

Helmut Vonhoegen

Neue Cube-Funktionen in Excel 2007

Vorabdruck eines Kapitels aus einem im Herbst erscheinenden Buches zu den Excel-Funktionen bei Galileo-Press

Ganz neu in Excel 2007 ist die Gruppe der Cube-Funktionen. Dabei geht es um die Möglichkeit, von Excel aus auf Daten zuzugreifen, die in mehrdimensionalen Datenstrukturen auf einem Datenbank-Server zusammengestellt sind, die **Cubes** genannt werden.

Solche "Datenwürfel" fassen Daten nach ganz unterschiedlichen Dimensionen zusammen: personenbezogene, sachliche, räumliche oder zeitliche. Dadurch wird es möglich, Querbezüge zwischen den Daten herzustellen und Fragen zu beantworten wie: Welche Artikel werden von einer bestimmten Altersgruppe besonders bevorzugt? Entwickeln sich die Verkäufe in allen Vertriebsgebieten gleich stark oder gibt es regionale Unterschiede? Welche Produktlinie liegt im Trend? Außerdem besteht die Möglichkeit, bestimmte Indikatoren einzurichten, die kritische Werte kontrollieren oder frühzeitig auf Fehlentwicklungen oder auf besondere Chancen aufmerksam machen.

Der Einsatz von Cubes ähnelt dem Einsatz von Pivottabellen in Excel, und es ist deshalb auch nicht verwunderlich, dass OLAP-Objekte direkt als Datenquelle für Pivottabellen verwendet werden können. Wenn es sich aber um sehr große Datenmengen handelt, ist es häufig sinnvoll, das Pivotieren schon auf dem Datenbankserver vorzunehmen oder wenigstens vorzubereiten, sodass die Weiterverarbeitung in Excel anschließend mit kleineren Datenmengen vorgenommen werden kann.

Anders als bei der Ausgabe von Cube-Daten in einer Pivottabelle erlauben nun die Cube-Funktionen auch eine freie Platzierung der aus dem Cube übernommenen Daten in einem Arbeitsblatt. Sie ergänzen insofern die Möglichkeiten, die die Pivottabellen bieten.

Voraussetzungen für den Einsatz von Cube-Funktionen

Um die Cube-Funktionen nutzen zu können, müssen allerdings einige Voraussetzungen geschaffen werden, die hier zunächst kurz skizziert werden sollen. Die SQL Server von Microsoft unterstützen seit der Version 7 mit den **Server Analysis Services** das **Online Analytical Processing**, kurz **OLAP**. Dabei handelt es sich um Analysemethoden für die Auswertung von umfangreichen Unternehmensdaten. Im Kern geht es dabei darum, über die bei den relationalen Datentabellen zwangsläufig vorgegebene Zweidimensionalität hinauszukommen. Dazu werden, getrennt von den normalen Datentransaktionen in den SQL Datenbanken, vorhandene Daten aus relationalen Datentabellen in einer Art Meta-Datenbank zu mehrdimensionalen Datenstrukturen zusammengeführt, für die die Bezeichnung **Cube** verwendet wird.

OLAP-Cubes

Für OLAP-Lösungen ist typisch, dass sie Informationen aus mehreren relationalen Datenquellen zusammenführen, um Datenbestände unter flexibel wählbaren Gesichtspunkten zusammenfassen (aggregieren) und auswerten zu können.

2006	Name	Bestandssummen	Mindestbestandssummen
	Bordeaux	260	200
	Bourgogne	230	170
2005	Name	Bestandssummen	Mindestbestandssummen
	Barolo	300	200
	Beaujolais	200	220
2004	Name	Bestandssummen	Mindestbestandssummen
	Barolo	300	200
	Beaujolais	200	220
	Brolio	230	200
	Chianti	120	100
	Freisa	120	100
	Grignolino	230	250
	Médoc	300	250
	Ortenau	340	300
	Ravello	200	150
	Valpolicella	300	250
	Gesamtergebnis	2340	2020

Kleiner Datenwürfel, bei dem die Zeit die dritte Dimension darstellt

Das technische Konzept für die OLAP-Datenbank basiert dabei auf dem Modell der Cubes, der mehrdimensionalen Datenwürfel. Natürlich handelt es sich bei diesem Begriff um ein Bild, das allerdings bei dreidimensionalen Datenstrukturen noch gut grafisch dargestellt werden kann, wie die Abbildung andeutet:

Business Intelligence Development Studio

Für den Umgang mit OLAP und Cubes steht als Teil einer professionellen SQL Server 2005-Installation mit **Business Intelligence Development Studio** eine spezielle Variante der Entwicklungsumgebung **Microsoft Visual Studio 2005** zur Verfügung, die entsprechende Vorlagen für **BI-Projekte** anbietet und umfangreiche Tools für ihr Design. Im Zentrum steht dabei der **Cube-Designer**. Es soll an dieser Stelle wenigstens angedeutet werden, welche Konzepte bei der Erstellung von Cubes in der Visual Studio-Umgebung maßgeblich sind, weil so die Syntax der im Folgenden beschriebenen Funktionen verständlich wird.

Dimensionen

Beim Entwurf von Cubes werden insbesondere die Begriffe **Dimension** und **Measure** unterschieden. Dimensionen sind maßgeblich für die Anordnung der Daten. Dabei kann es sich um sachliche Kriterien handeln, etwa die Anordnung von Produkten, wobei verschiedene Warengruppen oder Produktlinien als Zwischenebene dieser Dimension eingerichtet werden können. Eine zweite Dimension könnte die räumliche Verteilung der jeweiligen Geschäftstätigkeiten, eine dritte die zeitliche Verteilung darstellen, wobei wiederum mehrstufige Hierarchien von Elementen möglich sind, etwa Jahre <- Quartale <- Monate <- Wochen. Eine weitere Dimension könnte die personelle Anordnung sein: Kunden, Lieferanten etc.

Measures

Der Begriff **Measure**, der in den deutschen Texten manchmal auch als **Kennzahl** oder **Maßzahl** wiedergegeben wird, bezeichnet die Werte, auf die aus der Perspektive der

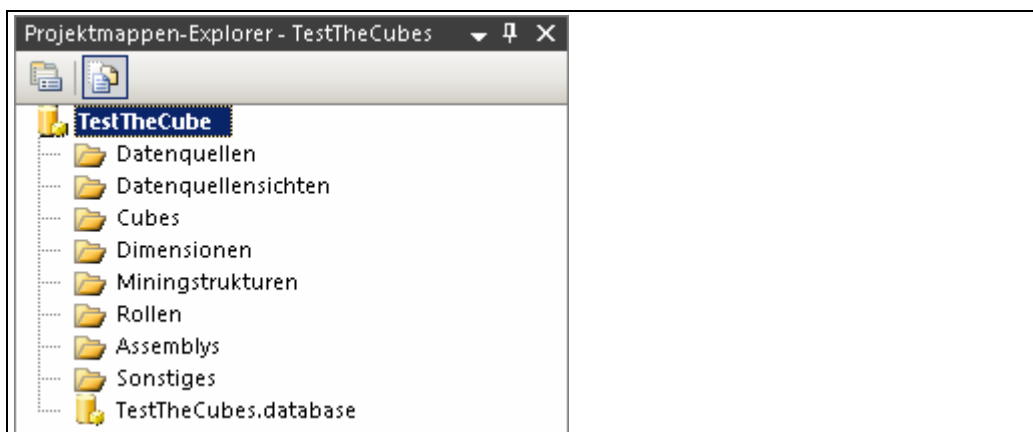
jeweiligen Dimensionen zugegriffen wird. In der letzten Abbildung sind es beispielsweise die Bestands- und Mindestbestandszahlen. Diese Werte können als Einzelwerte oder schon als aggregierte Werte vorliegen.

Die als **Measures** bezeichneten Werte werden innerhalb der **Analysis Services** von **SQL Server 2005** als Faktentabellen geführt, während die Elementwerte jeder Dimension jeweils in Dimensionstabellen vorgehalten werden. Die Verknüpfung erfolgt über Schlüssel, wobei der Primärschlüssel aus einer Dimensionstabelle als Fremdschlüssel in der Faktentabelle auftaucht. Dabei sind verschiedene Strukturschemas möglich, die einfachste Struktur wird als Sternschema bezeichnet. Dabei liegt für jede Dimension eine Tabelle vor und alle Dimensionstabellen sind mit der Faktentabelle verknüpft. Neben Dimensionen und Measures, die direkt aus vorhandenen Tabellen verfügbar sind, lassen sich zusätzlich berechnete Dimensionen und Measure erzeugen.

Ausnahmsweise wird in den folgenden Abschnitten auf die von Microsoft zur Verfügung gestellte Beispieldatenbank **Adventure Works DW** zurückgegriffen, um nicht zu viel Zeit mit der Beschreibung der Einrichtung von eigenen Datentabellen zu verlieren. Diese Beispieldaten sind komplex genug, um erste Erfahrungen im Einsatz von Cube-Funktionen zu gewinnen. Sie müssen allerdings zunächst separat installiert werden. Sie lassen sich im Microsoft Download Center über den Link **SQL Server 2005 Samples and Sample Databases** herunterladen. Die Installation wird per Doppelklick auf **SqlServerSamples.msi** gestartet.

