# TARGIT Calculations Expert

Level: Expert

TARGIT Decision Suite 2019.0 – document version 4.4 US

**Copyright**



**TARGIT A/S**

Gasværksvej 24

DK-9000 Aalborg

Denmark

Phone: (+45) 9623 1900

Telefax: (+45) 9623 1999

E-mail: info@targit.com

Internet: http://www.targit.com

# Contents

## Introduction

As a TARGIT Calculations Expert, you can really use the full potential in TARGIT.

You will be your company's number-cruncher by offering everyone in the organization clear insight into complicated data. First and foremost, you will be able to make dynamic ad-hoc calculations, which you can use for optimizing your intelligent Agents, comparisons and other active functions that give everyone in the organization the courage to act.

### Prerequisite

You need to be thoroughly versed in TARGIT, minimally equivalent to the TARGIT Fundamentals course. You will get the most out of the course if you have worked with your own data for a period of time.

### Goal

- After completing the course, you will be able to:
- Make advanced user-defined calculations
- Expand the boundaries for creativity in TARGIT

### Course subjects

- References in cross tables
- Calculations with advanced formula syntax
- Reference filters
- Many practical exercises that will equip you to fill the expert role

# Lesson 1: Cross Table References and Calculations.

In this lesson you will learn the **general syntax for referencing columns, rows and individual cells** within a Cross table.

All the standard **operators** and **functions** will be introduced including the use of some of them.

During the demo it will be shown how **mastering calculations** can help to **provide additional information** and to **enhance the overview** of information in a coherent Analysis.

## Summation / Recap

- The **general syntax** for Cross table references are:
  - **sum([x range], [y range], [m range]),** where x refers to columns, y to rows and m to measures.
- **Absolute** x and y references:
  - E.g. **d1, d2, d3** etc. – counting columns from top-to-bottom or rows from left-to-right.
  - E.g. **d-1, d-2, d-3** etc. – counting columns from bottom-to-top or rows from right-to-left.
- **Relative** x and y references:
  - E.g. **-2, -1, 0, 1, 2** etc. – zero refers to current column/row, negative integers refer to previous columns/rows and positive integers refer to subsequent columns/rows.
  - E.g. in a **calculation of difference between 2 columns the row reference will be 0** – meaning the calculation must be done in the **current row**.
  - E.g.. in a **calculation of totals per column the column reference will be 0** – meaning the calculation must be done in the **current column**.
- **Dimension** x and y references:
  - E.g. **@"[Reseller].[Denmark]"** – refers to the column/row with the dimension values "Reseller" and "Denmark" as first and second levels respectively in a hierarchical dimension. Using this reference you also need to pay attention to use of upper- and lowercase.

- **Measure** references:
  - E.g. **m1, m2, m3** etc. – referring to the first, second, third etc. of the inserted measures.
  - **0** referring to the current measure (e.g. calculating a total on different measures
- Reference **ranges**:
  - E.g. **d1:d-3** or **m1:m3** – use two references separated by a colon to define a range of columns, rows or measures.
  - Reference ranges can of course also be **relative**, e.g. **-2:0** meaning a **range** starting 2 columns/rows back and up to the current column/row
  - A "*classical*" reference range could be an **accumulation**: **d1:0** meaning from the first column/row to the current – or reverse **d-1:0** meaning from the last column/row to the current.

## Demo

The demo in Lesson 1 will demonstrate how to add customized calculations to Cross tables. In the demo we will make use of some of the available aggregation functions and operators.

During the course of building a coherent Analysis we will touch on some of the aspects of referencing cells in a Cross table:

- absolute references
- relative references
- reference ranges

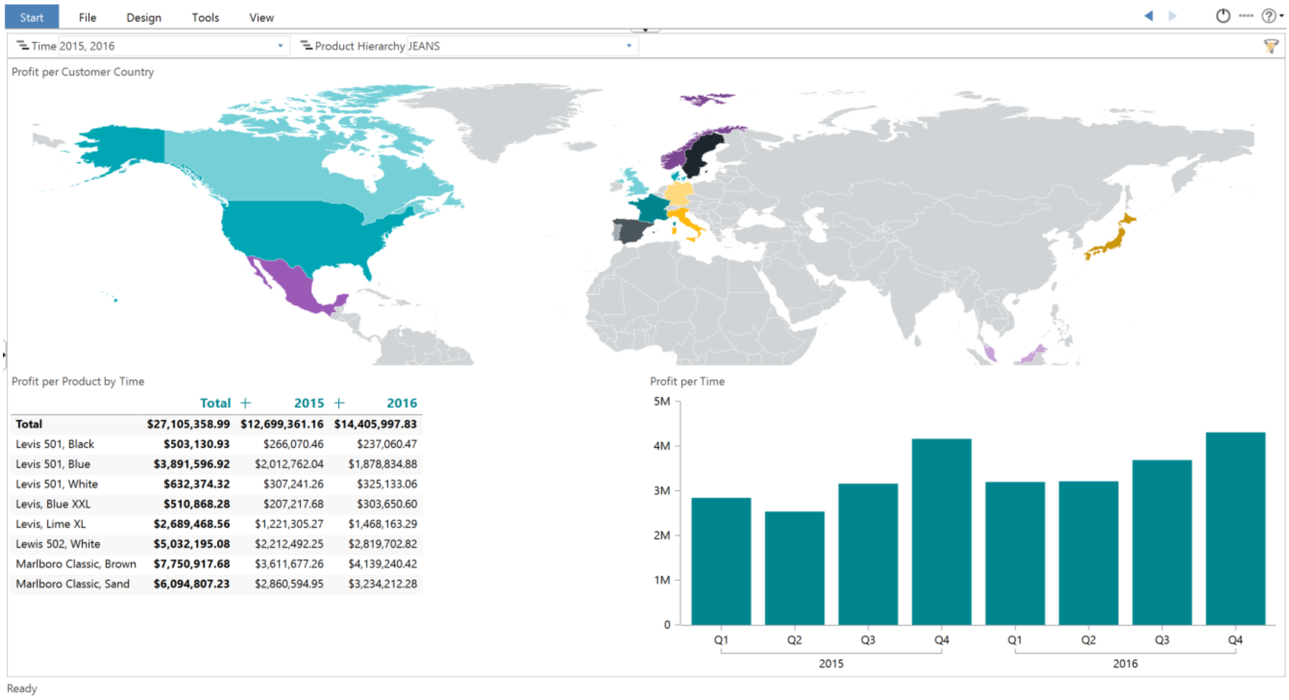
### A simple Analysis

A very simple analysis, without calculations and thus without the possibility to highlight important data issues, is not very helpful in providing useful information as will be demonstrated in the first part of this demo.

Start TARGIT BI Suite and create a new Analysis, **Revenue Analysis**, consisting of three objects:

- A Cross table, Profit per Product Hierarchy (Product) by Time Hierarchy (Year).
- An Area chart, Profit per Time Hierarchy (Quarter).
- A Map, Profit per Customer Country (Country).

Apply the global criteria **Time = 2015 and 2016. Product = JEANS.**



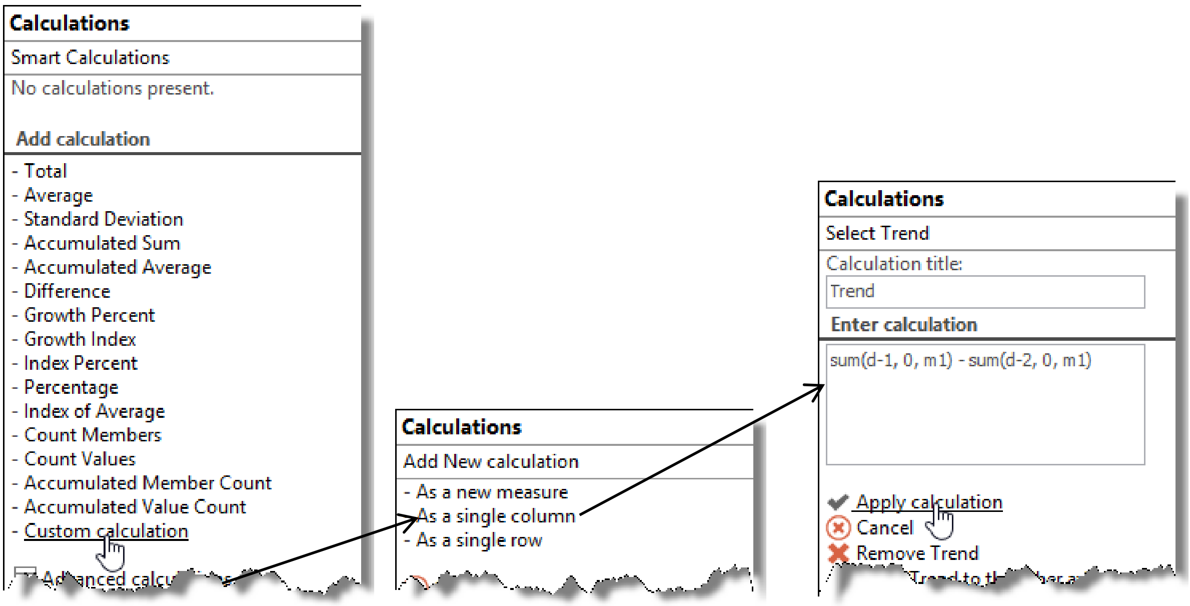

Although this is indeed a coherent Analysis (you can apply Drill down criteria from one object to the others) it is **not very successful** in supplying us with useful information upon which we can base our decisions.

The Analysis can be **dramatically enhanced** by applying calculations and a few features based on those calculations.

## An advanced Analysis

In this part of the demo the basic aspects of referencing columns, rows and measures are demonstrated by applying a calculation.

- Add a calculation, **Trend**, to the Cross table. Trend is calculated as the difference between the last two columns (2015 and 2016) in the Cross table:




- The formula **sum(d-1, 0, m1) - sum(d-2, 0, m1)** will subtract the values in the second last column (2015) from the values in the last column (2016).

- Add a **Growth** calculation, defined as the development (the Trend) expressed as a percentage of the second last column (2015): **sum(c1, 0, m1) / sum(d-2, 0, m1)**.

- Select the added Growth calculation to change the number format to **Percent**.

- Add a **Color Agent** to emphasize a negative or positive Growth:

| | |
|---|---|
| Color Growth **Red** | if **Growth** < 0 |
| Color Growth **Green** | if **Growth** > 0 |
| Color Growth **Yellow** | if **Growth** = 0 |

Now the Crosstab should look like this:

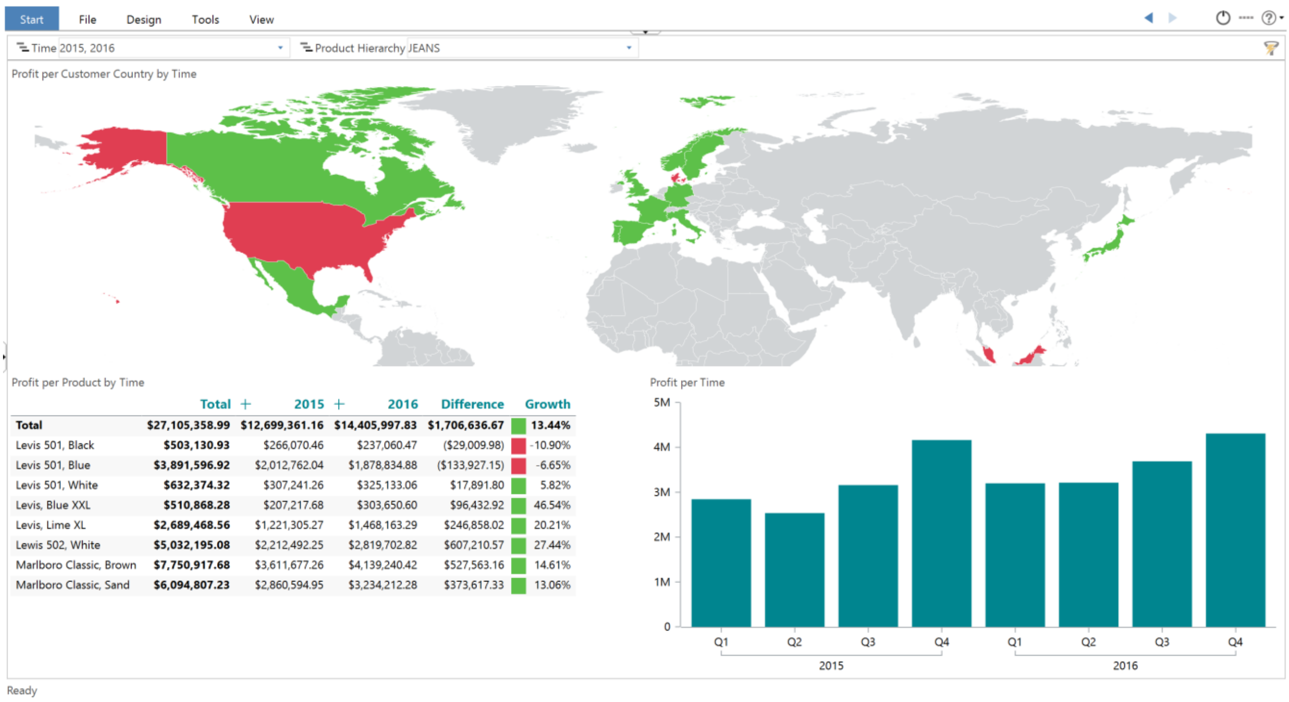
Color and Gauge Agents

| | |
|---|---|
| Growth | Positive values |
| Growth | Zero |
| Growth | Negative valu... |

Profit per Product by Time

| | Total + | 2015 + | 2016 | Difference | Growth |
|---|---|---|---|---|---|
| **Total** | **$27,105,358.99** | **$12,699,361.16** | **$14,405,997.83** | **$1,706,636.67** | **13.44%** |
| Levis 501, Black | **$503,130.93** | $266,070.46 | $237,060.47 | ($29,009.98) | -10.90% |
| Levis 501, Blue | **$3,891,596.92** | $2,012,762.04 | $1,878,834.88 | ($133,927.15) | -6.65% |
| Levis 501, White | **$632,374.32** | $307,241.26 | $325,133.06 | $17,891.80 | 5.82% |
| Levis, Blue XXL | **$510,868.28** | $207,217.68 | $303,650.60 | $96,432.92 | 46.54% |
| Levis, Lime XL | **$2,689,468.56** | $1,221,305.27 | $1,468,163.29 | $246,858.02 | 20.21% |
| Lewis 502, White | **$5,032,195.08** | $2,212,492.25 | $2,819,702.82 | $607,210.57 | 27.44% |
| Marlboro Classic, Brown | **$7,750,917.68** | $3,611,677.26 | $4,139,240.42 | $527,563.16 | 14.61% |
| Marlboro Classic, Sand | **$6,094,807.23** | $2,860,594.95 | $3,234,212.28 | $373,617.33 | 13.06% |

- **Copy** the Cross table (CTRL+C and CTRL+V).
- In the copied cross table, exchange the **Product** dimension with the **Customer Country(Country)** dimension.
- Change the object type to **Map**. (Delete the old Map object.).



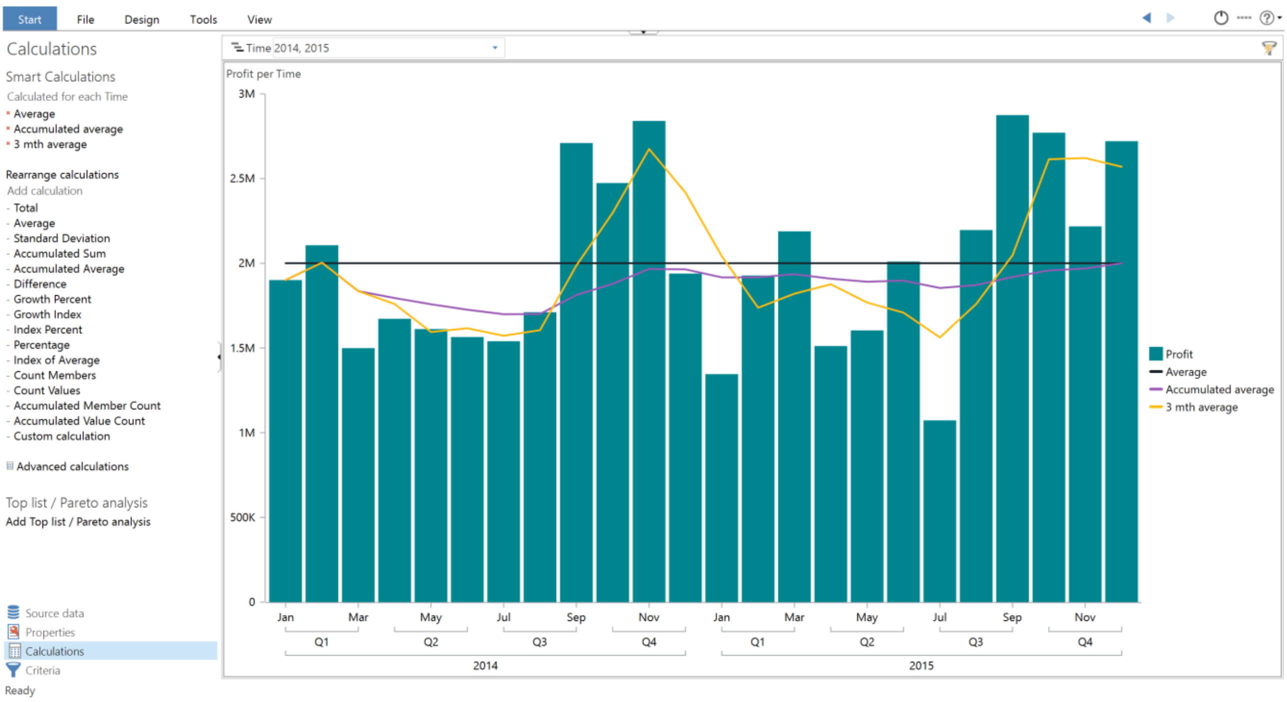

A Map object will always display the values of the *last* column in the underlying cross table, and this is why we are seeing the red and green colors expressing negative an positive Growth respectively on the map.The area chart is useful for Drill Down selections of interesting intervals of periods to be analyzed.

Now the rather useless Analysis has become a **strong platform for decision-making**, simply by adding a calculation and, as in this case, enhancing visibility with a Color Agent based on that calculation.

**More Calculations**

The last part of the demo will demonstrate reference **ranges** and **relative** references.

- Add a Bar chart to the Profit Analysis, **Profit per Time Hierarchy (Month)**.
- Add the following calculations to the Bar chart (all **as a single column**):
  - **Average = avg(d-1, all, m1)** (demonstrating the **all** reference range)
  - **Accumulated average = avg(d-1, d1:0, m1)** (demonstrating an absolute reference range)
  - **3 mth average = avg(d-1, -2:0, m1)** (demonstrating a relative reference)
- Change visualization for each of the three calculations to a **Line**.
- The final result should now look something like this:

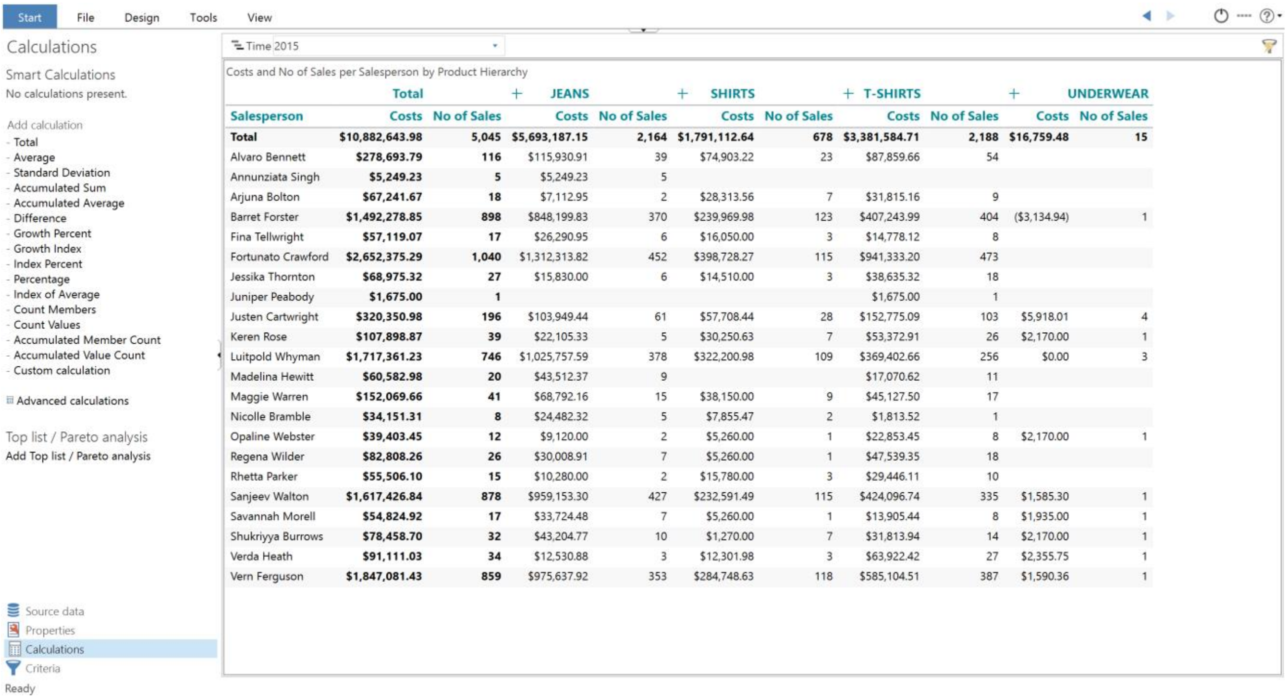


## Calculations as a new measure

In the third part of the demo we will work with calculations as a new measure and referencing a certain dimension value.

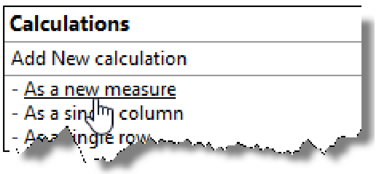
- Create a new **Cost analysis** with a crosstab showing **Costs** and **No of Sales** per **Salesperson** and by **Product Hierarchy(Product group).**

- Add **global criteria Time = 2015.**




| | Total | | JEANS | | SHIRTS | | T-SHIRTS | | UNDERWEAR | |
|---|---|---|---|---|---|---|---|---|---|---|
| Salesperson | Costs | No of Sales | Costs | No of Sales | Costs | No of Sales | Costs | No of Sales | Costs | No of Sales |
| **Total** | **$10,882,643.98** | **5,045** | **$5,693,187.15** | **2,164** | **$1,791,112.64** | **678** | **$3,381,584.71** | **2,188** | **$16,759.48** | **15** |
| Alvaro Bennett | **$278,693.79** | **116** | $115,930.91 | 39 | $74,903.22 | 23 | $87,859.66 | 54 | | |
| Annunziata Singh | **$5,249.23** | **5** | $5,249.23 | 5 | | | | | | |
| Arjuna Bolton | **$67,241.67** | **18** | $7,112.95 | 2 | $28,313.56 | 7 | $31,815.16 | 9 | | |
| Barret Forster | **$1,492,278.85** | **898** | $848,199.83 | 370 | $239,969.98 | 123 | $407,243.99 | 404 | ($3,134.94) | 1 |
| Fina Tellwright | **$57,119.07** | **17** | $26,290.95 | 6 | $16,050.00 | 3 | $14,778.12 | 8 | | |
| Fortunato Crawford | **$2,652,375.29** | **1,040** | $1,312,313.82 | 452 | $398,728.27 | 115 | $941,333.20 | 473 | | |
| Jessika Thornton | **$68,975.32** | **27** | $15,830.00 | 6 | $14,510.00 | 3 | $38,635.32 | 18 | | |
| Juniper Peabody | **$1,675.00** | **1** | | | | | $1,675.00 | 1 | | |
| Justen Cartwright | **$320,350.98** | **196** | $103,949.44 | 61 | $57,708.44 | 28 | $152,775.09 | 103 | $5,918.01 | 4 |
| Keren Rose | **$107,898.87** | **39** | $22,105.33 | 5 | $30,250.63 | 7 | $53,372.91 | 26 | $2,170.00 | 1 |
| Luitpold Whyman | **$1,717,361.23** | **746** | $1,025,757.59 | 378 | $322,200.98 | 109 | $369,402.66 | 256 | $0.00 | 3 |
| Madelina Hewitt | **$60,582.98** | **20** | $43,512.37 | 9 | | | $17,070.62 | 11 | | |
| Maggie Warren | **$152,069.66** | **41** | $68,792.16 | 15 | $38,150.00 | 9 | $45,127.50 | 17 | | |
| Nicolle Bramble | **$34,151.31** | **8** | $24,482.32 | 5 | $7,855.47 | 2 | $1,813.52 | 1 | | |
| Opaline Webster | **$39,403.45** | **12** | $9,120.00 | 2 | $5,260.00 | 1 | $22,853.45 | 8 | $2,170.00 | 1 |
| Regena Wilder | **$82,808.26** | **26** | $30,008.91 | 7 | $5,260.00 | 1 | $47,539.35 | 18 | | |
| Rhetta Parker | **$55,506.10** | **15** | $10,280.00 | 2 | $15,780.00 | 3 | $29,446.11 | 10 | | |
| Sanjeev Walton | **$1,617,426.84** | **878** | $959,153.30 | 427 | $232,591.49 | 115 | $424,096.74 | 335 | $1,585.30 | 1 |
| Savannah Morell | **$54,824.92** | **17** | $33,724.48 | 7 | $5,260.00 | 1 | $13,905.44 | 8 | $1,935.00 | 1 |
| Shukriyya Burrows | **$78,458.70** | **32** | $43,204.77 | 10 | $1,270.00 | 7 | $31,813.94 | 14 | $2,170.00 | 1 |
| Verda Heath | **$91,111.03** | **34** | $12,530.88 | 3 | $12,301.98 | 3 | $63,922.42 | 27 | $2,355.75 | 1 |
| Vern Ferguson | **$1,847,081.43** | **859** | $975,637.92 | 353 | $284,748.63 | 118 | $585,104.51 | 387 | $1,590.36 | 1 |

- Add a calculation (as a new measure):

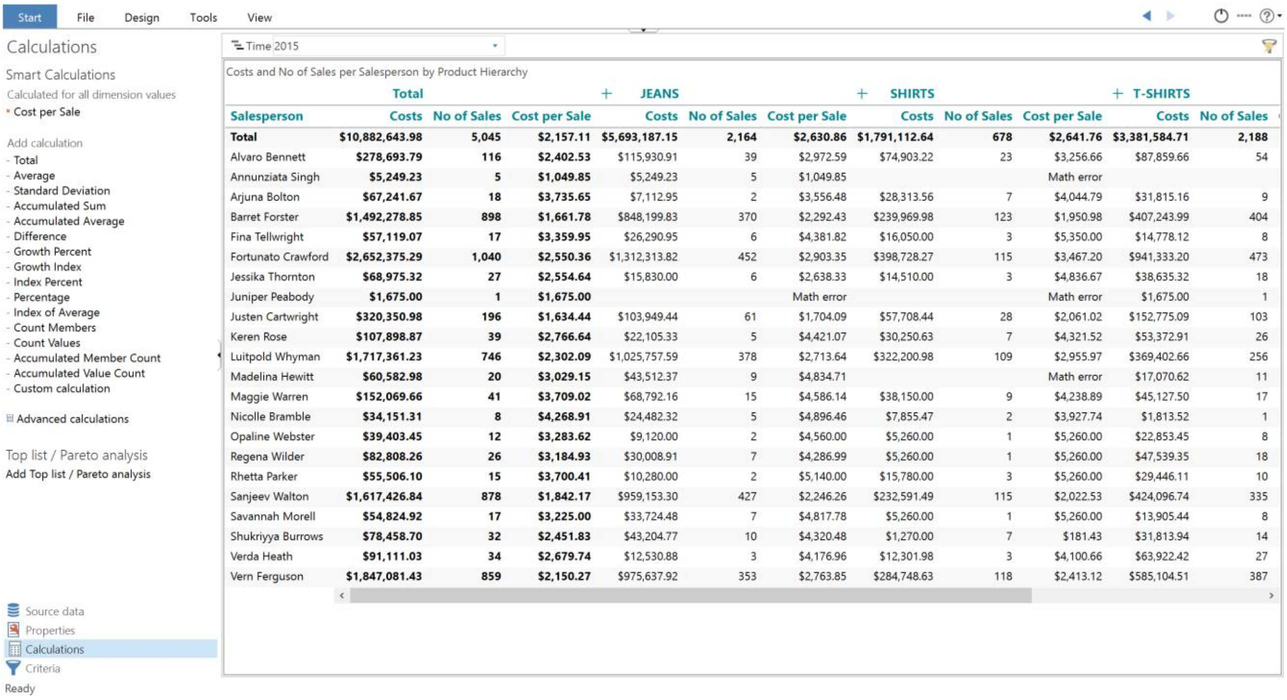



**Costs per Sale = sum(0, 0, m1) / sum(0, 0, m2)**

We have now added a new measure which can now be referred to if needed (in this case as **m3**).

Notice that when using calculation as a new measure you normally do **relative referencing** – we relate to the **current column and row**.

Now, the crosstab looks like this:



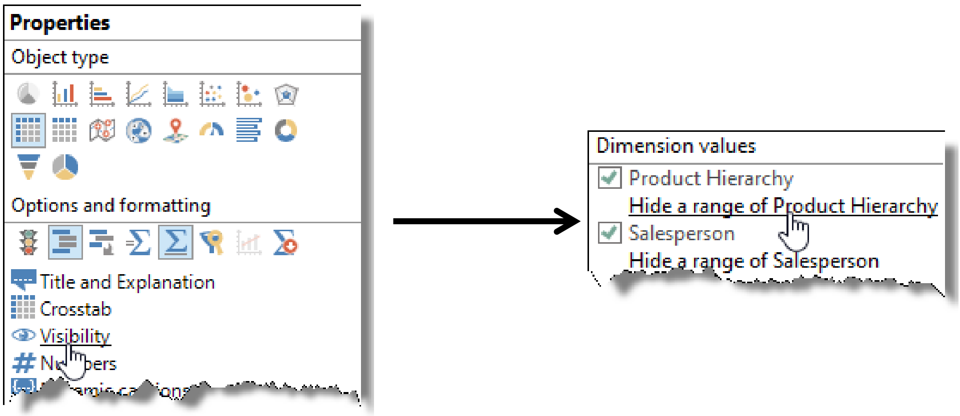

| | Total | | | JEANS | | | SHIRTS | | | T-SHIRTS | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Salesperson | Costs | No of Sales | Cost per Sale | Costs | No of Sales | Cost per Sale | Costs | No of Sales | Cost per Sale | Costs | No of Sales |
| **Total** | **$10,882,643.98** | **5,045** | **$2,157.11** | **$5,693,187.15** | **2,164** | **$2,630.86** | **$1,791,112.64** | **678** | **$2,641.76** | **$3,381,584.71** | **2,188** |
| Alvaro Bennett | **$278,693.79** | **116** | **$2,402.53** | $115,930.91 | 39 | $2,972.59 | $74,903.22 | 23 | $3,256.66 | $87,859.66 | 54 |
| Annunziata Singh | **$5,249.23** | **5** | **$1,049.85** | $5,249.23 | 5 | $1,049.85 | | | Math error | | |
| Arjuna Bolton | **$67,241.67** | **18** | **$3,735.65** | $7,112.95 | 2 | $3,556.48 | $28,313.56 | 7 | $4,044.79 | $31,815.16 | 9 |
| Barret Forster | **$1,492,278.85** | **898** | **$1,661.78** | $848,199.83 | 370 | $2,292.43 | $239,969.98 | 123 | $1,950.98 | $407,243.99 | 404 |
| Fina Tellwright | **$57,119.07** | **17** | **$3,359.95** | $26,290.95 | 6 | $4,381.82 | $16,050.00 | 3 | $5,350.00 | $14,778.12 | 8 |
| Fortunato Crawford | **$2,652,375.29** | **1,040** | **$2,550.36** | $1,312,313.82 | 452 | $2,903.35 | $398,728.27 | 115 | $3,467.20 | $941,333.20 | 473 |
| Jessika Thornton | **$68,975.32** | **27** | **$2,554.64** | $15,830.00 | 6 | $2,638.33 | $14,510.00 | 3 | $4,836.67 | $38,635.32 | 18 |
| Juniper Peabody | **$1,675.00** | **1** | **$1,675.00** | | | Math error | | | Math error | $1,675.00 | 1 |
| Justen Cartwright | **$320,350.98** | **196** | **$1,634.44** | $103,949.44 | 61 | $1,704.09 | $57,708.44 | 28 | $2,061.02 | $152,775.09 | 103 |
| Keren Rose | **$107,898.87** | **39** | **$2,766.64** | $22,105.33 | 5 | $4,421.07 | $30,250.63 | 7 | $4,321.52 | $53,372.91 | 26 |
| Luitpold Whyman | **$1,717,361.23** | **746** | **$2,302.09** | $1,025,757.59 | 378 | $2,713.64 | $322,200.98 | 109 | $2,955.97 | $369,402.66 | 256 |
| Madelina Hewitt | **$60,582.98** | **20** | **$3,029.15** | $43,512.37 | 9 | $4,834.71 | | | Math error | $17,070.62 | 11 |
| Maggie Warren | **$152,069.66** | **41** | **$3,709.02** | $68,792.16 | 15 | $4,586.14 | $38,150.00 | 9 | $4,238.89 | $45,127.50 | 17 |
| Nicolle Bramble | **$34,151.31** | **8** | **$4,268.91** | $24,482.32 | 5 | $4,896.46 | $7,855.47 | 2 | $3,927.74 | $1,813.52 | 1 |
| Opaline Webster | **$39,403.45** | **12** | **$3,283.62** | $9,120.00 | 2 | $4,560.00 | $5,260.00 | 1 | $5,260.00 | $22,853.45 | 8 |
| Regena Wilder | **$82,808.26** | **26** | **$3,184.93** | $30,008.91 | 7 | $4,286.99 | $5,260.00 | 1 | $5,260.00 | $47,539.35 | 18 |
| Rhetta Parker | **$55,506.10** | **15** | **$3,700.41** | $10,280.00 | 2 | $5,140.00 | $15,780.00 | 3 | $5,260.00 | $29,446.11 | 10 |
| Sanjeev Walton | **$1,617,426.84** | **878** | **$1,842.17** | $959,153.30 | 427 | $2,246.26 | $232,591.49 | 115 | $2,022.53 | $424,096.74 | 335 |
| Savannah Morell | **$54,824.92** | **17** | **$3,225.00** | $33,724.48 | 7 | $4,817.78 | $5,260.00 | 1 | $5,260.00 | $13,905.44 | 8 |
| Shukriyya Burrows | **$78,458.70** | **32** | **$2,451.83** | $43,204.77 | 10 | $4,320.48 | $1,270.00 | 7 | $181.43 | $31,813.94 | 14 |
| Verda Heath | **$91,111.03** | **34** | **$2,679.74** | $12,530.88 | 3 | $4,176.96 | $12,301.98 | 3 | $4,100.66 | $63,922.42 | 27 |
| Vern Ferguson | **$1,847,081.43** | **859** | **$2,150.27** | $975,637.92 | 353 | $2,763.85 | $284,748.63 | 118 | $2,413.12 | $585,104.51 | 387 |

- Use **visibility agents** to hide all **salespersons** who had a **total No of Sales less then 100 during the period.**

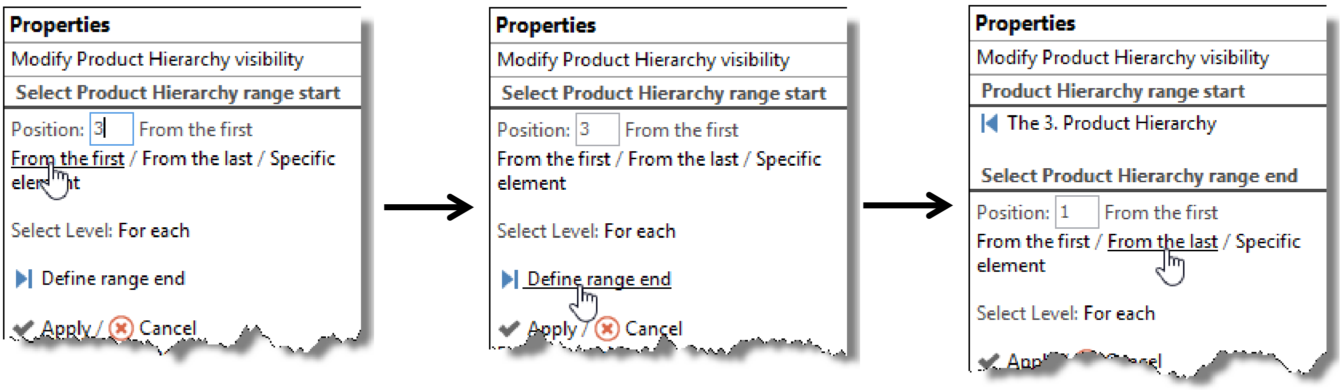
The **condition** for the **visibility agent** looks like this:

Use the **Visibility** formatting to hide the **Underwear** and **T-shirt** numbers.

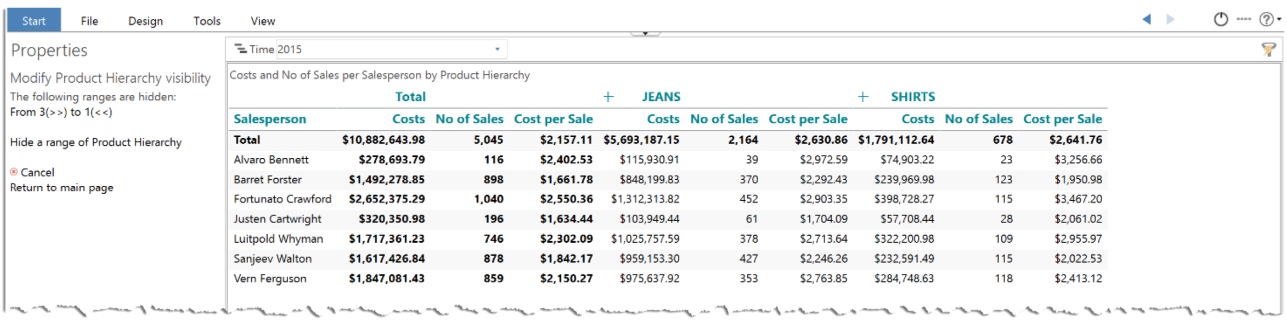
- Go to **Properties** and **Visibility** and choose **hide a range of Product Hierarchy.**




- Choose **Position 3 from the first** – and **Define range end** – **choose Position 1 From the last** – click **Apply.**

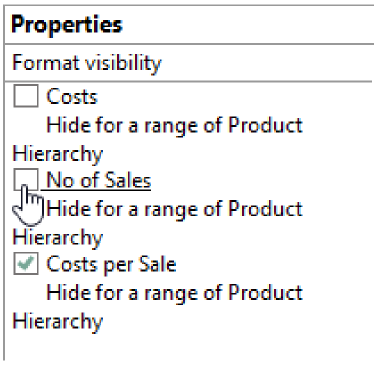



This will hide the last two columns of the cross tab and the resulting cross tab should now look something like this:

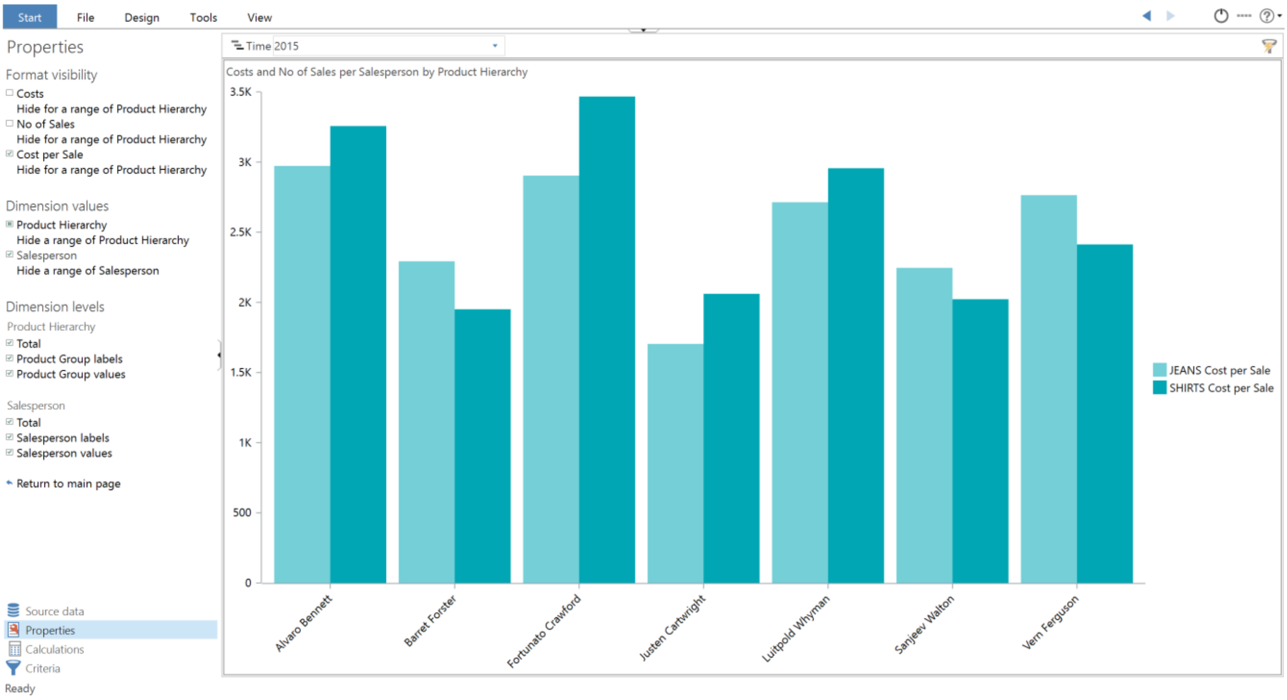



| | Total | | | JEANS | | | SHIRTS | | |
|---|---|---|---|---|---|---|---|---|---|
| Salesperson | Costs | No of Sales | Cost per Sale | Costs | No of Sales | Cost per Sale | Costs | No of Sales | Cost per Sale |
| Total | $10,882,643.98 | 5,045 | $2,157.11 | $5,693,187.15 | 2,164 | $2,630.86 | $1,791,112.64 | 678 | $2,641.76 |
| Alvaro Bennett | $278,693.79 | 116 | $2,402.53 | $115,930.91 | 39 | $2,972.59 | $74,903.22 | 23 | $3,256.66 |
| Barret Forster | $1,492,278.85 | 898 | $1,661.78 | $848,199.83 | 370 | $2,292.43 | $239,969.98 | 123 | $1,950.98 |
| Fortunato Crawford | $2,652,375.29 | 1,040 | $2,550.36 | $1,312,313.82 | 452 | $2,903.35 | $398,728.27 | 115 | $3,467.20 |
| Justen Cartwright | $320,350.98 | 196 | $1,634.44 | $103,949.44 | 61 | $1,704.09 | $57,708.44 | 28 | $2,061.02 |
| Luitpold Whyman | $1,717,361.23 | 746 | $2,302.09 | $1,025,757.59 | 378 | $2,713.64 | $322,200.98 | 109 | $2,955.97 |
| Sanjeev Walton | $1,617,426.84 | 878 | $1,842.17 | $959,153.30 | 427 | $2,246.26 | $232,591.49 | 115 | $2,022.53 |
| Vern Ferguson | $1,847,081.43 | 859 | $2,150.27 | $975,637.92 | 353 | $2,763.85 | $284,748.63 | 118 | $2,413.12 |

**Hide No of Sales** and **Costs** using the **Visibility** formatting and change **object type** to **Column chart.**




With a little further formatting, the result should look something like this:



### Direct reference to dimension member value

Add another crosstab to the analysis: **Costs and No of Sales per Customer Country (Country)**.

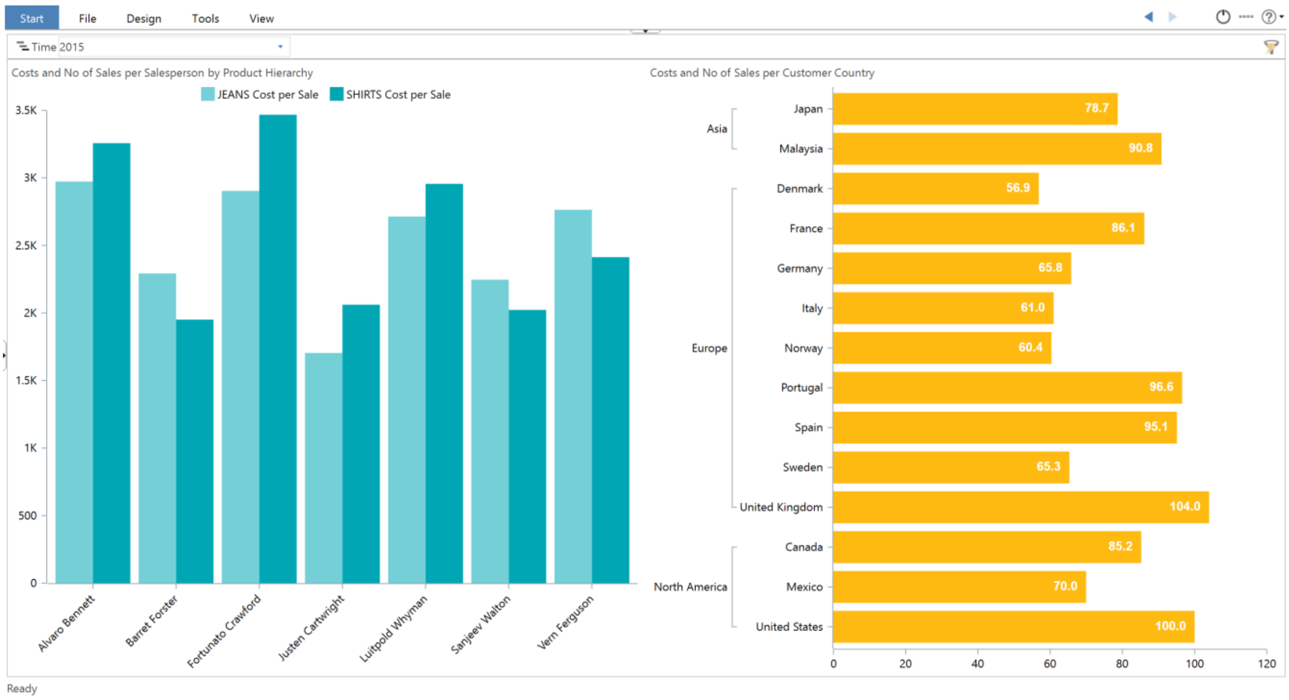
Add a **calculation as a single column per Customer Country**:

- **Costs per Sale = sum(d1, 0, m1) / sum(d1, 0, m2)**

Now we will calculate what the level of costs is in other countries compared to the American numbers – a kind of US-index.

- **US index: sum(c1, 0, m1) / sum(c1, @"[North America].[United States]", m1)**
- **Hide Costs, No of Sales** and **Costs per Sale** through the **Visibility** formatting option.
- **Change the object type** to **horizontal bar chart.**

The end result should look like this:



## Exercises Lesson 1

(Screenshots and exercises are based on version 2018.3 demo data. If you working on an earlier or later version you may need to subtract or add 1 year to achieve similar results.)

### Task 1

Create a new Analysis for calculation of Unit Prices. Generally Unit Price can be calculated as **Unit Price = Revenue / Units Sold**.

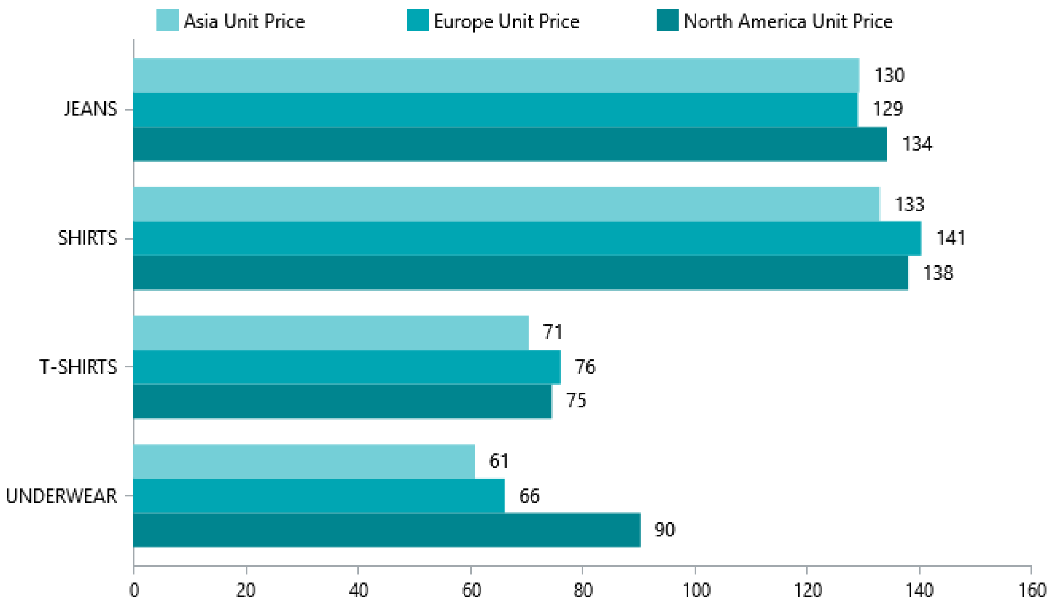
Start out with a Cross table **Revenue** and **Units Sold** per **Product Hierarchy (Product group)** by **Customer (Territory)**.

- Calculate the Unit Price to get a result like this:

Revenue and Units Sold per Product Hierarchy by Customer Territory

| | Total | | | Asia | | | Europe | | | North America | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Product Group** | **Revenue** | **Units Sold** | **Unit Price** | **Revenue** | **Units Sold** | **Unit Price** | **Revenue** | **Units Sold** | **Unit Price** | **Revenue** | **Units Sold** | **Unit Price** |
| **Total** | **182,541,552** | **1,723,000** | **106** | **44,397,608** | **459,900** | **97** | **106,447,337** | **981,000** | **109** | **31,696,607** | **282,100** | **112** |
| JEANS + | **95,985,049** | **737,500** | **130** | 20,177,380 | 155,800 | 130 | 59,483,070 | 460,300 | 129 | 16,324,598 | 121,400 | 134 |
| SHIRTS + | **29,135,231** | **210,500** | **138** | 5,898,649 | 44,300 | 133 | 15,939,790 | 113,400 | 141 | 7,296,792 | 52,800 | 138 |
| T-SHIRTS + | **56,980,759** | **768,600** | **74** | 18,230,101 | 258,300 | 71 | 30,765,884 | 403,400 | 76 | 7,984,774 | 106,900 | 75 |
| UNDERWEAR + | **440,513** | **6,400** | **69** | 91,479 | 1,500 | 61 | 258,592 | 3,900 | 66 | 90,442 | 1,000 | 90 |

- Add a horizontal **Bar chart** to display the Unit Prices for each of the territories (hint: Use the **Visibility** formatting option to hide the irrelevant data):



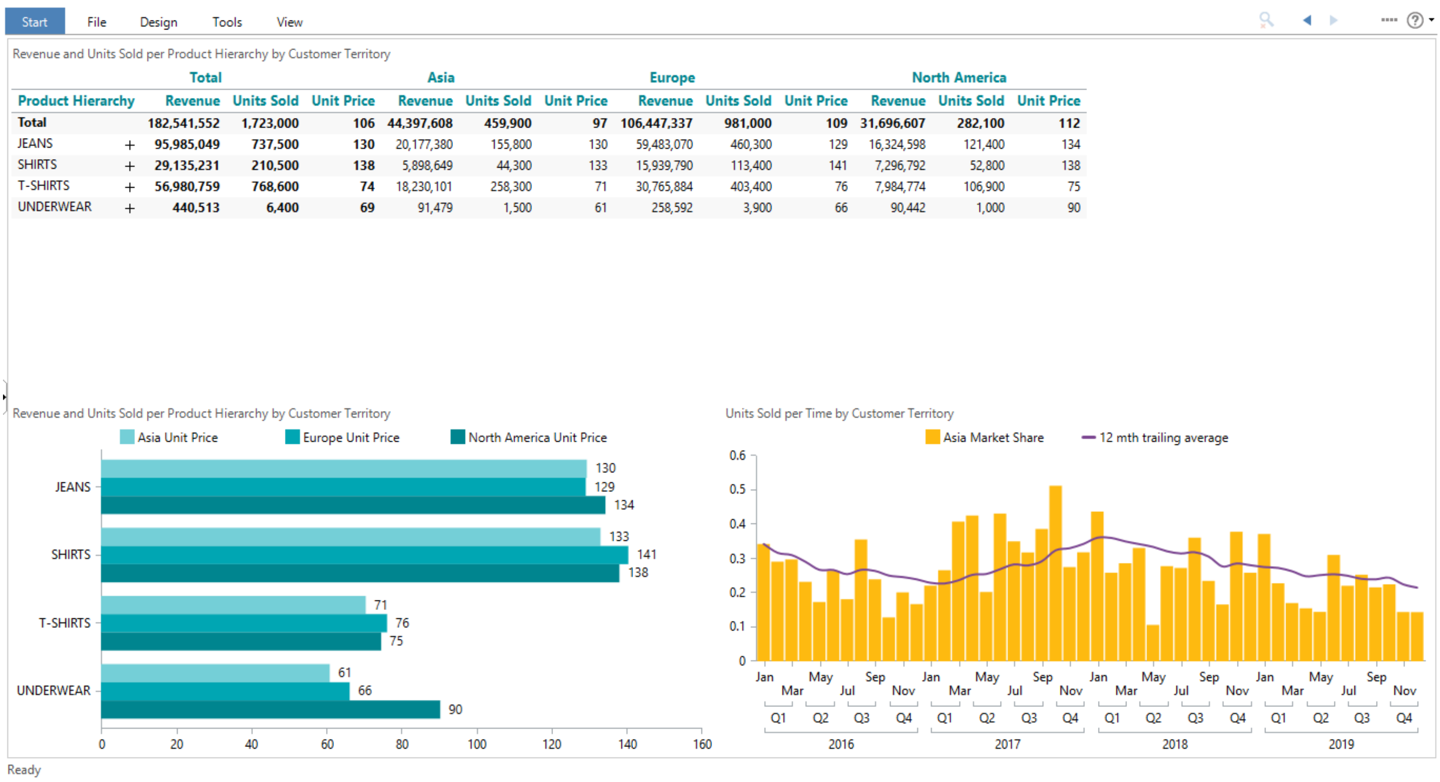
- Add a vertical Bar chart, **Units sold** per **Time Hierarchy (Month)** by **Customer (Territory)**.

- The bar chart must include and display these two calculations:

  - **Asia market share.** For each Month, the number of units sold in Asia must be calculated as a percentage of the **total** number of units sold within that Month.

  - **12 Months trailing average.** This calculation may also be referred to as a 'rolling' average. For each Month, this average is calculated as the average of the 12 Months ranging from 11 Months earlier than current Month until the current Month.

- The complete analysis should now look like this:




| Product Hierarchy | Total Revenue | Total Units Sold | Total Unit Price | Asia Revenue | Asia Units Sold | Asia Unit Price | Europe Revenue | Europe Units Sold | Europe Unit Price | North America Revenue | North America Units Sold | North America Unit Price |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Total | 182,541,552 | 1,723,000 | 106 | 44,397,608 | 459,900 | 97 | 106,447,337 | 981,000 | 109 | 31,696,607 | 282,100 | 112 |
| JEANS + | 95,985,049 | 737,500 | 130 | 20,177,380 | 155,800 | 130 | 59,483,070 | 460,300 | 129 | 16,324,598 | 121,400 | 134 |
| SHIRTS + | 29,135,231 | 210,500 | 138 | 5,898,649 | 44,300 | 133 | 15,939,790 | 113,400 | 141 | 7,296,792 | 52,800 | 138 |
| T-SHIRTS + | 56,980,759 | 768,600 | 74 | 18,230,101 | 258,300 | 71 | 30,765,884 | 403,400 | 76 | 7,984,774 | 106,900 | 75 |
| UNDERWEAR + | 440,513 | 6,400 | 69 | 91,479 | 1,500 | 61 | 258,592 | 3,900 | 66 | 90,442 | 1,000 | 90 |



- Save the analysis as **Lesson 1 Unit Price analysis**.

**Task 2**

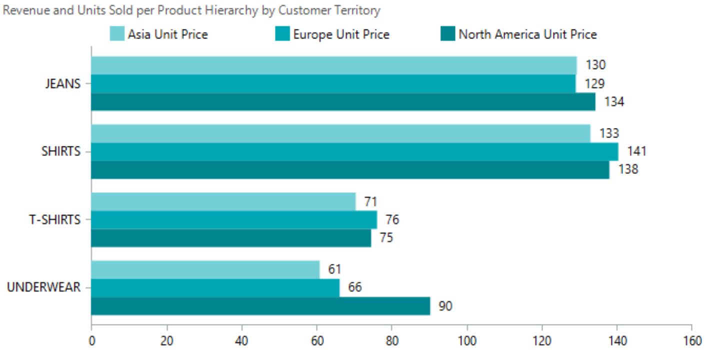
The 12 months trailing average calculation is not entirely correct when looking at the beginning of the time range – here, the calculation will be based on less than 12 months.

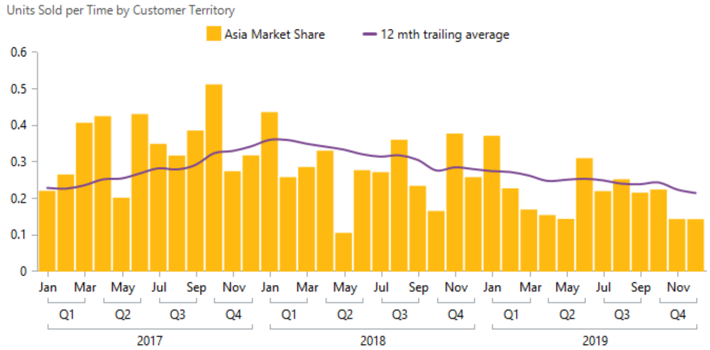
To correct this, at least in the beginning of the period, use **range visibility** to hide the first 12 months of the bar chart:

Start File Design Tools View

Revenue and Units Sold per Product Hierarchy by Customer Territory

| | Total | | | Asia | | | Europe | | | North America | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Product Hierarchy | Revenue | Units Sold | Unit Price | Revenue | Units Sold | Unit Price | Revenue | Units Sold | Unit Price | Revenue | Units Sold | Unit Price |
| Total | 182,541,552 | 1,723,000 | 106 | 44,397,608 | 459,900 | 97 | 106,447,337 | 981,000 | 109 | 31,696,607 | 282,100 | 112 |
| JEANS + | 95,985,049 | 737,500 | 130 | 20,177,380 | 155,800 | 130 | 59,483,070 | 460,300 | 129 | 16,324,598 | 121,400 | 134 |
| SHIRTS + | 29,135,231 | 210,500 | 138 | 5,898,649 | 44,300 | 133 | 15,939,790 | 113,400 | 141 | 7,296,792 | 52,800 | 138 |
| T-SHIRTS + | 56,980,759 | 768,600 | 74 | 18,230,101 | 258,300 | 71 | 30,765,884 | 403,400 | 76 | 7,984,774 | 106,900 | 75 |
| UNDERWEAR + | 440,513 | 6,400 | 69 | 91,479 | 1,500 | 61 | 258,592 | 3,900 | 66 | 90,442 | 1,000 | 90 |



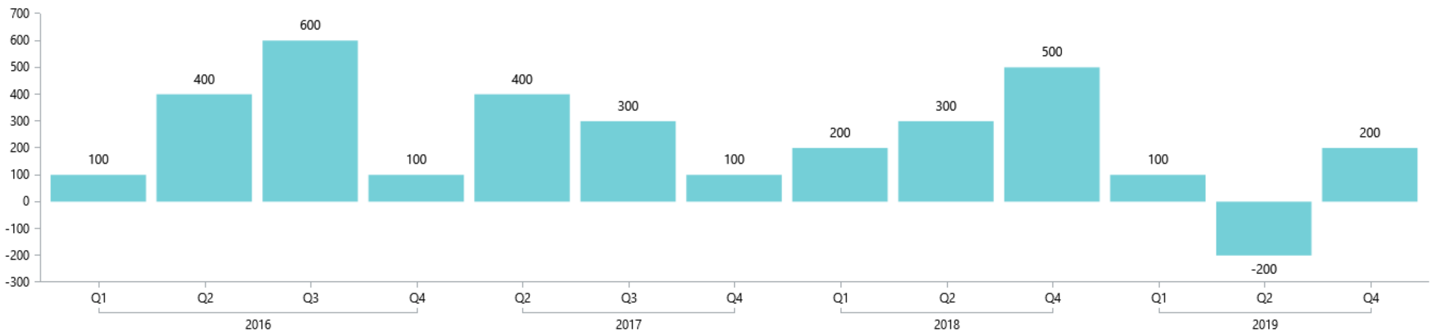




Ready

### Task 3

- Add a Bar chart to the **Unit Price** Analysis: **Units Sold per Time Hierarchy (Quarter)**.
- Add a calculation, **2017/Q4 Index.** This calculation is an Index calculation based on 4th Quarter of 2017. The number of Units Sold in this Quarter will be equal to an Index of 100 %, and the number of Units Sold in all other Quarters must be calculated as an Index in relation to 4th Quarter 2017.
- The calculation must be generalized to work with any criteria. I.e. 4th Quarter 2017 must be basis for the Index calculation no matter if this Quarter appears as the first, third or seventh row in the underlying cross table.
- The result, with **no criteria**, should look like this:


- Add the criterion **Salesperson = Nicolle Bramble** to achieve this result:



# Lesson 2: Operators, the 4th parameter and count/allcount

## General purpose

This lesson will introduce the complete list of available functions. Specifically we will be looking at if-then-else statements and how to use labels to improve the readability of our formulas.

## Summation / Recap

- **If-then-else** statements are used to check for certain conditions and to provide alternative results based on the different conditions. Specifically the if-then-else statement is useful to prevent a division-by-zero situation.
  An If-then-else can also be "nested" which means setting up more conditions (and actions when the conditions are met) inside one if-sentence.
  - **if [A] then [B] else [C]**
  - **if [A] then [B] else if [C] then [D] else [E]**
- With **labels** you can name specific expressions within your formula and refer to those labels from other places in the formula. This is especially useful when working with complex formulas, like the if-then-else statement, where one or more expressions may be used several times within the same formula.
  - **[label:] ([**expression]**)**

- The **4th parameter** can be used to address problems concerning **references out of range**. References to cells not available in the current dataset (e.g. a reference to previous month in the first month of a dataset). Such a reference would normally result in the error message "not defined", but with the 4th parameter you have the option to insert a value instead of the not-reachable value.

  The syntax could be: **sum(d1, -1, m1, 0)**. This reference to the previous row will (in case previous row is not-reachable) return a zero instead.

- **Count** counts all the values in a set of cells, while **allcount** counts all the cells regardless of content (**null-values** as well). This can be used to **validate** if all the wanted/required registrations are present in a dataset.

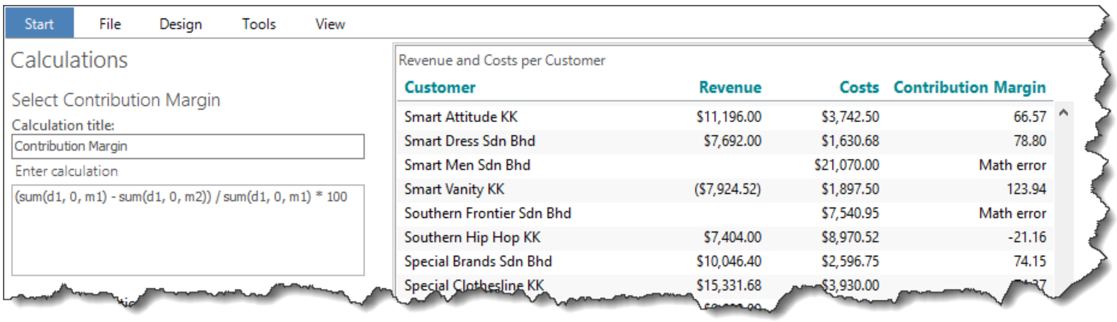
  The formula **count(d1, all, m1) % allcount(d1, all, m1)** will calculate the percentage of cells in the first column from the left containing numeric values.

## Demo

The demo will demonstrate a very common situation: A cross table that includes a custom calculation where a division-by-zero situation can occur.

Labels will added to the formula to enhance the readability.

- Create a new Analysis with a Cross table showing **Revenue and Costs per Customer**
- Add a new calculation **Contribution Margin** = (Revenue – Costs) / Revenue * 100.
- Apply the global criterion **Customer Country(Territory) = Asia.**

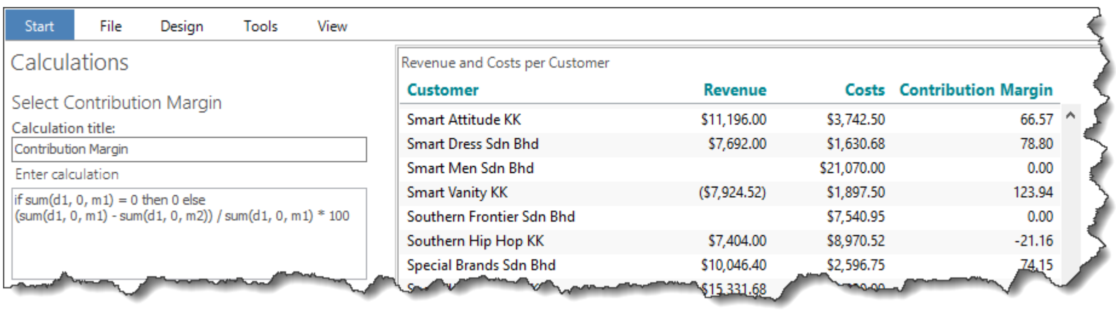



Notice the "Math errors" – which are caused by division by zero.

To fix this, we will implement an "if-then-else" statement in our calculations syntax.

### If-then-else

- In the formula, include an **if-then-else** statement to handle the division-by-zero issue. Insert "0" when a division by zero is attempted:



**A "nested" If-then-else**

Maybe more calculations need to be done depending on different conditions.

This can be solved using a "nested" if-sentence.

- Create a new **Bonus analysis** with a crosstab showing **Revenue per Salesperson**.
- Add a **dynamic criteria = Previous month**

Now we've got a basis for calculating the monthly bonus for the Salespersons.
The bonus is given according to these rules:

- **Revenue 0-50.000 = 0 in bonus**
- **Revenue > 50.000 = 5% of Revenue in bonus**
- **Revenue >= 100.000 and < 500.000 = 10% of Revenue in bonus**
- **Revenue >= 500.000 = 15% of Revenue in bonus**

This can be translated to Targit calculation syntax using one long **nested if sentence**:

***if sum(d1, 0, m1) >= 500000 then sum(d1, 0, m1) * 0,15 else***
***if sum(d1, 0, m1) >= 100000 then sum(d1, 0, m1) * 0,1 else***
***if sum(d1, 0, m1) >= 50000 then sum(d1, 0, m1) * 0,05 else 0***

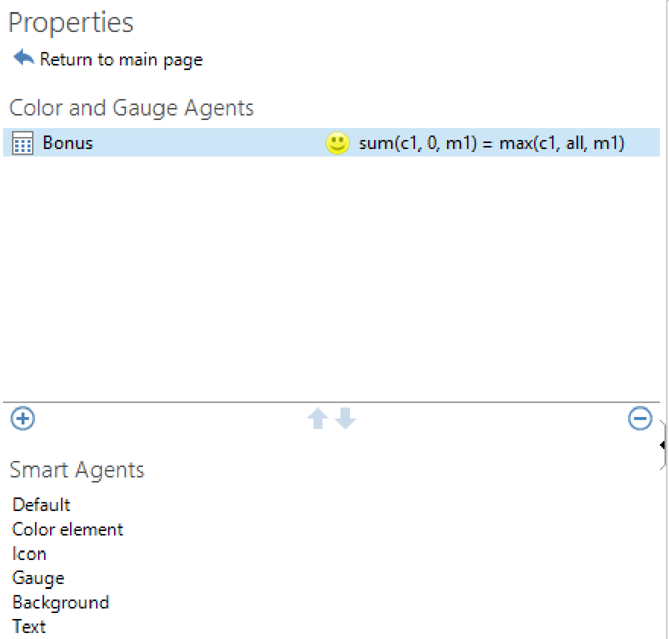
**Notice**: The last else takes care of those who do not meet any of the conditions and as a result has no bonus coming.

Adding **Labels** could make this sentence a lot more **readable**:

**if SALES:(sum(d1, 0, m1)) >= 500000 then SALES * 0,15 else**
**if SALES >= 100000 then SALES * 0,1 else**
**if SALES >= 50000 then SALES * 0,05 else 0**

Make sure that an **icon agent** highlights the salesperson who recieves the highest bonus.

With **June 2017** set as the dynamic date origin the crosstab should look like this:




Revenue per Salesperson

| Salesperson | Revenue | Bonus |
|---|---|---|
| Alvaro Bennett | $85,534.80 | $4,276.74 |
| Arjuna Bolton | $57,671.98 | $2,883.60 |
| Barret Forster | $510,872.70 | $76,630.91 |
| Fortunato Crawford | $117,295.63 | $11,729.56 |
| Jessika Thornton | $83,553.60 | $4,177.68 |
| Juniper Peabody | $7,784.40 | $0.00 |
| Justen Cartwright | $192,577.38 | $19,257.74 |
| Keren Rose | $17,550.00 | $0.00 |
| Luitpold Whyman | $562,811.60 | $84,421.74 |
| Madelina Hewitt | $30,388.80 | $0.00 |
| Maggie Warren | $30,108.00 | $0.00 |
| Opaline Webster | $53,362.40 | $2,668.12 |
| Regena Wilder | $54,568.80 | $2,728.44 |
| Rhetta Parker | $49,498.80 | $0.00 |
| Sanjeev Walton | $371,940.40 | $37,194.04 |
| Shukriyya Burrows | $58,188.00 | $2,909.40 |
| Verda Heath | $51,838.80 | $2,591.94 |
| Vern Ferguson | $799,886.36 | $119,982.95 |

## The 4th parameter

The 4th parameter is useful when **referencing out of range** in relation to the dataset that has been defined for the crosstab at hand.

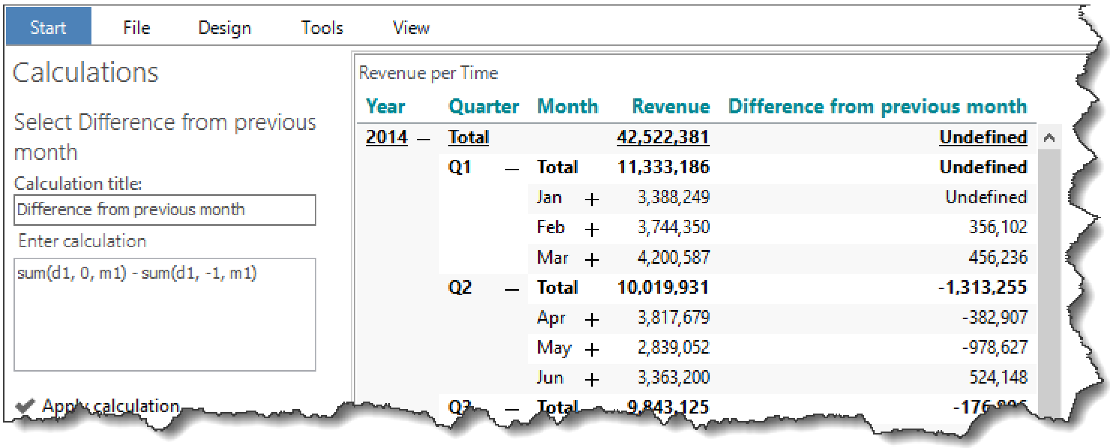
This will normally result in the error message "*not defined*". By using the 4th parameter you can insert a numeric value which will replace the error message.

An example:

- Create a new analysis **Monthdifference** with a crosstab showing **Revenue per Time Hierarchy (Month)**.
- Add a calculation showing the **difference between current month and the previous month**.

The formula could be: **sum(d1, 0, m1) – sum(d1, -1, m1)**

In this particular case this formular will return "*Undefined*" in the first row – because there is no previous month in the dataset – and the reference is out of range.

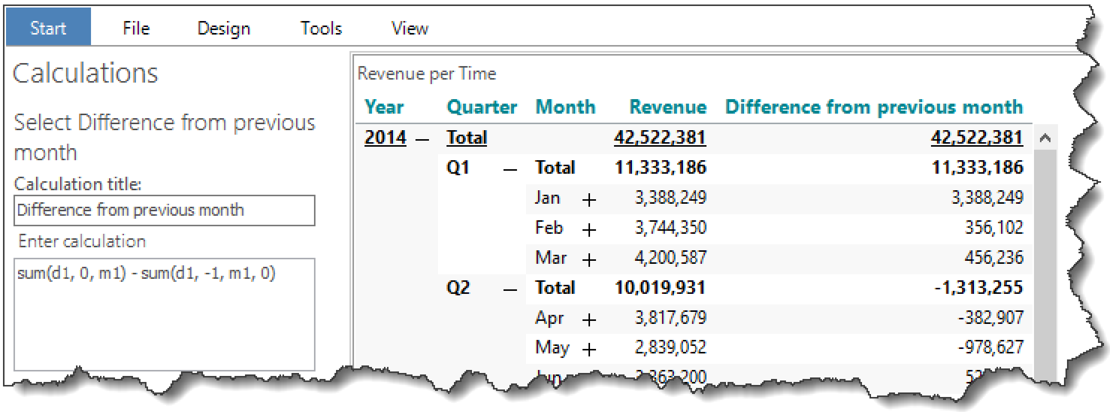



In this case the 4th parameter can be a solution.

New formula: **sum(d1, 0, m1) – sum(d1, -1, m1, 0)**

The last 0 indicates that the **value 0 should be inserted** in case of **reference out of range**.

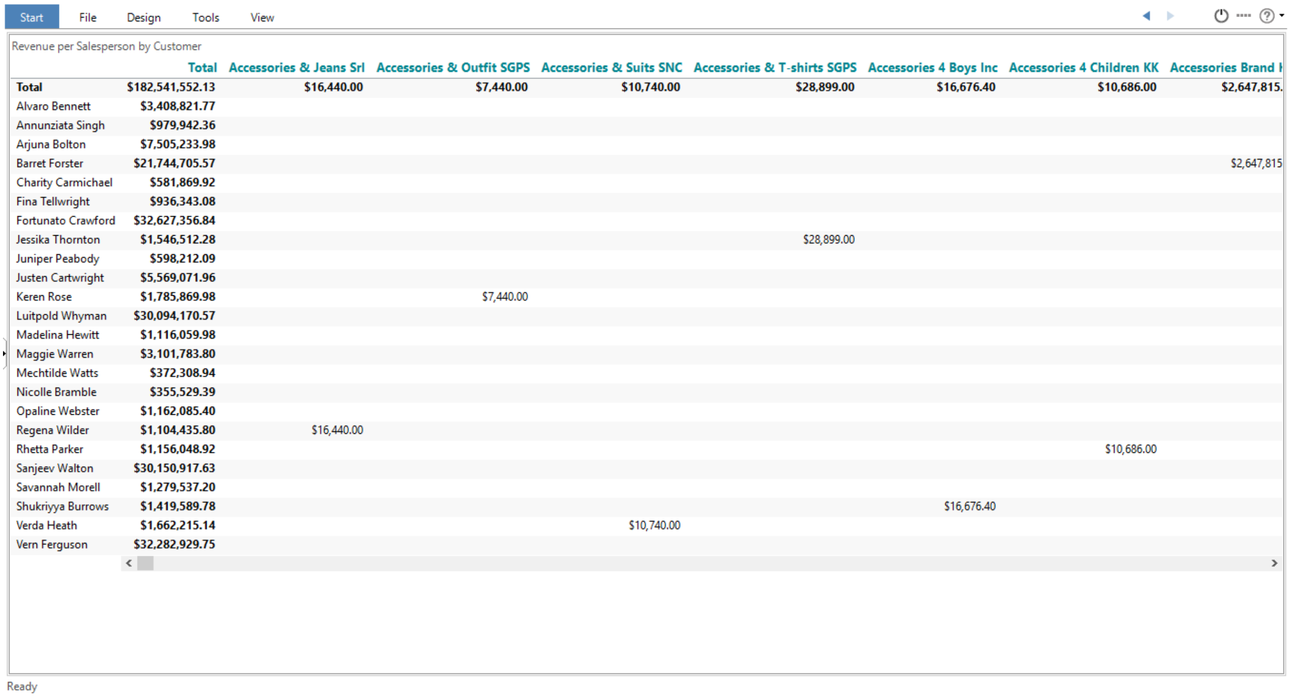
Now the first row (January) is calculated as if the previous month was 0.



**Count/Allcount**

We create another analysis **Customer Activity** with a **crosstab** showing **Revenue per Salesperson and by Customer**.

As this part of the crosstab shows, it is just a fraction of the Customers that each Salesperson gets his or hers Revenue from.

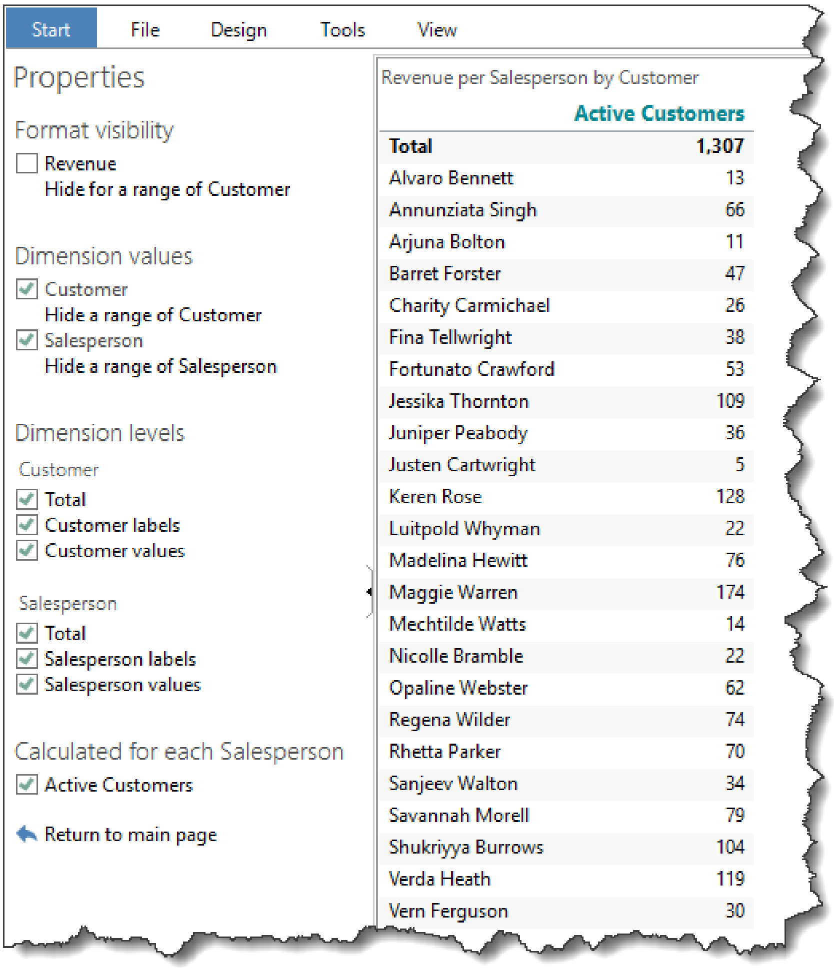



| | Total | Accessories & Jeans Srl | Accessories & Outfit SGPS | Accessories & Suits SNC | Accessories & T-shirts SGPS | Accessories 4 Boys Inc | Accessories 4 Children KK | Accessories Brand I |
|---|---|---|---|---|---|---|---|---|
| **Total** | $182,541,552.13 | $16,440.00 | $7,440.00 | $10,740.00 | $28,899.00 | $16,676.40 | $10,686.00 | $2,647,815. |
| Alvaro Bennett | $3,408,821.77 | | | | | | | |
| Annunziata Singh | $979,942.36 | | | | | | | |
| Arjuna Bolton | $7,505,233.98 | | | | | | | |
| Barret Forster | $21,744,705.57 | | | | | | | $2,647,815 |
| Charity Carmichael | $581,869.92 | | | | | | | |
| Fina Tellwright | $936,343.08 | | | | | | | |
| Fortunato Crawford | $32,627,356.84 | | | | | | | |
| Jessika Thornton | $1,546,512.28 | | | | $28,899.00 | | | |
| Juniper Peabody | $598,212.09 | | | | | | | |
| Justen Cartwright | $5,569,071.96 | | | | | | | |
| Keren Rose | $1,785,869.98 | | $7,440.00 | | | | | |
| Luitpold Whyman | $30,094,170.57 | | | | | | | |
| Madelina Hewitt | $1,116,059.98 | | | | | | | |
| Maggie Warren | $3,101,783.80 | | | | | | | |
| Mechtilde Watts | $372,308.94 | | | | | | | |
| Nicolle Bramble | $355,529.39 | | | | | | | |
| Opaline Webster | $1,162,085.40 | | | | | | | |
| Regena Wilder | $1,104,435.80 | $16,440.00 | | | | | | |
| Rhetta Parker | $1,156,048.92 | | | | | | $10,686.00 | |
| Sanjeev Walton | $30,150,917.63 | | | | | | | |
| Savannah Morell | $1,279,537.20 | | | | | | | |
| Shukriyya Burrows | $1,419,589.78 | | | | | $16,676.40 | | |
| Verda Heath | $1,662,215.14 | | | $10,740.00 | | | | |
| Vern Ferguson | $32,282,929.75 | | | | | | | |

First of all we want a count of how many Customers each Salesperson has covered in terms of Revenue.

- **Active customers** = **count(all, 0, m1)**

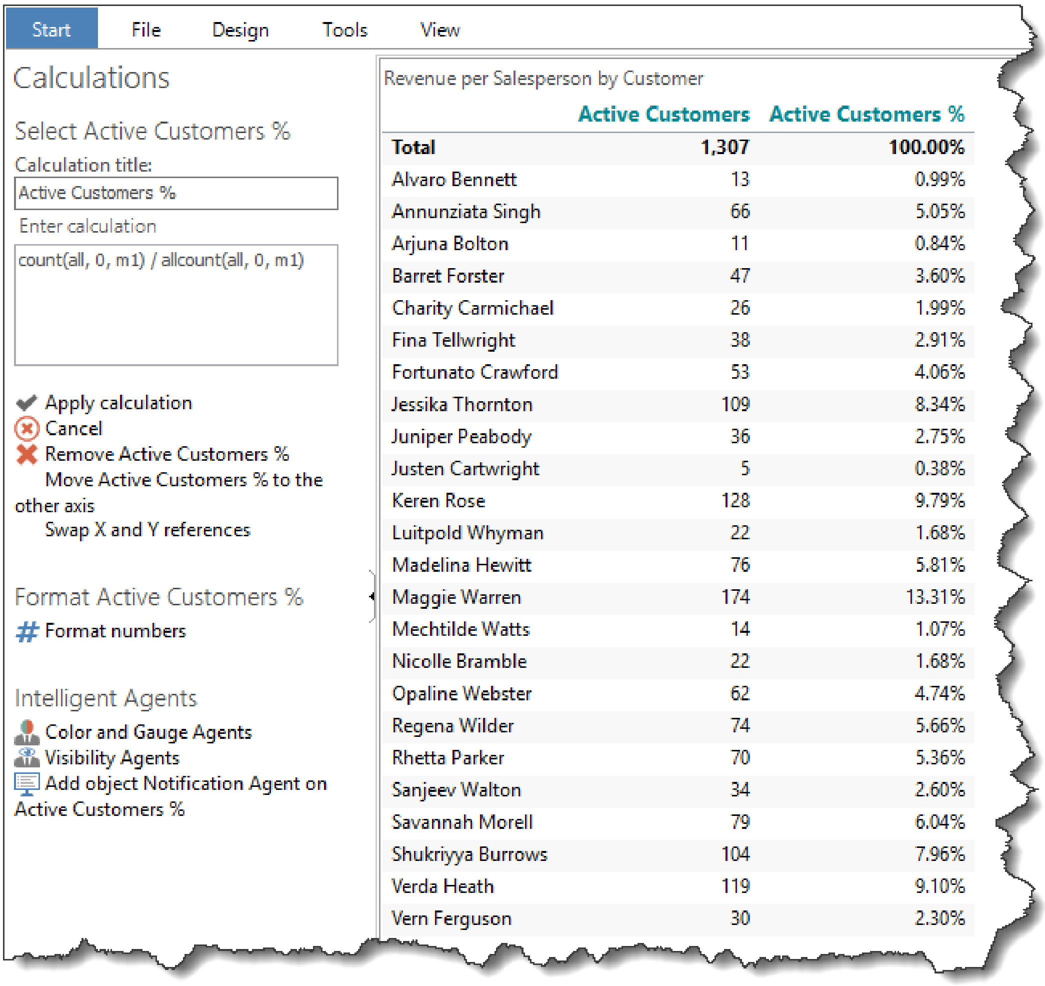
Hide, through Visibility formatting, everything but the calculation – this should make the crosstab look like this:



A refined variation of this counting could be to relate the count to an allcount (all possible customers).

- **Active Customer % = count(all, 0, m1) / allcount(all, 0, m1)**

With a bit of formatting the final crosstab should look like this:




| | Active Customers | Active Customers % |
|---|---|---|
| **Total** | **1,307** | **100.00%** |
| Alvaro Bennett | 13 | 0.99% |
| Annunziata Singh | 66 | 5.05% |
| Arjuna Bolton | 11 | 0.84% |
| Barret Forster | 47 | 3.60% |
| Charity Carmichael | 26 | 1.99% |
| Fina Tellwright | 38 | 2.91% |
| Fortunato Crawford | 53 | 4.06% |
| Jessika Thornton | 109 | 8.34% |
| Juniper Peabody | 36 | 2.75% |
| Justen Cartwright | 5 | 0.38% |
| Keren Rose | 128 | 9.79% |
| Luitpold Whyman | 22 | 1.68% |
| Madelina Hewitt | 76 | 5.81% |
| Maggie Warren | 174 | 13.31% |
| Mechtilde Watts | 14 | 1.07% |
| Nicolle Bramble | 22 | 1.68% |
| Opaline Webster | 62 | 4.74% |
| Regena Wilder | 74 | 5.66% |
| Rhetta Parker | 70 | 5.36% |
| Sanjeev Walton | 34 | 2.60% |
| Savannah Morell | 79 | 6.04% |
| Shukriyya Burrows | 104 | 7.96% |
| Verda Heath | 119 | 9.10% |
| Vern Ferguson | 30 | 2.30% |

## Exercises lesson 2

(Screenshots and exercises are based on version 2018.3 demo data. If you working on an earlier or later version you may need to subtract or add 1 year to achieve similar results.)

**Task 1**

- Open the **Unit Price** analysis that was created and saved during Lesson 1.
- Apply the global criteria **Salesperson = Nicolle Bramble**. This will produce Math Errors in the crosstab.

Revenue and Units Sold per Product Hierarchy by Customer Territory

| | Total | | | Asia | | | Europe | | | North America | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **Product Group** | **Revenue** | **Units Sold** | **Unit Price** | **Revenue** | **Units Sold** | **Unit Price** | **Revenue** | **Units Sold** | **Unit Price** | **Revenue** | **Units Sold** | **Unit Price** |
| **Total** | **355,529** | **3,100** | **115** | **15,732** | **100** | **157** | **314,471** | **2,600** | **121** | **25,326** | **400** | **63** |
| JEANS + | **181,228** | **1,300** | **139** | | | Math error | 163,228 | 1,200 | 136 | 18,000 | 100 | 180 |
| SHIRTS + | **53,377** | **300** | **178** | | | Math error | 53,377 | 300 | 178 | | | Math error |
| T-SHIRTS + | **120,924** | **1,500** | **81** | 15,732 | 100 | 157 | 97,866 | 1,100 | 89 | 7,326 | 300 | 24 |

- Include **if-then-else** statements in the calculation syntax to avoid the Math Errors you would otherwise see.
- Save the **Unit Price** analysis with these changes.

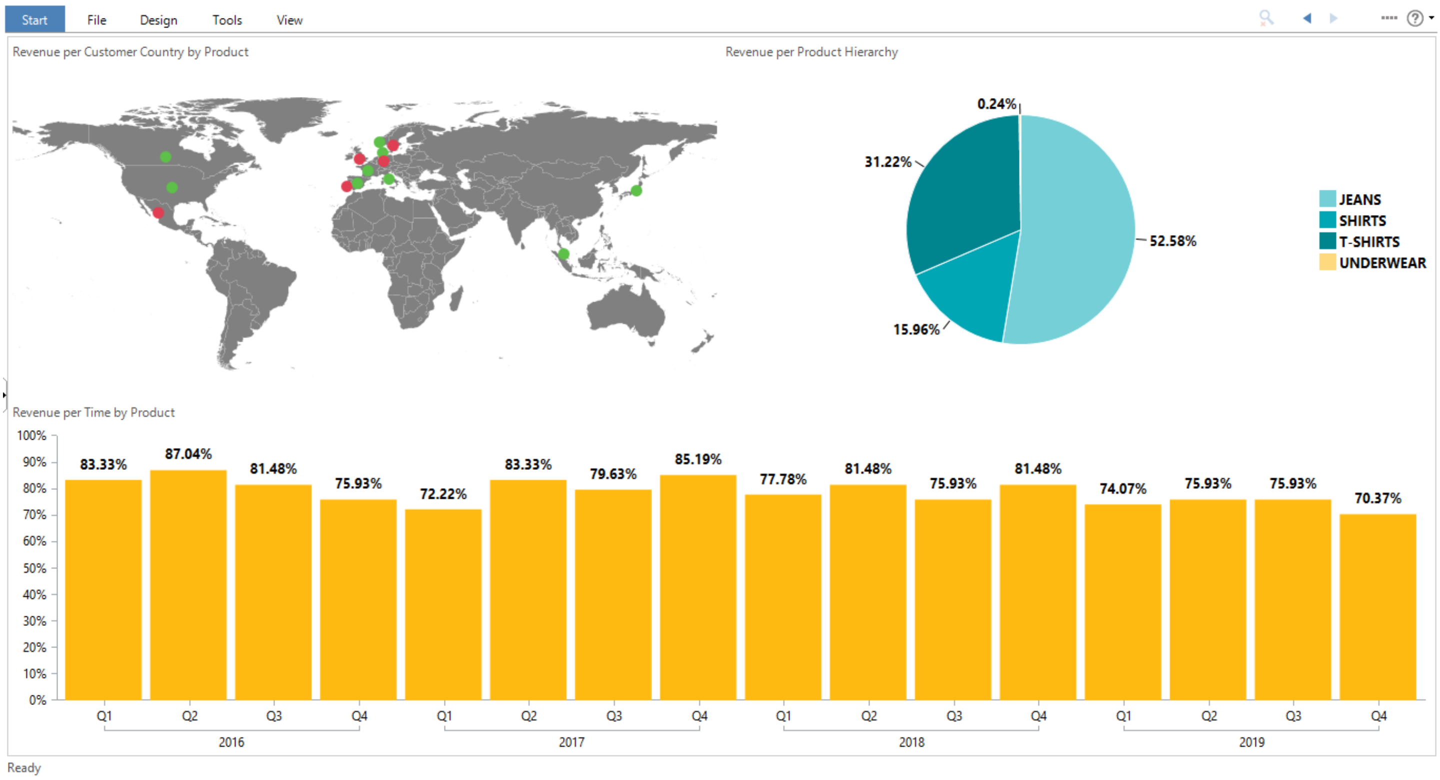
**Task 2**

- Create a new Analysis, **Product Suite Exploitation** (PSE), from which we will be able to tell how many products, out of the total number of products, that have been utilized in a given period or for a given country.
- A product is defined as being "exploited" whenever any revenue has been recorded for the product in a given period or for a given country. Positive, negative as well as 0 revenue counts as a recorded revenue.
- PSE may be expressed in percentage:
    - **PSE** = [number of products with a *non-NULL* revenue] / [*total* number of products].

- The Analysis must be based on **three objects**:
    - A **Pie chart** showing **Revenue** per **Product Hierarchy (Product Group)**.
    - A **Bar chart** showing **PSE** per **Time Hierarchy (Quarter)**.

      Tip: The basis for this object will be a cross table showing **Revenue** per **Time Hierarchy (Quarter)** by **Product (Product)**.
    - A **Map** (Map chart) showing **PSE per Country**.
        - Color Country **red** if **PSE < average PSE**.
        - Color Country **green** if **PSE >= average PSE**.

- The analysis should now look like this:




- Save the analysis as **Lesson 2 Product Suite Exploitation.**

# Lesson 3: Reference Modifiers – Visibility and Order

## General purpose

This lesson will teach the use and relevance of **Reference Modifiers**.

All aggregation functions are by default based on all the elements within their reference range.

For example **avg(d-1, all, m1)** will calculate the average of the measure m1 for **all** rows in the last column.

Now, if for example some of the rows had been hidden by a Visibility Agent, then a visibility modifier must be used to calculate the average of the **visible** rows only: **avg(d-1, all(visible), m1)**.

## Summation / Recap

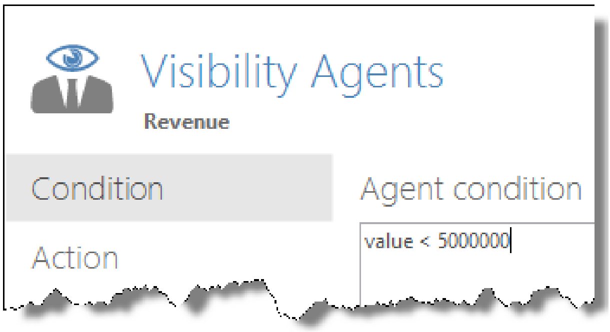
- A Reference Modifier is used to filter the elements within the reference range upon which the aggregation function must be based.
- There are three types of Reference Modifiers:
    - **Visibility modifiers** includes/excludes visible/hidden elements.
    - **Order modifiers** determines the order of elements when they are referenced – the current sorting order, as seen in the cross table, or the inherited sorting order, as sorted in the cube.
    - **Hierarchy modifiers** – to be focused on in the next lesson.

## Demo

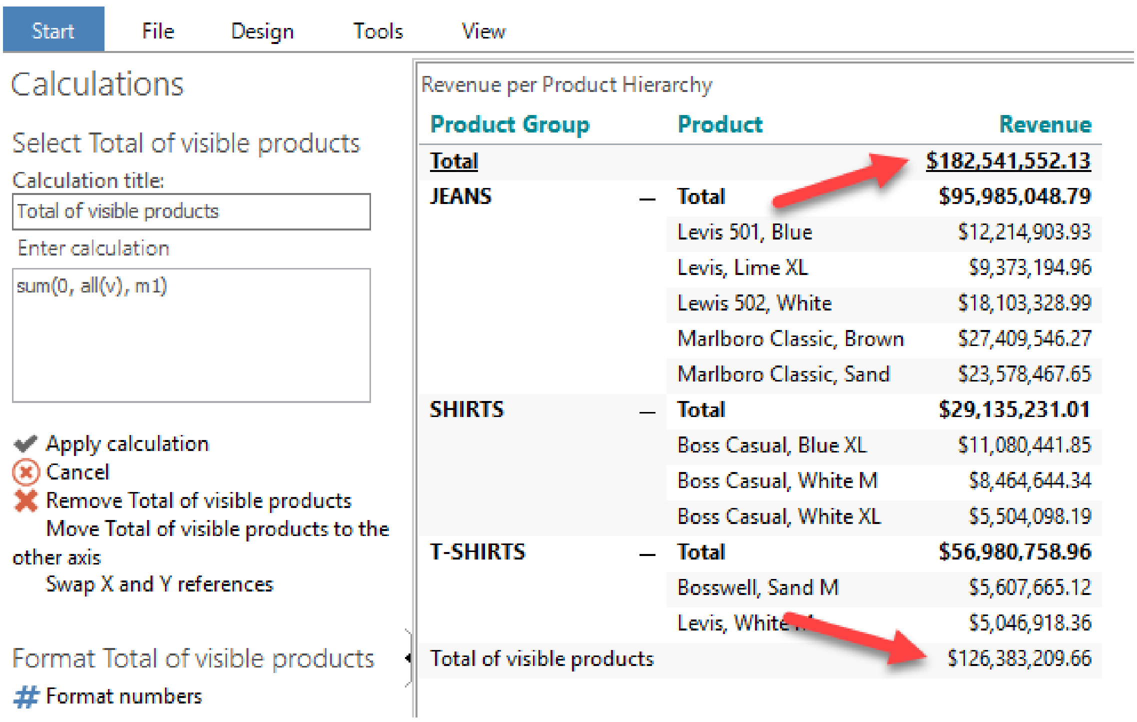
This demo will demonstrate examples of how to utilize the Visibility and Order Modifiers.

**Visibility Modifiers**

- Create a new Analysis with a Cross table showing **Revenue per Product Hierarchy(Product)**.
- Use a **Visibility Agent** to hide products with a **Revenue < 5.000.000**.



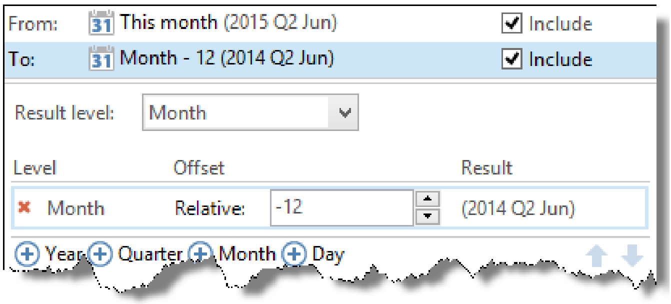

- Add a calculation: **Total of visible products**.



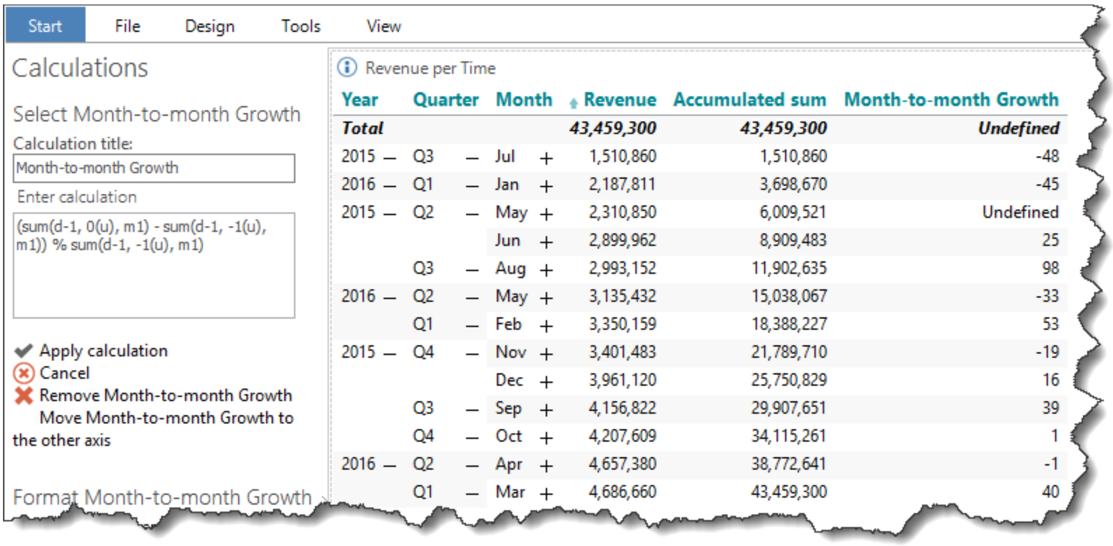
**Note:** The standard Grand total at the top of the table is the sum of visible *and* hidden rows, while the calculated total at the bottom of the table is the sum of the visible rows only.

**Order Modifiers**

- Create a new Analysis **Monthly Growth** with a Cross table showing **Revenue per Time Hierarchy (Month)**.
- Apply criteria **Time = Last 13 months**.



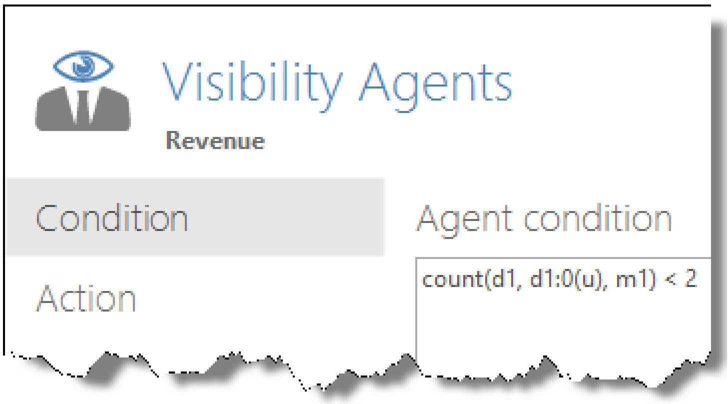

- Make an **ascending sort order** by right-clicking the Revenue column's header.
- From the Properties tab, Formatting, disable **Hierarchical collation**.
- Add two calculations:
    - **Accumulated sum, sorted**
        - sum(d-1, d1:0, m1)
    - **Month-to-month growth, unsorted**
        - (sum(d-1, 0(u), m1) – sum(d-1, -1(u), m1)) % sum(d-1, -1(u), m1)




| Year | Quarter | Month | Revenue | Accumulated sum | Month-to-month Growth |
|---|---|---|---|---|---|
| *Total* | | | *43,459,300* | *43,459,300* | *Undefined* |
| 2015 | Q3 | Jul | 1,510,860 | 1,510,860 | -48 |
| 2016 | Q1 | Jan | 2,187,811 | 3,698,670 | -45 |
| 2015 | Q2 | May | 2,310,850 | 6,009,521 | Undefined |
| | | Jun | 2,899,962 | 8,909,483 | 25 |
| | Q3 | Aug | 2,993,152 | 11,902,635 | 98 |
| 2016 | Q2 | May | 3,135,432 | 15,038,067 | -33 |
| | Q1 | Feb | 3,350,159 | 18,388,227 | 53 |
| 2015 | Q4 | Nov | 3,401,483 | 21,789,710 | -19 |
| | | Dec | 3,961,120 | 25,750,829 | 16 |
| | Q3 | Sep | 4,156,822 | 29,907,651 | 39 |
| | Q4 | Oct | 4,207,609 | 34,115,261 | 1 |
| 2016 | Q2 | Apr | 4,657,380 | 38,772,641 | -1 |
| | Q1 | Mar | 4,686,660 | 43,459,300 | 40 |

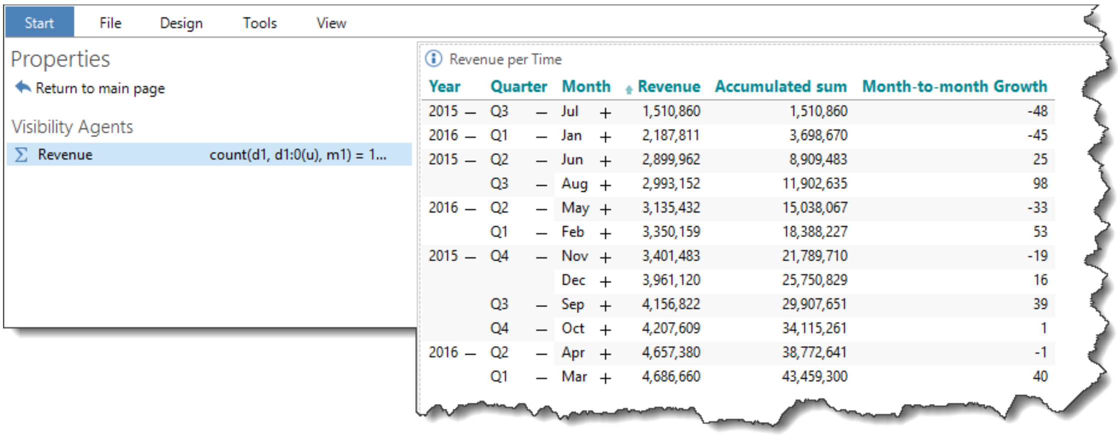
**Notice**: The monthly growth calculation should work **according to the original sorting order** to make sence – that's why the **u-parameter** is necessary.
The **earliest month** (which is only included in order to calculate growth on the second earliest month) is still visible in the analysis.

Add a **visibility agent** to hide the earliest month (again, according to the original sorting order).




The visibility agent counts the rows (unsorted) and counting the first row the condition is met and the row is hidden. After the first row the count will be > 1 and the condition is not met.

The final analysis should look like this:




| Year | Quarter | Month | Revenue | Accumulated sum | Month-to-month Growth |
|---|---|---|---|---|---|
| 2015 | Q3 | Jul | 1,510,860 | 1,510,860 | -48 |
| 2016 | Q1 | Jan | 2,187,811 | 3,698,670 | -45 |
| 2015 | Q2 | Jun | 2,899,962 | 8,909,483 | 25 |
| | Q3 | Aug | 2,993,152 | 11,902,635 | 98 |
| 2016 | Q2 | May | 3,135,432 | 15,038,067 | -33 |
| | Q1 | Feb | 3,350,159 | 18,388,227 | 53 |
| 2015 | Q4 | Nov | 3,401,483 | 21,789,710 | -19 |
| | | Dec | 3,961,120 | 25,750,829 | 16 |
| | Q3 | Sep | 4,156,822 | 29,907,651 | 39 |
| | Q4 | Oct | 4,207,609 | 34,115,261 | 1 |
| 2016 | Q2 | Apr | 4,657,380 | 38,772,641 | -1 |
| | Q1 | Mar | 4,686,660 | 43,459,300 | 40 |

Notice that it is possible to sort the last calculation, while the accumulated sum cannot be sorted – and the reason for this is the use of the u-parameter on the last calculation.

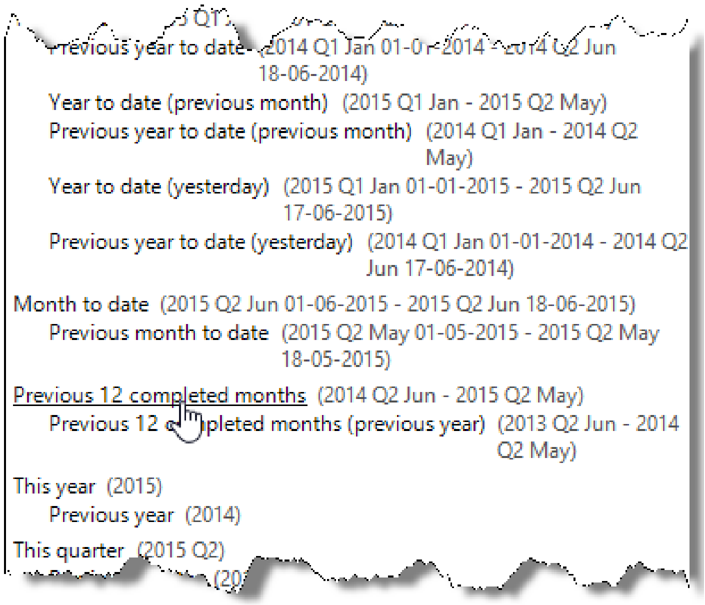
## Exercises lesson 3

(Screenshots and exercises are based on version 2018.3 demo data. If you working on an earlier or later version you may need to subtract or add 1 year to achieve similar results.)

### Task 1

In this exercise you are challenged with creating a new Analysis, **Advanced Profit**, to analyze 12 months data for the **Profit** measure.

- Add a global criterion (using Dynamic Time) that will limit our data to the previous 12 months (latest 12 concluded months):




- Add a Horizontal Bar Chart **Profit, 12 months Index**:
  - Calculate a **12 months index**, that calculates each month as a percentage of the **first** month in the selected period.

    Tip: Include the Order Modifier in the calculation, otherwise it will not be possible to sort it.

  - Make a **descending sort order** based on the calculation 12 mth index.

    Tip: Right-click the calculation's column header in order to sort it.

  - Apply **labels.**

- Add a Vertical Bar Chart **Profit, Top Customers of Total**:

- Top Customers (TC) are defined as customers having a total profit **greater than or equal to 50.000** for the selected 12 months period.

  Tip: The basic cross table should be defined with the Time Hierarchy (Month) dimension on the vertical axis, and the Customer dimension on the horizontal axis.

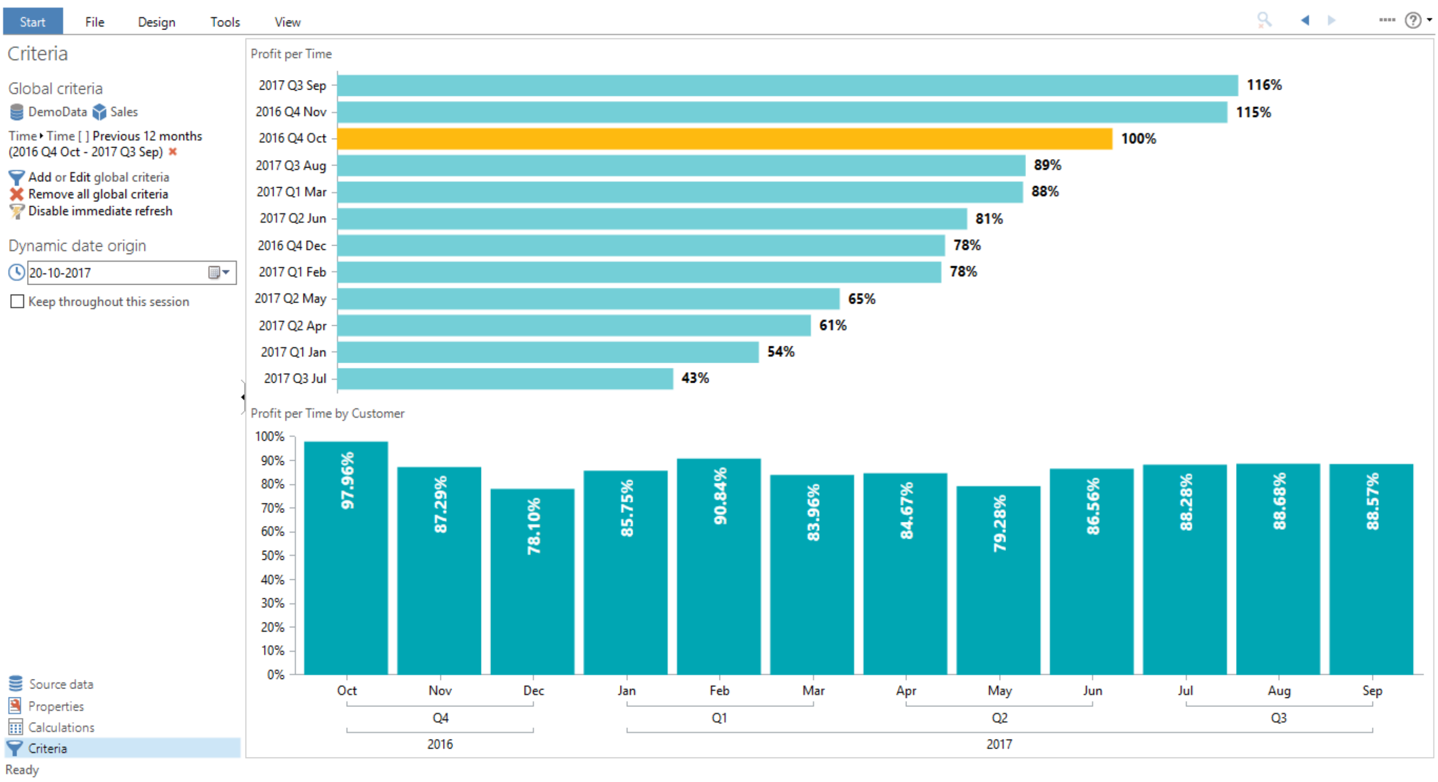
- Calculate a Top Customer percentage, **TC%**, that is the profit for Top Customers expressed as a percentage of the total profit for all customers.

- Tip: Customers contributing with *less* than 50.000 must be hidden. TC% is in other words the profit for *visible* customers expressed as a percentage of the *total* profit (profit for visible *and* hidden customers).

- Display TC% in the bar chart with **labels** applied.

- Let us simulate that we are currently in **20 October 2017**. This is done via the *Specify dynamic date origin* option in the Criteria Smartpad.

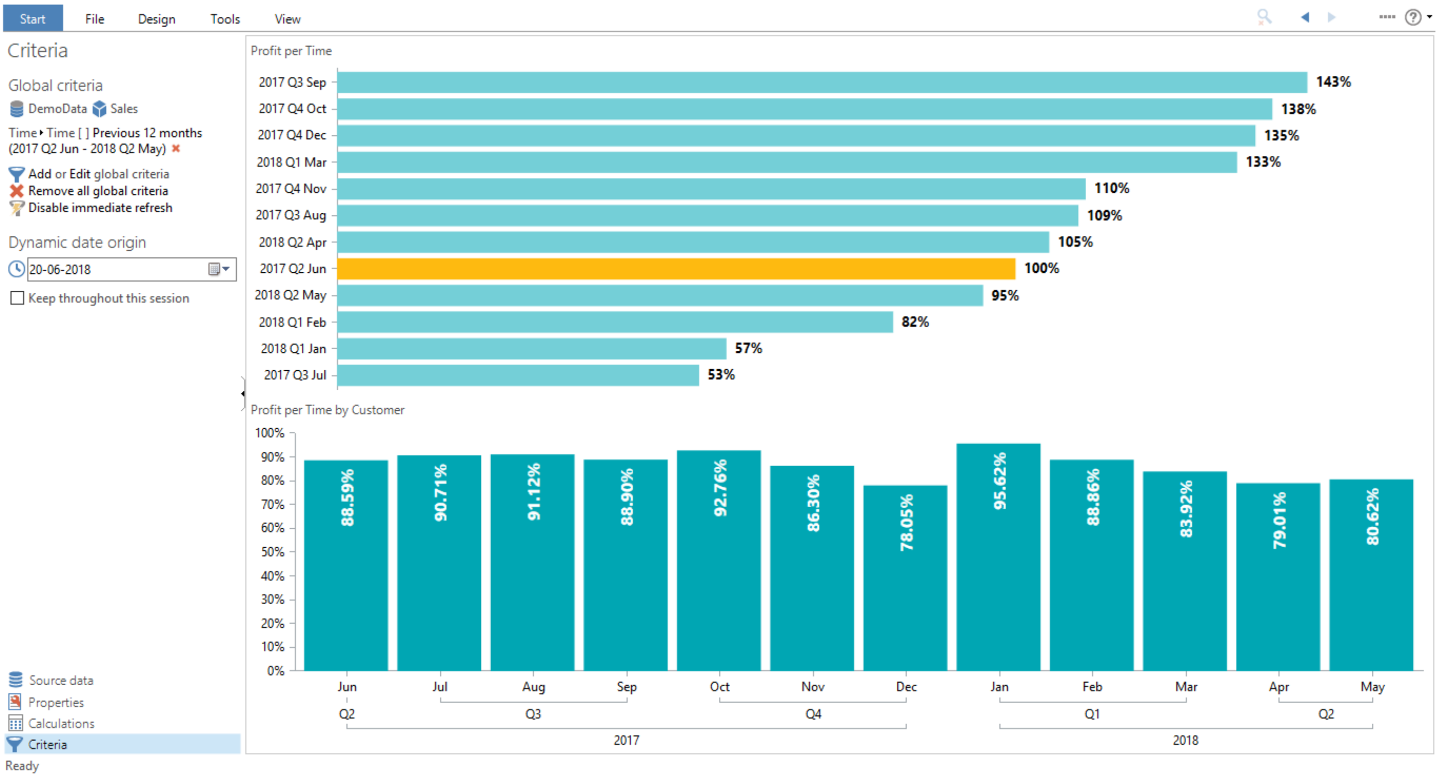
The result should now look like this:




Save the analysis as **Lesson 3 Advanced Profit analysis**.

How do you apply a different color to the first month, the 100% bar, in the horizontal bar chart?

Change the dynamic date origin to **20 June 2018** to get this result:



# Lesson 4: Reference Modifiers – Hierarchy

## General purpose

This lesson will teach the use and relevance of **Hierarchy Reference Modifiers**.

Hierarchy Modifiers will enable the user to refer to certain levels of data in a hierarchical dimension.

By default any calculation will be applied to its current level, which in most cases also makes sense: E.g. monthly averages are calculated on the Month level of the Time dimensions, quarterly averages are calculated on the Quarter level etc.

But in some instances it may be necessary to refer to a different level, e.g. to calculate the monthly revenue as a percentage of the yearly revenue.

## Summation / Recap

- A Reference Modifier is used to filter the elements within the reference range upon which the aggregation function must be based.
- **Hierarchy modifiers** are used to force references to specific levels in a hierarchical dimension.

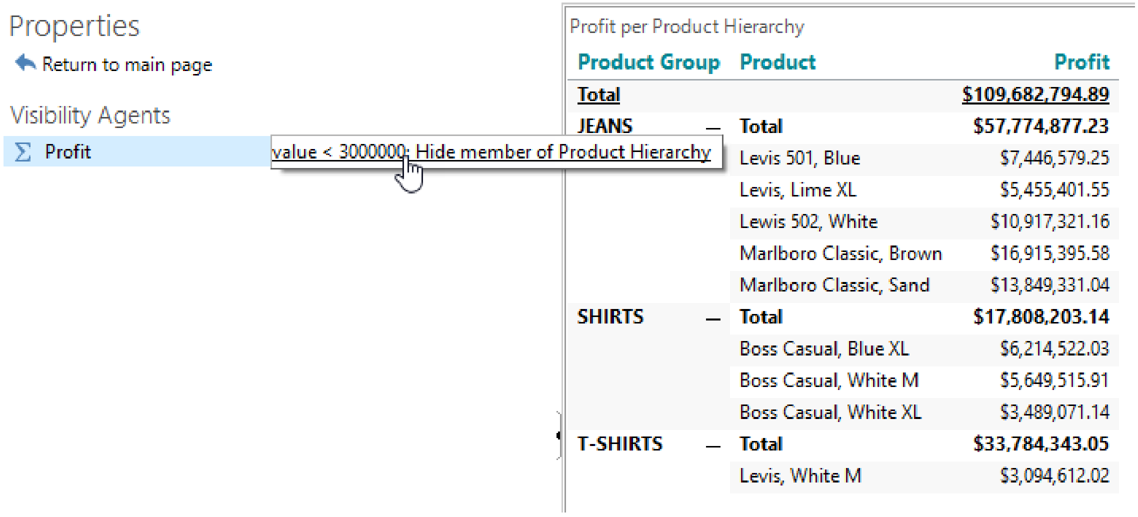
## Demo

This demo will demonstrate different examples of how to utilize the Hierarchy Modifiers.

### Child Modifier

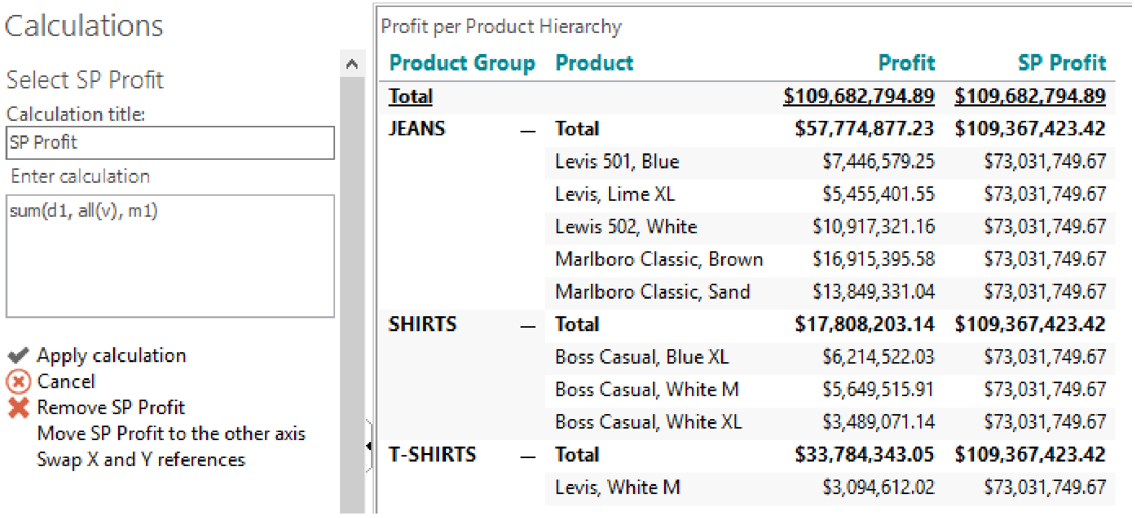
- Create a new Analysis with a Cross table showing **Profit per Product Hierarchy(Product)**.

- Use a **Visibility Agent** to hide all products where **Profit < 3.000.000**. So now we are only seeing the Significant Products (SP).




**Note:** The subtotals are still based on the sums for **all** products (visible **and** hidden) for each product group.

- Add a new calculation for the Significant Products, **SP Profit = sum(d1, all(v), m1)**, as a calculation for each product.



Calculations

Select SP Profit

Calculation title:

SP Profit

Enter calculation

sum(d1, all(v), m1)

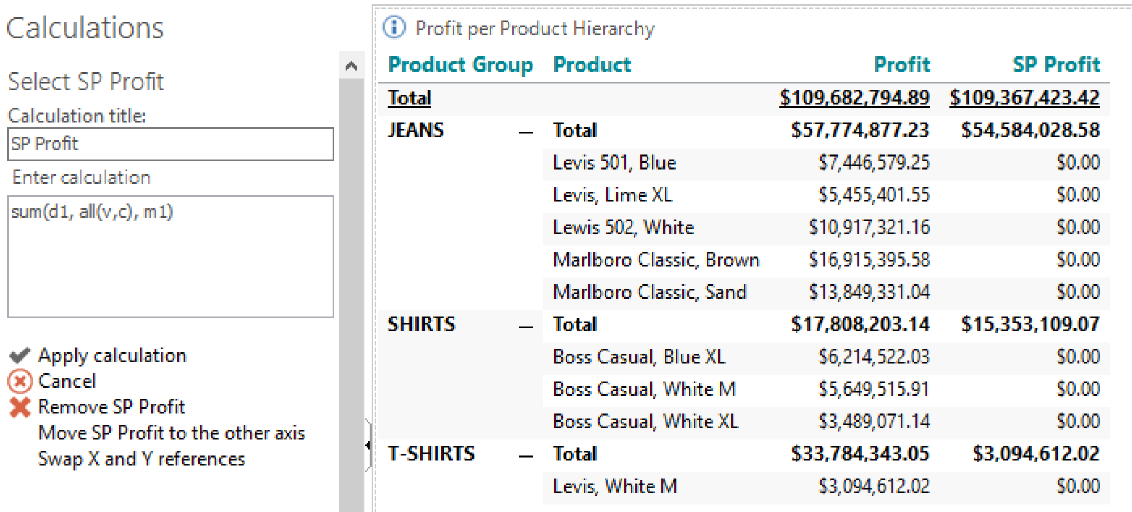
Apply calculation
Cancel
Remove SP Profit
Move SP Profit to the other axis
Swap X and Y references

Profit per Product Hierarchy

| Product Group | Product | Profit | SP Profit |
|---|---|---|---|
| Total | | $109,682,794.89 | $109,682,794.89 |
| JEANS – | Total | $57,774,877.23 | $109,367,423.42 |
| | Levis 501, Blue | $7,446,579.25 | $73,031,749.67 |
| | Levis, Lime XL | $5,455,401.55 | $73,031,749.67 |
| | Lewis 502, White | $10,917,321.16 | $73,031,749.67 |
| | Marlboro Classic, Brown | $16,915,395.58 | $73,031,749.67 |
| | Marlboro Classic, Sand | $13,849,331.04 | $73,031,749.67 |
| SHIRTS – | Total | $17,808,203.14 | $109,367,423.42 |
| | Boss Casual, Blue XL | $6,214,522.03 | $73,031,749.67 |
| | Boss Casual, White M | $5,649,515.91 | $73,031,749.67 |
| | Boss Casual, White XL | $3,489,071.14 | $73,031,749.67 |
| T-SHIRTS – | Total | $33,784,343.05 | $109,367,423.42 |
| | Levis, White M | $3,094,612.02 | $73,031,749.67 |

**Note:** The result for each product is the total Profit for all visible products (in the first column). The subtotal result for each Product group is the total Profit for all visible Product groups (in the first column).

- Add a Children modifier to the formula, **SP Profit = sum(d1, all(v,c), m1)**.



Calculations

Select SP Profit

Calculation title:

SP Profit

Enter calculation

sum(d1, all(v,c), m1)

Apply calculation
Cancel
Remove SP Profit
Move SP Profit to the other axis
Swap X and Y references

Profit per Product Hierarchy

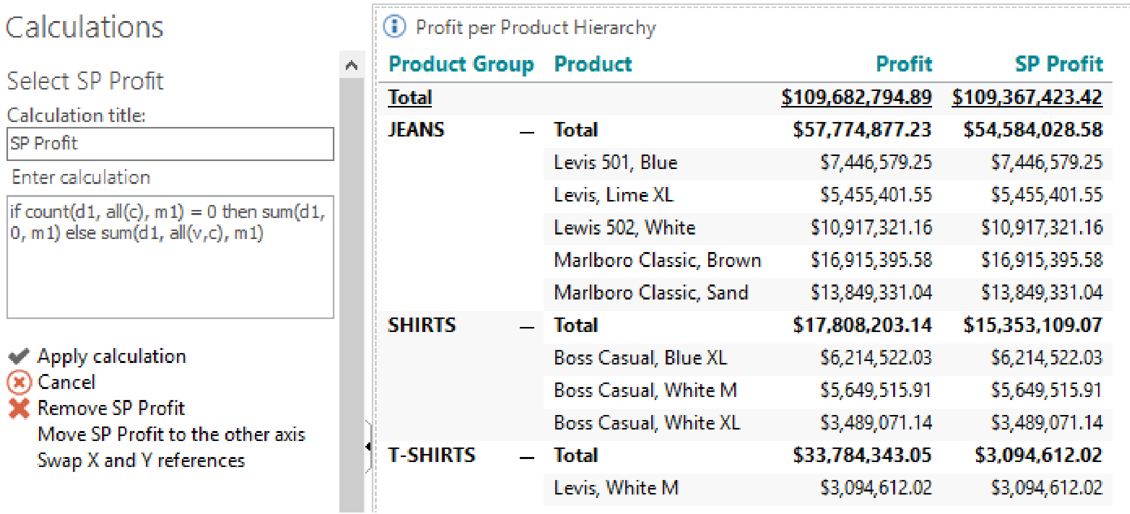
| Product Group | Product | Profit | SP Profit |
|---|---|---|---|
| Total | | $109,682,794.89 | $109,367,423.42 |
| JEANS – | Total | $57,774,877.23 | $54,584,028.58 |
| | Levis 501, Blue | $7,446,579.25 | $0.00 |
| | Levis, Lime XL | $5,455,401.55 | $0.00 |
| | Lewis 502, White | $10,917,321.16 | $0.00 |
| | Marlboro Classic, Brown | $16,915,395.58 | $0.00 |
| | Marlboro Classic, Sand | $13,849,331.04 | $0.00 |
| SHIRTS – | Total | $17,808,203.14 | $15,353,109.07 |
| | Boss Casual, Blue XL | $6,214,522.03 | $0.00 |
| | Boss Casual, White M | $5,649,515.91 | $0.00 |
| | Boss Casual, White XL | $3,489,071.14 | $0.00 |
| T-SHIRTS – | Total | $33,784,343.05 | $3,094,612.02 |
| | Levis, White M | $3,094,612.02 | $0.00 |

**Note:** The result for each product is now zero because the products have no children. The subtotal result for each group is now the total Profit for all visible Product group children, i.e. all visible products.

- Finally use an if-then-else statement to produce a satisfying result:

**if count(d1, all(c), m1) = 0 then sum(d1, 0, m1) else sum(d1, all(v,c), m1)**.

Or, in plain text: *If the current element has no children then just copy the element's value, otherwise calculate the sum of all its visible children.*

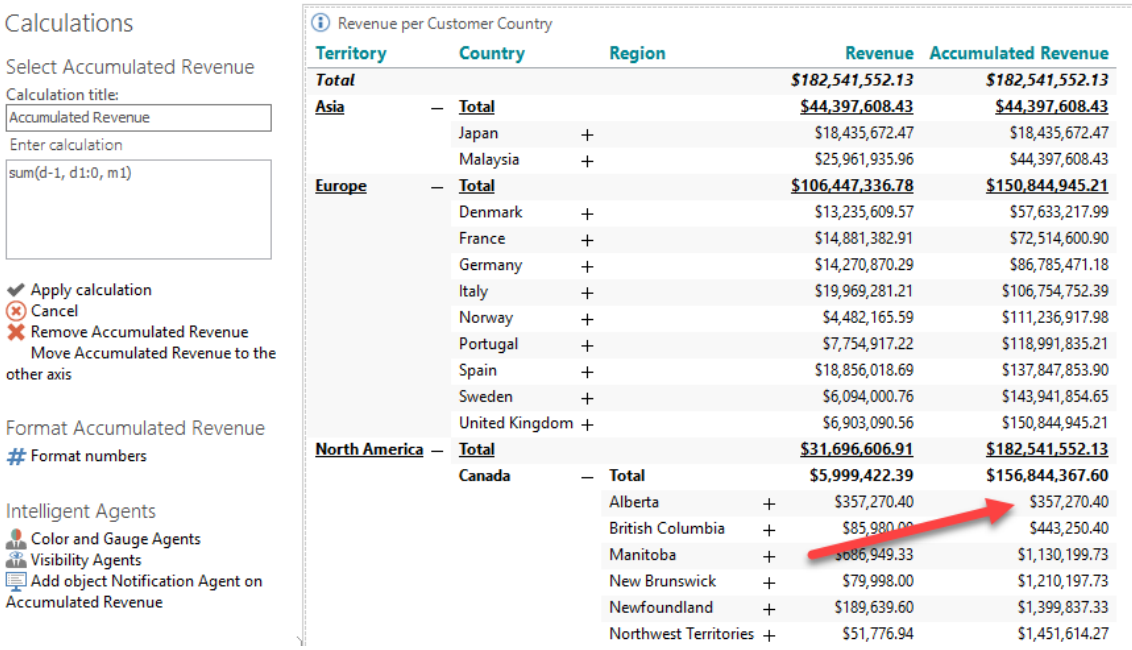



| Product Group | | Product | Profit | SP Profit |
|---|---|---|---|---|
| Total | | | $109,682,794.89 | $109,367,423.42 |
| JEANS | – | Total | $57,774,877.23 | $54,584,028.58 |
| | | Levis 501, Blue | $7,446,579.25 | $7,446,579.25 |
| | | Levis, Lime XL | $5,455,401.55 | $5,455,401.55 |
| | | Lewis 502, White | $10,917,321.16 | $10,917,321.16 |
| | | Marlboro Classic, Brown | $16,915,395.58 | $16,915,395.58 |
| | | Marlboro Classic, Sand | $13,849,331.04 | $13,849,331.04 |
| SHIRTS | – | Total | $17,808,203.14 | $15,353,109.07 |
| | | Boss Casual, Blue XL | $6,214,522.03 | $6,214,522.03 |
| | | Boss Casual, White M | $5,649,515.91 | $5,649,515.91 |
| | | Boss Casual, White XL | $3,489,071.14 | $3,489,071.14 |
| T-SHIRTS | – | Total | $33,784,343.05 | $3,094,612.02 |
| | | Levis, White M | $3,094,612.02 | $3,094,612.02 |

**Ragged Hierarchy Modifier**

A 'ragged' hierarchy is a hierarchy where different branches of the hierarchy have different numbers of levels.

- Create another Cross table, **Revenue per Customer Country(Country)**
- Add a calculation, **Accumulated sum = sum(d-1, d1:0, m1)**.
- Expand Canada, to simulate a **ragged hierarchy**.



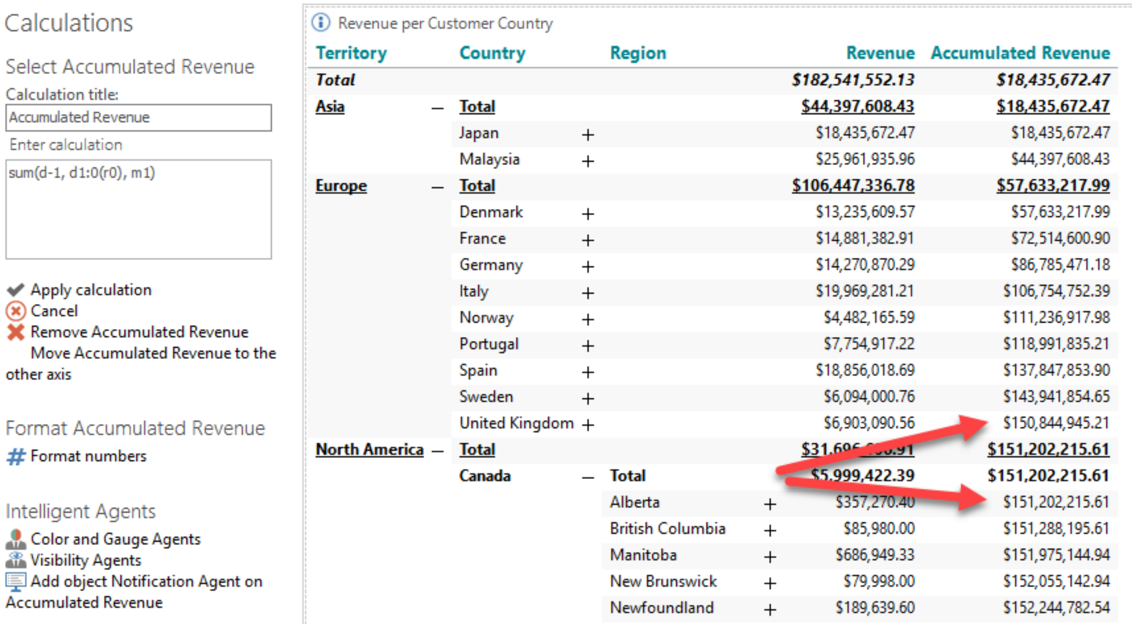

| Territory | Country | Region | Revenue | Accumulated Revenue |
|---|---|---|---|---|
| *Total* | | | *$182,541,552.13* | *$182,541,552.13* |
| Asia – | Total | | $44,397,608.43 | $44,397,608.43 |
| | Japan + | | $18,435,672.47 | $18,435,672.47 |
| | Malaysia + | | $25,961,935.96 | $44,397,608.43 |
| Europe – | Total | | $106,447,336.78 | $150,844,945.21 |
| | Denmark + | | $13,235,609.57 | $57,633,217.99 |
| | France + | | $14,881,382.91 | $72,514,600.90 |
| | Germany + | | $14,270,870.29 | $86,785,471.18 |
| | Italy + | | $19,969,281.21 | $106,754,752.39 |
| | Norway + | | $4,482,165.59 | $111,236,917.98 |
| | Portugal + | | $7,754,917.22 | $118,991,835.21 |
| | Spain + | | $18,856,018.69 | $137,847,853.90 |
| | Sweden + | | $6,094,000.76 | $143,941,854.65 |
| | United Kingdom + | | $6,903,090.56 | $150,844,945.21 |
| North America – | Total | | $31,696,606.91 | $182,541,552.13 |
| | Canada – | Total | $5,999,422.39 | $156,844,367.60 |
| | | Alberta + | $357,270.40 | $357,270.40 |
| | | British Columbia + | $85,980.00 | $443,250.40 |
| | | Manitoba + | $686,949.33 | $1,130,199.73 |
| | | New Brunswick + | $79,998.00 | $1,210,197.73 |
| | | Newfoundland + | $189,639.60 | $1,399,837.33 |
| | | Northwest Territories + | $51,776.94 | $1,451,614.27 |

**Note:** The accumulation "starts over" in the expanded part of the hierarchy. This is because the default Hierarchy Modifier (= no Hierarchy Modifier) is set to include elements on the same level as the **current** element.

- Add the 'ragged' modifier **r0** to the calculation:

**sum(d-1, d1:0(r0), m1)**.

- This will force the calculation to include elements at the lowest level in the hierarchy according to the current expansion of the table.




| Territory | Country | Region | Revenue | Accumulated Revenue |
|---|---|---|---|---|
| *Total* | | | *$182,541,552.13* | *$18,435,672.47* |
| Asia – | Total | | $44,397,608.43 | $18,435,672.47 |
| | Japan + | | $18,435,672.47 | $18,435,672.47 |
| | Malaysia + | | $25,961,935.96 | $44,397,608.43 |
| Europe – | Total | | $106,447,336.78 | $57,633,217.99 |
| | Denmark + | | $13,235,609.57 | $57,633,217.99 |
| | France + | | $14,881,382.91 | $72,514,600.90 |
| | Germany + | | $14,270,870.29 | $86,785,471.18 |
| | Italy + | | $19,969,281.21 | $106,754,752.39 |
| | Norway + | | $4,482,165.59 | $111,236,917.98 |
| | Portugal + | | $7,754,917.22 | $118,991,835.21 |
| | Spain + | | $18,856,018.69 | $137,847,853.90 |
| | Sweden + | | $6,094,000.76 | $143,941,854.65 |
| | United Kingdom + | | $6,903,090.56 | $150,844,945.21 |
| North America – | Total | | $31,69[illegible].91 | $151,202,215.61 |
| | Canada – | Total | $5,999,422.39 | $151,202,215.61 |
| | | Alberta + | $357,270.40 | $151,202,215.61 |
| | | British Columbia + | $85,980.00 | $151,288,195.61 |
| | | Manitoba + | $686,949.33 | $151,975,144.94 |
| | | New Brunswick + | $79,998.00 | $152,055,142.94 |
| | | Newfoundland + | $189,639.60 | $152,244,782.54 |

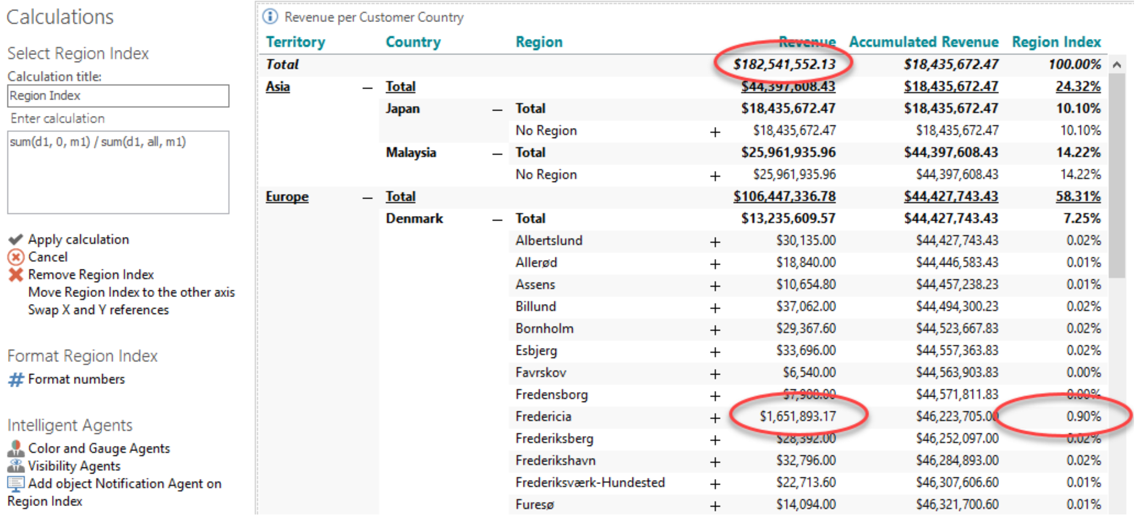
**Note:** The accumulation now continues despite the uneven hierarchy levels.

### Siblings Modifier

- **Expand all Countries** (e.g. by use of the '+' in the upper right corner of the object).

- Add another calculation, **Index per Region:**

  **sum(d1, 0, m1) / sum(d1, all, m1)**. Format as Percent.

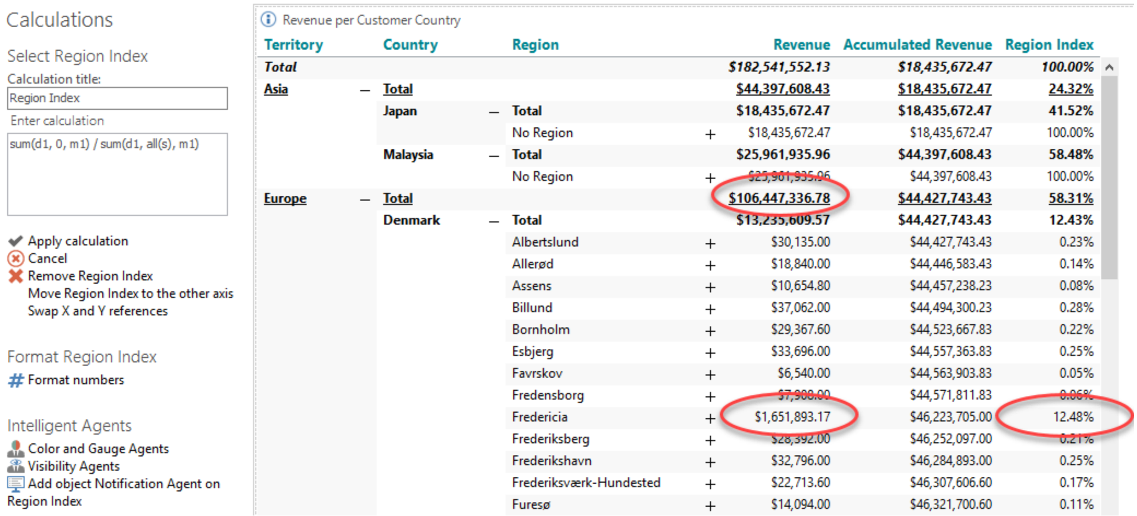



| Territory | Country | Region | Revenue | Accumulated Revenue | Region Index |
|---|---|---|---|---|---|
| Total | | | $182,541,552.13 | $18,435,672.47 | 100.00% |
| Asia | Total | | $44,397,608.43 | $18,435,672.47 | 24.32% |
| | Japan | Total | $18,435,672.47 | $18,435,672.47 | 10.10% |
| | | No Region | $18,435,672.47 | $18,435,672.47 | 10.10% |
| | Malaysia | Total | $25,961,935.96 | $44,397,608.43 | 14.22% |
| | | No Region | $25,961,935.96 | $44,397,608.43 | 14.22% |
| Europe | Total | | $106,447,336.78 | $44,427,743.43 | 58.31% |
| | Denmark | Total | $13,235,609.57 | $44,427,743.43 | 7.25% |
| | | Albertslund | $30,135.00 | $44,427,743.43 | 0.02% |
| | | Allerød | $18,840.00 | $44,446,583.43 | 0.01% |
| | | Assens | $10,654.80 | $44,457,238.23 | 0.01% |
| | | Billund | $37,062.00 | $44,494,300.23 | 0.02% |
| | | Bornholm | $29,367.60 | $44,523,667.83 | 0.02% |
| | | Esbjerg | $33,696.00 | $44,557,363.83 | 0.02% |
| | | Favrskov | $6,540.00 | $44,563,903.83 | 0.00% |
| | | Fredensborg | $7,908.00 | $44,571,811.83 | 0.00% |
| | | Fredericia | $1,651,893.17 | $46,223,705.00 | 0.90% |
| | | Frederiksberg | $28,392.00 | $46,252,097.00 | 0.02% |
| | | Frederikshavn | $32,796.00 | $46,284,893.00 | 0.02% |
| | | Frederiksværk-Hundested | $22,713.60 | $46,307,606.60 | 0.01% |
| | | Furesø | $14,094.00 | $46,321,700.60 | 0.01% |

> **Note:** When no Hierarchy Modifier has been applied, the profit of each Region will be calculated as a percentage of the equivalent of the grand total.

- Add the "siblings" modifier to the calculation:

  **sum(d-1, 0, m1) % sum(d-1, all(s), m1)**.



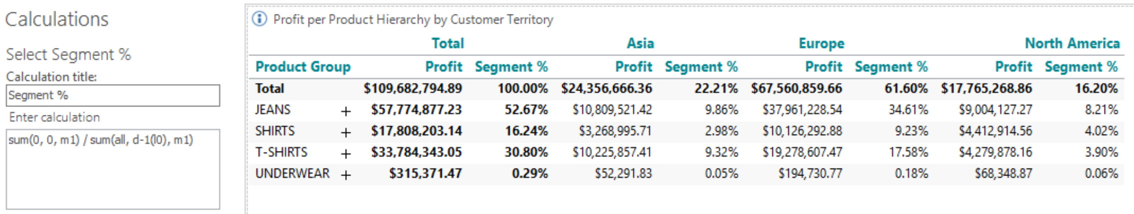

| Territory | Country | Region | Revenue | Accumulated Revenue | Region Index |
|---|---|---|---|---|---|
| Total | | | $182,541,552.13 | $18,435,672.47 | 100.00% |
| Asia | Total | | $44,397,608.43 | $18,435,672.47 | 24.32% |
| | Japan | Total | $18,435,672.47 | $18,435,672.47 | 41.52% |
| | | No Region | $18,435,672.47 | $18,435,672.47 | 100.00% |
| | Malaysia | Total | $25,961,935.96 | $44,397,608.43 | 58.48% |
| | | No Region | $25,961,935.96 | $44,397,608.43 | 100.00% |
| Europe | Total | | $106,447,336.78 | $44,427,743.43 | 58.31% |
| | Denmark | Total | $13,235,609.57 | $44,427,743.43 | 12.43% |
| | | Albertslund | $30,135.00 | $44,427,743.43 | 0.23% |
| | | Allerød | $18,840.00 | $44,446,583.43 | 0.14% |
| | | Assens | $10,654.80 | $44,457,238.23 | 0.08% |
| | | Billund | $37,062.00 | $44,494,300.23 | 0.28% |
| | | Bornholm | $29,367.60 | $44,523,667.83 | 0.22% |
| | | Esbjerg | $33,696.00 | $44,557,363.83 | 0.25% |
| | | Favrskov | $6,540.00 | $44,563,903.83 | 0.05% |
| | | Fredensborg | $7,908.00 | $44,571,811.83 | 0.06% |
| | | Fredericia | $1,651,893.17 | $46,223,705.00 | 12.48% |
| | | Frederiksberg | $28,392.00 | $46,252,097.00 | 0.21% |
| | | Frederikshavn | $32,796.00 | $46,284,893.00 | 0.25% |
| | | Frederiksværk-Hundested | $22,713.60 | $46,307,606.60 | 0.17% |
| | | Furesø | $14,094.00 | $46,321,700.60 | 0.11% |

> **Note:** By adding the "siblings" modifier, the profit of each Region will now be calculated as a percentage of the equivalent of the subtotals for each Country.

**Level Modifier**

- Create a cross table **Profit per Product Hierarchy by Customer Country**.
- Add a new calculated measure **Segment %** that calculates each Product/Customer Country combination as a percentage of the grand total profit.
- Instead of referring to all columns or all rows on a certain level it will often be relevant to refer to a single column or row at the 'All' level:

**sum(0, 0, m1) / sum(all, all, m1) -> sum(0, 0, m1) / sum(all, d-1(l0), m1)**




Profit per Product Hierarchy by Customer Territory

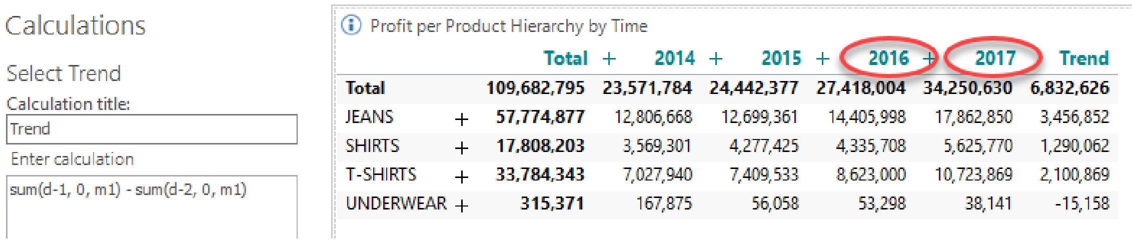
| | | Total | | Asia | | Europe | | North America | |
|---|---|---|---|---|---|---|---|---|---|
| Product Group | | Profit | Segment % | Profit | Segment % | Profit | Segment % | Profit | Segment % |
| Total | | $109,682,794.89 | 100.00% | $24,356,666.36 | 22.21% | $67,560,859.66 | 61.60% | $17,765,268.86 | 16.20% |
| JEANS | + | $57,774,877.23 | 52.67% | $10,809,521.42 | 9.86% | $37,961,228.54 | 34.61% | $9,004,127.27 | 8.21% |
| SHIRTS | + | $17,808,203.14 | 16.24% | $3,268,995.71 | 2.98% | $10,126,292.88 | 9.23% | $4,412,914.56 | 4.02% |
| T-SHIRTS | + | $33,784,343.05 | 30.80% | $10,225,857.41 | 9.32% | $19,278,607.47 | 17.58% | $4,279,878.16 | 3.90% |
| UNDERWEAR | + | $315,371.47 | 0.29% | $52,291.83 | 0.05% | $194,730.77 | 0.18% | $68,348.87 | 0.06% |

Another example of the level parameter being useful can be observed in the simple trend calculation, where the level parameter can make a calculation robust in case of hierarchy expansion.

- Add a crosstab – **Profit** per **Product Hierarchy (Product Group)** and by **Time Hierarchy (Year).**
- Calculate a difference between the last 2 columns:
  **Trend = sum(d-1, 0, m1) – sum(d-2, 0, m1)**

This calculation always **calculates the 2 last columns** – if the hierarchy is expanded the calculation will calculate the 2 last columns at the expanded level – as shown below:

Not expanded:




Profit per Product Hierarchy by Time

| | | Total + | 2014 + | 2015 + | 2016 + | 2017 | Trend |
|---|---|---|---|---|---|---|---|
| Total | | 109,682,795 | 23,571,784 | 24,442,377 | 27,418,004 | 34,250,630 | 6,832,626 |
| JEANS | + | 57,774,877 | 12,806,668 | 12,699,361 | 14,405,998 | 17,862,850 | 3,456,852 |
| SHIRTS | + | 17,808,203 | 3,569,301 | 4,277,425 | 4,335,708 | 5,625,770 | 1,290,062 |
| T-SHIRTS | + | 33,784,343 | 7,027,940 | 7,409,533 | 8,623,000 | 10,723,869 | 2,100,869 |
| UNDERWEAR | + | 315,371 | 167,875 | 56,058 | 53,298 | 38,141 | -15,158 |

Partially expanded (now the Trend calculation calculates the difference between Q4 and Q3 in 2015):

Profit per Product Hierarchy by Time

| | Total + | 2014 + | 2015 − | 2016 | | | | | + 2017 | Trend |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Total + | Q1 + | Q2 + | Q3 + | Q4 | | |
| **Total** | **109,682,795** | **23,571,784** | **24,442,377** | **27,418,004** | **5,468,189** | **6,497,142** | **7,088,793** | **8,363,880** | **34,250,630** | **1,275,088** |
| JEANS + | 57,774,877 | 12,806,668 | 12,699,361 | **14,405,998** | 3,197,781 | 3,212,369 | 3,687,897 | 4,307,951 | 17,862,850 | 620,054 |
| SHIRTS + | 17,808,203 | 3,569,301 | 4,277,425 | **4,335,708** | 890,224 | 1,084,790 | 1,079,182 | 1,281,512 | 5,625,770 | 202,330 |
| T-SHIRTS + | 33,784,343 | 7,027,940 | 7,409,533 | **8,623,000** | 1,359,807 | 2,175,582 | 2,325,694 | 2,761,918 | 10,723,869 | 436,224 |
| UNDERWEAR + | 315,371 | 167,875 | 56,058 | **53,298** | 20,377 | 24,401 | -3,979 | 12,500 | 38,141 | 16,479 |

This could be the intention – and in that case there is no problem, but if you want to make the calculation robust to expansion of the hierarchy you can use the level parameter.

**Trend = sum(d-1(l1), 0, m1) – sum(d-2(l1), 0, m1)**

Profit per Product Hierarchy by Time

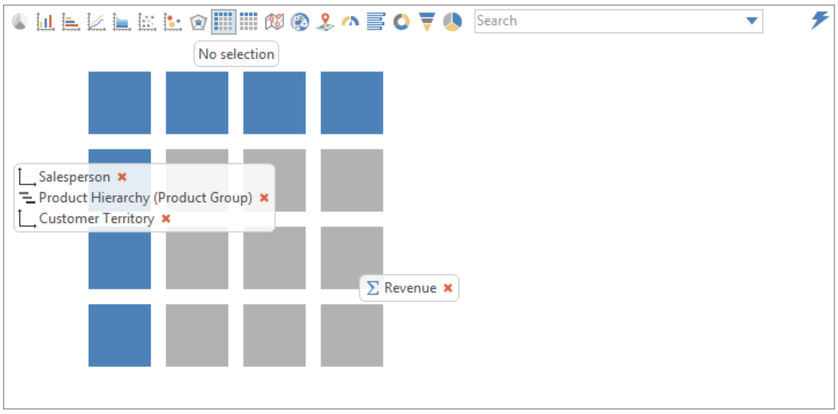
| | Total + | 2014 + | 2015 − | 2016 | | | | | + 2017 | Trend |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Total + | Q1 + | Q2 + | Q3 + | Q4 | | |
| **Total** | **109,682,795** | **23,571,784** | **24,442,377** | **27,418,004** | **5,468,189** | **6,497,142** | **7,088,793** | **8,363,880** | **34,250,630** | **6,832,626** |
| JEANS + | 57,774,877 | 12,806,668 | 12,699,361 | **14,405,998** | 3,197,781 | 3,212,369 | 3,687,897 | 4,307,951 | 17,862,850 | 3,456,852 |
| SHIRTS + | 17,808,203 | 3,569,301 | 4,277,425 | **4,335,708** | 890,224 | 1,084,790 | 1,079,182 | 1,281,512 | 5,625,770 | 1,290,062 |
| T-SHIRTS + | 33,784,343 | 7,027,940 | 7,409,533 | **8,623,000** | 1,359,807 | 2,175,582 | 2,325,694 | 2,761,918 | 10,723,869 | 2,100,869 |
| UNDERWEAR + | 315,371 | 167,875 | 56,058 | **53,298** | 20,377 | 24,401 | -3,979 | 12,500 | 38,141 | -15,158 |

As **level 1** in this case specifies the **year level** the calculation is now **robust to expansion**.

**Levels with multiple dimensions on the same axis**

- Add another crosstab looking at **Revenue per Salesperson, by Product Hierarchy(Product Group) and by Customer(Territory)**.

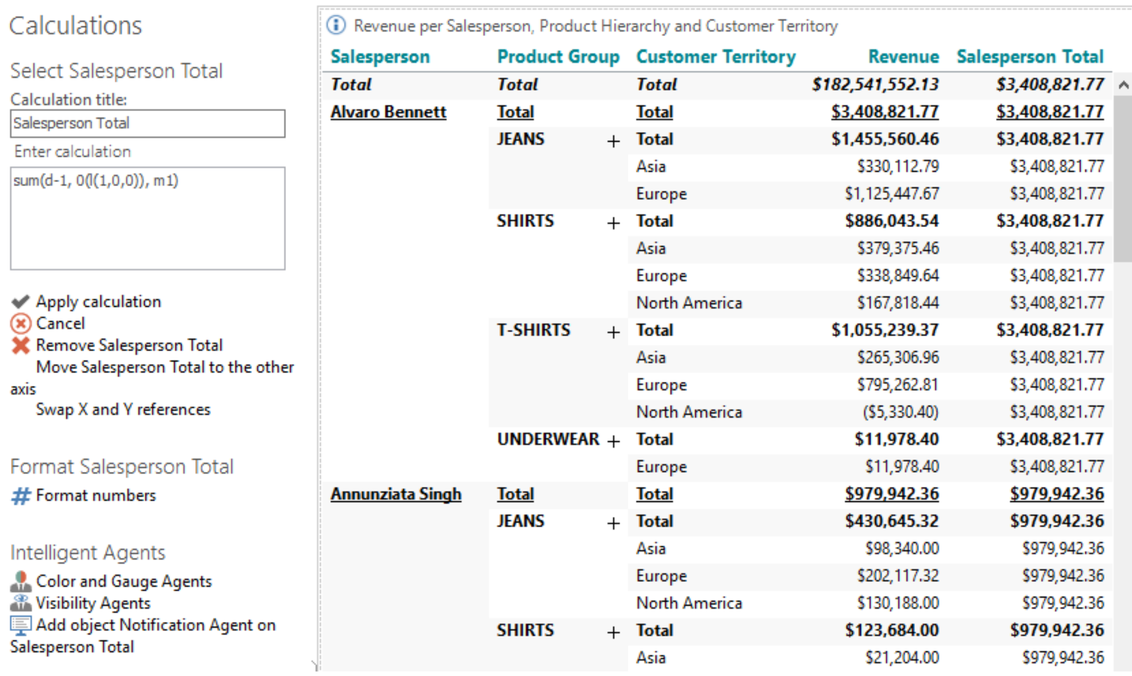
Now make sure that all 3 dimensions are on the same (vertical) axis as shown here:




This creates a special situation in relation to the level parameter. You can actually reference the level of each dimension in one sentence.

This example references the totals of the Salespersons:

**Salespersons totals: sum(d-1, 0(l(1, 0, 0)), m1)**



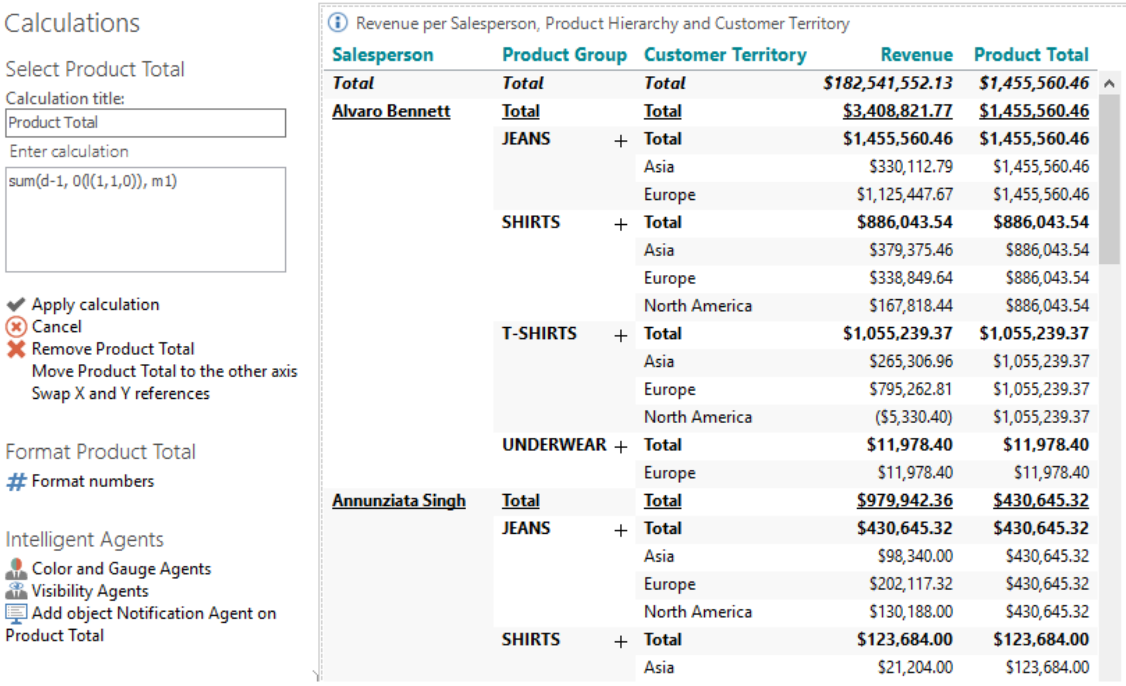

| Salesperson | Product Group | Customer Territory | Revenue | Salesperson Total |
|---|---|---|---|---|
| *Total* | *Total* | *Total* | *$182,541,552.13* | *$3,408,821.77* |
| Alvaro Bennett | Total | Total | $3,408,821.77 | $3,408,821.77 |
| | JEANS | Total | $1,455,560.46 | $3,408,821.77 |
| | | Asia | $330,112.79 | $3,408,821.77 |
| | | Europe | $1,125,447.67 | $3,408,821.77 |
| | SHIRTS | Total | $886,043.54 | $3,408,821.77 |
| | | Asia | $379,375.46 | $3,408,821.77 |
| | | Europe | $338,849.64 | $3,408,821.77 |
| | | North America | $167,818.44 | $3,408,821.77 |
| | T-SHIRTS | Total | $1,055,239.37 | $3,408,821.77 |
| | | Asia | $265,306.96 | $3,408,821.77 |
| | | Europe | $795,262.81 | $3,408,821.77 |
| | | North America | ($5,330.40) | $3,408,821.77 |
| | UNDERWEAR | Total | $11,978.40 | $3,408,821.77 |
| | | Europe | $11,978.40 | $3,408,821.77 |
| Annunziata Singh | Total | Total | $979,942.36 | $979,942.36 |
| | JEANS | Total | $430,645.32 | $979,942.36 |
| | | Asia | $98,340.00 | $979,942.36 |
| | | Europe | $202,117.32 | $979,942.36 |
| | | North America | $130,188.00 | $979,942.36 |
| | SHIRTS | Total | $123,684.00 | $979,942.36 |
| | | Asia | $21,204.00 | $979,942.36 |

**l(1, 0, 0)** means level 1 on the Salesperson dimension and level 0 on Product and Customer Country (level 0 being the "all" level).

So in short – totals **for Products and Customer Country but still within each Salesperson**.

To reference the Product subtotals the syntax would be:

**Product Subtotals = sum(d-1, 0(l(1, 1, 0)),m1)**




| Salesperson | Product Group | Customer Territory | Revenue | Product Total |
|---|---|---|---|---|
| *Total* | *Total* | *Total* | *$182,541,552.13* | *$1,455,560.46* |
| Alvaro Bennett | Total | Total | $3,408,821.77 | $1,455,560.46 |
| | JEANS + | Total | $1,455,560.46 | $1,455,560.46 |
| | | Asia | $330,112.79 | $1,455,560.46 |
| | | Europe | $1,125,447.67 | $1,455,560.46 |
| | SHIRTS + | Total | $886,043.54 | $886,043.54 |
| | | Asia | $379,375.46 | $886,043.54 |
| | | Europe | $338,849.64 | $886,043.54 |
| | | North America | $167,818.44 | $886,043.54 |
| | T-SHIRTS + | Total | $1,055,239.37 | $1,055,239.37 |
| | | Asia | $265,306.96 | $1,055,239.37 |
| | | Europe | $795,262.81 | $1,055,239.37 |
| | | North America | ($5,330.40) | $1,055,239.37 |
| | UNDERWEAR + | Total | $11,978.40 | $11,978.40 |
| | | Europe | $11,978.40 | $11,978.40 |
| Annunziata Singh | Total | Total | $979,942.36 | $430,645.32 |
| | JEANS + | Total | $430,645.32 | $430,645.32 |
| | | Asia | $98,340.00 | $430,645.32 |
| | | Europe | $202,117.32 | $430,645.32 |
| | | North America | $130,188.00 | $430,645.32 |
| | SHIRTS + | Total | $123,684.00 | $123,684.00 |
| | | Asia | $21,204.00 | $123,684.00 |

Totals on the Customer Country level – but within each Salesperson/Product Group.

## Exercises lesson 4

(Screenshots and exercises are based on version 2018.3 demo data. If you working on an earlier or later version you may need to subtract or add 1 year to achieve similar results.)

Create a new **Top and Bottom analysis** designed to show a list of the **3 best selling** and **3 worst selling Products within each Product Group.**

- The basis is a cross tab showing **Revenue** per **Product**.
- Add visibility agents to **hide Product Groups with less than 6 Products**.
- Add visibility agents to hide products **not** in the top/bottom 3 categories.

Tip: It might help to **sort** the list by Revenue.

Tip: The Min and Max functions are not relevant to solve this exercise. Make sure that **subtotals** and **grand totals** are **visible** and **correct** in the final crosstab.

Tip: It will probably be necessary to calculate **intermediate results** to get correct subtotals and totals.

- Add **icon agents** to highlight the **top 3** and **bottom 3** within each Product Group.

The final crosstab should look like this:

Revenue per Product Hierarchy

| Product Group | | Product | | Corrected Revenue |
|---|---|---|---|---|
| **JEANS** | — | **Total** | | **$74,396,949.90** |
| | | Marlboro Classic, Brown | ↑ | $27,409,546.27 |
| | | Marlboro Classic, Sand | ↑ | $23,578,467.65 |
| | | Lewis 502, White | ↑ | $18,103,328.99 |
| | | Levis, Blue XXL | ↓ | $1,846,966.84 |
| | | Levis 501, White | ↓ | $1,836,566.10 |
| | | Levis 501, Black | ↓ | $1,622,074.04 |
| **T-SHIRTS** | — | **Total** | | **$15,057,029.41** |
| | | Bosswell, Sand M | ↑ | $5,607,665.12 |
| | | Levis, White M | ↑ | $5,046,918.36 |
| | | Bosswell, White XXL | ↑ | $4,386,201.35 |
| | | Armani, White L | ↓ | $6,972.00 |
| | | Boss, Grey XXL | ↓ | $4,883.19 |
| | | Bosswell, Black M | ↓ | $4,389.40 |
| **SHIRTS** | — | **Total** | | **$25,829,207.16** |
| | | Boss Casual, Blue XL | ↑ | $11,080,441.85 |
| | | Boss Casual, White M | ↑ | $8,464,644.34 |
| | | Boss Casual, White XL | ↑ | $5,504,098.19 |
| | | Boss Casual, Sand S | ↓ | $382,815.47 |
| | | Boss Casual, Sand XL | ↓ | $333,521.40 |
| | | Boss Casual, Blue L | ↓ | $63,685.92 |

# Appendix

## Extra exercise

- Create a new Analysis, **Observation List**, that will be useful in pointing out low-selling, or otherwise questionable products within a given period of time.

- The Analysis must be able to analyze data in any range of periods, e.g. Quarterly or Monthly periods across one or multiple years.

- Products apply to the Observation List when any of these conditions are met:

    - If the product's Revenue, in any period is **negative**.

    - If the product's Revenue, in half or more of the analyzed periods is **null** (blank).

    - If the product revenue in the **last** period is less than the average revenue of the product across all analyzed periods.

- Furthermore, the list must contain a calculation of each product's total revenue as a percentage of the total revenue within the associated group. (E.g. "Boss, Casual Blue L" total revenue as a percentage of the total revenue of all products in the SHIRTS group.)

- With the global criterion, **Time = 2016**, and with Quarterly periods the Observation List should look like this:

| Revenue per Product Hierarchy by Time | | 2016 | | | | | Percentage of Product Groups |
|---|---|---|---|---|---|---|---|
| | | Total | Q1 | Q2 | Q3 | Q4 | |
| JEANS | Levis, Blue XXL | **504,441** | 132,241 | 82,853 | 199,175 | 90,172 | 2.06% |
| SHIRTS | Boss Casual, Blue L | **23,213** | | | 23,213 | | 0.31% |
| | Boss Casual, Blue S | **100,706** | 10,567 | 90,138 | | | 1.34% |
| T-SHIRTS | Armani, White S | **103,400** | 35,649 | 32,994 | 19,079 | 15,679 | 0.66% |
| | Armani, White XL | **21,364** | -1,048 | -5,684 | 10,187 | 17,909 | 0.14% |
| | Boss Casual, Blue M | **14,040** | | | | 14,040 | 0.09% |
| | Boss, Grey XXL | **4,883** | | 4,883 | | | 0.03% |
| | Bosswell, Black M | **7,784** | -764 | 8,549 | | | 0.05% |
| | Bosswell, Blue L | **41,587** | | | 17,625 | 23,962 | 0.26% |
| | Bosswell, Blue M | **402,450** | 55,934 | 128,092 | 128,092 | 90,333 | 2.56% |
| | Bosswell, White L | **132,412** | 48,504 | 48,017 | 24,237 | 11,654 | 0.84% |
| | Levis, Blue XXL | **103,896** | | 78,780 | | 25,116 | 0.66% |
| | Levis, White M | **1,432,993** | 346,416 | 360,510 | 392,301 | 333,766 | 9.13% |
| | Levis, White XL | **795,009** | 124,281 | 301,010 | 204,718 | 165,000 | 5.06% |
| **UNDERWEAR** | **Total** | **79,349** | **30,701** | **36,605** | **-7,150** | **19,192** | **0.17%** |
| | Armani, Boxer Grey L | **30,610** | 17,160 | 7,636 | -7,150 | 12,964 | 38.58% |
| | Armani, Boxer Grey X | **48,739** | 13,541 | 28,969 | | 6,229 | 61.42% |

- .... and with Monthly periods:

| Revenue per Product Hierarchy by Time | | 2016 | | | | | | | | | | | | | | | | | Percentage of Product Groups |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Total | Q1 | | | | Q2 | | | | Q3 | | | | Q4 | | | | |
| | | | Total | Jan | Feb | Mar | Total | Apr | May | Jun | Total | Jul | Aug | Sep | Total | Oct | Nov | Dec | |
| JEANS | Levis 501, Black | 418,116 | **134,590** | 59,190 | 57,304 | 18,096 | **54,288** | | 36,192 | 18,096 | **68,308** | | 18,096 | 50,212 | **160,931** | 89,575 | 36,068 | 35,287 | 1.71% |
| | Levis, Blue XXL | 504,441 | **132,241** | 48,578 | 51,277 | 32,386 | **82,853** | 80,154 | 2,699 | | **199,175** | 83,663 | 52,847 | 62,666 | **90,172** | 39,866 | 16,193 | 34,113 | 2.06% |
| SHIRTS | Boss Casual, Blue L | 23,213 | | | | | | | | | **23,213** | | 23,213 | | | | | | 0.31% |
| | Boss Casual, Blue S | 100,706 | **10,567** | | | 10,567 | **90,138** | 45,743 | 44,395 | | | | | | | | | | 1.34% |
| | Boss Casual, Sand S | 120,666 | | | | | **48,266** | | 48,266 | | **24,133** | 24,133 | | | **48,266** | | | 48,266 | 1.60% |
| | Boss Casual, Sand XL | 84,377 | **30,950** | 15,475 | | 15,475 | **24,024** | | | 24,024 | | | | | **29,403** | 13,928 | | 15,475 | 1.12% |
| | Boss Casual, White XL | 1,442,514 | **233,163** | 65,645 | 41,985 | 125,533 | **406,230** | 218,543 | 167,032 | 20,654 | **297,351** | 76,488 | 124,037 | 96,826 | **505,770** | 111,132 | 319,820 | 74,818 | 19.16% |
| T-SHIRTS | Armani, Sand S | 1,105,883 | **158,631** | | 15,463 | 143,169 | **348,665** | 121,311 | 62,088 | 165,266 | **229,179** | 53,364 | 71,431 | 104,384 | **369,407** | 152,271 | 128,573 | 88,563 | 7.04% |
| | Armani, White S | 103,400 | **35,649** | 434 | 13,452 | 21,762 | **32,994** | 23,915 | | 9,079 | **19,079** | 9,422 | 9,656 | | **15,679** | | | 15,679 | 0.66% |
| | Armani, White XL | 21,364 | **-1,048** | -8,489 | | 7,441 | **-5,684** | | -5,684 | | **10,187** | | | 10,187 | **17,909** | | | 17,909 | 0.14% |
| | Boss Casual, Blue M | 14,040 | | | | | | | | | | | | | **14,040** | | | 14,040 | 0.09% |
| | Boss Casual, Grey XL | 144,037 | **18,814** | 7,363 | | 11,450 | **29,453** | | 7,363 | 22,090 | **50,585** | | 14,726 | 35,859 | **45,186** | 15,524 | 21,132 | 8,529 | 0.92% |
| | Boss, Grey XXL | 4,883 | | | | | **4,883** | | 4,883 | | | | | | | | | | 0.03% |
| | Bosswell, Black M | 7,784 | **-764** | | -8,549 | 7,784 | **8,549** | 8,549 | | | | | | | | | | | 0.05% |
| | Bosswell, Blue L | 41,587 | | | | | | | | | **17,625** | | | 17,625 | **23,962** | 23,962 | | | 0.26% |
| | Bosswell, Blue M | 402,450 | **55,934** | 36,598 | 3,050 | 16,286 | **128,092** | 18,299 | 91,494 | 18,299 | **128,092** | 54,896 | 36,598 | 36,598 | **90,333** | 63,182 | 31,450 | -4,299 | 2.56% |
| | Bosswell, White L | 132,412 | **48,504** | 7,098 | 26,602 | 14,804 | **48,017** | 12,464 | 17,776 | 17,776 | **24,237** | 1,039 | 6,232 | 16,966 | **11,654** | 6,232 | 5,422 | | 0.84% |
| | Bosswell, White XXL | 1,253,454 | **267,244** | 60,464 | 43,253 | 163,527 | **305,422** | 174,673 | 35,474 | 95,274 | **292,671** | 53,929 | 116,318 | 122,423 | **388,117** | 167,656 | 161,390 | 59,071 | 7.98% |
| | Levis, Blue L | 74,350 | | | | | **24,944** | | | 24,944 | **23,556** | | 23,556 | | **25,849** | | 25,849 | | 0.47% |
| | Levis, Blue XXL | 103,896 | | | | | **78,780** | 25,116 | 28,548 | 25,116 | | | | | **25,116** | | 25,116 | | 0.66% |
| | Levis, Lime M | 141,968 | **17,722** | | | 17,722 | **17,722** | | | 17,722 | | | | | **106,525** | 88,804 | | 17,722 | 0.90% |
| | Levis, Sand XXL | 80,275 | **17,722** | | | 17,722 | | | | | **17,722** | | | 17,722 | **44,832** | 29,640 | 13,053 | 2,139 | 0.51% |
| | Levis, White M | 1,432,993 | **346,416** | | 111,796 | 234,620 | **360,510** | 170,988 | 25,506 | 164,016 | **392,301** | 65,745 | 132,199 | 194,357 | **333,766** | 134,031 | 127,621 | 72,114 | 9.13% |
| | Levis, White M 600 | 446,200 | **124,829** | 71,487 | 30,785 | 22,558 | **104,486** | 36,488 | 38,923 | 29,075 | **87,617** | | 50,236 | 37,381 | **129,268** | 72,893 | 32,869 | 23,505 | 2.84% |
| | Levis, White XL | 795,009 | **124,281** | -41,574 | 4,670 | 161,185 | **301,010** | 131,242 | 85,716 | 84,053 | **204,718** | | 126,269 | 78,449 | **165,000** | 104,295 | 32,687 | 28,018 | 5.06% |
| **UNDERWEAR** | **Total** | **79,349** | **30,701** | **6,770** | | **23,930** | **36,605** | **28,969** | **7,636** | | **-7,150** | **-7,150** | | | **19,192** | | | **19,192** | **0.17%** |
| | Armani, Boxer Grey L | 30,610 | **17,160** | | | 17,160 | **7,636** | | 7,636 | | **-7,150** | -7,150 | | | **12,964** | | | 12,964 | 38.58% |
| | Armani, Boxer Grey X | 48,739 | **13,541** | 6,770 | | 6,770 | **28,969** | 28,969 | | | | | | | **6,229** | | | 6,229 | 61.42% |

## Extra exercise – continued

- Make visible subtotals on the Product Group level.
- Make sure that the subtotals display the sum of the visible products only.
- With **Time = 2016** and Quarterly periods the Observation List should look like this:

Revenue per Product Hierarchy by Time

| | | – 2016 | | | | | Percentage of Product Groups |
|---|---|---|---|---|---|---|---|
| | | Total + | Q1 + | Q2 + | Q3 + | Q4 | |
| Product Group | Product | Revenue | Revenue | Revenue | Revenue | Revenue | |
| Total | | 47,772,999 | 10,224,630 | 11,596,532 | 10,920,155 | 15,031,683 | 100.00% |
| JEANS – | Total | 504,441 | 132,241 | 82,853 | 199,175 | 90,172 | 1.06% |
| | Levis, Blue XXL | 504,441 | 132,241 | 82,853 | 199,175 | 90,172 | 2.06% |
| SHIRTS – | Total | 123,918 | 10,567 | 90,138 | 23,213 | 0 | 0.26% |
| | Boss Casual, Blue L | 23,213 | 0 | 0 | 23,213 | 0 | 0.31% |
| | Boss Casual, Blue S | 100,706 | 10,567 | 90,138 | 0 | 0 | 1.34% |
| T-SHIRTS – | Total | 3,059,819 | 608,972 | 957,151 | 796,238 | 697,458 | 6.40% |
| | Armani, White S | 103,400 | 35,649 | 32,994 | 19,079 | 15,679 | 0.66% |
| | Armani, White XL | 21,364 | -1,048 | -5,684 | 10,187 | 17,909 | 0.14% |
| | Boss Casual, Blue M | 14,040 | 0 | 0 | 0 | 14,040 | 0.09% |
| | Boss, Grey XXL | 4,883 | 0 | 4,883 | 0 | 0 | 0.03% |
| | Bosswell, Black M | 7,784 | -764 | 8,549 | 0 | 0 | 0.05% |
| | Bosswell, Blue L | 41,587 | 0 | 0 | 17,625 | 23,962 | 0.26% |
| | Bosswell, Blue M | 402,450 | 55,934 | 128,092 | 128,092 | 90,333 | 2.56% |
| | Bosswell, White L | 132,412 | 48,504 | 48,017 | 24,237 | 11,654 | 0.84% |
| | Levis, Blue XXL | 103,896 | 0 | 78,780 | 0 | 25,116 | 0.66% |
| | Levis, White M | 1,432,993 | 346,416 | 360,510 | 392,301 | 333,766 | 9.13% |
| | Levis, White XL | 795,009 | 124,281 | 301,010 | 204,718 | 165,000 | 5.06% |
| UNDERWEAR – | Total | 79,349 | 30,701 | 36,605 | -7,150 | 19,192 | 0.17% |
| | Armani, Boxer Grey L | 30,610 | 17,160 | 7,636 | -7,150 | 12,964 | 38.58% |
| | Armani, Boxer Grey X | 48,739 | 13,541 | 28,969 | 0 | 6,229 | 61.42% |

## Functions in the Targit Syntax

Here is a list of available functions.

| Function | Description |
|---|---|
| **Sum (default)** | