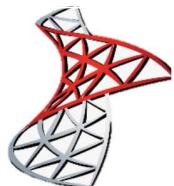


# PowerBI DAX – eine erweiterte Einführung

## Minimale technische Voraussetzungen



SQL Express Server (SQL SVR)



SQL Server Management Studio (SSMS)



AdventureWorksDW2016CTP3.bak als Import

## Minimale inhaltliche Voraussetzungen

Grundverständnis von Funktionen wie in Excel

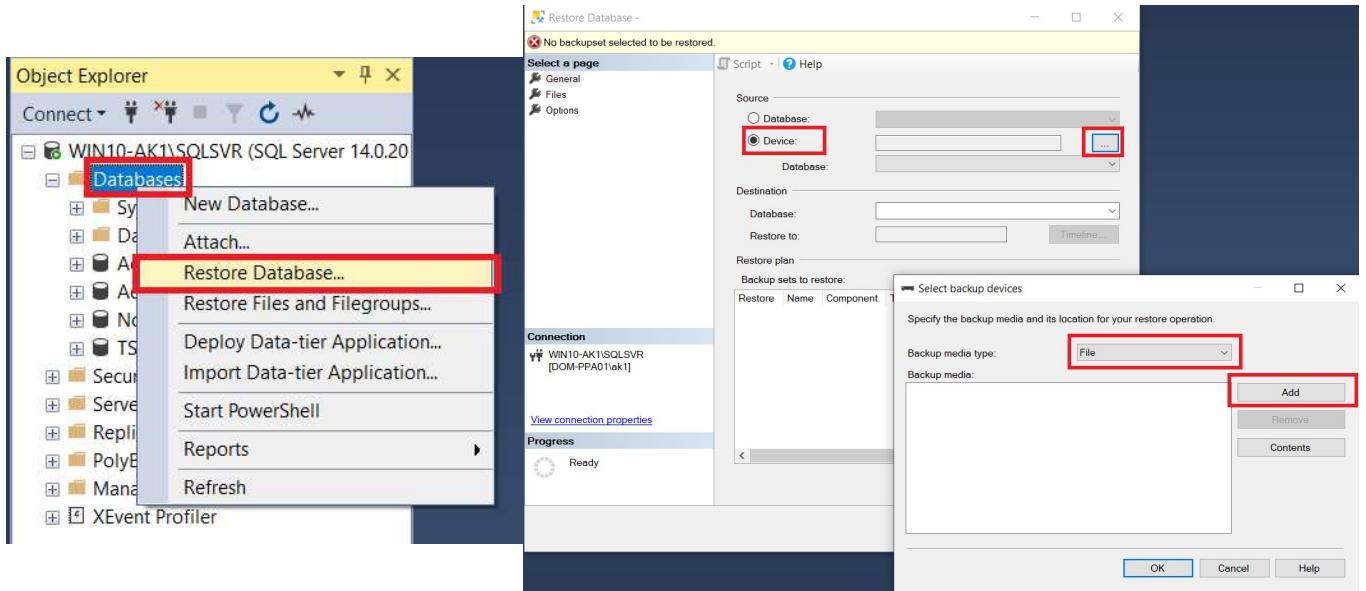
```
SUMX (  
    Sales,  
    Sales[Price] * Sales[Units]  
)
```

————— DAX Function  
————— Table in the data model  
————— Columns in the Sales table

# Datengrundlage schaffen

## Datenbank einpflegen

1. SSMS starten und **AdventureWorksDW2016CTP3.bak** als Import auswählen



2. Etwaige Optimierungen durchführen zwecks **Speicherort** oder dem Schließen von **Connections**.
3. Da **dbo** noch nicht **autorisiert** ist, muss das nachgeholt werden, sonst könnte es als Trainer haken.

Als Teilnehmer sehr schnell eine Fehlermeldung, wenn man ein **Scheme** anlegen will.

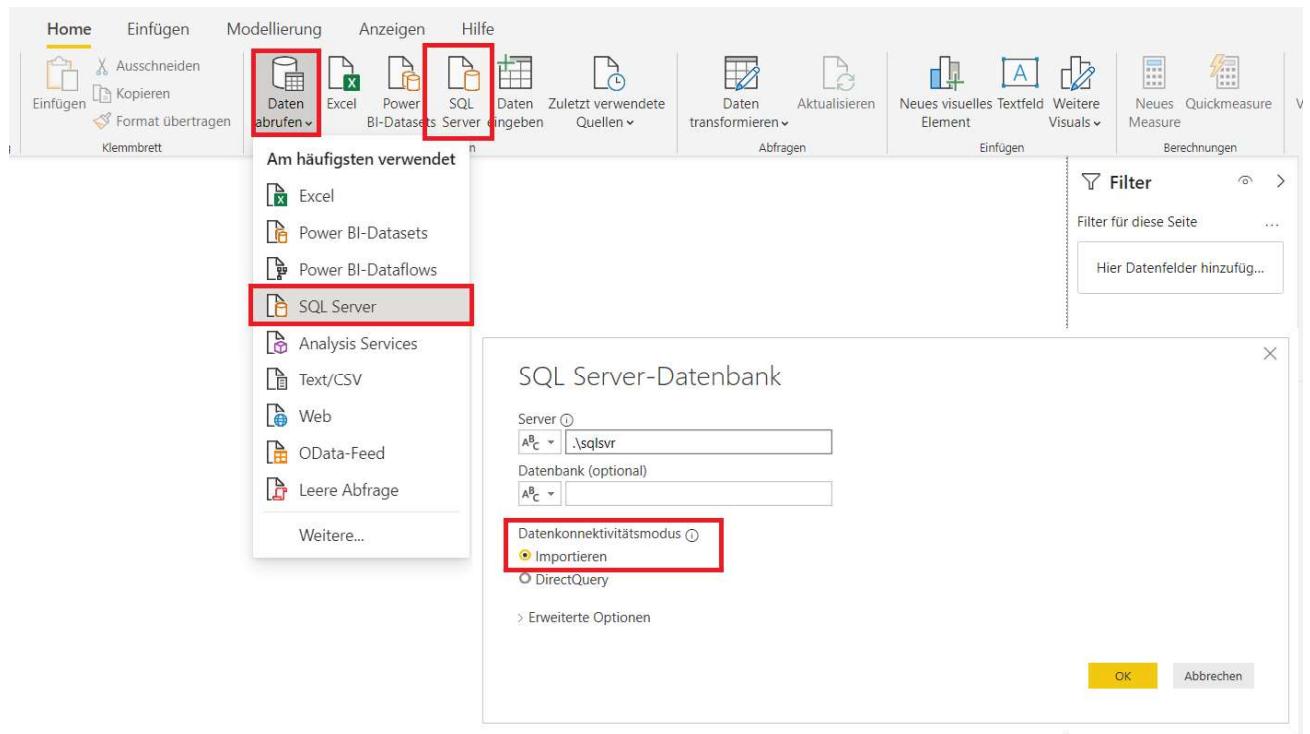
```
use [YourDatabaseName] EXEC sp_changecreator 'account'
```

Im konkreten Fall wäre das für mich:

```
use [AdventureWorksDW2016CTP3] EXEC sp_changecreator 'ak1'
```

4. Ferner muss ein **Script** ausgeführt werden, um **volle Zugriffsrechte** zu erlangen. Diese Schritte sind nicht notwendig, sollte man "nur" die Daten in PowerBI einbinden wollen, aber wenn man die Daten "vorstellen" möchte im **SSSMS**, dann könnten Fehlermeldungen erscheinen.

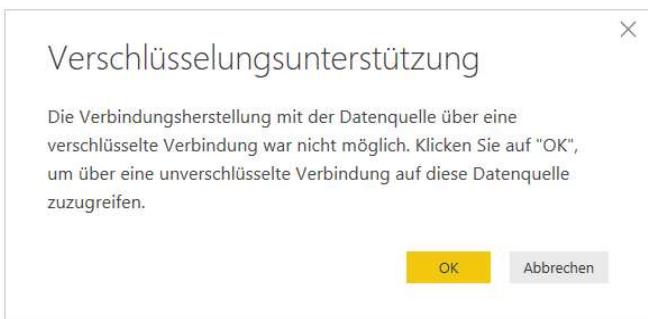
## Daten in PowerBI einlesen



1. Verbindung zum **SQL SVR** aufbauen. Unbedingt **Importieren** wählen, da sonst einige Measures eine Fehlermeldung ausgeben, dass sie nur bei importierten Daten funktionieren.
2. Es kann die **Windows-Authentifizierung** genommen werden, also die aktuellen Anmelddaten.



3. Die Test-DBs sind im Regelfall ohne **Zertifikate** oder weitere **Verschlüsselung** zugänglich.

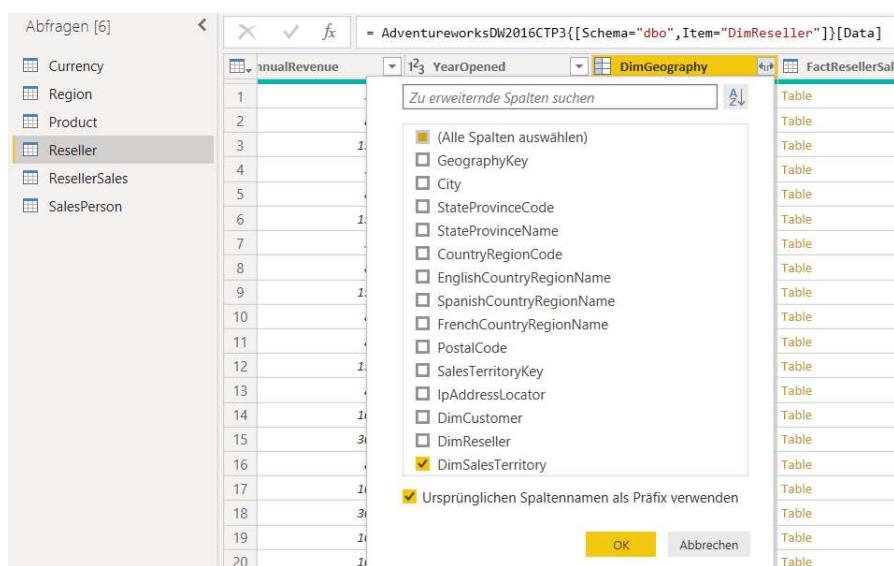


## Tabellen und Spalten auswählen

1. Folgende **Tabellen** auswählen und in der **Transformations**-Ebene umbenennen.

Tabellen-Name im Original	Neuer Name für einfachere Schreibweise
DimCurrency	Currency
DimProduct	Product
DimReseller	Reseller
DimEmployee	SalesPerson
FactResellerSales	ResellerSales
FactCurrencyRate	CurrencyRate

2. Bewusst **keine** verknüpften Tabellen und auch **nicht** DimDate auswählen!
3. Um langatmiges **RELATED()** unnötig zu machen, ent-pivotisiert man am besten gleich in der Tabelle **Reseller** die Spalte **DimGeography** zur **DimSalesTerritory** und in dieser wiederum...



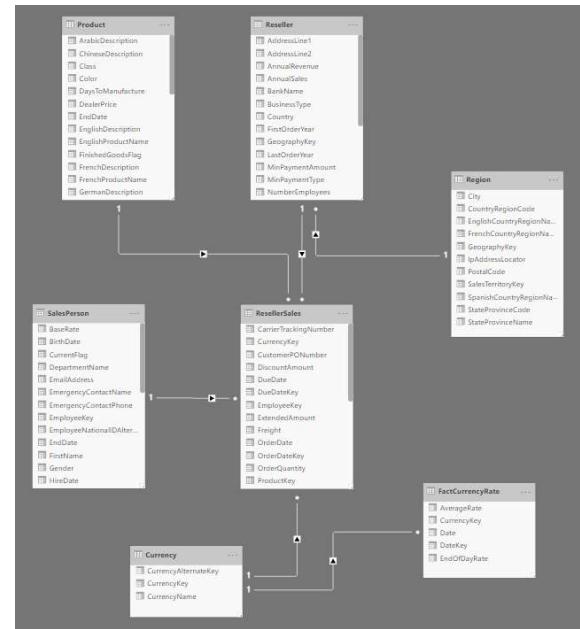
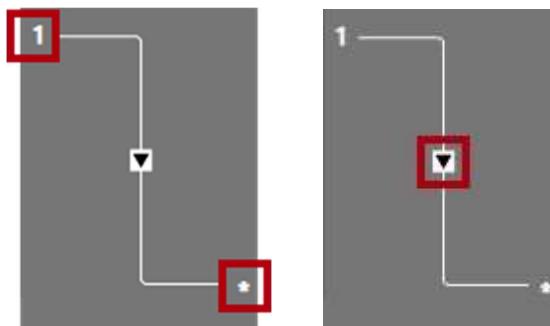
4. ... die Spalten **SalesTerritoryRegion** , **SalesTerritoryCountry** und **SalesTerritoryGroup** . Außerdem sollte man eine **Neubenamung** danach vornehmen; später besser lesbares DAX.

Spalten-Name im Original	Neuer Name für einfachere Schreibweise
SalesTerritoryRegion	Region
SalesTerritoryCountry	Country

5. Die **SalesTerritoryGroup** kann prinzipiell den Namen behalten, muss aber eben gekürzt werden, da im Original **DimGeography.DimSalesTerritory.SalesTerritoryGroup**

### Beziehungen überprüfen und anpassen

1. Wechseln in der linken oberen Ecke zur **Modellansicht**.
2. Das **Datenmodell** besteht aus sieben Tabellen, die nach dem rechten Schema entsprechende Beziehungen aufbauen.
3. Besonders zu beachten ist, dass die **Beziehungen** vom Typ [1:n] sind und die **Kreuzfeldrichtung** aktuell einseitig.



Nun könnte man im Sinne der Optimierung alle nicht relevanten Spalten **ausblenden**, um die Übersicht zu wahren und in der **Visualisierungs-** sowie in der **Modellierungs-Ebene** die verwendeten Spalten besser zu sehen.

# Erstellen einer Datumstabelle

Es gibt einen eleganten Weg, um im Datenmodell sich eine Datumstabelle automatisch schreiben zu lassen. Allerdings gibt es auch "interessante Phänomene" am Anfang einer solchen Erstellung, die mit wenigen Klicks gelöst werden können, aber eben zunächst ein Stirnrunzeln im Sinne von PowerBI bewirken können.

Stirnrunzeln senden

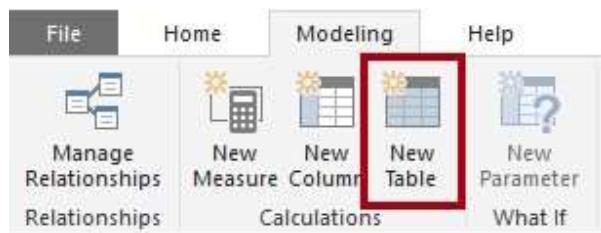
Schließen

## Erstellen einer berechneten Tabelle CALENDARAUTO

1. Wechseln zur Datenansicht.



2. Klicke im **Menüband Modellierung** innerhalb der Gruppe **Berechnungen** auf **Neue Tabelle**.



3. Gebe in der Bearbeitungsleiste ein:

### DAX

```
Date =  
CALENDARAUTO()
```



Die Funktion **CALENDARAUTO()** gibt eine einspaltige Tabelle zurück, die aus Datumswerten besteht. Das Verhalten "auto" scannt alle Datumsspalten des Datenmodells, um die frühesten und neuesten Datumswerte zu ermitteln, die im Datenmodell gespeichert sind. Anschließend wird für jedes Datum innerhalb dieses Bereichs eine Zeile erstellt, wodurch der Bereich in beide Richtungen erweitert wird, um sicherzustellen, dass vollständige Datenjahre gespeichert werden. Diese Funktion kann ein einzelnes optionales Argument verwenden, das die letzte Monatsnummer eines Jahres ist. Wenn nicht angegeben, ist der Wert 12, was bedeutet, dass Dezember der letzte Monat des Jahres ist.

---

*Tipp: "Leerraum" (d. h. NewLine und Tabs) in Layout-Formeln sorgen für in einem intuitiven und leicht lesbaren DAX-Text und sind besonders empfohlen, insbesondere wenn Formeln lang und komplex sind.*

*Deutsches Format: SHIFT + Enter*

*Englisches Format: CTRL + Enter*

---

- Beachte nun die Spalte mit den Datumswerten.

Date
01.01.1939 00:00:00
02.01.1939 00:00:00
03.01.1939 00:00:00
04.01.1939 00:00:00
05.01.1939 00:00:00
06.01.1939 00:00:00

**← was wir am Ende bekamen**

Date
01/01/2016 00:00:00
01/02/2016 00:00:00
01/03/2016 00:00:00
01/04/2016 00:00:00
01/05/2016 00:00:00
01/06/2016 00:00:00

**was wir vorher erwartet hätten →**

Die angezeigten Daten werden mit **regionalen Einstellungen** formatiert, falls PowerBI bzw. Betriebssystem (**de-DE**) "dd.mm.yyyy", ansonsten eben (**en-US**) "mm/dd/yyyy". Das erklärt aber nicht die Unterschiede zwischen den Datumsangaben der Bestellwerte und der **CALENDARAUTO**.

---

*Hinweis: CALENDARAUTO evaluiert stets ALLE Tabellen auf potentielle Datumswerte, weil ein **Kartesisches Produkt** aufgespannt wird **im Datenmodell** und dieses erkennt, dass es Mitarbeiter gibt, die laut Modell im Jahre 1939 geboren worden sind.*

TABELLE: Date (27.759 Zeilen)

*Eigentlich hätten **1096 Datenzeilen** generiert werden sollen, also **3 Jahre**.*

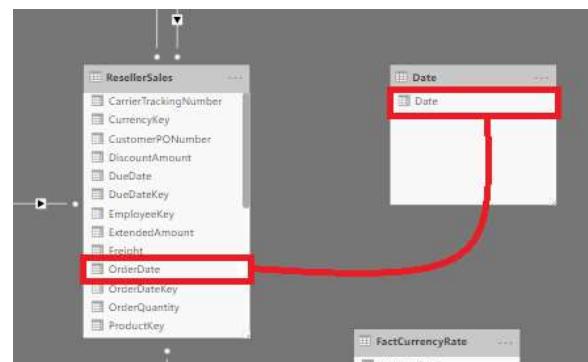
TABLE: Date (1,096 rows)

- Um also die Relevanz der Date-Tabelle zu definieren, muss die **fehlende Beziehung** gesetzt werden:

**ResellerSales.OrderDate = Date.Date**

Das wäre aber nur die Hälfte der Lösung!

Nun erstelle man noch ein paar Spalten (geht prinzipiell auch danach) und zeige noch einen inhaltlichen Fehler, bevor man das macht.



## Erstellen berechneter Spalten

1. Wähle **Neue Spalte** aus, entweder im **Kontext-Menü** der Tabelle oder in der **Registerkarte**.
2. CURRENT YIELD YEAR

### DAX

```
YEAR =  
"CY" & JAHR( [Datum] )
```

3. Überprüfen, ob die neue Spalte hinzugefügt wurde.

Date	Year
01/01/2016 00:00:00	CY2016
01/02/2016 00:00:00	CY2016
01/03/2016 00:00:00	CY2016
01/04/2016 00:00:00	CY2016
01/05/2016 00:00:00	CY2016
01/06/2016 00:00:00	CY2016

4. QUARTER / MONTH / DAY

### DAX

```
Quarter =  
[Year] & " Q"  
& IF (  
    MONTH ( 'Date'[Date] ) <= 3;  
    1;  
    IF (  
        MONTH ( 'Date'[Date] ) <= 6;  
        2;  
        IF (  
            MONTH ( 'Date'[Date] ) <= 9;  
            3;  
            4  
        )  
    )  
)
```

## DAX

Month =

FORMAT ( [Date].[Date] ; "YYYY MMM" )

## DAX

Day =

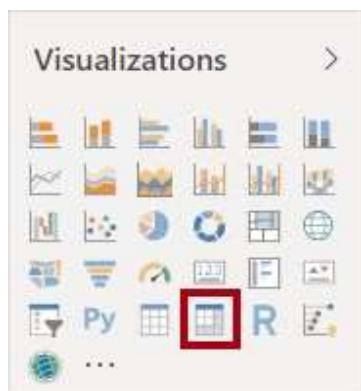
FORMAT( [Date] ; "YYYY MMM, DD")

Im Rohzustand muss im Regelfall '**Date**'.**[Date]** geschrieben werden. Wenn die Tabelle in einer Beziehung ist, dann eher **[Date].****[Date]**. Und wenn am Ende die Date-Type-Table existiert, dann **[Date]**, weil dann feststeht, wo die Primärspalte des Datumsfeldes liegt, was alles vereinfacht.

5. Das Ergebnis sollte so aussehen

Date	Year	Quarter	Month	Day
01/01/2016 00:00:00	CY2016	CY2016 Q1	2016 Jan	2016 Jan, 01
01/02/2016 00:00:00	CY2016	CY2016 Q1	2016 Jan	2016 Jan, 02
01/03/2016 00:00:00	CY2016	CY2016 Q1	2016 Jan	2016 Jan, 03
01/04/2016 00:00:00	CY2016	CY2016 Q1	2016 Jan	2016 Jan, 04
01/05/2016 00:00:00	CY2016	CY2016 Q1	2016 Jan	2016 Jan, 05
01/06/2016 00:00:00	CY2016	CY2016 Q1	2016 Jan	2016 Jan, 06

6. Um die Berechnungen zu überprüfen, wechsel zur Berichtsansicht.
7. Nutze das **Matrix Visual** zur Verifikation.



8. Ziehe im Bereich **Felder** aus der Tabelle **Date** die Felder **Year** und **Month** in die **Zeilen**. So wird eine Pseudo-Hierarchie gebaut, die jedem Jahr den passenden Monat zuordnet.

The screenshot shows the PowerBI Fields pane. On the left, under 'Visualizations', there are various chart icons. In the center, the 'Fields' pane has a search bar at the top. Below it, the 'Date' hierarchy is expanded, showing 'Year', 'Month', 'Day', and 'Quarter'. The 'Year' node is highlighted with a red box. On the right, the 'Rows' section is set up with 'Year' and 'Month' selected. A red arrow points from the 'Add data fields here' button in the Visualizations pane to the 'Month' field in the Rows section.

9. Neben dem **Matrix Visual** das **DropDown** auf die darunter-liegende Ebene **erweitern**.



10. Die Monate sind **alphabetisch und nicht chronologisch** sortiert, aber passend zugeordnet.

Year
CY2016
2016 Apr
2016 Aug
2016 Dec
2016 Feb
2016 Jan
2016 Jul
2016 Jun
2016 Mar
2016 May
2016 Nov
2016 Oct
2016 Sep

---

*Standardmäßig sortieren Textwerte alphabetisch, Zahlen sortieren von der kleinsten zur größten und Datumsangaben werden von frühester bis letzter sortiert. Das hier war allerdings eine STRING-Verkettung mittels & und damit das Ergebnis auch wieder Text.*

---

11. Um die **Reihenfolge** der Monatsfelds anzupassen, wechsel zur Datenansicht.

12. Die Spalte **MonthKey** ist die Grundlage einer erzwungenen Inhaltsreihenfolge.

#### DAX

```
MonthKey =  
( YEAR ( [Date] ) * 100 )  
+ MONTH ( [Date] )
```

*Diese Formel berechnet einen numerischen Wert für jede Jahres-Monats-Kombination.*

13. **Numerische Werte** weisen über 201601 dem Wert Januar 2016 eine **Eindeutigkeit** zu.

Date	Year	Quarter	Month	Day	MonthKey
01/01/2016 00:00:00	CY2016	CY2016 Q1	2016 Jan	2016 Jan, 01	201601
01/02/2016 00:00:00	CY2016	CY2016 Q1	2016 Jan	2016 Jan, 02	201601
01/03/2016 00:00:00	CY2016	CY2016 Q1	2016 Jan	2016 Jan, 03	201601
01/04/2016 00:00:00	CY2016	CY2016 Q1	2016 Jan	2016 Jan, 04	201601
01/05/2016 00:00:00	CY2016	CY2016 Q1	2016 Jan	2016 Jan, 05	201601
01/06/2016 00:00:00	CY2016	CY2016 Q1	2016 Jan	2016 Jan, 06	201601

14. Stellen Bereich **Felder** sicher, dass das Feld **Month** ausgewählt ist und klicke im **Menüband Modellierung** innerhalb der Gruppe **Sortieren** auf **Nach Spalte sortieren**, und wähle dann **MonthKey** aus. So wird eine Spalte nach dem numerischen Wert einer anderen Spalte sortiert.

The screenshot shows the PowerBI desktop interface with the 'Modeling' ribbon tab selected. In the 'Sort by Column' dropdown, 'MonthKey' is highlighted with a red box. To the right, a vertical list of sorting options is displayed, with 'MonthKey' also highlighted with a red box. The list includes: Month (Default) (selected), Date, Day, MonthKey, Quarter, and Year.

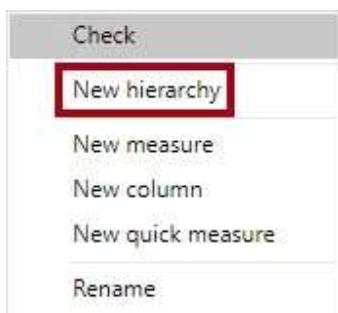
- Wiederhole das mit dem Feld **Day**, sodass es nach dem Feld **Date** sortiert wird. Funktioniert zwar im Regelfall auch so, aber das erzeugt eine weitere Eindeutigkeit der Zuordnung.
- Um das Feld **MonthKey** auszublenden, klicke im Bereich **Felder** mit der rechten Maustaste auf das Feld **MonthKey**, und wähle dann **Ausblenden** aus. Alternativ geht das auch in der **Modellansicht**.



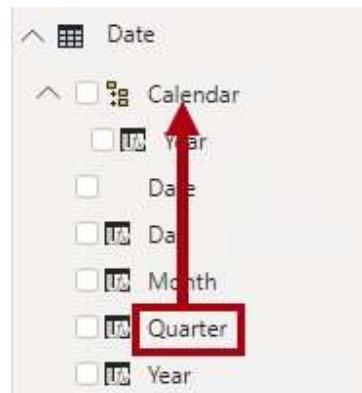
Die Spalte ist weiterhin im Datenmodell, wird aber jetzt für Berichtsautoren ausgeblendet.

## Erstellen einer DatumsHierarchie

- Um eine Hierarchie zu erstellen, klicke im Bereich **Felder** mit der rechten Maustaste auf das Feld **Year**, und wählen Sie dann **Neue Hierarchie** aus.



- Unbedingt die **Hierarchie umbenennen**, da Standardnamen für Hierarchien stets nicht so brillant. Ich habe jetzt mal den Namen **Kalender** gewählt.
- Um der Hierarchie eine Ebene hinzuzufügen, wähle bspw. **Quartal**, und lege es in der **Kalenderhierarchie** ab.
- Füge eine neues **Matrix Visual** hinzu und dort wiederum die Hierarchie. Ändere es so ab, das **Jahr** und **Monat** dort zu sehen sind.



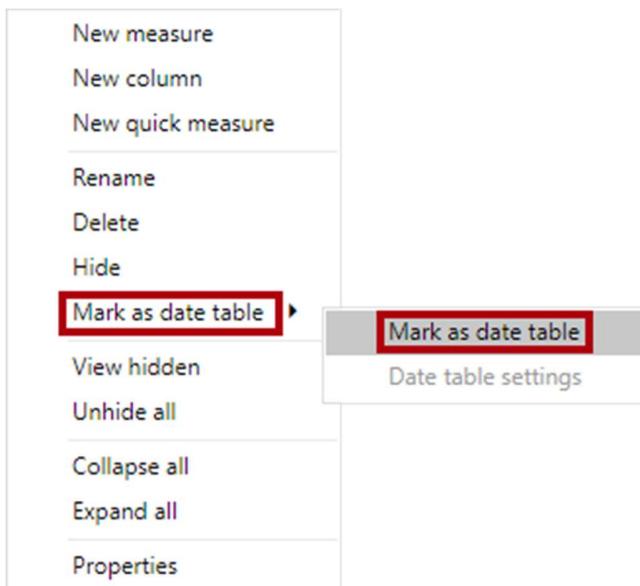
5. Erweitere im Matrix Visual die Jahre, um alle Monate anzuzeigen.



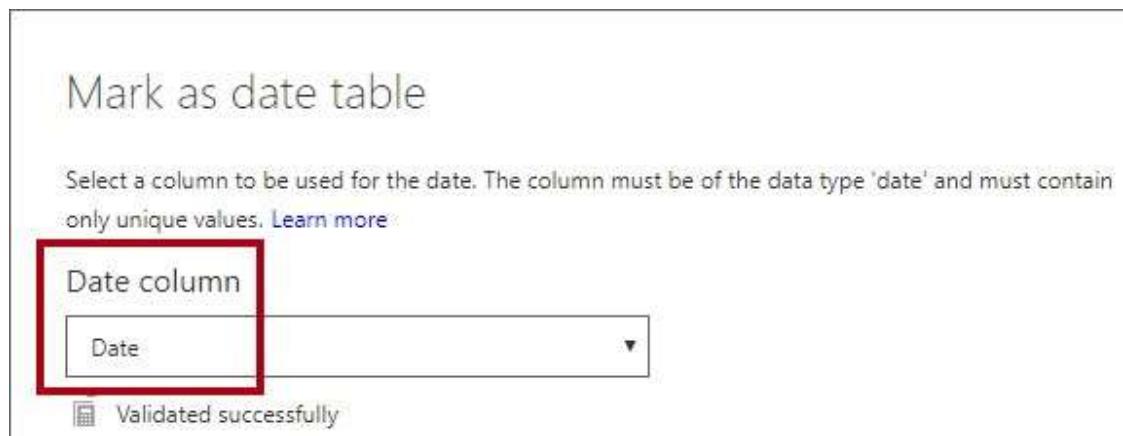
### Markieren der Datumstabelle

So bescheuert der Satz auch klingt: jetzt definieren wir die **Datumstabelle** als Datumstabelle (Date-Type), um sie als Datumstabelle verwenden zu können – damit wird der Standard überschrieben.

1. Im Bereich **Felder** mit der rechten Maustaste auf die Tabelle **Date** (nicht Feld!), und wähle dann **Als Datumstabelle markieren | Markieren Sie als Datumstabelle**.



2. Wählen Sie im Fenster in der Dropdownliste **Datumsspalte Date** aus.



3. Speicher die PowerBI Datei.

← super-wichtig, um Datenmodell zu evaluieren!