Ein Cube-Projekt anlegen

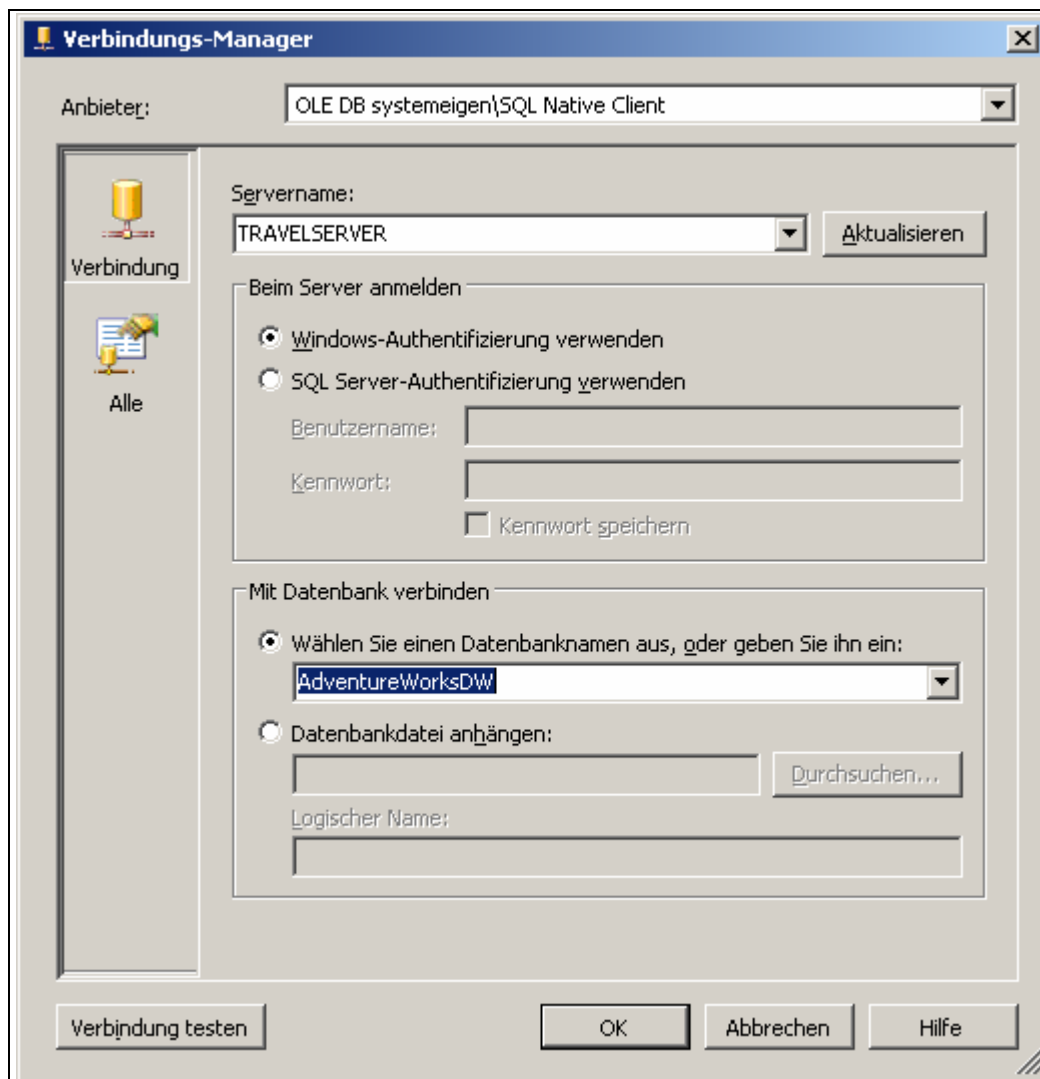
Ist die Datenbank installiert, kann über das Tool **SQL Server Business Intelligence Development Studio** ein Projekt mit dem Projekttyp **Business Intelligence-Projekte** angelegt werden, um darin einen Cube zu definieren und bereitzustellen, auf den die unten beschriebenen Cube-Funktionen zugreifen können. Im **Projektmappen-Explorer** erscheint ein vorgegebener Ordnerbaum für die Komponenten eines solchen Projekts.



Komponenten für ein Cube-Projekt

Die Abfolge bis zur Bereitstellung eines Cubes soll hier nur angedeutet werden. Der erste Schritt ist in der Regel die Definition mindestens einer Datenquelle, die das Material für den vorgesehenen Cube liefert. Dabei wird die **Verbindungszeichenfolge**

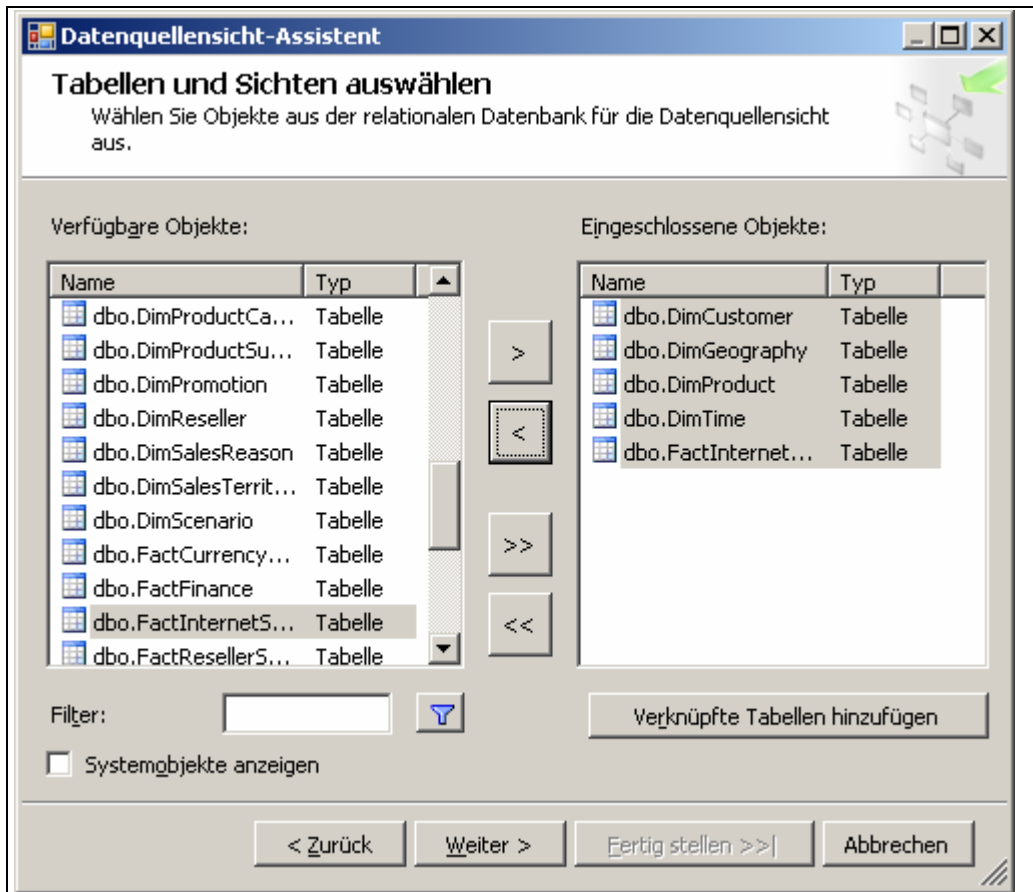
festgelegt, die bei der Eröffnung der Verbindung benötigt wird. Im Dialog **Verbindungs-Manager** wird unter **Anbieter** der Datenprovider abgefragt. Hier wird die Einstellung **OLE DB systemeigen\SQL Native Client** übernommen.



Herstellen der Verbindung zur Beispieldatenbank

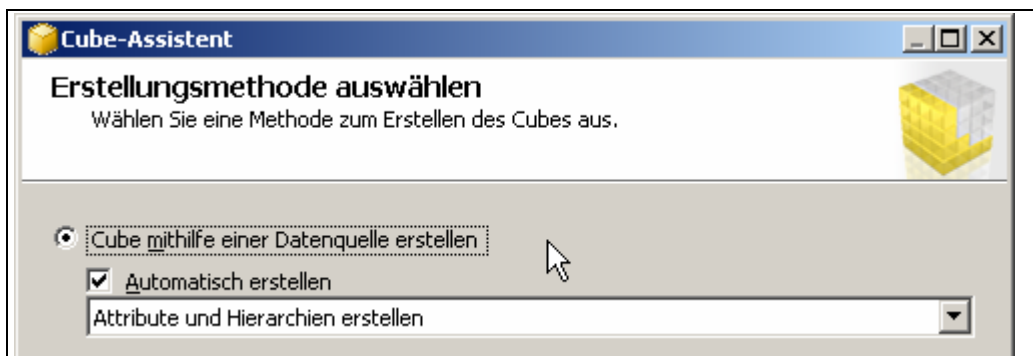
Unter **Servername** kann der Computername, localhost oder die IP-Adresse eingegeben werden. Für das Anmeldeverfahren wird die Windows-Authentifizierung verwendet. Wenn die Verbindung zum Server gelungen ist, kann die Beispieldatenbank **AdventureWorksDW** ausgewählt werden.

Der nächste Schritt ist die Festlegung einer brauchbaren Datenquellsicht. Dazu wird über das Kontextmenü von **Datenquellsichten** die Option **Neue Datenquellsicht** aufgerufen. Im **Datenquellsicht-Assistenten** können die für den Cube vorgesehen Dimensionentabellen und Faktentabellen bestimmt werden. Die ersteren sind an dem vorgegebenen Präfix **Dim** zu erkennen, letztere an dem Präfix **Fact**.



Auswahl der für den Cube vorgesehenen Tabellen

Nach diesen Vorbereitungen lässt sich aus der ausgewählten Sicht mit der Option **Neuer Cube**, die über das Kontextmenü von **Cubes** angeboten wird, ein Cube erstellen.



