DeltaMaster clicks
156

BISSANTZ

November/Dezember 2017

Standardauswahl in relationalen Anwendungen dynamisch bestimmen

Liebe Datenanalysten,

zu den großen Neuerungen in DeltaMaster 6 gehört ein wesentlich erweiterter Unterbau für relationale Anwendungen. Diese benötigen keinen Datenwürfel (Cube), sondern kommen mit den Tabellen einer relationalen Datenbank aus, zum Beispiel Microsoft SQL Server, Oracle, SAP HANA, IBM DB2 oder MySQL. Dem Management, den Berichtsempfängern, wird das oft gar nicht auffallen, denn das meiste funktioniert so, wie man DeltaMaster eben kennt. Unter der Oberfläche gibt es natürlich Unterschiede, schon sprachlich, denn die relationalen Datenbanken werden mit SQL abgefragt statt mit MDX. Das wirkt sich auch etwa auf die dynamische Konfiguration des Filterkontexts aus. In der letzten Ausgabe hatten wir diese für OLAP beschrieben, in dieser Ausgabe geht es um relationale Anwendungen – und es geht technischer zu: Wir wenden uns an Berichtsredakteure, die solche Anwendungen einrichten. Das Ergebnis ist ganz und gar nicht technisch, sondern nur ein weiterer Baustein in einem automatisierten Reporting: nämlich individuelle Voreinstellungen, wie die Anwender sie brauchen.

Herzliche Grüße
Ihr Team von Bissantz & Company

Auch in relationalen Anwendungen lässt sich eine Standardauswahl (Default-Auswahl) definieren. Die Wirkung im Präsentationsmodus ist die gleiche wie bei OLAP-Datenbanken und wie in den DeltaMaster clicks 155 beschrieben: Beim ersten Öffnen eines Berichts führt DeltaMaster die Abfragen für die Default-Auswahl aus und setzt in jeder Dimension die entsprechenden Filter (Sicht). Damit wird es möglich, Filter automatisch vorzubelegen, auch nach dynamischen Kriterien, insbesondere abhängig vom aktuellen Datum, vom aktuellen Benutzer oder von den verfügbaren Daten.

Online-Hilfe
Direkt in DeltaMaster:
Menü ⓘ/*Hilfe* oder Taste *F1*

Support-Hotline
support@bissantz.de
Tel. +49 911 935536-700

DeltaMaster clicks
Archiv und Abo:
clicks.bissantz.de

Kundenportal
Aktuelle DeltaMaster-Version:
www.bissantz.de/login

Blogs
Bissantz denkt nach
blog.bissantz.de

Bella berät – die meisten Diagramme sind für die Katz
bella-beraet.de

Auf die Würfel, fertig, los – wie wir Ihren Daten Beine machen
crew.bissantz.de

Bissantz forscht –
Neues aus unseren Laboren
forschung.bissantz.de

Schulungen
Über 60 Schulungstage rund um DeltaMaster und Microsoft SQL Server/Analysis Services.
www.bissantz.de/trainings

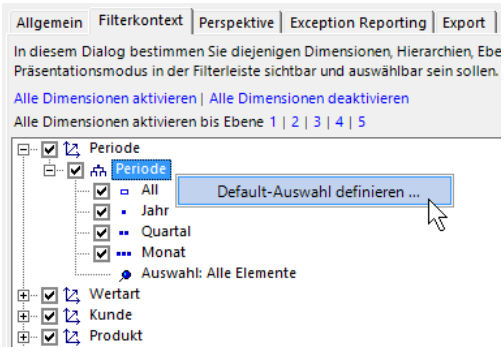
Veranstaltungen
Erleben Sie DeltaMaster live – zum Beispiel auf Kundentreffen, Fachseminaren, Informationstagen, Kongressen oder Messen.
www.bissantz.de/events

Andere Sprachen, andere Strukturen

Unterschiede gibt es in der Syntax, aber auch konzeptionell. Denn anders als in MDX-Datenbanken kennen SQL-Datenbanken keine Hierarchien, keine Ebenen und damit keine standardisierten Ausdrucksmöglichkeiten, um hierarchische Strukturen zu beschreiben. Gerade auf die kommt es aber an, gerade die benötigt man in der Managementinformation, um Erkenntnisse aus Daten zu ziehen! DeltaMaster schließt diese Lücke und bietet Mechanismen an, um multidimensionale Denk- und Arbeitsweisen auch mit relationalen Datenbanken umzusetzen. Das geschieht mit SQL und einigen zusätzlichen Regeln, nach denen DeltaMaster das relationale Abfrageergebnis in die gewünschten multidimensionalen Konstrukte überträgt.

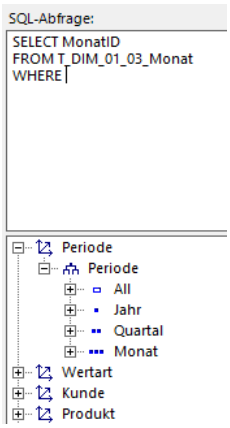
Berichtseigenschaften, Filterkontext

Die Default-Auswahl gehört zum *Filterkontext*. Dieser lässt sich auf einer eigenen Registerkarte in den *Berichtseigenschaften* (Kontextmenü von Berichten im Bearbeitungsmodus) bearbeiten, wie bei MDX-Datenbanken. Über das Kontextmenü einer Hierarchie können Sie die *Default-Auswahl definieren*.

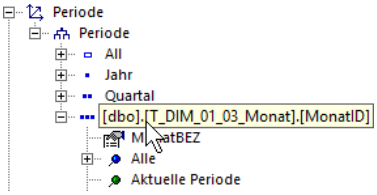


Dazu öffnet sich der bekannte *SQL-Editor*. Im oberen Teil des Dialogs geben Sie die Abfrage ein, im unteren Teil sind alle Dimensionen, Hierarchien, Ebenen und Elemente sowie die Analysewerte aufgeführt.

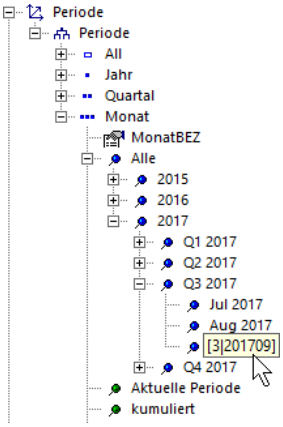
Wenn Sie bei gedrückter *Alt*-Taste mit der Maus auf einen Eintrag zeigen, blendet DeltaMaster in einem Tooltipp technische Informationen ein. Diese werden in SQL-Abfragen benötigt und lassen sich per Drag-and-drop oder per Doppelklick in das Eingabefeld übernehmen.



Für Ebenen zeigt DeltaMaster den qualifizierten Namen des Schlüsselfelds an. Dieser besteht aus dem Schema (in der Abbildung: dbo), dem Tabellennamen (T_DIM_01_03_Monat) und dem Feldnamen (MonatID).



Für Elemente zeigt DeltaMaster die Ordnungszahl der Hierarchieebene, also gewissermaßen deren Tiefe, sowie den Schlüssel an. In der Abbildung ist der Monat die Ebene 3 (die oberste Ebene ist immer 0) und der Schlüssel lautet 201709.



SQL-Abfragen

Wie die Abfragen für die Default-Auswahl zu formulieren sind, richtet sich nach der betreffenden Dimension. Wir unterscheiden

- reguläre, ebenenbasierte Dimensionen, die sich aus den Spalten einer oder mehrerer Tabellen speisen, sowie
- virtuelle Zeitdimensionen, die auf einem Datumsfeld, etwa vom Datentyp Date oder Datetime, aufbauen.

Die virtuelle Zeitdimension ist eine Spezialität von DeltaMaster. In relationalen Anwendungen erspart sie einige typische Arbeitsschritte der Datenaufbereitung: Anstatt gespeicherte Datumsangaben auf separate Spalten für Tage, Monate, Quartale, Jahre usw. zu verteilen, wie es bei einer regulären Hierarchie erforderlich wäre, kann DeltaMaster das Datumsfeld direkt verarbeiten und konstruiert selbsttätig eine dazu passende, virtuelle Hierarchie, ähnlich wie bei der sogenannten Serverzeitdimension von Analysis Services.

Für beide Arten von Dimensionen gilt: Im Abfrageergebnis kommt es nur auf die erste Spalte an, weitere Spalten werden ignoriert. Diese erste Spalte enthält die Werte, die als Elementschlüssel interpretiert werden sollen, und zwar als Zeichenketten (Strings, zum Beispiel in den Datentypen Char oder Varchar) oder als Zahlen (zum Beispiel Integer).

In den meisten Anwendungsfällen soll genau ein Element die Default-Auswahl werden, nicht mehrere. Die Abfrage muss dazu genau einen Wert zurückliefern (eine Spalte, eine Zeile). Um sicherzustellen, dass eine Abfrage genau eine Zeile zurückgibt und somit genau ein Element ausgewählt wird, können Sie die Anzahl der Ergebniszeilen begrenzen (in Microsoft SQL Server etwa mit einer Abfrage wie „SELECT TOP 1 FROM …“), falls sich dies nicht schon aus der Abfragelogik ergibt, beispielsweise aus der WHERE-Klausel.