---

*PowerBI Desktop versteht nun, dass diese Tabelle Datum (Uhrzeit) definiert. Dies ist wichtig, wenn man sich auf Zeitintelligenz-Berechnungen verlassen will.*

*Der Datumstabellenentwurf wurde nun abgeschlossen. Es wurde noch keine Beziehung ausgewertet, um sicherzustellen, dass eine entsprechende Filterweitergabe stattfindet. Deshalb ist die Beziehung auch nur die halbe Lösung wie oben erwähnt!*

*Tipp: Dieser Entwurfsansatz ist geeignet, falls keine Datumstabelle in Quellen vorhanden ist. Wenn allerdings Zugriff auf ein Datawarehouse besteht oder eine allgemeinverbindliche Datumsdimensionstabelle geladen werden kann, sollte man dies tun, anstatt die Datumslogik in dem Datenmodell neu zu definieren. Sonst kommt eventuell Mist raus!*

---

Im Rohzustand musste noch im Regelfall '**Date'[Date]**' geschrieben werden. Wenn die Tabelle in einer **Beziehung** ist und nun die **Date-Type-Table** existiert, dann kann **[Date]** geschrieben werden, weil dann feststeht, wo die Primärspalte des Datumsfeldes liegt, was alles vereinfacht.

# Erstellen von Measures

## Erkunden von zusammenfassenden Spalten

1. Erweitere im Bereich **Felder** die Tabelle **Reseller Sales**. Dort gibt es Felder mit dem  $\Sigma$  -Symbol

The screenshot shows the 'Fields' pane in Power BI. A tree view is open under 'Reseller Sales'. Several fields are highlighted with red boxes: 'Cost Amount', 'Quantity', 'Sales Amount', and 'Unit Price'. Each of these highlighted fields has a small red box around its name and the mathematical symbol  $\Sigma$  to its left.

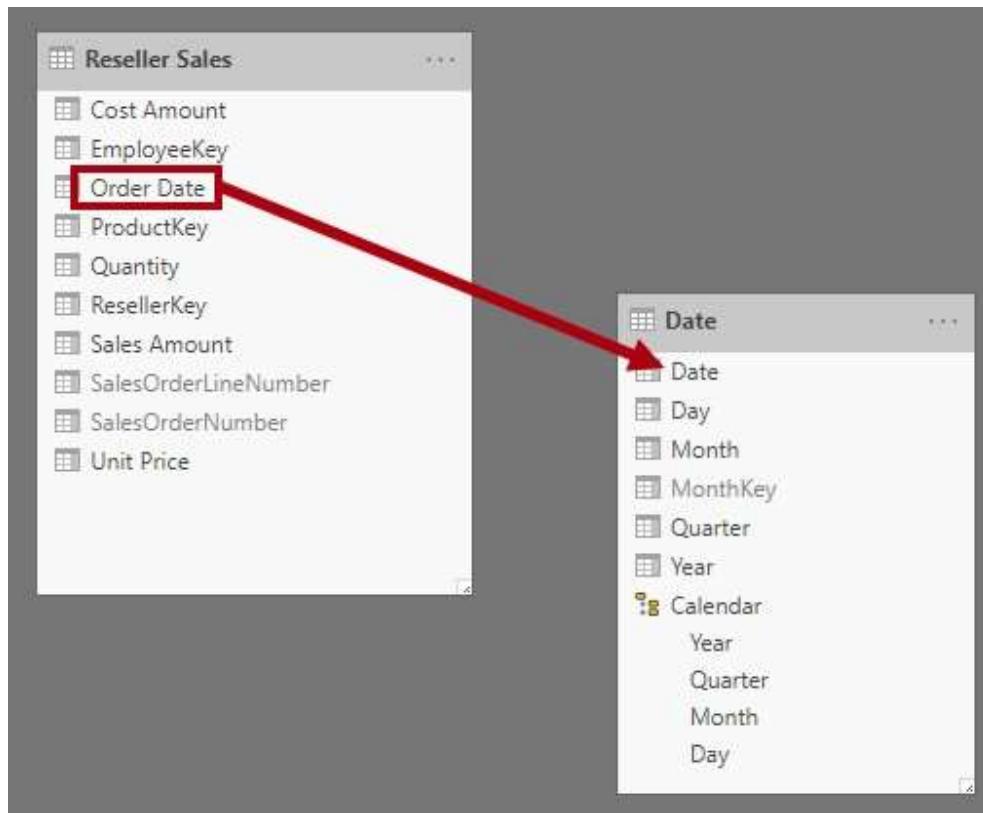
Diese Felder sind zusammenfassende Spalten. Diese Spalten haben immer einen numerischen Datentyp. Wenn sie in Visuals verwendet werden, werden sie standardmäßig zusammenfassen, um ein skalares Wertergebnis zu erzeugen. Dies kann durch die Verwendung einer Aggregatmethode wie Summe, Anzahl, min, max oder Average erreicht werden. Das Datenmodell kann eine Eigenschaft festlegen, um zu bestimmen, welche Aggregatmethode standardmäßig verwendet wird.

2. Hineinziehen der Umsätze sorgt dafür, dass identische Werte überall stehen.

The screenshot shows the 'Fields' pane and a table view side-by-side. In the Fields pane, the 'Sales Amount' field is selected and highlighted with a red box. A red arrow points from this highlighted field in the pane to the corresponding column in the table below. The table has columns for 'Year' and 'Sales Amount'. The data shows the same value, '\$80,450,596.9823', for every row from 'CY2016' down to '2016 May'. This visualizes how selecting a summarizing field creates uniform values across the data.

Year	Sales Amount
CY2016	\$80,450,596.9823
2016 Jan	\$80,450,596.9823
2016 Feb	\$80,450,596.9823
2016 Mar	\$80,450,596.9823
2016 Apr	\$80,450,596.9823
2016 May	\$80,450,596.9823

3. Nun hole man das Setzen der Beziehung nach, wie sie oben bereits erwähnt wurde.



4. Die Spalte **OrderDate** muss nicht sichtbar bleiben, da Filter nun auf die Spalte **Date** der Tabelle **Date** angewendet werden können; so war es in der **Date-Type-Table** definiert worden.
5. Nun sieht es bedeutend besser aus:

Year	Sales Amount
CY2016	\$16,288,441.7675
2016 Jan	\$489,328.5787
2016 Feb	\$1,538,408.3122
2016 Mar	\$1,165,897.0778
2016 Apr	\$844,720.9963
2016 May	\$2,324,135.7975

Year	Sales Amount	Average of Unit Price
CY2016	\$16,288,441.7675	\$748.6784
2016 Jan	\$489,328.5787	\$655.5874
2016 Feb	\$1,538,408.3122	\$758.9356
2016 Mar	\$1,165,897.0778	\$741.8509
2016 Apr	\$844,720.9963	\$677.4532
2016 May	\$2,324,135.7975	\$752.3062

The screenshot shows two tables side-by-side. The first table displays sales amounts by month for CY2016. The second table displays sales amounts and the average unit price by month for CY2016. A context menu is open over the third column of the second table, specifically over the 'Average of Unit Price' header. The menu is titled 'Values' and includes options: Remove field, Rename, Move to, Conditional formatting, Remove conditional formatting, Sum (checked), Average, Minimum, Maximum, Count (Distinct), Count, Standard deviation, Variance, and Median. The 'Sum' option is checked with a green checkmark.

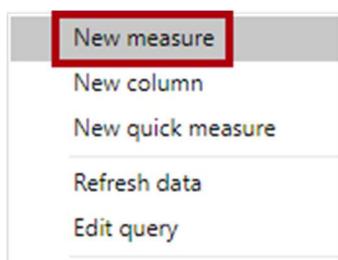
---

*Tipp: Sichtbare numerische Spalten ermöglichen es Berichtsauteuren zu entscheiden, wie eine Spalte zusammengefasst wird (oder nicht). Dies kann zu einer inhaltlich grob falschen Berichterstattung führen. Vorzugsweise sollte man die Dinge jedoch nicht dem Zufall überlassen und man entscheide sich bewusst dafür, reguläre Spalten auszublenden und stattdessen die durch Measures definierte Aggregationslogik verfügbar zu machen.*

---

## Erstellen von Measures

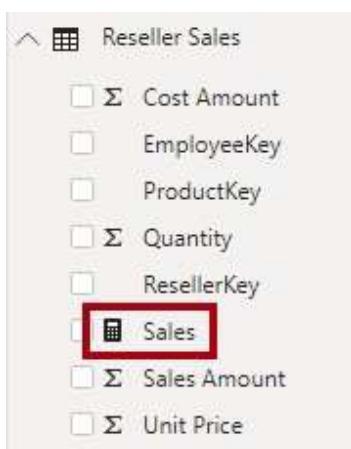
1. Klicke im Bereich **Felder** mit der rechten Maustaste auf die Tabelle **ResellerSales**, um dort neue Kennzahlen in Form von **Measures** zu generieren. So entstehen nicht zufällig an unbeabsichtigter Stelle diese, wodurch man regelmäßig beim Löschen und Neu-Erstellen ist.



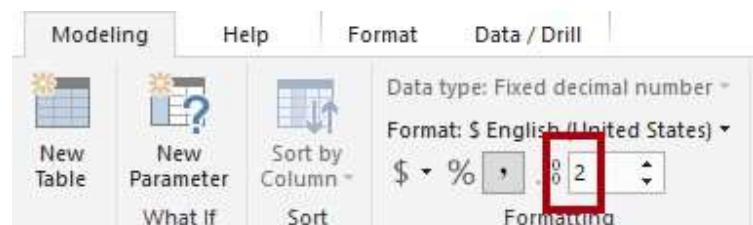
### DAX

```
Sales =  
SUM( [SalesAmount] )
```

2. Die neue Kennzahl wurde hinzugefügt und scheint erst einmal das gleiche zu machen als die AutoAggregation davor. Woraus besteht also der Mehrwert? Man kann diesen Measures ein **definiertes Format** zuweisen.



*Tipp: Es empfiehlt sich, immer sofort nach dem Erstellen eines Measures ein geeignetes Format dafür festzulegen.*



3. Legt auf diese Weise zwei Dezimalstellen fest.

4. Versuche, die Aggregationsmethode für die Kennzahl zu ändern... dies ist nicht möglich.

*Tipp: Bei Auswertung einzelner Teile eines Datums kann es mitunter die Lösung sein, **keine Hierarchie** zu bauen, um jeden Aspekt einzeln ansprechen zu können. Kommt auf einen Versuch an, was sich am Ende als Lösung herausstellt...*

5. Erstelle nun Measures für **Kosten** und **Durchschnittlichen Preis**, wobei jeweils zwei Dezimalstellen auszuweisen wären.

#### DAX

```
Cost =
SUM( [TotalProductCost] )
```

#### DAX

```
AvgPrice =
AVERAGE( [UnitPrice] )
```

6. Füge beide neuen Measures dem **Matrix Visual** hinzu.

Year	Sales	Cost	Avg Price
CY2016	\$16,288,441.77	\$16,297,676.82	\$748.68
2016 Jan	\$489,328.58	\$472,295.14	\$655.59
2016 Feb	\$1,538,408.31	\$1,469,459.58	\$758.94
2016 Mar	\$1,165,897.08	\$1,108,483.14	\$741.85
2016 Apr	\$844,721.00	\$817,129.53	\$677.45
2016 May	\$2,324,135.80	\$2,236,063.12	\$752.31

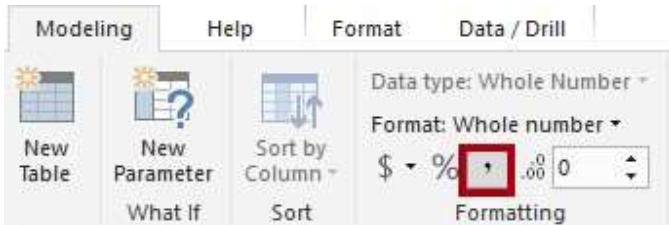
7. Auf die gleiche Weise erstelle man Kennzahlen für "**Aufträge**" und "**Auftragszeilen**" und formatieren Sie sie jeweils mit dem Tausender trennzeichen.

#### DAX

```
Orders =  
DISTINCTCOUNT( [SalesOrderNumber] )
```

#### DAX

```
OrderLines =  
COUNTROWS( 'ResellerSales' )
```



Die **DISTINCTCOUNT()**-Funktion zählt Aufträge nur duplikatfrei. Die **Funktion COUNTROWS()** arbeitet über eine Tabelle. In diesem Fall wird die Anzahl der Aufträge durch Zählen der unterschiedlichen **SalesOrderNumber-Spaltenwerte** berechnet, während die Anzahl der Auftragspositionen einfach die Anzahl der Tabellenzeilen ist (1 Zeile= 1 Auftrag).

Year	Sales	Cost	Avg Price	Orders	Order Lines
CY2016	\$16,288,441.77	\$16,297,676.82	\$748.68	739	8,459
2016 Jan	\$489,328.58	\$472,295.14	\$655.59	38	352
2016 Feb	\$1,538,408.31	\$1,469,459.58	\$758.94	75	785
2016 Mar	\$1,165,897.08	\$1,108,483.14	\$741.85	60	593
2016 Apr	\$844,721.00	\$817,129.53	\$677.45	40	499
2016 May	\$2,324,135.80	\$2,236,063.12	\$752.31	90	1,106

8. Und nun ein paar Kennzahlen für Gewinn , **Gewinnspanne** in % und in Kopie für die Optik.

## DAX

Profit =

[Sales] - [Cost]

## DAX

Profit % Margin =

DIVIDE( [Profit] ; [Sales] )

## DAX

Profit % Optik =

IF(

ISINSCOPE('Date'[Month]);

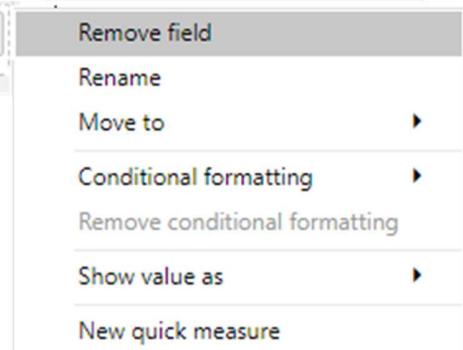
[Profit % Margin]

)

Die **DIVIDE()-Funktion** teilt zwei Ausdrücke, vorausgesetzt, dass das zweite Argument zu einer Zahl ungleich Null führt. Wenn das zweite Argument zu Null oder leer (fehlend) führt, wird die Funktion leer zurückgegeben. Ein optionales drittes Argument kann bereitgestellt werden, um einen alternativen Wert zurückzugeben, falls die Division fehlschlägt. Und die **ISINSCOPE()-Funktion** testet, ob die Monatsspalte die aktuelle Ebene in einer Hierarchie von Ebenen ist – so werden alle Zellen mit **100%** als Inhalt ausgeblendet, was für die **Bedingte Formatierung** besonders chic ist.

Year	Sales	Cost	Avg Price	Orders	Order Lines	Profit	Profit Margin
'2016	\$16,288,441.77	\$16,297,676.82	\$748.68	739	8,459	(\$9,235.05)	-0.06%
2016 Jan	\$489,328.58	\$472,295.14	\$655.59	38	352	\$17,033.44	3.48%
2016 Feb	\$1,538,408.31	\$1,469,459.58	\$758.94	75	785	\$68,948.73	4.48%
2016 Mar	\$1,165,897.08	\$1,108,483.14	\$741.85	60	593	\$57,413.93	4.92%
2016 Apr	\$844,721.00	\$817,129.53	\$677.45	40	499	\$27,591.47	3.27%
2016 May	\$2,324,135.80	\$2,236,063.12	\$752.31	90	1,106	\$88,072.68	3.79%

9. **Bedingte Formatierung** ist eine Eigenschaft des **Feldes** und kann dort formatiert werden. In diesem konkreten Beispiel wäre das bewusste Wählen eines **Minimums** von **-0,05** und eines **Maximums** von **0,05** am besten, um optische Verzerrungen zu reduzieren. Und nicht vergessen, die **100%** auszublenden!



## Profi-Tipps fürs Arbeiten mit Measures

Wenn alle regulären Spalten in einer Tabelle ausgeblendet sind und nur noch sichtbare Kennzahlen vorhanden sind, wird die Tabelle zu einer **Measuregruppentabelle** heraufgestuft. Dies ändert nichts an der Bewertung der Maßnahmen. Es positioniert einfach die Tabelle an der Spitze der Feldliste und schmückt den Tabellennamen mit einem anderen Symbol. Die Verfasser des Berichts sollten darin geschult werden, Measures bewusst nur aus der **Measure Group** zu beziehen.

1. In der Tabelle **Reseller Sales** die **CTRL-Taste gedrückt halten**, um alle sichtbaren Spalten auszuwählen und dann auszublenden.

The screenshot shows two panes from PowerBI Desktop. On the left is the 'Properties' pane, which includes sections for General, Synonyms, Description, Display folder, and a toggle switch labeled 'Is hidden' which is set to 'On'. On the right is the 'Fields' pane, which has a search bar at the top. Below it, the 'Reseller Sales' table is expanded, showing its columns: Avg Price, Cost, Order Lines, Orders, Profit, Profit Margin, Sales, and Currency. The 'Is hidden' switch in the Properties pane is highlighted with a red box, and the 'Reseller Sales' table in the Fields pane is also highlighted with a red box.

2. Klicke oben im Bereich **Felder** auf den Pfeil, um den Bereich zu reduzieren, kurz warten und dann erneut erweitern. Die Tabelle **ResellerSales** befindet sich jetzt oben in der Liste.

3. Erst beim Speicher der PowerBI Datei werden die Einstellungen final.

---

Häufig herrscht Verwirrung darüber, ob eine Datenmodellberechnung eine berechnete Spalte oder Kennzahl sein muss. Sie sind nie austauschbar. Eine berechnete Spalte ermöglicht das Filtern und Gruppieren, da es sich um einen ErgebnisVektor handelt, während eine Kennzahl eine Zusammenfassung ermöglicht, da Skalarer Wert am Ende.

---

Die Entscheidung, numerische Spalten auszublenden, hängt vom **Zweck des Datenmodells** ab. Das Ausblenden numerischer Spalten verringert die Möglichkeit für Berichtsautoren, Spaltenwerte auf unterschiedliche Weise zu aggregieren, es sei denn, sie werden explizit in einer Kennzahl definiert.

Das **AvgPrice-Measure** wäre ein solches Beispiel. Durch das Ausblenden der Spalte **UnitPrice** ist es nicht mehr möglich, die (Roh-) Daten in der Spalte unangemessen darzustellen (z. B. durch Summierung der Preise, was komplett sinnbefreit wäre). Es kann aber anders herum Sinn machen, wenn der Ersteller mal in einem Visual den Median des Stückpreises benötigt, da diese Kennzahl im Datenmodell nicht vorhanden ist, es aber auch keinen großen Aufguss geben soll, nur um mal etwas auf den kurzen Dienstweg auszuwerten.

Wenn der PowerBI Report mittels einer Live-Connection mit dem Datenmodell im **PowerBI-Service** verbunden ist, können Measures im Bericht erstellt werden, ohne dieses Problem zu tangieren, da man die Daten gewissermaßen "abonniert", aber keinen Einfluss mehr auf Ausgestaltung zwecks Datenpflege, Beziehungen oder sonst einen Aspekt des ETL-Prozesses zu haben.

YEAR	Sales	Cost	AvgPrice	Order	Profit	Profit % Margin	Profit % Optik
□ FJ2013	<b>33.574.834,16 €</b>	<b>34.066.704,16 €</b>	<b>396,84 €</b>	<b>1708</b>	<b>-491.870,01 €</b>	<b>-1,46 %</b>	
2013 Jan	4.212.971,51 €	4.463.792,72 €	342,20 €	185	-250.821,21 €	-5,95 %	
2013 Feb	4.047.574,04 €	4.325.499,07 €	343,41 €	176	-277.925,03 €	-6,87 %	
2013 Mai	3.510.948,73 €	3.479.959,75 €	412,52 €	176	30.988,99 €	0,88 %	
2013 Apr	3.483.161,40 €	3.460.754,70 €	395,81 €	178	22.406,70 €	0,64 %	
2013 Nov	3.416.234,85 €	3.431.054,15 €	406,70 €	180	-14.819,30 €	-0,43 %	
2013 Okt	3.314.600,78 €	3.340.896,76 €	391,86 €	179	-26.295,97 €	-0,79 %	
2013 Aug	2.738.653,62 €	2.732.009,69 €	485,27 €	174	6.643,92 €	0,24 %	
2013 Jul	2.699.300,79 €	2.687.669,51 €	477,84 €	174	11.631,28 €	0,43 %	
2013 Mrz	2.282.115,88 €	2.285.346,81 €	378,65 €	99	-3.230,94 €	-0,14 %	
2013 Sep	2.206.725,22 €	2.201.759,06 €	378,91 €	93	4.966,16 €	0,23 %	
2013 Jun	1.662.547,32 €	1.657.961,95 €	433,50 €	94	4.585,38 €	0,28 %	
□ FJ2012	<b>28.193.631,53 €</b>	<b>27.277.732,50 €</b>	<b>396,81 €</b>	<b>1277</b>	<b>915.899,04 €</b>	<b>3,25 %</b>	
2012 Jan	3.601.190,71 €	3.468.825,70 €	372,75 €	139	132.365,02 €	3,68 %	
2012 Apr	3.053.616,33 €	2.919.120,37 €	412,48 €	133	134.695,96 €	4,41 %	
2012 Feb	2.885.359,20 €	2.735.886,23 €	365,30 €	111	149.472,97 €	5,18 %	
2012 Okt	2.880.752,68 €	2.765.546,09 €	400,51 €	134	115.206,59 €	4,00 %	
2012 Dez	2.665.650,54 €	2.847.507,65 €	350,56 €	94	-181.857,11 €	-6,82 %	
2012 Jul	2.384.846,59 €	2.308.083,78 €	488,76 €	132	76.762,81 €	3,22 %	
2012 Mai	2.185.213,21 €	2.073.508,49 €	407,08 €	114	111.704,72 €	5,11 %	
2012 Nov	1.987.872,71 €	1.887.464,16 €	390,32 €	102	100.408,55 €	5,05 %	
2012 Sep	1.865.278,43 €	1.782.337,13 €	369,78 €	74	82.941,31 €	4,45 %	
2012 Mrz	1.802.154,21 €	1.724.517,59 €	382,00 €	73	77.636,63 €	4,31 %	
2012 Aug	1.563.955,06 €	1.497.549,93 €	450,03 €	106	66.405,15 €	4,25 %	
2012 Jun	1.317.541,83 €	1.267.385,38 €	428,87 €	65	50.156,45 €	3,81 %	

# Erzwingen der Sicherheit auf Zeilenebene

## Erstellen einer Rolle

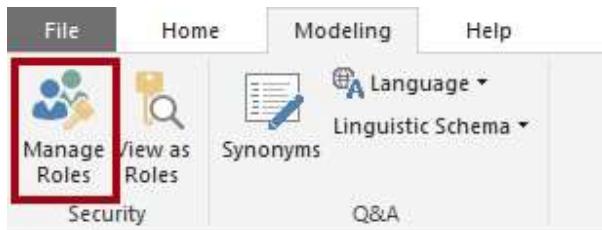
In dieser Aufgabe erstellen Sie eine Rolle, um die Vertriebsdaten für Wiederverkäufer basierend auf der Vertriebsregion zu begrenzen, zu der der aktuelle Benutzer gehört.

1. Die Tabelle **SalesPerson** hat Spalten für **E-Mail-Adresse** und **SalesTerritoryKey** enthält.

EmployeeKey	Salesperson	Email Address	SalesTerritoryKey
272	Stephen Jiang	s.jiang@adventureworks.com	11
281	Michael Blythe	m.blythe@adventureworks.com	2
282	Linda Mitchell	l.mitchell@adventureworks.com	4
283	Jillian Carson	j.carson@adventureworks.com	3
284	Garrett Vargas	g.vargas@adventureworks.com	6

Jeder Verkäufer verwendet eine E-Mail-Adresse, um sich beim **PowerBI-Service** zu authentifizieren, und ist einer einzelnen Vertriebsregion zugewiesen. Filter, die auf diese Tabelle angewendet werden, werden an die **ResellerSales** weitergegeben.

2. Unter **Modellierung** innerhalb der **Sicherheitsgruppe** wähle **Rollen verwalten | Erstellen**.



3. Neue Rolle **Salespeople** unter Verwendung der Tabelle **SalesPerson** einen **Filter hinzufügen | [SalesTerritoryKey]**

The screenshot shows the 'Manage roles' dialog in Power BI desktop. On the left, under 'Roles', the 'Salespeople' role is selected and highlighted with a red box. On the right, a 'Tables' pane lists several tables: Currency, Date, Product, Region, Reseller, Reseller Sales, and Salesperson. The 'Salesperson' table is also highlighted with a red box.

4. Ersetzen im Bereich **Tabellenfilter DAX-Ausdruck** den gesamten Ausdruck

## DAX

```
[SalesTerritoryKey] =  
LOOKUPVALUE (  
    Salesperson [SalesTerritoryKey] ;  
    [EmailAddress] ;  
    USERNAME()  
)
```

Die **USERNAME()**-Funktion ruft den **USER Principal Name** (UPN) des authentifizierten Benutzers ab. Die Funktion **LOOKUPVALUE()** ruft den **SalesTerritoryKey-Wert** für die Zeile ab, die einen E-Mail-Adresswert hat, der mit dem UPN des Benutzers übereinstimmt. Dieser Ausdruck hat den Effekt, dass die Tabelle **Reseller** nach den Verkäufern gefiltert wird, die derselben Verkaufsregion wie der aktuelle Benutzer im Office365 zugewiesen sind, ob nun im SharePoint oder im PowerBI aufgerufen.

5. Klicke auf **Speichern**. Schließen allein reicht nicht!



## Testen der Rolle

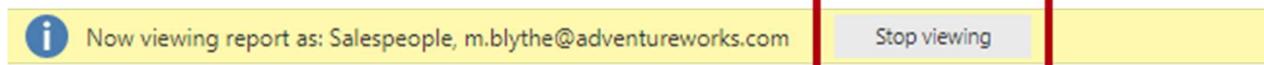
Gesamtumsatz für **CY2016** beträgt **\$ 16.288.441,77** für alle Vertriebsregionen.

Year	Sales	Cost
<b>CY2016</b>	<b>\$16,288,441.77</b>	\$16,297,676.82
2016 Jan	\$489,328.58	\$472,295.14
2016 Feb	\$1,538,408.31	\$1,469,459.58
2016 Mar	\$1,165,897.08	\$1,108,483.14
2016 Apr	\$844,721.00	\$817,129.53
2016 May	\$2,324,135.80	\$2,236,063.12

Wähle unter **Modellierung** innerhalb der **Sicherheitsgruppe | Ansicht als Rollen** aus.



Diese Einstellung testet nur die Salespeople-Rolle für eine Person.

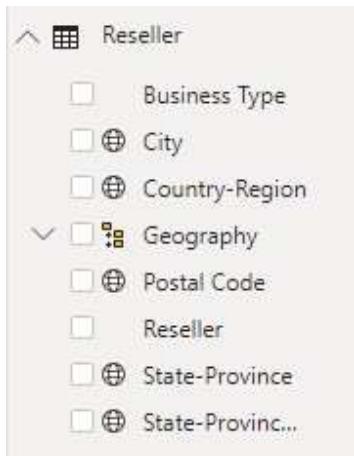


# SWITCH CASE Selector

## Erstellen einer berechneten Spalte

In diesem Vorgang erstellen Sie eine berechnete Spalte, die auf Kennzahlen verweist.

1. Der Measure wird in der Tabelle **Reseller** erstellt.



### DAX

```
Rating =  
SWITCH (  
    TRUE ;  
    ISBLANK ( [Orders] ) || [Orders] < 5 ; "Very Low" ;  
    [Orders] < 10 ; "Low" ;  
    [Sales] < 750000 ; "Middle" ;  
    "High"  
)
```

Die **SWITCH()-Funktion** testet mehrere Fälle, um jeden Fall mit **TRUE** zu vergleichen, bis eine Übereinstimmung gefunden wird. Ein Fall besteht aus zwei Argumenten: dem Test und dem Ergebnis, sollte der Test zu **TRUE** ausgewertet werden.

---

Die Tests innerhalb von **SWITCH** werden mittels **DAX Operatoren** gebildet; diese leiten sich von den Logischen Schaltern innerhalb des C++ ab und sind hier verlinkt:

<https://docs.microsoft.com/de-de/dax/dax-operator-reference>

---

Im konkreten Fall handelt es sich bei **||** um ein Boolean **OR**, um zwischen **NUL** und einem Wert kleiner 5 zu testen, und der Wert "Hoch" ist ein **CATCH ALL - Block** und zwingend empfohlen!

---

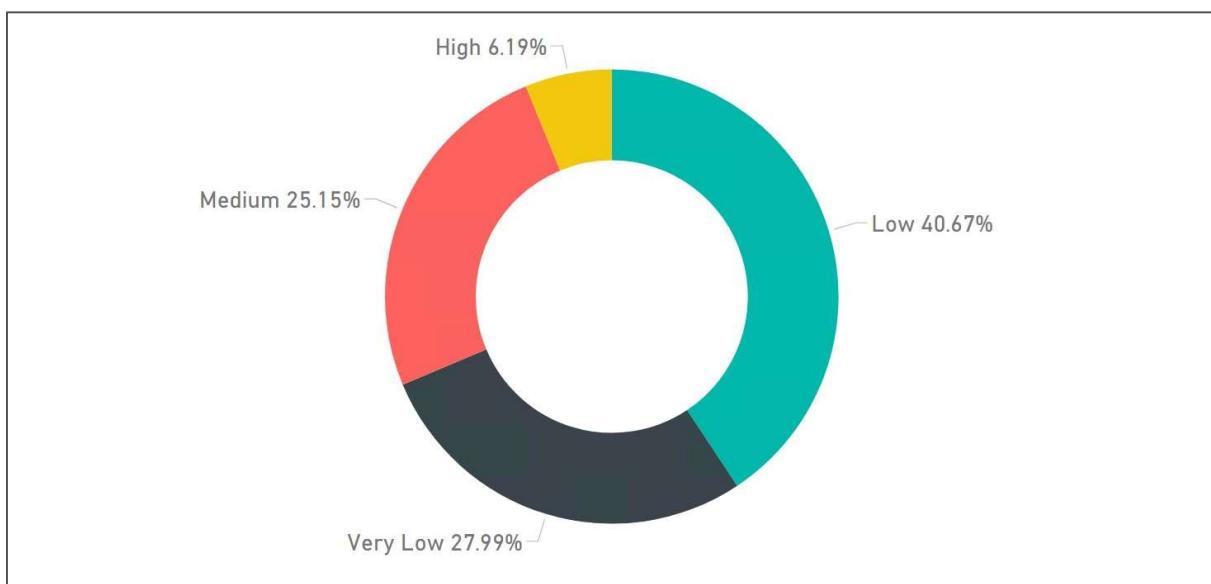
Diese berechnete Spalte bezieht sich auf **Measures**, um eine Bewertung der Umsätze **zeilenweise** durchzuführen, d. h. die Formel wird für jede Zeile in **Reseller** ausgewertet. Measures werden jedoch regulär im **Filterkontext** ausgewertet, von dem es hier aber keiner angegeben wurde. Die Auswertung von Measures innerhalb einer berechneten Spalte wird einem **Kontextübergang** unterzogen, bei dem der **Zeilenkontext** als Filterkontext angewendet wird. Dadurch wird sichergestellt, dass die Measures für jeden Wiederverkäufer (Reseller) in der Tabelle pro Zeile ausgewertet werden.

Dieses abstrakte Konzept ist wichtig zu verstehen, da es grundsätzlich zwei verschiedene Arten der **Berechneten Werte** gibt. Der Zeilenkontext gibt einen **Ergebnis-Vektor** zurück und der Filterkontext einen **Skalaren Wert** als Resultat der Berechnung.

---

State-Province	Country-Region	Postal Code	Loyalty Class
California	United States	94536	Low
California	United States	93010	Very Low
California	United States	94583	Low
California	United States	94587	Low
California	United States	90245	Low
California	United States	95035	Low
California	United States	92625	Very Low

6. Zur Visualisierung wähle das **Donut Visual** aus.



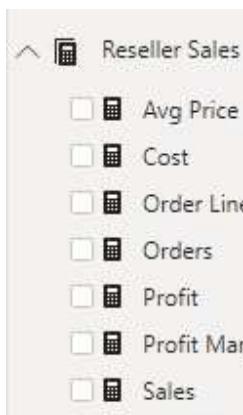
# Verwenden des Filterkontexts

1. Zunächst erstelle eine **Region-Hierarchie** in der Tabelle **Reseller**.



Bei Adventure Works sind die Vertriebsregionen in Gruppen, Länder und Regionen gegliedert. Alle Länder mit Ausnahme der Vereinigten Staaten haben eine Region, die nach dem Land benannt ist. Da die Vereinigten Staaten ein so großes Verkaufsgebiet sind, sind sie in fünf Regionen unterteilt. Die NA-Region repräsentiert die Unternehmenszentrale.

SalesTerritoryGroup
Europe
France
France
Germany
Germany
United Kingdom
United Kingdom
North America
Canada
Canada
United States
Central
Northeast
Northwest
Southeast
Southwest
Pacific
Australia
Australia
Total



2. Erweitere nun die Tabelle **ResellerSales**.

In dieser Übung fügen wir dieser Tabelle viele Measures hinzu, die den Filterkontext bearbeiten, um Ergebnisse zu erzielen, die über die vereinfachte Aggregation von Spalten hinausgehen.

*Tipp:* Um zu verhindern, dass man unbeabsichtigt in einer anderen Tabelle die Measures anlegt, empfiehlt es sich, eines der Measures der Tabelle anzuklicken, sodass diese grau hinterlegt ist, und danach ein neues Measure zu erstellen.

- 3.

## DAX

```
Sales All Region =
CALCULATE ( [Sales]; REMOVEFILTERS ( Region ) )
```

Die **CALCULATE()-Funktion** wird zum Bearbeiten des Filterkontexts verwendet. Das erste Argument nimmt einen Ausdruck oder eine Kennzahl (ein Measure ist nur ein benannter Ausdruck). Nachfolgende Argumente ermöglichen das Ändern des Filterkontexts. Die **Funktion REMOVEFILTERS()** entfernt aktive Filter. Es kann entweder keine Argumente oder eine Tabelle, eine Spalte oder mehrere Spalten als Argument verwenden.

In dieser Formel wertet Umsatzkennzahl in einem geänderten Filterkontext aus, der alle Filter entfernt, die auf die Tabelle **Reseller** angewendet werden.

- Das Einfügen dieses Measures führt wieder zu einem "interessanten Phänomen".

Sales All Region	← was wir am Ende bekamen	Sales All Region
\$80,450,596.9823		1.594.335,38 €
\$80,450,596.9823		1.594.335,38 €
\$80,450,596.9823		1.594.335,38 €
\$80,450,596.9823		67.985.726,81 €
\$80,450,596.9823		14.377.925,60 €
\$80,450,596.9823		14.377.925,60 €
\$80,450,596.9823		53.607.801,21 €
\$80,450,596.9823		7.906.008,18 €
\$80,450,596.9823		6.932.842,01 €
\$80,450,596.9823		12.435.076,00 €

- Dieser Wert wird sich ganz am Ende ändern, wenn **weitere Measures** drum herum gebaut sind, der PowerBI Report gespeichert, geschlossen und wieder geöffnet wurde... und ein paar Anpassungen, die davor gemacht werden, wie z.B. **CULTURE**, **Währungseinheit**, **Dezimalen**, ...

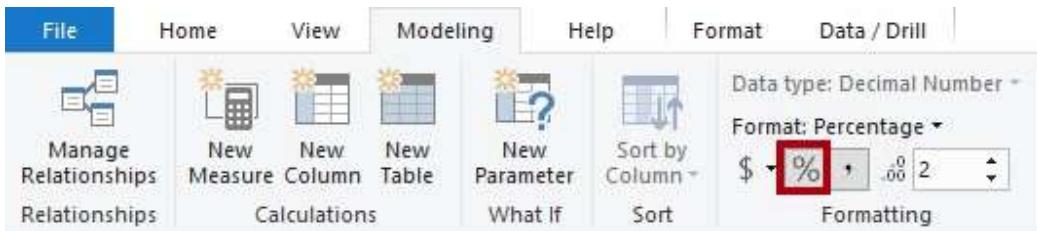
Aktuell wird die Summe aller Regionsverkäufe für jede Region, jedes Land (Zwischensumme) und jede Gruppe (Zwischensumme) berechnet. Wieso wird aber überall dieselbe Wert angegeben? Das folgende Measure wird uns einen Anhaltspunkt für den Grund dahinter geben.

#### DAX

```
Sales % All Region =  
DIVIDE (  
    [Sales];  
    CALCULATE (  
        [Sales];  
        REMOVEFILTERS ( Region )  
    )  
)
```

Die **Funktion DIVIDE()** teilt die Sales-Kennzahl (nicht nach Filterkontext geändert) durch die Sales-Kennzahl in einem geänderten Kontext, der alle auf die Tabelle **Region** angewendeten Filter entfernt durch **REMOVEFILTERS()**.

- Nach dem Hinzufügen den Inhalt als **Prozentsatz formatieren**.



7. Neue Darstellung ergibt, dass jedes Element die Filterung gegenüber sich selbst anwendet.

SalesTerritoryGroup	Sales All Region	Sales % All Region
Europe	10.870.534,80 €	100,00 %
France	4.607.537,94 €	100,00 %
France	4.607.537,94 €	100,00 %
Germany	1.983.988,04 €	100,00 %
Germany	1.983.988,04 €	100,00 %
United Kingdom	4.279.008,83 €	100,00 %
United Kingdom	4.279.008,83 €	100,00 %
North America	67.985.726,81 €	100,00 %
Canada	14.377.925,60 €	100,00 %
Canada	14.377.925,60 €	100,00 %
United States	53.607.801,21 €	100,00 %
Central	7.906.008,18 €	100,00 %
Northeast	6.932.842,01 €	100,00 %
Northwest	12.435.076,00 €	100,00 %
Southeast	7.867.416,23 €	100,00 %
Southwest	18.466.458,79 €	100,00 %
Pacific	1.594.335,38 €	100,00 %
Australia	1.594.335,38 €	100,00 %
Australia	1.594.335,38 €	100,00 %
Total	80.450.596,98 €	100,00 %

8. Ein anderes Measure löst dieses Problem, indem der **Filter korrekt gesetzt und ausgewertet** wird.

## DAX

Sales % Country =

VAR pattern =

IF(

```
ISINSCOPE(Reseller[Region]) ;
    DIVIDE( [Sales] ;
        CALCULATE(
            [Sales] ; REMOVEFILTERS(Reseller[Region])
        )
    )
)
RETURN
IF( pattern = 1 ;
    "" ;
    pattern
)
```

## DAX

```
Sales % Country =  
VAR pattern =  
IF(  
    ISINSCOPE(Reseller[Region]);  
    DIVIDE( [Sales] ;  
        CALCULATE(  
            [Sales] ; REMOVEFILTERS(Reseller[Region]))  
    )  
)  
)  
RETURN  
IF( pattern = 1 ;  
    "";  
    pattern  
)
```

← hier wird eine Variable "pattern" definiert  
← Falls ...  
← ... die **Reseller[Region]** sichtbar ist, ...  
← ..., dann teile die **aktuellen Umsätze** ...  
← ... als **zeilenweise** Auswertung ...  
← ... der Umsätze, auf dem ein **Region-Filter**  
die **relativen 100 %** jeweils auf Höhe der  
**Länder** ansetzt und die **Regionen als Anteil**  
in **Prozent** berechnet, ...  
← ... , um dieses **Resultat** auszugeben, damit es ...  
← ... **überprüft** wird, ob wir 100% ausgeben, ...  
← ... , um es durch **leeren String** zu ersetzen ...  
← ... oder es **in die aktuelle Zeile** zu schreiben.

- Beachten Sie, dass die Formel "**Sales % Country**" leicht von der Formel "**Sales %**" abweicht.

## DAX

```
Sales % All Region =  
DIVIDE (  
    [Sales];  
    CALCULATE (  
        [Sales];  
        REMOVEFILTERS ( Region ))  
)
```

Der Nenner ändert den Filterkontext, indem der Filter für die Region-Spalte der Tabelle **Region** entfernt werden, nicht aber für alle Spalten der Tabelle. Dies bedeutet, dass alle Filter, die auf die Gruppen- oder Länderspalten angewendet werden, beibehalten werden. Dadurch wird ein Ergebnis erzielt, das den **Umsatz in Prozent des Landes** korrekt darstellt.

10. Beachte, dass nur die Regionen der Vereinigten Staaten einen Wert erzeugen, der nicht 100%

SalesTerritoryGroup	Sales	Sales % All Region	Sales % Country
□ Europe	10.870.534,80 €	100,00 %	
□ France	4.607.537,94 €	100,00 %	
France	4.607.537,94 €	100,00 %	
□ Germany	1.983.988,04 €	100,00 %	
Germany	1.983.988,04 €	100,00 %	
□ United Kingdom	4.279.008,83 €	100,00 %	
United Kingdom	4.279.008,83 €	100,00 %	
□ North America	67.985.726,81 €	100,00 %	
□ Canada	14.377.925,60 €	100,00 %	
Canada	14.377.925,60 €	100,00 %	
□ United States	53.607.801,21 €	100,00 %	
Southwest	18.466.458,79 €	34,45 %	34,45 %
Northwest	12.435.076,00 €	23,20 %	23,20 %
Central	7.906.006,18 €	14,75 %	14,75 %
Southeast	7.867.416,23 €	14,68 %	14,68 %
Northeast	6.932.842,01 €	12,93 %	12,93 %
□ Pacific	1.594.335,38 €	100,00 %	
□ Australia	1.594.335,38 €	100,00 %	
Australia	1.594.335,38 €	100,00 %	
Gesamt	80.450.596,98 €	100,00 %	

beträgt, da dieser Regionen besitzt, die nicht bereits identisch mit dem Land lauten.

Um nun nicht mit 100% optisch überfrachtet zu werden, blendet man mit dem **Variablen-Trick [pattern]** all jene Elemente aus, die den Wert 1 = 1.00 = 100% zurückgeben.

*Eingebettet in IF0 wird ISINSCOPE() verwendet, um zu testen, ob die Regionsspalte die Ebene in einer Hierarchie von Ebenen ist. Wenn TRUE, wird die DIVIDE()-Funktion ausgewertet. Das Fehlen eines falschen Teils bedeutet, dass leer zurückgegeben*

wird, wenn sich die Regionsspalte nicht im Gültigkeitsbereich befindet; es werden also initial weniger Werte berechnet, unter denen der [pattern] ausgewertet wird.

Würde man nicht gegen den [pattern] testen, würden sämtliche Region-Werte ausgegeben werden, da die nur dann einen Wert zurückgibt, wenn sich eine Region im Gültigkeitsbereich befindet.

11. Nun wird derselbe Ansatz für **Sales % Group** berechnet. Dies geschieht in mehreren Stufen bis zum gewünschten Ergebnis. Man erstelle also einen passendes Measure, als Prozentsatz formatiert:

### DAX

```
Sales % Group =
DIVIDE (
    [Sales];
    CALCULATE (
        [Sales];
        REMOVEFILTERS ( Region [Region] ) ; Region [Country]
    )
)
```

SalesTerritoryGroup	Sales % Group
□ Europe	100,00 %
□ France	42,39 %
France	42,39 %
□ Germany	18,25 %
Germany	18,25 %
□ United Kingdom	39,36 %
United Kingdom	39,36 %
□ North America	100,00 %
□ Canada	21,15 %
Canada	21,15 %
□ United States	78,85 %
Southwest	27,16 %
Northwest	18,29 %
Central	11,63 %
Southeast	11,57 %
Northeast	10,20 %
□ Pacific	100,00 %
□ Australia	100,00 %
Australia	100,00 %
Gesamt	100,00 %

Um einen Umsatz als Prozentsatz der Gruppe zu erzielen, können **zwei Filter zusammen** angewendet werden, um die Filter auf zwei Spalten effektiv zu entfernen, hier Region und Country.

12. Füge die **Kennzahl "Sales % Group"** zum Matrix-Visual hinzu, das Ergebnis zu vergleichen.

13. Um die Lesbarkeit dieser Kennzahl erneut zu verbessern, ergänze den Ansatz mit **ISINSCOPE()**:

### DAX

Sales % Group =

IF(

ISINSCOPE(Reseller[Region]);

DIVIDE( [Sales] ;

CALCULATE(

[Sales] ;

REMOVEFILTERS(Reseller[Region] ; Reseller[Country])

)

)

)

SalesTerritoryGroup	Sales % Group
Europe	
France	42,39 %
Germany	18,25 %
United Kingdom	39,36 %
North America	
Canada	21,15 %
United States	
Southwest	27,16 %
Northwest	18,29 %
Central	11,63 %
Southeast	11,57 %
Northeast	10,20 %
Pacific	
Australia	
Australia	100,00 %
Gesamt	

14. Beachte, dass **Sales % Group** jetzt nur dann einen Wert zurückgibt, wenn sich eine Region im Gültigkeitsbereich befindet. Dies soll nun um das Land ergänzt werden:

### DAX

Sales % Group =

IF(

ISINSCOPE(Reseller[Region]) || ISINSCOPE(Reseller[Country]) ;

DIVIDE( [Sales] ;

CALCULATE(

[Sales] ;

REMOVEFILTERS(Reseller[Region] ; Reseller[Country])

)

)

)

SalesTerritoryGroup	Sales % Group
Europe	
France	42,39 %
Germany	18,25 %
United Kingdom	39,36 %
North America	
Canada	21,15 %
United States	78,85 %
Southwest	27,16 %
Northwest	18,29 %
Central	11,63 %
Southeast	11,57 %
Northeast	10,20 %
Pacific	
Australia	100,00 %
Australia	100,00 %
Gesamt	

Nun sind die Zahlen korrekt, was die **Country** anbelangt, weil Prozentanteile pro **Territory** berechnet werden, die sich zu 100% passend aufsummieren. Nun sind leider die **Regions** in den Vereinigten Staat relativ Quatsch, weil diese sich auf **78,86%** aufsummieren und nicht auf **100%**, also nicht korrekterweise die **relativen Anteile** innerhalb der Region, sondern absoluten Werte als relativen Anteil der Region innerhalb des Landes **in Relation zum Beitrag an der SalesTerritory**.

15. Dies soll nun korrigiert werden, indem man mittels **[pattern]** die "inkorrekt" Werte ausblendet.

### DAX

Sales % Group =

**VAR pattern =**

IF(

ISINSCOPE(Reseller[Region]) || ISINSCOPE(Reseller[Country]) ;

DIVIDE( [Sales] ;

CALCULATE(

[Sales] ;

REMOVEFILTERS(Reseller[Region] ; Reseller[Country])

)

)

)

RETURN

IF(

ISINSCOPE(Reseller[Region]) ;

"" ;

**pattern**

)

SalesTerritoryGroup	Sales % Group
Europe	
France	42,39 %
Germany	18,25 %
United Kingdom	39,36 %
North America	
Canada	21,15 %
United States	78,85 %
Southwest	
Northwest	
Central	
Southeast	
Northeast	
Pacific	
Australia	100,00 %
Gesamt	

16. Im Anschluss werden die passenden Elemente zu einem Gesamt-Ergebnis zusammen-gefügt werden, um **jeweils 100% pro Hierarchie-Ebene** die Werte zu berechnen und auszugeben. Dazu schreibt man ein **Measure Analysis**, welches die beiden Teile passend vereinigt:

### DAX

Analysis =

IF(

ISBLANK([Sales % Country]) ;

[Sales % Group] ;

[Sales % Country]

)

SalesTerritoryGroup	Sales % Country	Sales % Group	Analysis
Europe			
France	42,39 %	42,39 %	
Germany	18,25 %	18,25 %	
United Kingdom	39,36 %	39,36 %	
North America			
Canada	21,15 %	21,15 %	
United States	78,85 %	78,85 %	
Southwest	34,45 %	34,45 %	
Northwest	23,20 %	23,20 %	
Central	14,75 %	14,75 %	
Southeast	14,68 %	14,68 %	
Northeast	12,93 %	12,93 %	
Pacific			
Australia	100,00 %	100,00 %	
Gesamt			

Der Filterkontext ist also wichtig, um eine hierarchische Navigation zu erreichen, die ihrerseits die Zwischensummen berechnet durch Entfernen einiger Spalten aus dem Filterkontext.

# Arbeiten mit Iterator-Funktionen

In dieser Aufgabe erstellen Sie eine Kennzahl, die eine Iteratorfunktion verwendet.

1. Das Measure **TOP5 Sales** berechnet die Umsatzzahlen der 5 größten Bestellsummen pro Hierarchie-Ebene (= Zeilenkontext), in unserem Fall die Regionen, Länder und SalesTerritory. Der so entstandene Ausdruck wäre auf zwei Dezimalstellen zu formatieren, da **CURRENCY** im Inhalt.

```
DAX  
TOP5 SALES =  
SUMX(  
    TOPN(  
        5 ;  
        SUMMARIZE(  
            'ResellerSales' ;  
            Reseller[ResellerKey] ;  
            [SalesOrderNumber] ;  
            "Sales" ;  
            [Sales]  
        );  
        [Sales] ; DESC  
    )  
;  
    [Sales]  
)
```

**Iterator-Funktionen** lassen sich durch das nachfolgende **X** im Funktionsnamen leicht erkennen. Jede Aggregatfunktion verfügt über eine solche. Iterator-Funktionen nehmen **immer eine Tabelle** als erstes Argument und ein **Ausdrucks-Argument**. Die Funktion iteriert über jede Zeile in der so bestimmten Tabelle und wertet den Ausdruck für jede Zeile mithilfe des Zeilenkontextes als Aggregation aus. Entsprechend der Iterator-Funktion wird das Auswertungsergebnis für jede Zeile aggregiert (z.B. summiert die **SUMX()-Funktion** alle Ergebnisse auf, die **AVERAGEX()-Funktion** berechnet den Durchschnitt aller Ergebnisse usw.). Iterator-Funktionen unterstützen eine komplexere Aggregationslogik, die über die Aggregation einer einzelnen Spalte oder Tabelle hinausgeht. Sie stellt damit **das Analogon zur CURSOR-Logik im TSQL** dar, nur eben in DAX.

Diese Kennzahl verwendet die **Funktion SUMX()**. Die zu iterierende Tabelle wird von der **TOPN()-Funktion** zurückgegeben, die die obersten fünf Zeilen zurückgibt, die von der **SUMMARIZE()-Funktion** zurückgegeben werden. Die **Funktion SUMMARIZE()** selbst fasst die Tabelle **ResellerSales** zusammen und gruppiert intern nach **ResellerKey** (also je Mitarbeiter) und **SalesOrderNumber**, um den Umsatz **Sales** pro Zeile zu bestimmen und ihn in einem **Ausdruck "Sales"** abzuspeichern, der aus dem **Measure [Sales]** seine Inhalte bezieht. Vor der **TOPN**-Bewertung wird der Inhalt **ASC** oder hier eben **DESC** sortiert. Am Ende werden die so ermittelten 5 Ergebnisse zeilenweise bestimmt und mittels **SUMX** ausgewertet.

Zusammenfassend lässt sich sagen, dass diese Kennzahl die Summe der fünf höchsten Auftragswerte für einen Wiederverkäufer zurückgibt, um nun die passende Relation zu bestimmen.

---

*Tipp: Es ist nicht ungewöhnlich, dass DAX-Formeln viele Funktionen innerhalb der Funktion verschachteln, weshalb **WhiteSpace-Einzüge** wichtig sind, um die Formel zu interpretieren.*

---

2. Nun wieder die üblichen Tricks mit **[pattern]** und **ISINSCOPE()**, um relevante Anteile anzuzeigen:

### DAX

TOP5 Analysis =

VAR pattern =

IF(

```
    ISINSCOPE( 'Date'[Month] ) ;
    DIVIDE(
        [TOP5 Sales] ;
        [Sales]
    )
)
```

RETURN

IF(

```
    pattern = 1 ;
    "";
    pattern
)
```

YEAR	Sales	TOP5 Sales	TOP5 Analysis
□ FJ2010	<b>489.328,58 €</b>	<b>190.639,64 €</b>	
2010 Dez	489.328,58 €	190.639,64 €	38,96 %
□ FJ2011	<b>18.192.802,71 €</b>	<b>636.254,25 €</b>	
2011 Jan	1.538.408,31 €	416.688,69 €	27,09 %
2011 Mrz	2.010.618,07 €	400.656,78 €	19,93 %
2011 Mai	4.027.080,34 €	543.554,28 €	13,50 %
2011 Jul	713.116,69 €	245.146,06 €	34,38 %
2011 Aug	3.356.069,34 €	420.900,80 €	12,54 %
2011 Sep	882.899,94 €	268.165,54 €	30,37 %
2011 Okt	2.269.116,71 €	465.093,18 €	20,50 %
2011 Nov	1.001.803,77 €	266.077,14 €	26,56 %
2011 Dez	2.393.689,53 €	598.884,23 €	25,02 %
□ FJ2012	<b>28.193.631,53 €</b>	<b>726.815,54 €</b>	
2012 Jan	3.601.190,71 €	543.393,96 €	15,09 %
2012 Feb	2.885.359,20 €	640.504,36 €	22,20 %
2012 Mrz	1.802.154,21 €	445.334,65 €	24,71 %
2012 Apr	3.053.816,33 €	476.171,51 €	15,59 %
2012 Mai	2.185.213,21 €	455.664,68 €	20,65 %
2012 Jun	1.317.541,83 €	382.339,15 €	29,02 %
2012 Jul	2.384.846,59 €	402.471,28 €	16,68 %
2012 Aug	1.563.955,08 €	360.974,69 €	23,08 %
2012 Sep	1.865.278,43 €	458.407,73 €	24,58 %
2012 Okt	2.880.752,68 €	496.358,22 €	17,23 %
2012 Nov	1.987.872,71 €	462.211,35 €	23,25 %
2012 Dez	2.665.650,54 €	562.581,64 €	21,10 %
□ FJ2013	<b>33.574.834,16 €</b>	<b>649.975,62 €</b>	
2013 Jan	4.212.971,51 €	582.076,72 €	13,82 %
2013 Feb	4.047.574,04 €	528.503,88 €	13,06 %
2013 Mrz	2.282.115,88 €	484.652,35 €	21,24 %
2013 Apr	3.483.161,40 €	530.856,82 €	15,24 %
2013 Mai	3.510.948,73 €	413.833,07 €	11,79 %
2013 Jun	1.662.547,32 €	341.995,36 €	20,57 %
Gesamt	<b>80.450.596,98 €</b>	<b>768.691,72 €</b>	

Dies muss gemacht werden, weil ansonsten **jede Zeile** ausgewertet wird, also auch welche aus dem **Jahr 1939** oder leere Zeilen ohne Umsätze wie im **Dezember 2013**, was zu einem Wert von 100% führt, der in dem konkreten Zusammenhang sinnbefreit ist. In der Abbildung sieht man, dass die Jahresangaben falsch zusammensummiert werden, weil hier die 5 größten Umsätze der letzten 12 Monate ausgewertet werden und nicht die einzelnen Summen ausgewertet werden.

3. Sortiere die erste **Spalte [Sales] DESC** durch Anklicken der Spalte im Visual.

Im Rahmen der Selbstübung möge der geneigte Leser die **TOP5 Sales** mit **ISINSCOPE()** so anpassen, dass dieser nur auf Monatsebene ausgewertet wird. Anschließend ein neues Measure ausloben, das diese Zahlen pro Jahr aufsummiert. Anschließend passend teilen und dann diese zwei Spalten wieder zusammenfügen zu einer gemeinsamen Auswertung.

4. Es soll eine alternative Anwendung für dieses Measure gezeigt werden. Dazu erstelle man ein **Table-Visual**, das aus **Reseller[ResellerName]** , **[Sales]** und **[TOP5 Sales]** besteht. Kontextbezogen ist nun der **ISINSCOPE()** auf den jeweiligen Wiederverkäufer anzuwenden.

Da es auch 100%-Elemente gibt, sollen diese außen vorgehalten werden. Dazu muss die üblichen **[pattern]** und **ISINSCOPE()** angewendet werden. Darüber hinaus sind als **Universal-Filter** all jene auszublenden, in der Spalte [TOP 5 Reseller % All Orders] keinen Wert zurückgeben:

### DAX

```
TOP5 Reseller % All Orders =
```

```
VAR pattern =
```

```
IF(
```

```
    ISINSCOPE(Reseller[ResellerName]);
```

```
    DIVIDE(
```

```
        [TOP5 Sales];
```

```
        [Sales]
```

```
)
```

```
)
```

```
RETURN
```

```
IF(
```

```
    pattern <> 1;
```

```
    pattern
```

```
)
```

Filter

Filter für dieses Visual

...

**TOP5 Reseller % All Or**

ist nicht leer

Elemente anzeigen, deren W...

ist nicht leer

Und    Oder

Filter anwenden

ResellerName	Sales	TOP5 Sales	TOP5 Reseller % All Orders
Original Bicycle Supply Company	416.653,48 €	209.652,49 €	50,32 %
Every Bike Shop	223.285,39 €	112.735,35 €	50,49 %
Small Bike Shop	424.512,70 €	218.926,77 €	51,57 %
Finer Riding Supplies	290.158,93 €	150.030,19 €	51,71 %
Rapid Bikes	397.553,27 €	207.642,73 €	52,23 %
Catalog Store	373.658,47 €	195.217,10 €	52,24 %
Resale Services	492.362,76 €	259.448,45 €	52,69 %
Vigorous Exercise Company	841.908,77 €	449.664,51 €	53,41 %
Totes & Baskets Company	816.755,58 €	438.662,85 €	53,71 %
Latest Sports Equipment	724.299,64 €	389.049,90 €	53,71 %
Brakes and Gears	877.107,19 €	475.278,81 €	54,19 %
Highway Bike Shop	68.492,10 €	37.208,39 €	54,33 %
New and Used Bicycles	395.413,13 €	215.506,31 €	54,50 %
Paint Supply	395.867,27 €	216.202,62 €	54,61 %
Larger Cycle Shop	537.528,10 €	294.881,63 €	54,86 %
Painters Bicycle Specialists	107.591,67 €	59.190,99 €	55,01 %

# Zeit-Funktionen innerhalb von DAX

## Running Total pro Jahr

1. Grundlage ist erneut das Matrix-Visual, das nach Jahr und Monat gruppiert ist.
2. **Running Total Funktionen** lauten auf **YTD = Year To Date** und wird zeilenweise ausgewertet:

### DAX

Sales YTD =  
TOTALYTD([Sales], 'Date'[Date] )

Die Funktion **TOTALYTD()** wertet den Ausdruck **[Sales]** über eine bestimmte Datumsspalte aus. Die Datumsspalte **muss zu einer Datumstabelle** gehören, die als **Datumstabelle markiert** ist. Die Funktion hat als dritte Komponente eine Option, die das letzte Datum eines Jahres darstellt. Das Fehlen dieses Datums bedeutet, dass der 31. Dezember das letzte Datum des Jahres ist.

Dadurch wird nun jeder Monat mit einem Wert besetzt: in den Monaten, in denen nichts hinzukommt, bleibt der Wert unverändert; ansonsten wird der aktuelle Wert hinzuaddiert.

Die **TOTALYTD()-Funktion** führt Filtermanipulationen durch, insbesondere Zeitfilterbearbeitung. Um beispielsweise die YTD-Verkäufe für März 2016 (den dritten Monat des Jahres) zu berechnen, werden alle Filter in der Datumstabelle entfernt und durch einen neuen Filter von Datumsangaben ersetzt, die zu Beginn des Jahres beginnen (1. Januar 2016) beginnen und sich bis zum letzten Datumszeitraum (31. März 2016) erstrecken.