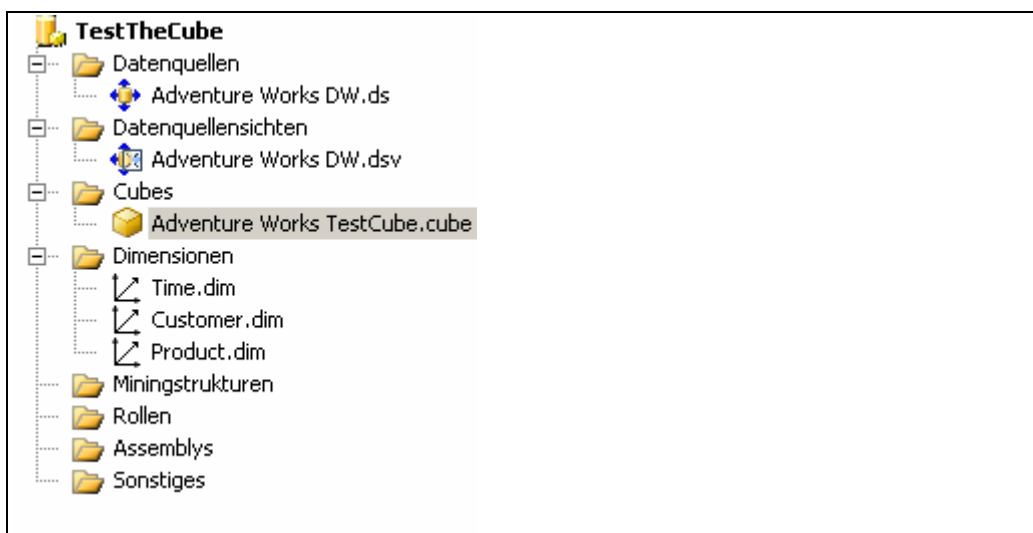
Wahl der automatischen Erstellungsmethode

Für einen Cube müssen mindestens eine Faktentabelle und auch mindestens eine Dimensionstabelle verfügbar sein.

Tabellen		Diagramm	
Name	<input checked="" type="checkbox"/> Fakt	<input checked="" type="checkbox"/> Dimension	
dbo.DimCustomer	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
dbo.DimGeography	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
dbo.DimProduct	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
dbo.DimTime	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
dbo.FactInternetSales	<input checked="" type="checkbox"/>	<input type="checkbox"/>	

Der Cube-Assistent schlägt die Nutzung der Tabellen vor

Im nächsten Schritt lassen sich aus den Spalten der angegebenen Faktentabelle die auswählen, die tatsächlich als **Measures** für den Cube verwendet werden sollen. Zum Schluss muss noch ein Name für den Cube vergeben werden. Im **Projektmappen-Explorer** sind schließlich alle für das Projekt benötigten Komponenten sichtbar.



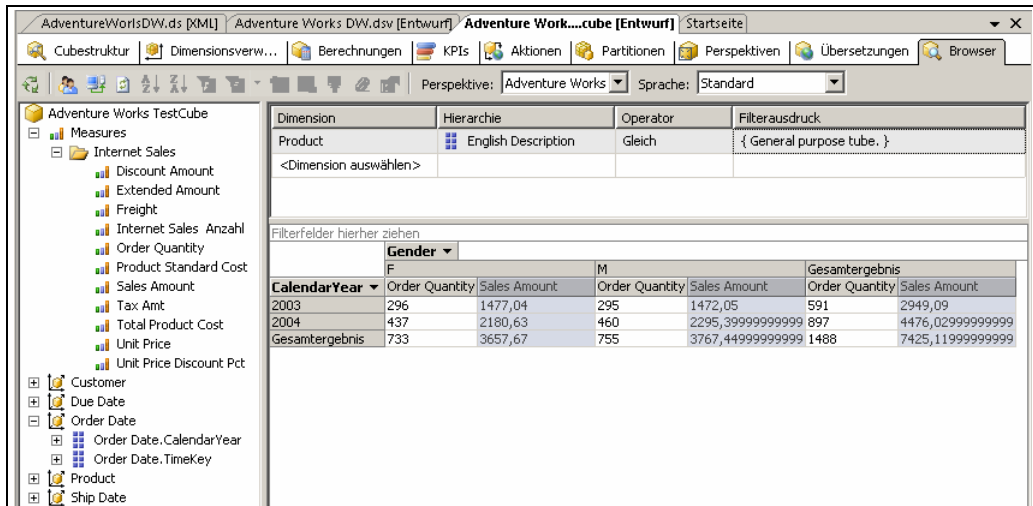
Der Cube und die einzelnen Dimensionen im Projektmappen-Explorer

Bereitstellen des Cubes

Der letzte Schritt ist die Verarbeitung und Bereitstellung des Cubes. Beim Verarbeiten der Objekte werden die für Dimensionen und Measures ausgewählten Daten zusammengestellt bzw. berechnet, in den Cube kopiert und entsprechend gespeichert, sodass der Cube für Anwendungen zur Verfügung steht, die auf die entsprechende Instanz des SQL-Servers zugreifen wollen. Beim Bereitstellen werden die definierten Objekte in einer Instanz von **Analysis Services** verfügbar gemacht. Zunächst geschieht diese Bereitstellung auf dem Entwicklungsserver. Ist alles geprüft, kann dann die Bereitstellung auf einem Produktionsserver erfolgen.

Gestartet wird die Verarbeitung und Bereitstellung über **Erstellen • <Projektname> erstellen**. Die vorgegebene Speicheremethode ist **MOLAP**, d.h. dass die Daten aus den verwendeten Tabellen samt der berechneten Daten in den Cube kopiert werden.

Nach der Bereitstellung des Cubes kann auch das vom **Cube-Designer** zur Verfügung gestellte Register **Browser** genutzt werden, um innerhalb der Visual Basic-Umgebung Daten im Cube in einer Form anzuzeigen, die große Ähnlichkeit mit der Darstellung der Pivottabellen in Excel hat.



Anzeige von Daten aus dem Cube auf dem Register Browser

Die Abfragesprache MDX

Während für die Abfrage von relationalen Datentabellen üblicherweise SQL verwendet wird, gilt für die Handhabung von mehrdimensionalen Datenstrukturen eine spezielle Abfragesprache, die als **Multidimensional Expressions (MDX)** bezeichnet wird. Da diese Ausdrücke als Argumente für die Cube-Funktionen benötigt werden, soll MDX hier wenigstens ansatzweise beschrieben werden. MDX ist eine von Microsoft geförderte Abfragesprache für multidimensionale Datenbanken.

Die Syntax einer SELECT-Anweisung in MDX gleicht der SQL-Syntax, geht aber von einem anderen Datenmodell aus, da ja statt mit zweidimensionalen Tabellen mit mehrdimensionalen Cubes gearbeitet wird. Die generelle Syntax sieht so aus:

```
SELECT [<Achsenspezifikation>
      [, <Achsenspezifikation>...]]
FROM [<Cubespezifikation>]
[WHERE [<Slicerspezifikation>]]
```

Die Anweisung arbeitet anstelle der Spalten in SQL mit Abfrageachsen, wobei die ersten fünf Achsen über Aliasnamen angesprochen werden können. Jede Achsenspezifikation definiert eine Dimension. Die Syntax ist <Menge> ON <Name der Achse>. Für die ersten fünf Achsen werden die Aliasnamen COLUMNS | ROWS | PAGES | SECTIONS | CHAPTERS verwendet. Die folgende Anweisung benutzt beispielsweise COLUMNS und ROWS als Aliasnamen für Achsen:

```
SELECT {[Measures].[Sales Amount],
       [Measures].[Tax Amount] } ON COLUMNS,
       {[Date].[Fiscal].[Fiscal Year].&[2004],
       [Date].[Fiscal].[Fiscal Year].&[2005] } ON ROWS
```

```
FROM [Adventure Works]
WHERE ( [Sales Territory].[Southwest] )
```

Alternativ können die Achsen auch über ihre Position angesprochen werden; dazu wird mit einem nullbasierten Index gearbeitet: `AXIS(0)` ist also die erste Achse. Die Nummerierung muss immer fortlaufend sein:

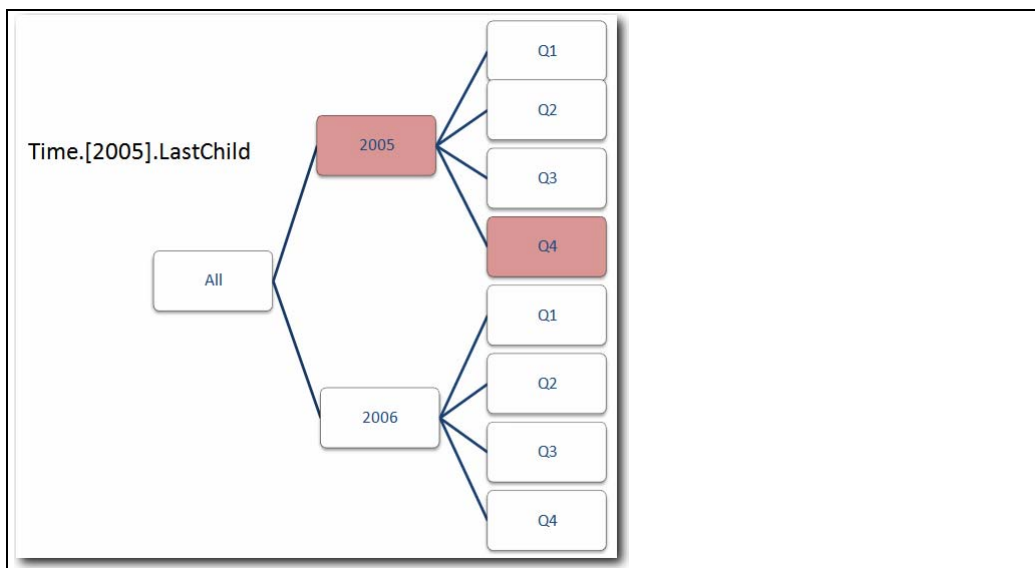
```
SELECT { [Measures].[Unit Sales], [Measures].[Store Sales]
        } ON AXIS(0),
        { [Time].[1997], [Time].[1998] } ON AXIS(1)
```

In der `FROM`-Klausel kann jeweils ein Cube angegeben werden. Die `FROM`-Klausel bestimmt den Cube-Kontext. Dabei kann es sich um einen kompletten Cube oder auch um einen Teilcube handeln.

Während in SQL die `WHERE`-Klausel im Sinne eines Filters darüber entscheidet, welche Zeilen angezeigt werden, wird die `WHERE`-Klausel in MDX verwendet, um die Daten über eine bestimmte Dimensionen oder ein Mitglied derselben einzuschränken. Das wird auch als **Slicing** bezeichnet. Die `WHERE`-Klausel legt dabei die Slicerachse fest und liefert so in jedem Fall eine klar abgegrenzte Untermenge.