Werden mehrere Zeilen zurückgeliefert, behandelt DeltaMaster dies als Mehrfachauswahl. In diesem Fall müssen die Elemente zur selben Dimensionsebene gehören; unterschiedliche Ebenen sind nicht möglich. Doppelte Zeilen vermeiden Sie in Microsoft SQL Server mit dem Zusatz DISTINCT („SELECT DISTINCT <Feld> FROM …“).

Reguläre Dimensionen

Wichtig für die Syntax in regulären Dimensionen ist, ob eine Hierarchieebene mitangegeben wird. Dies hängt wiederum davon ab, wie viele Ebenen die Dimension hat.

- Bei Dimensionen mit ein oder zwei Ebenen brauchen Sie die Ebene nicht anzugeben.

  Hat eine Dimension genau eine Ebene, so sucht DeltaMaster das Element ebenda. Beispiele dafür sind die Wertarten (Plan, Ist, …) oder etwa Währungen. In diesen Fällen genügt es, schlicht den Elementschlüssel zu selektieren:

  ```
  SELECT 'P' -- Element „Plan“ in der Dimension Wertart
  ```

  So könnte eine Default-Auswahl in der Dimension Wertart definiert sein; darin mag „P“ der Schlüssel für das Element Plan sein. In der Praxis würden Sie natürlich nicht ein konstantes Element selektieren, sondern dieses dynamisch ermitteln, siehe folgender Abschnitt; feststehende Elemente verwenden wir hier nur zur Veranschaulichung.

  Hat eine Dimension zwei Ebenen, eine Top- und eine Hauptebene, so sucht DeltaMaster das Element automatisch in der Hauptebene, sodass diese nicht angegeben werden muss. Auch in diesem Fall genügt also das Selektieren des Schlüssels. In unserer Referenzanwendung Chair sind die Stoffgruppen und die Vertriebsteams Beispiele für solche Dimensionen.

- Bei Dimensionen mit mehr als zwei Ebenen ist die Angabe der Ebene erforderlich.

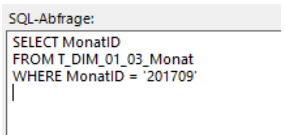
  Dazu muss in der Abfrage ein Alias zugewiesen oder das Schlüsselfeld spezifiziert werden:

  ```
  SELECT '201709' AS MonatID
  ```

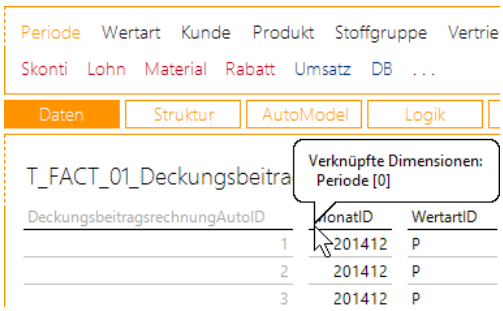
  Oder:

  ```
  SELECT MonatID
  FROM T_DIM_01_03_Monat
  WHERE MonatID = '201709'
  ```

  Den Namen des Schlüsselfelds (in beiden Beispielen: MonatID) sowie den Tabellennamen (im unteren Beispiel: T_DIM_01_03_Monat) ermitteln Sie im *SQL-Editor*, wie oben beschrieben (*Alt*+Mauszeiger, Übernehmen per Drag-and-drop oder Doppelklick).



Um Abfragen zu formulieren, ist es manchmal hilfreich, einen Blick in die Daten zu werfen. Die Tabellen mit ihren Schlüsselfeldern und den Ausprägungen können Sie beim *Modellieren* auf der Seite *Daten* einsehen. Der groß geschriebene Tabellenname (T_FACT_01_Deckungsbeitrag) dient zugleich als Menü, mit dem Sie zu den anderen Tabellen umschalten können. In Spaltenköpfen gibt ein Tooltipp an, wie die Spalte verwendet wird (*Alt*+Mauszeiger).



Datum, Benutzer, Analysewerte

In den obigen Beispielen haben wir konstante Schlüssel angegeben, um die Darstellung zu vereinfachen – Sinn und Zweck der Default-Auswahl ist es aber ja, den Schlüssel dynamisch zu bestimmen, in Abhängigkeit von veränderlichen Bedingungen. Besonders oft benötigt man das Systemdatum und den aktuellen Benutzernamen. In Microsoft SQL Server erhält man diese Informationen mit den Funktionen GETDATE() bzw. SYSTEM_USER, in anderen Datenbanken gibt es Ähnliches. Die Rückgabewerte sind so in Zeichenketten umzuformen, dass sie den Elementschlüsseln in der betreffenden Dimension entsprechen.

