('Bounce','Y Box'), FOR year IN (2002,2003,2004)]
   = sales[CV(product), CV(year)-1] *1.05,
   sales['All_Products', FOR year IN (2002,2003,2004)] = sales['Bounce',
CV(year)] + sales['Y Box
ORDER BY country, product, year
```

Kombination mit analytischen Funktionen

In der Kombination mit analytischen Funktionen gilt folgende Einschränkung:
Im SELECT oder ORDER BY ist keine analytische Funktion erlaubt.
In der Model Klauser selber, also im PARTITION, MEASURES oder RULES sind analytische Funktionen aber erlaubt. Sie können dort mit einem Alias versehen werden, das dann im SELECT oder ORDER BY verwendet werden kann.
Nachfolgend ein Beispiel, das diese Möglichkeiten zeigt. Was wird gemacht? Mit der Analytischen Funktion LAG definieren wird unter MEASURES die neue Kennzahl sales_last_year. Die noch nicht definierten Zellen werden dann unter RULES noch definiert. Im SELECT sprechen wir diese Kennzahl über ihr Alias an. Weiter berechnen wir das jährliche Wachstum, indem wir die beiden Kennzahlen sales und sales_last_year kombinieren.
Da wir für die Jahre 2002 bis 2004 ein Wachstum von 5% angenommen haben, muss diese Auswertung für diese Jahre natürlich ein Wachstum von 5% ausweisen, was auch geschieht.

```
SELECT SUBSTR(country, 1, 20) country,
      SUBSTR(product, 1, 15) product, year, sales,
      sales_last_year,
      (sales-sales_last_year)/sales_last_year * 100 grow
FROM sales_view
WHERE country IN ('Italy', 'Japan')
  AND product IN ('Bounce','Y Box')
MODEL
  PARTITION BY (country) DIMENSION BY (product, year)
  MEASURES (sales sales,
            LAG(sales) OVER (PARTITION BY country, product
                             ORDER BY year asc) sales_last_year)
  RULES
  (sales[FOR product IN ('Bounce','Y Box'), FOR year IN (2002,2003,2004)]
   = sales[CV(product), CV(year)-1] *1.05,
   sales['All_Products', FOR year IN (2002,2003,2004)] =
     sales['Bounce', CV(year)] + sales['Y Box
   sales_last_year[FOR product IN ('Bounce','Y Box'),
                   FOR year IN (2002,2003,2004)]  =
     sales[CV(product), CV(year)-1],
   sales_last_year['All_Products', FOR year IN (2002,2003,2004)] =
     sales['Bounce', CV(year)-1] + sales['Y Box', CV(year)-1])
ORDER BY country, product, year
```

COUNTRY	PRODUCT	YEAR	SALES	SALES_LAST_YEAR	GROW
Italy	All_Products	2002	90356.54	86053.85	5.00
Italy	All_Products	2003	94874.37	90356.54	5.00
Italy	All_Products	2004	99618.09	94874.37	5.00
Italy	Bounce	1999	2474.78		
Italy	Bounce	2000	4333.69	2474.78	75.11
Italy	Bounce	2001	4846.30	4333.69	11.83
Italy	Bounce	2002	5088.62	4846.30	5.00
Italy	Bounce	2003	5343.05	5088.62	5.00
Italy	Bounce	2004	5610.20	5343.05	5.00
Italy	Y Box	1999	15215.16		
Italy	Y Box	2000	29322.89	15215.16	92.72
Italy	Y Box	2001	81207.55	29322.89	176.94
Italy	Y Box	2002	85267.93	81207.55	5.00
Italy	Y Box	2003	89531.32	85267.93	5.00
Italy	Y Box	2004	94007.89	89531.32	5.00
Japan	All_Products	2002	100735.35	95938.43	5.00
Japan	All_Products	2003	105772.12	100735.35	5.00
Japan	All_Products	2004	111060.73	105772.12	5.00
Japan	Bounce	1999	2961.30		
Japan	Bounce	2000	5133.53	2961.30	73.35
Japan	Bounce	2001	6303.60	5133.53	22.79
Japan	Bounce	2002	6618.78	6303.60	5.00
Japan	Bounce	2003	6949.72	6618.78	5.00
Japan	Bounce	2004	7297.20	6949.72	5.00
Japan	Y Box	1999	22161.91		
Japan	Y Box	2000	45690.66	22161.91	106.17
Japan	Y Box	2001	89634.83	45690.66	96.18
Japan	Y Box	2002	94116.57	89634.83	5.00
Japan	Y Box	2003	98822.40	94116.57	5.00
Japan	Y Box	2004	103763.52	98822.40	5.00

Fazit

Mit der SQL Model Klausel hat Oracle eine SQL-Erweiterung eingeführt, die sehr vielseitig und hilfreich ist, wenn in multidimensionalen Abfragen zusätzliche Werte berechnet werden müssen. Im Gegensatz zu Excel kennt Oracle keine Mengenbeschränkung. Durch die Möglichkeit der Joker wie ANY und CV sind die Auswertemöglichkeiten klar flexibler als in Excel. Dazu kommt, dass die Formeln alle gesammelt unter dem Schlüsselwort RULES definiert sind. Dies erleichtert die Lesbarkeit.
Kurz kann SQL Model Klause mit den Worten verständlich, flexibel, mächtig und schnell beschrieben werden.

Literatur

- Oracle Database Data Warehousing Guide, 10g Release 1 (10.1), Part No. B10736-01, Chapter 22

Kontaktadresse:

Andrea Kennel
Trivadis AG
Europa-Strasse 5
CH-8152 Glattbrugg

Telefon: +41(0)1-808 70 20
Fax: +49(0)1-808 70 21
E-Mail andrea.kennel@trivadis.com
Internet: www.trivadis.com