Attribute und Attributhierarchien

Dimensionen bestehen aus Gruppen von Attributen, die auf Spalten in den verwendeten Dimensionstabellen basieren. Diese Gruppen sind in Form von Hierarchien geordnet.



Beispiel für die Navigation in der Zeithierarchie

Um Elemente auf den verschiedenen Ebenen einer Dimension innerhalb einer Anweisung ansprechen zu können, wird eine Schreibweise verwendet, bei der Bezeichner durch eckige Klammern gekennzeichnet werden. Punkte trennen Ebenen innerhalb einer Hierarchie. Dabei können untergeordnete Elemente entweder explizit angegeben werden, etwa:

```
[Date].[Calendar Year].[CY 2005]
```


oder mithilfe von MDX-Funktionen:

```
[Date].[Calendar Year].LastChild
```

Die eckigen Klammern sind notwendig, wenn der Name eines Elements eine Zahl oder ein Leerzeichen enthält. Die Klammern dürfen aber auch bei einfachen Namen verwendet werden. Der Bezug auf Elemente einer Dimension kann statt über den Namen auch über entsprechende Schlüssel hergestellt werden. Das ist sinnvoll, wenn die Namen innerhalb mehrerer Dimensionen nicht eindeutig sind. Schlüssel werden durch ein vorangestelltes Ampersand(&)-Zeichen gekennzeichnet, etwa:

```
[Time].[1nd half].&[Q2].
```

Elemente können auch durch Berechnungen aus vorhandenen Werten neu gebildet werden. Zur Definition solcher Elemente für eine Abfrage wird das Schlüsselwort WITH verwendet:

```
WITH MEMBER [Measures].[Special Discount] AS [Measures].[Discount Amount] * 1.5
```

In diesem Fall wird das neue Element [Measures].[Special Discount], dessen voll qualifizierter Name zuerst angegeben wird, auf der Basis eines bereits vorhandenen Elements [Measures].[Discount Amount] durch Multiplikation mit dem Faktor gebildet. Das berechnete Element gilt nur für die so spezifizierte Abfrage.

Tupel und Mengen

Ein Tupel innerhalb einer multidimensionalen Datenstruktur ist eine geordnete Kollektion von je einem Element für jede Dimension, die in einer Abfrage verwendet wird. Dabei können auch vorgegebene Elemente einbezogen werden. Tupel, die nur aus einem Element einer Dimension bestehen, können in einer vereinfachten Syntax angesprochen werden, etwa

```
Time.[1nd half]
```

Bei Tupeln, die Elemente verschiedener Dimensionen enthalten, wird mit Klammern gearbeitet:

```
(Time.[2nd half], Route.nonground.air)
```

Mehrere Tupel können eine Menge als geordnete Kollektion bilden. Dies geschieht bei der Spezifikation von Achsen. Mengenausdrücke werden durch geschweifte Klammern gekennzeichnet:

```
{(Time.[1st half], Route.nonground.air), (Time.[2nd half], Route.nonground.sea)}
```

Um einen Bereich von Elementen anzugeben, kann mit dem Bereichsoperator gearbeitet werden:

```
{[1st quarter]:[4th quarter]}
```

Mengen können aber auch durch MDX-Funktionen geliefert werden.

```
{Time.Children}
```

liefert beispielsweise die untergeordneten Elemente zu Time.

MDX ist eine umfangreiche Sprache, die hier nicht in allen Details beschrieben werden kann. Neben zahlreichen Operatoren und Funktionen steht noch eine Reihe von Anweisungen zur Verfügung. Die möglichen Anweisungen sind in drei Gruppen unterteilt: Anweisungen für Skripte, Anweisungen für die Datendefinition und Anweisungen für die Datenbearbeitung.

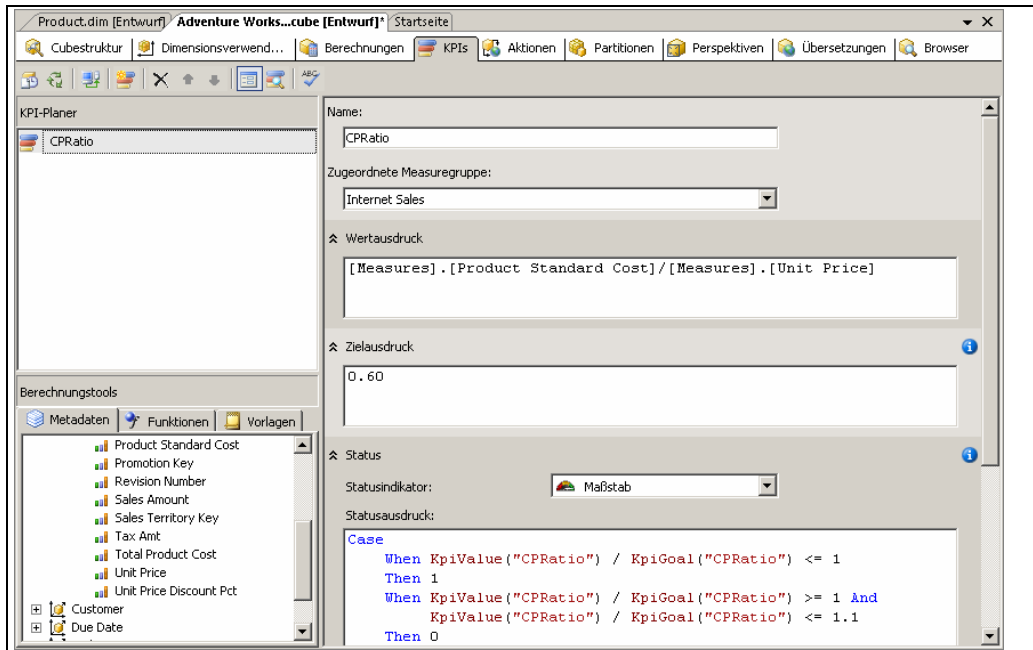
Skripte werden verwendet, um innerhalb des Cubes bestimmte Berechnungsprozesse durchzuführen. Dabei werden die folgenden Anweisungen verwendet:

Anweisung	Beschreibung
CALCULATE	Berechnet einen Teilcube.
CASE	Erlaubt die bedingte Rückgabe von Werten aus mehreren Vergleichen.
EXISTING	Erzwingt die Auswertung der angegebenen Menge im aktuellen Kontext.
FREEZE	Fixiert die Werte eines Teilcubes auf die aktuell vorhandenen Werte.
IF	Bedingte Anweisung
SCOPE	Beschränkt eine Anweisung auf einen Teil des Cubes.

Tabelle 1: Skriptanweisungen in MDX

Die komplette MDX-Referenz finden Sie unter <http://msdn2.microsoft.com/de-de/library/ms145556.aspx>.

Key Performance Indikatoren



Definition von KPIs in SQL Server Business Intelligence Development Studio

Eine spezielle Funktion, die in einem Cube eingerichtet werden kann, ist ein mit grafischen Hinweisen verbundener Indikator, dessen Anzeige über einen Statusausdruck gesteuert wird. Sinn und Zweck solcher **KPIs** ist es, auffällige Hinweise auf relevante betriebswirtschaftliche Kennzahlen zu generieren, die über Erfolg und Misserfolg von Geschäftsprozessen entscheiden. Für die Definition von KPIs steht in **SQL Server Business Intelligence Development Studio** ein spezielles Register zur Verfügung, wie die Abbildung zeigt.

Die Schaltfläche **Neuer KPI** blendet ein Formular ein, das zunächst einen Namen und die Measuregruppe abfragt, mit der der Indikator verknüpft werden soll. Unter **Wertausdruck** wird ein Wert oder ein Ausdruck angegeben, der beobachtet und mit einem Zielausdruck verglichen werden soll. Ein einfaches Beispiel wäre etwa das Verhältnis der Produktkosten zum Preis der Einheit. Unter **Zielausdruck** wird der angestrebte Planwert oder ein Ausdruck, der diesen Wert liefert, eingegeben.

Für die Anzeige des erreichten Status kann dann ein grafischer **Statusindikator** gewählt werden. Die Anzeige in diesem Indikator wird über einen **Statusausdruck** gesteuert, der Werte zwischen 1 und -1 liefert. 1 steht für sehr gut, -1 für sehr schlecht. Der Ausdruck kann ein MDX-Skriptfragment sein mit verschiedenen Prüfpunkten, die bestimmten Positionen des Indikators entsprechen. Ein einfaches Beispiel:

```
Case
  When
    KpiValue("CPRatio")/KpiGoal("CPRatio")<= 1
    Then 1
  When
    KpiValue("CPRatio ")/KpiGoal("CPRatio")>=1
```

```

And
KpiValue("CPRatio ")/KpiGoal("CPRatio")<=1.1
Then 0
Else-1
End

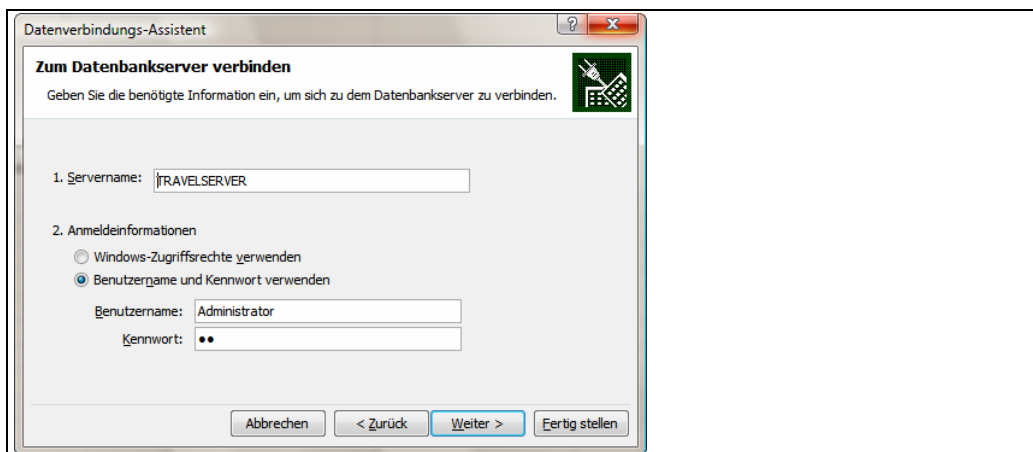
```

Zusätzlich können noch ein **Trendindikator** (ebenfalls Werte zwischen 1 und -1) und ein **Trendausdruck** eingegeben werden. Die hierüber erzeugten Daten können über die unten beschriebene Funktion CUBE-KPIELEMENT() abgefragt werden.

Definieren einer Verbindung zu einem Analysis Services Server

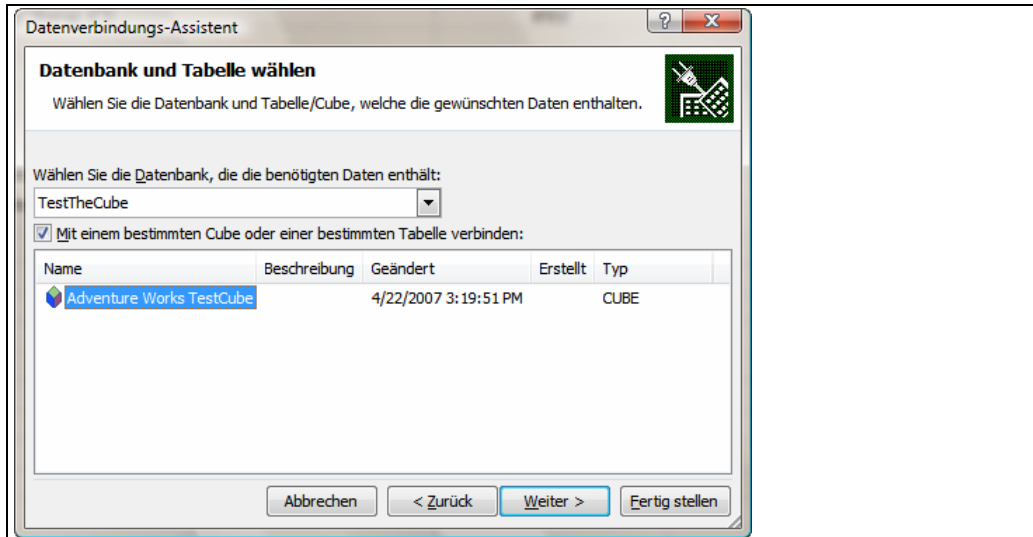
Wird auf einem SQL Server 2005 ein Cube bereitgestellt, kann von Excel 2007 aus eine Verbindung zu diesem Cube eingerichtet werden, die anschließend für die Cube-Funktionen zur Verfügung steht.

Dazu wird auf dem Register **Daten** die Schaltfläche **Externe Daten abrufen** und anschließend die Schaltfläche **Aus anderen Quellen** verwendet. In der Menüleiste wird die Option **Von Analysis Services** angeboten, die den **Datenverbindungs-Assistenten** startet.



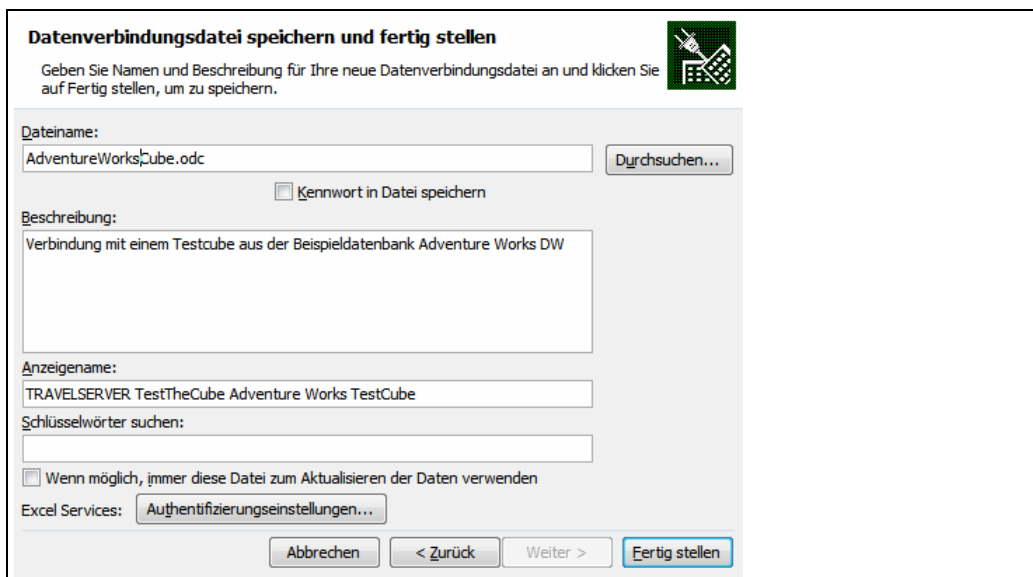
Aufbau einer Verbindung zu einem SQL-Server

Im ersten Dialog werden der **Servername** und die **Anmeldeinformationen** eingegeben. Kommt die Verbindung zustande, lassen sich anschließend die Analysis Services Datenbank und der darin enthaltene Cube auswählen. Der Datenbankname entspricht in diesem Fall dem Namen des Projekts, das mit dem BI-Tool angelegt wurde.



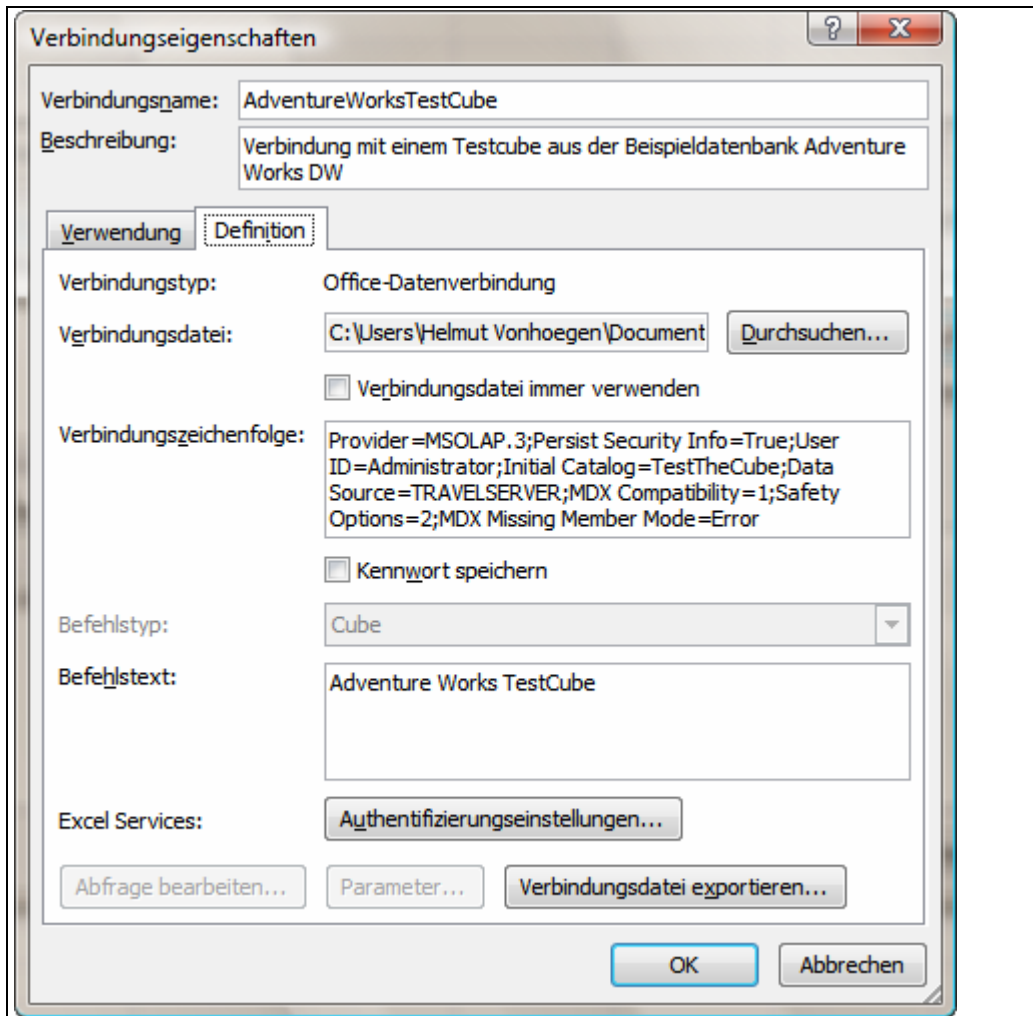
Auswahl von Datenbank und Cube

Die Daten zu der Verbindung werden in einer **.odc**-Datei gespeichert, sodass immer wieder darauf zurückgegriffen werden kann. Der Verbindungsname wird in fast allen Cube-Funktionen als erstes Argument benötigt.



Speichern der Datenverbindungsdatei

Über **Daten • Verbindungen • Verbindungen** kann jederzeit der Dialog **Arbeitsmappenverbindungen** geöffnet werden, in dem die mit der Arbeitsmappe verknüpften Verbindungsdefinitionen verwaltet werden. Die Schaltfläche **Eigenschaften** zeigt die Detaildaten zu einer ausgewählten Verbindung:



Verbindungseigenschaften

Offline-Cubes

Alternativ zur Online-Verbindung mit einem Cube auf dem Server können Daten auch offline aus so genannten **CUB**-Dateien übernommen werden, die vorher aus den Server-Daten generiert wurden. Mithilfe einer solchen **CUB**-Datei lassen sich Daten lokal verfügbar machen, insbesondere für den Fall, dass die Verbindung zum Server, auf dem die Daten normalerweise gepflegt werden, unterbrochen ist. Die **CUB**-Datei enthält in der Regel eine Untermenge der Quelldaten aus einer **OLAP**-Datenbank.

Um eine **CUB**-Datei zu erstellen, muss zunächst über eine Online-Verbindung eine Pivottable mit den Daten aus einem OLAP-Cube erstellt werden. Dazu wird eine Pivottable über **Einfügen • Tabellen • PivotTable** mit der Option **Externe Datenquelle verwenden** angelegt. Über die Schaltfläche **Verbindung auswählen** wird die vorher definierte Verbindung angegeben. Die folgende Abbildung zeigt eine Pivottable mit Daten aus der Beispieldatenbank.

	A	B	C	D
2		Werte		
3	City	Order Quantity	Sales Amount	Total Product Cost
4	Bellingham	566	207.613,25	120.333,94
5	Berkeley	567	258.138,46	151.590,40
6	Berlin	583	260.930,63	153.039,96
7	Burien	579	230.359,50	136.125,44
8	Chula Vista	583	233.416,33	136.764,15
9	Cliffside	918	211.608,90	122.661,02
10	Concord	559	210.445,86	123.631,61
11	London	1579	802.810,30	471.901,12
12	Paris	1174	539.725,80	317.765,69
13	Shawnee	563	128.835,06	74.975,18
14	Gesamtergebnis	7671	3.083.884,09	1.808.788,50
15				
16				
17				
18				
19				
20				

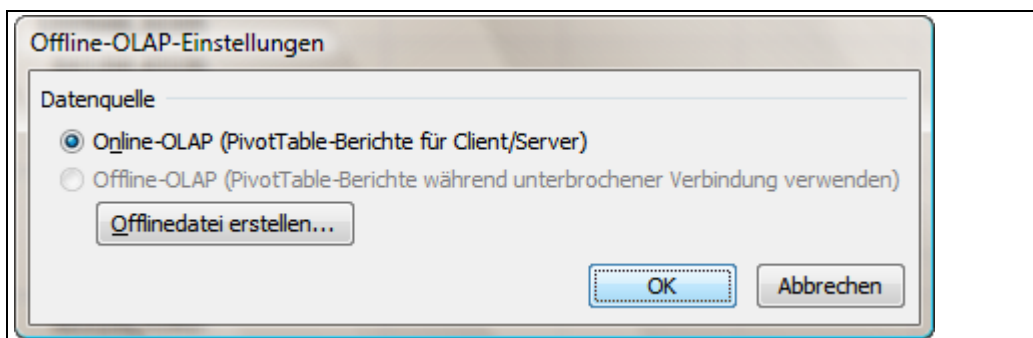
OLAP-Daten in einer PivotTable

Um nun aus den online verfügbaren Daten eine CUB-Datei zu erstellen, wird, wenn die Pivot-Tabelle ausgewählt ist, über das **Optionen**-Register in der Gruppe **Tools** die Schaltfläche **OLAP-Tools** benutzt.



Die Schaltfläche OLAP-Tools

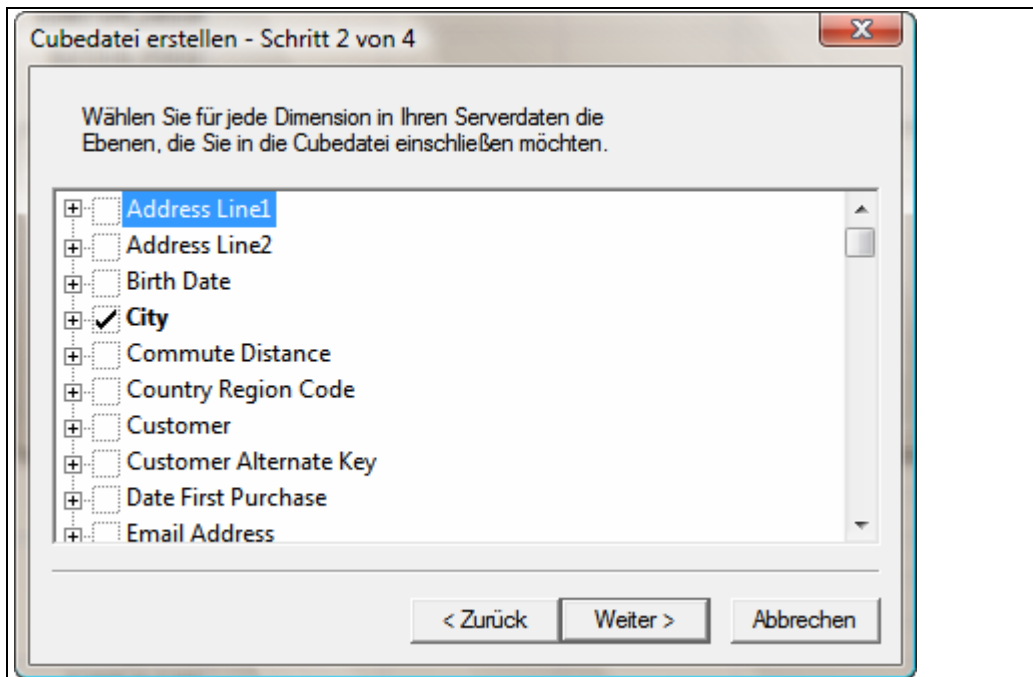
Aus dem Menü wird die Option **Offline-OLAP** verwendet und im Dialog die Schaltfläche **Offlinedatei erstellen**.



Dialog für die Erstellung eines Offline-Cubes

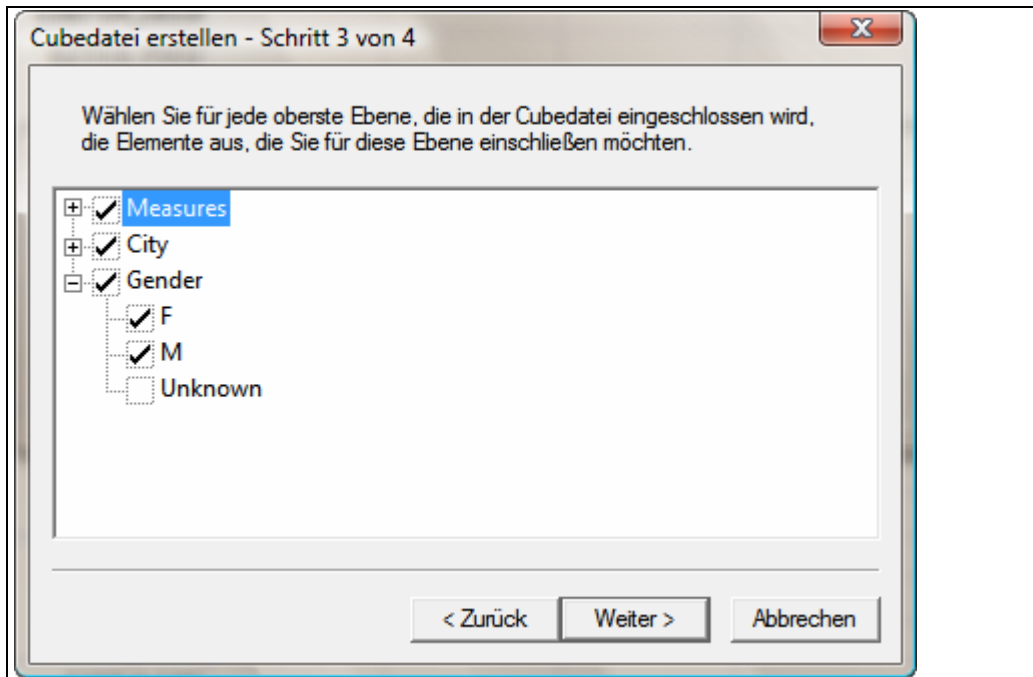
Damit wird der **Offlinecube-Assistent** aufgerufen. Zunächst werden die Dimensionen des Cubes angegeben, die in den Offlinecube mit aufgenommen werden sollen. In der Regel sollten alle Dimensionen übernommen werden, für die gruppierte Elemente

vorhanden sind. Innerhalb der Dimensionen lassen sich die Ebenen auswählen. Zwar können Zwischenebenen nicht übersprungen werden, aber es ist möglich, die unteren Ebenen wegzulassen, wenn der Umfang des Cubes eingeschränkt werden soll.



Auswahl der Dimensionen und Ebenen

Nur bei den Dimensionen, die nicht mit einem Plusfeld ausgestattet sind, müssen alle Ebenen übernommen werden. Diese Dimensionen können also nur ganz ein- oder ausgeschlossen werden.



Auswahl der Elemente pro Ebene

Im nächsten Schritt können die gewünschten Elemente zu den Measures gewählt werden. Hierbei handelt es sich in der Regel um die Datenfelder, die numerische Werte enthalten. Mindestens ein Measure-Eintrag muss ausgewählt werden, weil sonst die zugeordneten Dimensionen keine Daten aufweisen. Die Eigenschaftsfelder, die den Elementen eines OLAP-Cubes zugeordnet sind, werden automatisch in die Cubedatei aufgenommen. Zuletzt muss nur noch der Name und der Speicherort der **CUB**-Datei angegeben werden.

CUB-Dateien können einerseits direkt als Quelldaten für PivotTable-Berichte – und PivotCharts verwendet werden, andererseits besteht seit Excel 2007 auch die Möglichkeit, Daten daraus direkt über die neuen Cube-Funktionen auszuwerten.

Es gibt übrigens auch noch die Möglichkeit, eine **CUB**-Datei beispielsweise über den Weg der Abfrage einer ACCESS-Datei mithilfe der altbekannten Komponente **MSQuery** zu erstellen. Dabei wird das Ergebnis der Abfrage in eine **CUB**-Datei gespeichert.

Die Verwendung vorhandener **CUB**-Dateien für Pivot-Berichte ist einfach. Sie brauchen nur die **CUB**-Datei als externe Datenquelle anzugeben.

Cube-Formeln automatisch erzeugen

Wenn Sie aus OLAP-Daten eine Pivottabelle angelegt haben, gibt es auch noch die Möglichkeit, diese Tabelle in einen normalen Zellbereich zurück zu verwandeln und dabei gleichzeitig automatisch Cube-Funktionen zu generieren, die dafür sorgen, dass die aktuell angezeigten Daten als Ergebnis der entsprechenden Formeln ausgegeben werden.

Solange eine Zelle in der Pivottabelle ausgewählt ist, wird dazu über die schon genannte Schaltfläche **OLAP_Tools** die Option **In Formeln konvertieren** verwendet.

Soll die gesamte Pivottable konvertiert werden, muss die Option **Berichtsfiler konvertieren** aktiviert werden, ansonsten werden nur die Spalten- und Zeilenbereiche konvertiert und der Berichtsfiler kann weiter wie in einer Pivottable genutzt werden.

Die Abbildung zeigt die oben noch als PivotTable angezeigte Tabelle nun als normalen Zellbereich mit lauter Cube-Funktionen.

B4					
=CUBEWERT("TestTheCube Adventure Works";\$A4:\$B\$3)					
A	B	C	D	E	F
2	Werte				
3	City	Order Quantity	Sales Amount	Total Product Cost	
4	Bellingham	566	207.613,25	120.333,94	
5	Berkeley	567	258.138,46	151.590,40	
6	Berlin	583	260.930,63	153.039,96	
7	Burien	579	230.359,50	136.125,44	
8	Chula Vista	583	233.416,33	136.764,15	
9	Cliffside	918	211.608,90	122.661,02	
10	Concord	559	210.445,86	123.631,61	
11	London	1579	802.810,30	471.901,12	
12	Paris	1174	539.725,80	317.765,69	
13	Shawnee	563	128.835,06	74.975,18	
14	Gesamtergebnis	7671	3.083.884,09	1.808.788,50	
15					

Die konvertierte PivotTable

Die folgende Abbildung zeigt einige der generierten Formeln. Die CUBEWERT()-Funktionen verwenden jeweils eine absolute Zeilen- und Spaltenadresse, d. h. es handelt sich um eine Überkreuzauswertung, wie sie auch in der Pivottable vorgenommen wird.

A	B
2	Werte
3	City
4	=CUBELEMENT("TestTheCube Adventure Works";["Customer].[City].&{Bellingham}")
5	=CUBELEMENT("TestTheCube Adventure Works";["Customer].[City].&{Berkeley}")
6	=CUBELEMENT("TestTheCube Adventure Works";["Customer].[City].&{Berlin}")

Die generierten Formeln

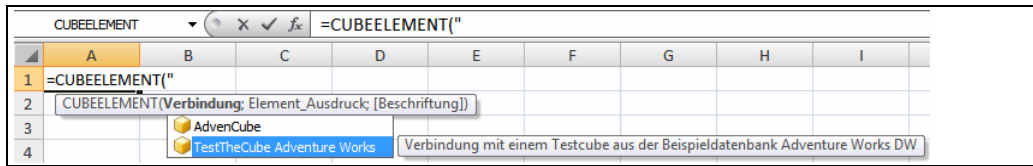
Besonderheiten der Cube-Funktionen

Anders als bei den sonstigen Excel-Funktionen liefern Cube-Funktionen mit Ausnahme von CUBEMENGENANZAHL() jeweils zwei Ergebnisse, ein Ergebnis, das im jeweiligen Zellbereich ausgegeben wird und ein internes Ergebnis, das verwertet wird, wenn eine Cube-Funktion als Argument für eine andere Cube-Funktion verwendet wird. Beispielsweise liefert die Funktion CUBEMENGE() einen internen Wert, der dann von der Funktion CUBEMENGENANZAHL() ausgewertet werden kann.

Wieder mit Ausnahme von CUBEMENGENANZAHL() verwenden alle Funktionen als erstes Argument den Namen der Verbindung zu dem für die Abfrage vorgesehenen Cube.

Die im Kapitel 1 bereits beschriebene Funktion **AutoVervollständigen bei Formeln** ist bei der Eingabe von Cube-Funktionen besonders hilfreich. Zuerst wird die Eingabe des Verbindungsnamens unterstützt, wobei immer nur die für die Anwendung verfügb-

baren Verbindungen angeboten werden, sobald hinter der öffnenden Klammer die Anführungszeichen eingegeben sind.



Die definierten Verbindungen werden angeboten

Besonders nützlich ist dann die Unterstützung bei der Zusammenstellung der als Argument benötigten MDX-Ausdrücke. Nach der Eingabe von Anführungszeichen oder einem Punkt in einem solchen Ausdruck werden die aktuell möglichen Optionen angeboten. Die innerhalb der Ausdrücke möglichen Funktionsnamen wie Children, Parent oder Crossjoin müssen allerdings manuell eingetragen werden.

Beispiel für eine Lösung mit Cube-Funktionen

Da die Cube-Funktionen selten einzeln eingesetzt werden, soll an dieser Stelle an einem übersichtlichen Beispiel das Zusammenspiel der wichtigsten Funktionen demonstriert werden. Die Abbildung zeigt eine Tabelle, in der für ein bestimmtes Jahr die Verkaufssumme bei einem Produkt für die zehn besten Kunden ausgegeben wird.

A4		=CUBELEMENT("TestTheCube Adventure Works"; "[Measures].[Sales Amount]")					
A	B	C	D	E	F	G	H
1							
2	CUBE-Abfrage						
3							
4	Sales Amount	2004	Cycling Cap				
5							
6	Rang	Top Kunden	Cycling Caps 2004				
7	1	Diaz	287,68				
8	2	Torres	206,77				
9	3	Hernandez	197,78				
10	4	Jenkins	188,79				
11	5	Xu	188,79				
12	6	Russell	179,80				
13	7	Sanchez	179,80				
14	8	Gonzalez	170,81				
15	9	Li	170,81				
16	10	Rodriguez	170,81				
17							
18	Von 376 Kunden insgesamt						
19							

Beispiel für eine Cube-Abfrage

Um diese Lösung zu realisieren wird zunächst in den drei Zellen in Zeile 4 mit CUBELEMENT()-Funktionen gearbeitet. Die Formeln sehen so aus:

```
=CUBELEMENT("TestTheCube Adventure Works"; "[Measures].[Sales Amount]")
```

```
=CUBELEMENT("TestTheCube Adventure Works"; "[Order Date].[Fiscal Year].[All].[2004]")
```

```
=CUBELEMENT("TestTheCube Adventure
Works";"[Product].[Model Name].[All].[Cycling
Cap]")
```

Die erste Funktion liefert das [Measures]-Element [Sales Amount]. Die zweite Funktion liefert ein untergeordnetes Element der Zeitdimension, hier ein bestimmtes Jahr. Die dritte Funktion liefert ein Element aus der Dimension [Product]. Alle drei Funktionen sorgen dafür, dass in der jeweiligen Zelle der Name des untersten Elements als Text angezeigt wird. Gleichzeitig aber stellen diese Funktionen die Metadaten für die mithilfe der MDX-Ausdrücke angesprochenen Elemente zur Verfügung, sodass andere Cube-Funktionen diese Daten nutzen können.

Die Werte in der Spalte mit der Bezeichnung Rang werden in diesem Fall einfach manuell eingetragen. In der Zelle B6 ist eine Formel abgelegt, die eine Menge von Elementen spezifiziert:

```
=CUBEMENGE("TestTheCube Adventure
Works";"[Customer].[Last
Name].[All].children";"Top Kunden";2;C6)
```

Diese Formel liefert den Zugang zu der Liste der Nachnamen der Kunden. Die Anzeige in der Zelle wird als drittes Argument der Funktion eingegeben. Das vierte Argument sorgt für eine absteigende Sortierung, das fünfte Argument gibt über den Zellbezug auf C6 an, auf welche Elemente sich die Sortierung bezieht.

In der Zelle C6 wird ein Tupel bestimmt, das die Ergebnisse der in den Zellen A4:C4 abgelegten CUBELEMENT()-Funktionen nun kombiniert und die darin angesprochenen Dimensionen zusammenführt. Die Sortierung bezieht sich also auf die Verkaufssumme für das ausgewählte Produkt im Jahr 2004.

```
=CUBELEMENT("TestTheCube Adventure Works";A4:C4;"Cycling
Caps 2004")
```

Die Bereichsangabe A4:C4 wird also automatisch als eine korrekte Tupel-Definition in MDX übersetzt.

Um nun die Namen aus der Kundenliste im Tabellenblatt anzuzeigen, wird mit der Funktion CUBERANGELEMENT() gearbeitet. Die Formel für den ersten Kunden lautet:

```
=CUBERANGELEMENT("TestTheCube Adventure Works";$B$6;$A7)
```

Das zweite Argument ist ein absoluter Bezug auf die Mengendefinition. Das dritte Argument übernimmt mit einem teilabsoluten Bezug die Rangzahl auf Spalte A. Diese Formel kann also ohne weiteres nach unten kopiert werden, um die restlichen Top-Kunden anzuzeigen.

Nun fehlt noch die Funktion für die Anzeige der einzelnen Werte pro Kunde. Diese besorgt die CUBEWERT()-Funktion. Diese Funktion liefert als Measure-Daten die zusammengefassten Verkaufswerte, wobei das zweite und das dritte Argument die jeweilige Schnittmenge bestimmen. Das zweite Argument ist ja ein Bezug auf die CUBERANGELEMENT()-Funktion, die den einzelnen Kunden festlegt. Das dritte Argument ist jedes Mal ein Bezug auf die Tupel-Definition in C6.

```
=CUBEWERT("TestTheCube Adventure Works";$B7;C$6)
```

Zur Ergänzung ist in A18 noch eine Formel eingetragen, die anzeigt, wieviele Kunden die Formel in B6 insgesamt geliefert hat:

```
= "Von " & CUBEMENGENANZAHL(B6) & " Kunden insgesamt "
```

1.1 Referenz der Cube-Funktionen

CUBEELEMENT()

CUBEMEMBER()

Syntax: CUBEELEMENT(**Verbindung**; **Element_Ausdruck**;
Beschriftung)

Beispiel: =CUBEELEMENT("AdventureWorks";
"[Measures].[Sales Amount]")

Verbindung ist der Name der Verbindung zum Cube. Element_Ausdruck ist ein gültiger MDX-Ausdruck, der ein eindeutiges Element im Cube ergibt. Stattdessen kann auch ein Zellbereich oder Array angegeben werden, der ein gültiges Tupel darstellt. Die Funktion gibt ein Element oder Tupel aus dem Cube zurück. Auf diese Weise wird zugleich überprüft, ob das mithilfe von Element_Ausdruck angegebene Element oder Tupel im Cube vorhanden ist. Beschriftung ist eine Zeichenfolge, die anstelle der sonst in der Zelle angezeigten Elementnamen erscheinen soll. (Bei Tupeln erscheint als Vorgabe immer die Beschriftung des letzten Elements.)

CUBEELEMENTEIGENSCHAFT()

CUBEMEMBERPROPERTY()

Syntax: CUBEELEMENTEIGENSCHAFT(**Verbindung**;
Element_Ausdruck; **Eigenschaft**)

Beispiel: =CUBEELEMENTEIGENSCHAFT("Adventure
Works"; "[Order Date].[Fiscal Year].
[All].[2005]"; B10)

Die Funktion liefert den Wert einer Eigenschaft des über Element_Ausdruck angegebenen Elements im Cube. Damit wird geprüft, ob ein entsprechender Elementname im Cube vorhanden ist. Wenn ja, wird die für dieses Element angegebene Eigenschaft zurückzugeben. Das Argument Eigenschaft enthält den Namen der Eigenschaft oder einen Zellbezug, der diesen Namen enthält.

CUBEKPIELEMENT()

CUBEKPIMEMBER()

Syntax: CUBEKPIELEMENT(**Verbindung**; **KPI_Name**;
KPI_Eigenschaft; Beschriftung)

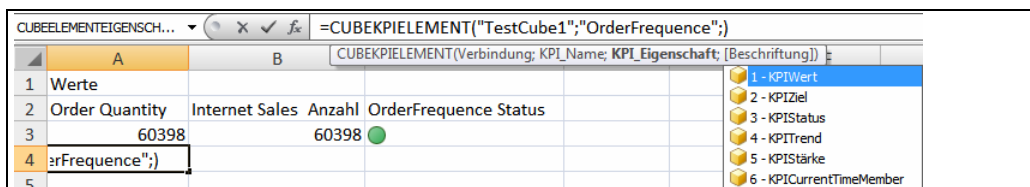
Beispiel: =CUBEKPIELEMENT("AdventureWorks";
"CPRatio"; 3; "Kosten/Preis-Relation")

Diese Funktion gibt die angegebene KPI_Eigenschaft eines Key Performance Indicators (KPI) zurück und zeigt KPI_Name und Eigenschaft in der Zelle an. Wird ein Text für Beschriftung angegeben, wird dieser stattdessen in der

Zelle angezeigt. Ein solcher Indikator ist ein messbares Maß, wie z. B. der Quartalsgewinn oder die vierteljährliche Mitarbeiterfluktuation, womit die Leistungsfähigkeit eines Unternehmens beurteilt werden kann. Für das Argument `KPI_Eigenschaft` sind folgende Werte möglich:

Zahl	Konstante	Beschreibung
1	KPIValue	Aktueller Wert
2	KPIGoal	Zielwert
3	KPIStatus	Zustand des KPI zu einem bestimmten Zeitpunkt
4	KPI_Trend	Measure des Werts über einen Zeitraum
5	KPIWeight	Gewichtung des Indikators
6	KPICurrentTimeMember	Zeitraumen des Indikators

Wird `KPIValue` verwendet, wird nur der `KPI_Name` in der Zelle angezeigt. Die Eingabe der Werte wird durch die Funktion **AutoVervollständigen** unterstützt.




Die möglichen Werte für `KPI_Eigenschaft` werden bei der Eingabe angeboten

Enthält der Cube einen solchen KPI, kann mit einer Kombination von `CUBEKPIELEMENT()` und `CUBEVALUE()` gearbeitet werden, wie die folgende Tabelle zeigt:

A4		=CUBEKPIELEMENT("TestTheCube Adventure Works";"CPRatio";1;"Kosten/Preis-Relation")			
	A	B	C	D	E
1					
2	KPI-Anzeige				
3					
4	Kosten/Preis-Relation	0,588507222			
5	Kosten/Preis-Zielwert	0,6			
6	Kosten/Preis-Relation-Status	1			
7					

Anzeige von KPI-Daten in Excel 2007-05-09

In der Spalte A wird die `CUBEKPIELEMENT()`-Funktion mit unterschiedlichen Werten für das Argument `KPI_Eigenschaft` verwendet. Dabei wird jeweils eine passende Beschriftung angegeben. In der Spalte B wird mithilfe von `CUBEWERT()`-Funktionen auf die Funktionen in Spalte A Bezug genommen, um die den verschiedenen Eigenschaften entsprechenden Werte auszulesen. Die folgende Abbildung zeigt die verwendeten Formeln:

4	=CUBEKPIELEMENT("TestTheCube Adventure Works";"CPRatio";1;"Kosten/Preis-Relation")	=CUBEWERT("TestTheCube Adventure Works";A\$4)
5	=CUBEKPIELEMENT("TestTheCube Adventure Works";"CPRatio";2;"Kosten/Preis-Zielwert")	=CUBEWERT("TestTheCube Adventure Works";A\$5)
6	=CUBEKPIELEMENT("TestTheCube Adventure Works";"CPRatio";3;"Kosten/Preis-Relation-Status")	 =CUBEWERT("TestTheCube Adventure Works";A\$6)

Die Formeln zum Auslesen der KPI-Werte

Durch eine bedingte Formatierung kann der Statuswert noch hervorgehoben werden.

CUBEMENGE()

CUBESET()

Syntax: CUBEMENGE (**Verbindung**; **Menge_Ausdruck**;
Beschriftung; Sortierreihenfolge;
Sortieren_nach)

Beispiel: =CUBEMENGE ("AdventureWorks"; "[Customer].
[Last Name].[All].children"; "Top Kunden"
; 2; C6)

Diese Funktion sendet einen Mengenausdruck für einen bestimmten Satz von Elementen oder Tupeln an den Cube, der die entsprechenden Daten zusammenstellt und an Excel liefert. Menge_Ausdruck ist entweder ein MDX-Ausdruck oder ein Zellbezug auf einen solchen. Das Argument kann auch durch den Bezug auf einen Zellbereich zusammengestellt werden.

Mit Beschriftung kann eine andere Beschriftung gewählt werden als die im Cube vorhandene Beschriftung. Außerdem kann eine Sortierreihenfolge und ein Sortierfeld angegeben werden.

Sortierreihenfolge bestimmt die Art der auszuführenden Sortierung und kann folgende Werte annehmen:

Zahl	Konstante	Beschreibung
0	SortNone	Keine Änderung der Reihenfolge
1	SortAscending	Sortieren in aufsteigender Reihenfolge
2	SortDescending	Sortieren in absteigender Reihenfolge
3	SortAlphaAscending	Sortieren in alphabetisch aufsteigender Reihenfolge
4	Sort_Alpha_Descending	Sortieren in alphabetisch absteigender Reihenfolge
5	Sort_Natural_Ascending	Sortieren in natürlich aufsteigender Reihenfolge
6	Sort_Natural_Descending	Sortieren in natürlich absteigender Reihenfolge

Der Standardwert ist 0. Wird 1 oder 2 verwendet, muss das Argument Sortieren_nach angegeben werden, in allen anderen Fällen ist es nicht erforderlich. Bei einer alphabetischen Sortierung für eine Menge mit Tupeln wird jeweils anhand des letzten Elements in jedem Tupel sortiert. Sortieren_nach ist eine Textzeichenfolge des Werts, nach dem sortiert werden soll.

CUBEMENGENANZAHL()

CUBESETCOUNT()

Syntax: CUBEMENGENANZAHL(**Menge**)

Beispiel: =CUBEMENGENANZAHL(CUBEMENGE(
"AdventureWorks"; "[Customer].[Last Name]
.[All].children"; "Top Kunden"; 2; C6))

Die Funktion liefert die Anzahl der Elemente in einer Menge, die durch das Argument Menge bestimmt ist. Dieses Element besteht aus einer CUBEMENGE()-Funktion oder einem Zellbezug auf eine solche Funktion.

CUBERANGELEMENT()

CUBERANKEDMEMBER()

Syntax: CUBERANGELEMENT(**Verbindung**;
Menge_Ausdruck; **Rang**; Beschriftung)

Beispiel: =CUBERANGELEMENT("AdventureWorks";
\$B\$6; \$A7)

Gibt das mit dem Wert für Rang angegebene Element in einer Menge zurück. 1 liefert den besten Wert, 2 den zweitbesten Wert etc. Wird verwendet, um mindestens ein Element in einer Menge zurückzugeben, die meist durch die Funktion CUBEMENGE() definiert ist oder durch einen Bezug auf eine Zelle, die eine solche Funktion enthält.

CUBEWERT()

CUBEVALUE()

Syntax: CUBEWERT(**Verbindung**; **Element_Ausdruck1**;
Element_Ausdruck2...)

Beispiel: =CUBEWERT("AdventureWorks"; \$A4; D\$3)

Die Funktion liefert den aggregierten Wert für die durch einen oder mehrere Elementausdrücke eingeschränkten Elemente.

Weblinks zum Thema:

SQL Server Developer Center von Microsoft

<http://msdn2.microsoft.com/de-de/sql/default.aspx>

BI-Lösungen

<http://www.microsoft.com/sql/solutions/bi/default.mspx>

SQL Online Dokumentation

<http://msdn2.microsoft.com/de-de/library/ms130214.aspx>

Zur Installation der SQL-Datenbank-Beispiele

<http://msdn2.microsoft.com/de-de/library/ms161556.aspx>

ADOMD.NET

<http://msdn2.microsoft.com/en-us/library/ms123483.aspx>

MDX-Referenz

<http://msdn2.microsoft.com/de-de/library/ms145556.aspx>

Webcasts zu Business Intelligence

<http://www.microsoft.com/events/series/sqlserverbi.msp>

ZU XMLA

<http://www.xmlforanalysis.com/>