Beispiele:

```
SELECT CAST(YEAR(GETDATE()) AS VARCHAR) +
RIGHT('0' + CAST(MONTH(GETDATE()) AS VARCHAR), 2)
AS MonatID -- für Periodenbezeichnungen wie „201709“ als Zeichenkette

SELECT SYSTEM_USER AS BenutzerID -- einschließlich Domäne

SELECT SUBSTRING(SYSTEM_USER,
CHARINDEX('\', SYSTEM_USER) + 1, LEN(SYSTEM_USER)) AS BenutzerID -- ohne Domäne
```

Die letzten beiden Beispiele setzen voraus, dass die Benutzer als eigene Dimension modelliert sind, sodass man das einzustellende Element direkt aus dem Namen des Systembenutzers übernehmen kann. Häufig ist stattdessen eine Zuordnung gefragt, zum Beispiel zwischen Benutzername und Vertriebsteam, Produkt- oder Kundengruppe. In einfachen Fällen, mit wenigen Zuordnungen, können diese in der Default-Auswahl erfasst werden, etwa so:

```
SELECT CASE
WHEN SYSTEM_USER = 'chair\baumann' THEN 'V2'
WHEN SYSTEM_USER = 'chair\hohlmaier' THEN 'V1'
END
AS VertriebID
```

Für größere Anwendungen empfiehlt es sich, solche Zuordnungen in der Datenbanktabelle zu hinterlegen.

Auch das Einbeziehen von Analysewerten kann die Default-Auswahl dynamisch machen. So ermittelt die folgende Abfrage den letzten Monat, für den bereits ein (positiver) Ist-Umsatz erfasst wurde. Dadurch passt sich die Vorauswahl in der Periodendimension fortlaufend an die Bewegungsdaten an.

```
SELECT TOP 1 MonatID
FROM T_FACT_01_Deckungsbeitragsrechnung
WHERE WertartID = 'I' AND Umsatz > 0
ORDER BY MonatID DESC
```

In dieser Abfrage greift DeltaMaster auf die Fakttabelle zu. Diese ist beim *Modellieren* einzusehen, wie oben gezeigt.

Virtuelle Zeitdimension

Bei einer virtuellen Zeitdimension braucht DeltaMaster immer einen Hinweis, welche Ebene (Zeitgranularität) gemeint ist, zum Beispiel Tag oder Monat. Mit SQL lässt sich das nicht ausdrücken, daher unterstützt DeltaMaster eine eigene Syntax: Vor dem Datum im Textformat ist mit einem Delta-

Master-spezifischen Code die gewünschte Ebene anzugeben, gefolgt von einem Hochkomma als Trennzeichen. Beispiel:

```
SELECT '3''04/28/2017 00:00:00'
```

Diese Abfrage stellt als Default-Auswahl in der Periodendimension den April 2017 ein, ausgehend vom Datum 28. April 2017. Intern arbeitet DeltaMaster mit einer kulturunabhängigen Darstellung des Datums (Invariant Culture). Diese ist nicht regions- oder benutzerspezifisch und verwendet das amerikanische Datumsformat „MM/TT/JJJJ“, wie im Beispiel. Das erste und das letzte Apostroph sind die Begrenzungszeichen (Delimiter) für Zeichenketten in SQL. Der Code 3 bedeutet, dass der Monat gemeint ist. Das Apostroph hinter dem Code ist doppelt angegeben, weil es innerhalb der Begrenzungszeichen steht und daher maskiert werden muss (Escape).

DeltaMaster kennt die folgenden Codes für die Zeitgranularität:

Code	Ebene
0	Jahr
2	Quartal
3	Monat
4	Woche
5	Tag

In der Anwendung kombinieren Sie die Codes natürlich nicht mit einem festen Datum, sondern mit einer dynamischen SQL-Funktion wie GETDATE(), um das aktuelle Tagesdatum bzw. die Systemzeit des Rechners zu verarbeiten. Die Abfragen sehen dann aus wie diese:

```
SELECT '5''' + CONVERT(CHAR(10), GETDATE(), 101) + ' 00:00:00' -- Tag auswählen

SELECT '3''' + CONVERT(CHAR(10), GETDATE(), 101) + ' 00:00:00' -- Monat auswählen
```

Aufgrund des internen Aufbaus der virtuellen Zeitdimension kann das Datum leicht aus der Systemzeit abgeleitet werden, daher fällt die Abfrage kürzer aus als im Beispiel aus dem vorherigen Abschnitt: Anstatt Zeichenketten zusammenzusetzen, kann man die Systemzeit mit der Funktion CONVERT in das benötigte Format umwandeln; in Microsoft SQL Server hat dieses den Code 101. Um die Abfrage unabhängig von der Tageszeit zu machen, werden nur die ersten zehn Stellen der Systemzeit berücksichtigt, also das Datum, und eine „neutrale“ Uhrzeit angehängt.

Ein Alias, Feldbezeichnungen, Tabellennamen oder Ähnliches sind nicht erforderlich, da die virtuelle Zeitdimension nicht auf verschiedenen Datenbankfeldern basiert.