YEAR	Sales	Sales YTD
□ FJ2010	489.328,58 €	489.328,58 €
2010 Dez	489.328,58 €	489.328,58 €
□ FJ2011	18.192.802,71 €	18.192.802,71 €
2011 Jan	1.538.408,31 €	1.538.408,31 €
2011 Feb		1.538.408,31 €
2011 Mrz	2.010.618,07 €	3.549.026,39 €
2011 Apr		3.549.026,39 €
2011 Mai	4.027.080,34 €	7.576.106,73 €
2011 Jun		7.576.106,73 €
2011 Jul	713.116,69 €	8.289.223,42 €
2011 Aug	3.356.069,34 €	11.645.292,76 €
2011 Sep	882.899,94 €	12.528.192,71 €
2011 Okt	2.269.116,71 €	14.797.309,42 €
2011 Nov	1.001.803,77 €	15.799.113,19 €
2011 Dez	2.393.689,53 €	18.192.802,71 €
□ FJ2012	28.193.631,53 €	28.193.631,53 €
2012 Jan	3.601.190,71 €	3.601.190,71 €

## Auswertung gegen den Vergleichszeitraum

1. Nun soll der Wert nicht auf gleicher Höhe ausgegeben werden, sondern 12 Monate später. Die Logik lautet:

Aktuelles Datum - 12 Monate im parallelen Zeitraum soll der berechnete Umsatz ausgewertet werden, um verschoben eingefügt zu werden:

## DAX

```

Sales Last Year =
VAR SalesPriorYear =
    CALCULATE (
        [Sales];
        PARALLELPERIOD (
            'Date'[Date];
            -12;
            MONTH
        )
    )
RETURN
    SalesPriorYear

```

YEAR	Sales	Sales YTD	Sales Last Year
□ FJ2010	489.328,58 €	489.328,58 €	
2010 Dez	489.328,58 €	489.328,58 €	
□ FJ2011	18.192.802,71 €	18.192.802,71 €	489.328,58 €
2011 Jan	1.538.408,31 €	1.538.408,31 €	
2011 Feb		1.538.408,31 €	
2011 Mrz	2.010.618,07 €	3.549.026,39 €	
2011 Apr		3.549.026,39 €	
2011 Mai	4.027.060,34 €	7.576.106,73 €	
2011 Jun		7.576.106,73 €	
2011 Jul	713.116,69 €	8.289.223,42 €	
2011 Aug	3.356.069,34 €	11.645.292,76 €	
2011 Sep	882.899,94 €	12.528.192,71 €	
2011 Okt	2.269.116,71 €	14.797.309,42 €	
2011 Nov	1.001.803,77 €	15.799.113,19 €	
2011 Dez	2.393.689,53 €	18.192.802,71 €	489.328,58 €
□ FJ2012	28.193.631,53 €	28.193.631,53 €	18.192.802,71 €
2012 Jan	3.601.190,71 €	3.601.190,71 €	1.538.408,31 €
2012 Feb	2.885.359,20 €	6.486.549,91 €	

Die Formel **Sales Last Year** deklariert eine Variable, um die Formellogik zu vereinfachen, und effizienter sein, wenn ein Ausdruck mehrmals innerhalb der Formel ausgewertet werden muss (was der Fall sein wird).

Der **Variable SalesPriorYear** wird ein Ausdruck mit berechnetem Umsatz in einem geänderten Kontext zugewiesen, der die Funktion **PARALLELPERIOD()** verwendet, um 12 Monate von jedem Datum im Filterkontext zurück zu verschieben.

Nachdem nun der "schwierige Teil" der Formel getestet wurde, kann nun das Maß mit der endgültigen Formel überschreiben werden, die das **tatsächliche Wachstumsergebnis** berechnet.

2. Dazu ziehe vom aktuellen Umsatz auf Zeilenhöhe des Umsatz des Vergleichszeitraums und teile danach durch den aktuellen Umsatz, um einen Trend in % des Wachstums zu erhalten.
3. Bitte nicht verunsichern lassen, wenn für den **2017 Dez** ein Wert von **389.18%** herauskommt. Dies bedeutet lediglich, dass der Umsatz im Januar 2017 eine fast 4-fache Verbesserung gegenüber dem Vorjahresumsatz darstellt. Außerdem sollte man die Werte passend filtern auf die relevanten Jahre.

YEAR	Sales	Sales YTD	Sales Last Year
□ FJ2010	489.328,58 €	489.328,58 €	
2010 Dez	489.328,58 €	489.328,58 €	
□ FJ2011	18.192.802,71 €	18.192.802,71 €	
2011 Dez	2.393.689,53 €	18.192.802,71 €	389,18 %
□ FJ2012	28.193.631,53 €	28.193.631,53 €	

Filter

YEAR  
ist FJ2011, FJ2012, FJ...

Filtertyp  
Einfaches Filtern

FJ2010    365  
 FJ2011    365  
 FJ2012    366  
 FJ2013    365

Einfachauswahl erforderlich

## DAX

```

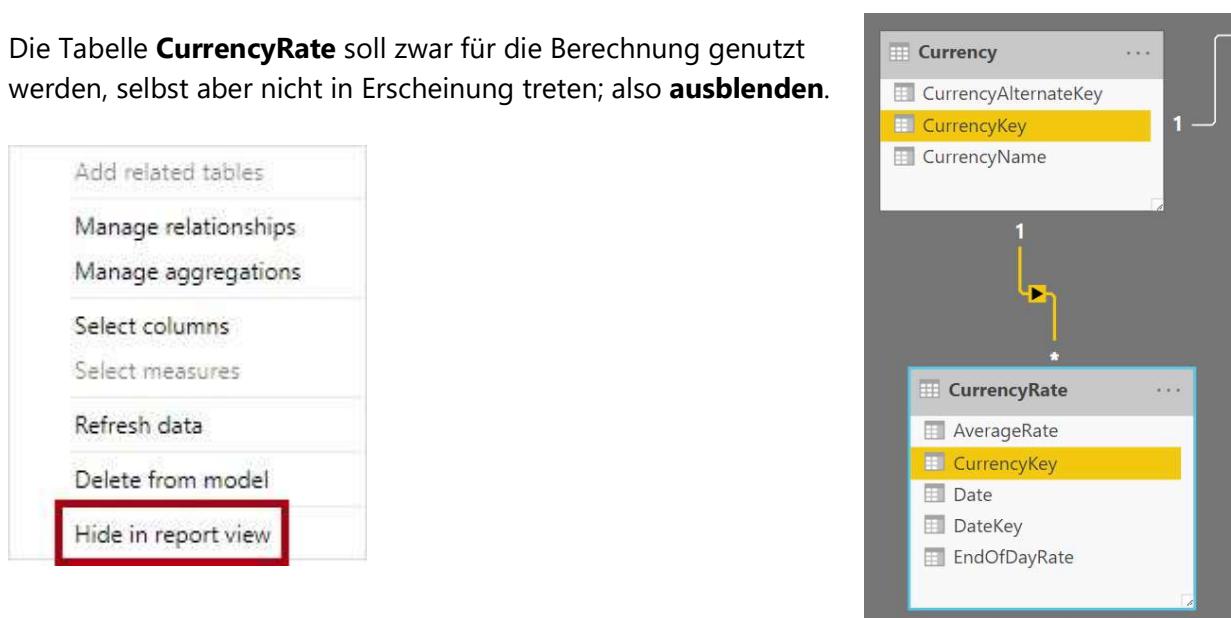
Sales Last Year =
VAR SalesPriorYear =
IF(
    ISINSCOPE('Date'[Month]) ;
    CALCULATE (
        [Sales];
        PARALLELPERIOD (
            'Date'[Date];
            -12;
            MONTH
        )
    )
)
RETURN
DIVIDE(
    [Sales] - SalesPriorYear ;
    SalesPriorYear
)

```

YEAR	Sales	Sales YTD	Sales Last Year
□ FJ2010	489.328,58 €	489.328,58 €	
2010 Dez	489.328,58 €	489.328,58 €	
□ FJ2011	18.192.802,71 €	18.192.802,71 €	
2011 Jan	1.538.408,31 €	1.538.408,31 €	
2011 Feb		1.538.408,31 €	
2011 Mrz	2.010.618,07 €	3.549.026,39 €	
2011 Apr		3.549.026,39 €	
2011 Mai	4.027.080,34 €	7.576.106,73 €	
2011 Jun		7.576.106,73 €	
2011 Jul	713.116,69 €	8.289.223,42 €	
2011 Aug	3.356.069,34 €	11.645.292,76 €	
2011 Sep	882.899,94 €	12.528.192,71 €	
2011 Okt	2.269.116,71 €	14.797.309,42 €	
2011 Nov	1.001.803,77 €	15.799.113,19 €	
2011 Dez	2.393.689,53 €	18.192.802,71 €	389,18 %
□ FJ2012	28.193.631,53 €	28.193.631,53 €	
2012 Jan	3.601.190,71 €	3.601.190,71 €	134,09 %
2012 Feb	2.885.359,20 €	6.486.549,91 €	
2012 Mrz	1.802.154,21 €	8.288.704,13 €	-10,37 %
2012 Apr	3.053.816,33 €	11.342.520,45 €	
2012 Mai	2.185.213,21 €	13.527.733,67 €	-45,74 %
2012 Jun	1.317.541,83 €	14.845.275,50 €	
2012 Jul	2.384.846,59 €	17.230.122,09 €	234,43 %
2012 Aug	1.563.955,06 €	18.794.077,17 €	-53,40 %
2012 Sep	1.865.278,43 €	20.659.355,61 €	111,27 %
2012 Okt	2.880.752,68 €	23.540.108,29 €	26,95 %
2012 Nov	1.987.872,71 €	25.527.980,99 €	98,43 %
2012 Dez	2.665.650,54 €	28.193.631,53 €	11,36 %
□ FJ2013	33.574.834,16 €	33.574.834,16 €	
2013 Jan	4.212.971,51 €	4.212.971,51 €	16,99 %
2013 Feb	4.047.574,04 €	8.260.545,55 €	40,28 %

# Implementieren der Währungsumrechnung

Die Tabelle **CurrencyRate** soll zwar für die Berechnung genutzt werden, selbst aber nicht in Erscheinung treten; also **ausblenden**.



1. Wähle im Bereich **Felder** die Tabelle **Currency** aus und überprüfe die Daten aus Sinnhaftigkeit.

Die Tabelle enthält den Währungscode und -namen sowie einen Formatzeichenfolgenwert (nützlich für das Formatieren von Zahlen und Datumsangaben).

---

*Die **FormatString-Spaltenwerte** werden in Formatierungen von Währungswerten verwendet, einschließlich Währungssymbolen und Dezimalstellen.*

---

2. Nun das gleiche für die Tabelle **CurrencyRate**.

---

*Es gibt eine Zeile für jede Währung und jedes Datum. Der Tagesendkurs für USD ist immer 1, da die **Basiswährung des Datenmodells in diesem Falle USD** ist.*

---

Nun ist es aber so, dass in einem Deutschen PowerBI alles in EURO ausgegeben wird, was ja inhaltlich grob falsch ist, wenn keine passende Umrechnung stattfindet. Es sollten also die Werte beispielhaft in USD umgerechnet werden, wodurch alle Werte angepasst werden sollen, die ...

- einen Umrechnungskurs vorweisen in dem betreffenden Zeitraum, sonst leer bleiben
- keine Umrechnung innerhalb von den Vereinigten Staaten bewirken, aber \$ benutzen
- nur dort Werte zurückgeben soll, wo die Angabe eines Wertes überhaupt sinnvoll ist

3. Es werden nun mehrere Measures gebaut: eine gibt den Wert in der Originalwährung wieder, ein weiterer die nach Tagessatz passende Umrechnung in USD.

```
DAX

Sales Origin Currency =
SUMX(
    CurrencyRate;
    CALCULATE(
        DIVIDE(
            [Sales];
            SUM(CurrencyRate[EndOfDayRate])
        )
    )
)
```

SalesTerritoryGroup	Sales All Region	Sales Origin Currency
Europe	10.870.534,80 €	11.200.146.819,43 €
France	4.607.537,94 €	5.383.044.308,27 €
France	4.607.537,94 €	5.383.044.308,27 €
Germany	1.983.988,04 €	2.489.630.998,06 €
Germany	1.983.988,04 €	2.489.630.998,06 €
United Kingdom	4.279.008,83 €	3.327.471.513,10 €
United Kingdom	4.279.008,83 €	3.327.471.513,10 €
North America	67.985.726,81 €	87.603.777.465,67 €
Canada	14.377.925,60 €	25.525.943.664,26 €
Canada	14.377.925,60 €	25.525.943.664,26 €
United States	53.607.801,21 €	62.077.833.801,41 €
Central	7.906.008,18 €	9.155.157.469,78 €
Northeast	6.932.842,01 €	8.028.231.053,14 €
Northwest	12.435.076,00 €	14.399.818.008,12 €
Southeast	7.867.416,23 €	9.110.467.988,67 €
Southwest	18.466.458,79 €	21.384.159.281,72 €
Pacific	1.594.335,38 €	3.366.500.529,81 €
Australia	1.594.335,38 €	3.366.500.529,81 €
Australia	1.594.335,38 €	3.366.500.529,81 €
Gesamt	80.450.596,98 €	102.170.424.814,92 €

Die **Funktion SUMX()** iteriert über alle kontextabhängigen Zeilen der **CurrencyRate** (inklusive Schaltjahre), die den **Tagesendkurs** für jedes Datum speichern. Für jede Zeile teilt der Ausdruck die Verkäufe für dieses Datum durch den Tagesendsatz für dieses Datum, deshalb **CALCULATE()**.

Die Tagesendrate muss innerhalb einer Aggregatfunktion unterbrochen werden, da Ausdrücke nicht direkt auf eine Spalte verweisen können. Da in der **CurrencyRate** genau eine Zeile pro Währungs-Datum-Kombination gespeichert wird, wird auch **SUM()** verwendet, da sie weiß, dass sie immer nur einen einzelnen Wert im passenden Filterkontext summiert. Nachdem die bis zu 366 täglich umgerechneten Verkäufe berechnet wurden, summiert die **SUMX()-Funktion** das **Antwort-Array** zusammen, um den gesamten umgerechneten Betrag zu erzeugen.

4. Es gibt mehrere Schönheitsfehler: zum einen wird in der Spalte **Gesamt alle Währungen** zusammengefasst werden, was nicht sinnvoll ist. Unterschiedliche Währungsbeträge sind nicht additiv, d.h. es macht keinen Sinn, australische Dollarbeträge zu US-Dollar-Beträgen hinzuzufügen. Außerdem wurden auch Werte, die bereits in USD sind, umgerechnet in USD, was sinnfrei ist.

Die Funktion **HASONEVALUE()** bestimmt, ob sich ein einzelner **CurrencyKey-Wert** im Kontext befindet. Wenn **TRUE**, wird das ursprüngliche Measure **Sales** ausgewertet. Bei **FALSE** wird die **Sales Origin Currency** für den Umrechnungskurs in USD ausgewertet.

SalesTerritoryGroup	Sales All Region	Sales Origin Currency	Sales Origin Corrected
□ Europe	10.870.534,80 €	11.200.146.819,43 €	4.767.474.724,67 €
□ France	4.607.537,94 €	5.383.044.308,27 €	4.767.474.724,67 €
France	4.607.537,94 €	5.383.044.308,27 €	4.767.474.724,67 €
□ Germany	1.983.988,04 €	2.489.630.998,06 €	1.983.988,04 €
Germany	1.983.988,04 €	2.489.630.998,06 €	1.983.988,04 €
□ United Kingdom	4.279.008,83 €	3.327.471.513,10 €	4.279.008,83 €
United Kingdom	4.279.008,83 €	3.327.471.513,10 €	4.279.008,83 €
□ North America	67.985.726,81 €	87.603.777.465,67 €	62.077.833.801,41 €
□ Canada	14.377.925,60 €	25.525.943.664,26 €	14.377.925,60 €
Canada	14.377.925,60 €	25.525.943.664,26 €	14.377.925,60 €
□ United States	53.607.801,21 €	62.077.833.801,41 €	53.607.801,21 €
Central	7.906.008,18 €	9.155.157.469,78 €	7.906.008,18 €
Northeast	6.932.842,01 €	8.028.231.053,14 €	6.932.842,01 €
Northwest	12.435.076,00 €	14.399.818.008,12 €	12.435.076,00 €
Southeast	7.867.416,23 €	9.110.467.988,67 €	7.867.416,23 €
Southwest	18.466.458,79 €	21.384.159.281,72 €	18.466.458,79 €
□ Pacific	1.594.335,38 €	3.366.500.529,81 €	1.594.335,38 €
□ Australia	1.594.335,38 €	3.366.500.529,81 €	1.594.335,38 €
Australia	1.594.335,38 €	3.366.500.529,81 €	1.594.335,38 €
Gesamt	80.450.596,98 €	102.170.424.814,92 €	66.845.308.526,08 €

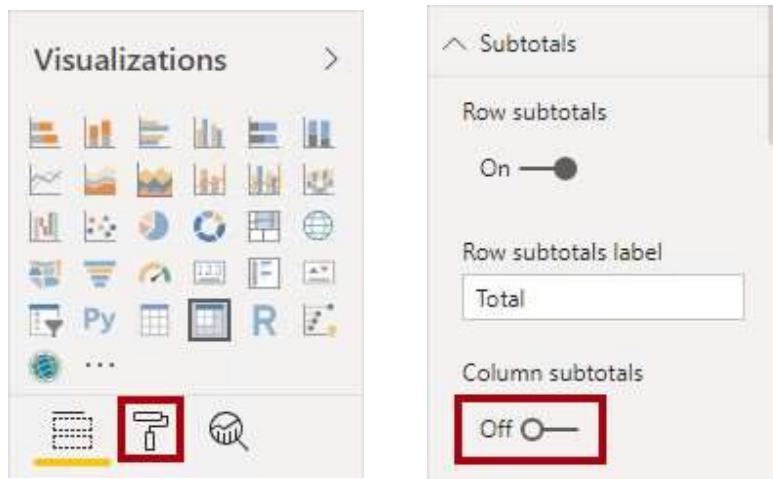
## DAX

Sales Origin Corrected =

```
IF (
    HASONEVALUE (CurrencyRate[CurrencyKey]);
    [Sales];
    CALCULATE (
        [Sales Origin Currency];
        'Currency'[CurrencyAlternateKey] = "USD"
    )
)
```

*Tipp: Sie müssten die Berichtsbenutzer darauf bringen, dass die Spalte **Sales All Region** implizit die Basiswährungen benutzt, ohne diese anzugeben. Ein bevorzugter Ansatz wäre, diese Spalte aus dem Visual zu entfernen. Minimum-Lösung wäre, die **SubTotals zu entfernen**, da die Zwischensummen keinen Sinn machen und zu viel summieren.*

5. Die Zwischensummen sind recht schnell entfernt:



6. Um die Formatierung auf die Währungswerte anzuwenden, erstelle ein passendes Measure:

#### DAX

```
Sales Origin Formated =
IF (
    HASONEVALUE ( FactCurrencyRate[CurrencyKey] );
    FORMAT (
        [Sales Origin Corrected];
        "$#,0.00"
    )
)
```

Die **VALUES()-Funktion** gibt eine Tabelle mit Kontextwerten für eine bestimmte Spalte zurück. Wenn die Funktion eine einzeilige Tabelle zurückgibt, kann sie in einem Ausdruck verwendet werden. In diesem Fall wird garantiert ein einzelner Wert zurückgegeben, da die **HASONEVALUE()-Funktion** verwendet wurde, um dies zu bestimmen. Dieses Measure formatiert daher die **[Sales]** mithilfe der **Funktion FORMAT()** und stellt einen einzelnen **CurrencyKey** zur Verfügung als Ziel-Währung. **Der Datentyp dieser Kennzahl ist STRING.**

# Endkonfiguration

**Zeilen**

Year	▼ X
Month	▼ X

**Spalten**

Hier Datenfelder hinzufügen	
-----------------------------	--

**Werte**

Sales	▼ X
Cost	▼ X
Avg Price	▼ X
Orders	▼ X
Order Lines	▼ X
Profit	▼ X
Profit % Margin	▼ X
Profit % Optic	▼ X

Year	Sales	Cost	Avg Price	Orders	Order Lines	Profit	Profit % Margin	Profit % Optic
CY2010	489.328,58 €	472.295,14 €	655,59 €	38	352	17.033,44 €	3,48 %	3,48 %
2010 Dez	489.328,58 €	472.295,14 €	655,59 €	38	352	17.033,44 €	3,48 %	3,48 %
CY2011	18.192.802,71 €	18.163.382,58 €	682,26 €	773	9830	29.420,14 €	0,16 %	0,16 %
2011 Jan	1.538.408,31 €	1.469.459,58 €	758,94 €	75	785	68.948,73 €	4,48 %	4,23 %
2011 Mrz	2.010.618,07 €	1.925.612,68 €	712,42 €	100	1092	85.005,40 €	4,23 %	4,23 %
2011 Mai	4.027.080,34 €	3.869.140,83 €	744,85 €	153	1909	157.939,51 €	3,92 %	3,92 %
2011 Jul	713.116,69 €	690.429,39 €	808,94 €	40	377	22.687,30 €	3,18 %	3,18 %
2011 Aug	3.356.069,34 €	3.227.247,65 €	882,50 €	143	1519	128.821,69 €	3,84 %	3,84 %
2011 Sep	882.899,94 €	854.645,89 €	732,25 €	37	494	28.254,06 €	3,20 %	3,20 %
2011 Okt	2.269.116,71 €	2.181.837,45 €	761,30 €	85	1112	87.279,26 €	3,85 %	3,85 %
2011 Nov	1.001.803,77 €	1.067.008,21 €	552,95 €	68	819	-605.204,44 €	-60,41 %	-60,41 %
2011 Dez	2.393.689,53 €	2.338.000,90 €	350,74 €	72	1723	55.688,63 €	2,33 %	2,33 %
CY2012	28.193.631,53 €	27.277.732,50 €	396,81 €	1277	22133	915.899,04 €	3,25 %	3,25 %
2012 Jan	3.601.190,71 €	3.468.825,70 €	372,75 €	139	2964	132.365,02 €	3,68 %	3,68 %
2012 Feb	2.885.359,20 €	2.735.886,23 €	365,30 €	111	2185	149.472,97 €	5,18 %	5,18 %
2012 Mrz	1.802.154,21 €	1.724.517,59 €	382,00 €	73	1406	77.636,63 €	4,31 %	4,31 %
2012 Apr	3.053.816,33 €	2.919.120,37 €	412,48 €	133	2345	134.695,96 €	4,41 %	4,41 %
2012 Mai	2.185.213,21 €	2.073.508,49 €	407,08 €	114	1732	111.704,72 €	5,11 %	5,11 %
2012 Jun	1.317.541,83 €	1.267.385,38 €	428,87 €	65	983	50.156,45 €	3,81 %	3,81 %
2012 Jul	2.384.846,59 €	2.308.083,78 €	488,76 €	132	1664	76.762,81 €	3,22 %	3,22 %
2012 Aug	1.563.955,08 €	1.497.549,93 €	450,03 €	106	1215	66.405,15 €	4,25 %	4,25 %
2012 Sep	1.865.278,43 €	1.782.337,13 €	369,78 €	74	1449	82.941,31 €	4,45 %	4,45 %
2012 Okt	2.880.752,68 €	2.765.546,09 €	400,51 €	134	2306	115.206,59 €	4,00 %	4,00 %
2012 Nov	1.987.872,71 €	1.867.464,16 €	390,32 €	102	1698	100.408,55 €	5,05 %	5,05 %
2012 Dez	2.665.650,54 €	2.847.507,65 €	350,56 €	94	2186	-181.857,11 €	-6,82 %	-6,82 %
CY2013	33.574.834,16 €	34.066.704,16 €	396,84 €	1708	28540	-491.870,01 €	-1,46 %	-1,46 %
2013 Jan	4.212.971,51 €	4.463.792,72 €	342,20 €	185	3754	-250.821,21 €	-5,95 %	-5,95 %
2013 Feb	4.047.574,04 €	4.325.499,07 €	343,41 €	176	3760	-277.925,03 €	-6,87 %	-6,87 %
2013 Mrz	2.282.115,88 €	2.285.346,81 €	378,65 €	99	1810	-3.230,94 €	-0,14 %	-0,14 %
2013 Apr	3.483.161,40 €	3.460.754,70 €	395,81 €	178	2904	22.406,70 €	0,64 %	0,64 %
Gesamt	80.450.596,98 €	79.980.114,38 €	444,43 €	3796	60855	470.482,60 €	0,58 %	0,58 %

SalesTerritoryGroup	Sales All Region	Sales % All Region	Sales % Country	Sales % Group	Sales Origin Currency	Sales Origin Currency Single	Sales Origin Currency Formatted Single
Europe	10.870.534,80 €	100,00 %			11.200.146.819,43 €	4.767.474,724,67 €	
France	4.607.537,94 €	100,00 %			5.383.044.308,27 €	4.767.474,724,67 €	
Germany	1.983.988,04 €	100,00 %			2.489.630.998,06 €	\$2.489.630.998,06	
United Kingdom	4.279.008,83 €	100,00 %			3.327.471.513,10 €	\$3.327.471.513,10	
North America	67.985.726,81 €	100,00 %			87.603.777.465,67 €	62.077.833.801,41 €	
Canada	14.377.925,60 €	100,00 %			25.525.943.664,26 €	25.525.943.664,26 €	
United States	53.607.801,21 €	100,00 %			25.525.943.664,26 €	25.525.943.664,26 €	
Central	7.906.008,18 €	100,00 %			9.155.157.469,78 €	\$9.155.157.469,78	
Northeast	6.932.842,01 €	100,00 %			8.028.231.053,14 €	\$8.028.231.053,14	
Northwest	12.435.076,00 €	100,00 %			8.028.231.053,14 €	\$8.028.231.053,14	
Southeast	7.867.416,23 €	100,00 %			21.384.159.281,72 €	\$21.384.159.281,72	
Southwest	18.466.458,79 €	100,00 %			14.399.818.008,12 €	\$14.399.818.008,12	
Pacific	1.594.335,38 €	100,00 %			3.366.500.529,81 €	\$3.366.500.529,81	
Australia	1.594.335,38 €	100,00 %			3.366.500.529,81 €	\$3.366.500.529,81	
Gesamt	80.450.596,98 €	100,00 %			102.170.424.814,92 €	66.845.308.526,08 €	

Advanced DAX in PowerBI Desktop

Andreas Krause

44

**Werte**

ResellerName	✓ X
Sales	✓ X
Sales Top 5 Reseller Ord	✓ X
Sales Top 5 Reseller Ord	✓ X

ResellerName	Sales	Sales Top 5 Reseller Orders	Sales Top 5 Reseller Orders % All Orders
Year-Round Sports	148.659,27 €	148.638,75 €	99,99 %
Valley Bicycle Specialists	140.130,84 €	140.110,32 €	99,99 %
Some Discount Store	109.074,76 €	108.990,77 €	99,92 %
Rental Bikes	40.581,18 €	40.527,18 €	99,87 %
Worthwhile Activity Store	130.597,78 €	130.417,65 €	99,86 %
Future Bikes	11.038,03 €	11.021,76 €	99,85 %
Tandem Sales and Service	24.970,54 €	24.925,55 €	99,82 %
Certified Sports Supply	13.427,69 €	13.403,41 €	99,82 %
Outdoor Sports Supply	162.313,29 €	161.840,79 €	99,71 %
Versatile Sporting Goods Company	8.140,75 €	8.116,46 €	99,70 %
Price-Cutter Discount Bikes	64.700,47 €	64.471,17 €	99,65 %
A Great Bicycle Company	9.055,29 €	9.018,04 €	99,59 %
Suburban Cycle Shop	78.538,69 €	78.175,86 €	99,54 %
Commendable Bikes	6.962,38 €	6.929,98 €	99,53 %
Reliable Retail Center	8.476,68 €	8.436,31 €	99,52 %
Serious Cycles	289.489,20 €	287.929,98 €	99,46 %
Active Transport Inc.	88.245,87 €	87.646,93 €	99,32 %
Roving Sports	17.320,33 €	17.193,44 €	99,27 %
North Bike Company	22.090,04 €	21.926,91 €	99,26 %
Distance Bikes	3.611,24 €	3.584,52 €	99,26 %
Metro Bike Mart	149.121,22 €	148.004,67 €	99,25 %
Plastic Parts Company	11.138,73 €	11.055,32 €	99,25 %
Genuine Bike Shop	43.911,15 €	43.578,18 €	99,24 %
Executive Gift Store	146.256,57 €	145.122,93 €	99,22 %
<b>Gesamt</b>	<b>80.450.596,98 €</b>	<b>768.691,72 €</b>	

**Zeilen**

Year	✓ X
Month	✓ X

**Spalten**

Hier Datenfelder hinzufügen

**Werte**

Sales	✓ X
Sales YTD	✓ X
Sales YoY Growth	✓ X

Year	Sales	Sales YTD	Sales YoY Growth
<b>CY2010</b>	<b>489.328,58 €</b>	<b>489.328,58 €</b>	
2010 Dez	489.328,58 €	489.328,58 €	
<b>CY2011</b>	<b>18.192.802,71 €</b>	<b>18.192.802,71 €</b>	<b>3.617,91 %</b>
2011 Jan	1.538.408,31 €	1.538.408,31 €	
2011 Feb	1.538.408,31 €		
2011 Mrz	2.010.618,07 €	3.549.026,39 €	
2011 Apr		3.549.026,39 €	
2011 Mai	4.027.080,34 €	7.576.106,73 €	
2011 Jun		7.576.106,73 €	
2011 Jul	713.116,69 €	8.289.223,42 €	
2011 Aug	3.356.069,34 €	11.645.292,76 €	
2011 Sep	882.899,94 €	12.528.192,71 €	
2011 Okt	2.269.116,71 €	14.797.309,42 €	
2011 Nov	1.001.803,77 €	15.799.113,19 €	
2011 Dez	2.393.689,53 €	18.192.802,71 €	389,18 %
<b>CY2012</b>	<b>28.193.631,53 €</b>	<b>28.193.631,53 €</b>	<b>54,97 %</b>
2012 Jan	3.601.190,71 €	3.601.190,71 €	134,09 %
2012 Feb	2.885.359,20 €	6.486.549,91 €	
2012 Mrz	1.802.154,21 €	8.288.704,13 €	-10,37 %
2012 Apr	3.053.816,33 €	11.342.520,45 €	
2012 Mai	2.185.213,21 €	13.527.733,67 €	-45,74 %
2012 Jun	1.317.541,83 €	14.845.275,50 €	
2012 Jul	2.384.846,59 €	17.230.122,09 €	234,43 %
2012 Aug	1.563.955,08 €	18.794.077,17 €	-53,40 %
2012 Sep	1.865.278,43 €	20.659.355,61 €	111,27 %
2012 Okt	2.880.752,68 €	23.540.108,29 €	26,95 %
2012 Nov	1.987.872,71 €	25.527.980,99 €	98,43 %
2012 Dez	2.665.650,54 €	28.193.631,53 €	11,36 %
<b>CY2013</b>	<b>33.574.834,16 €</b>	<b>33.574.834,16 €</b>	<b>19,09 %</b>
2013 Jan	4.212.971,51 €	4.212.971,51 €	16,99 %
2013 Feb	1.617.571,21 €	5.830.545,72 €	16,00 %
<b>Gesamt</b>	<b>80.450.596,98 €</b>		<b>0,00 %</b>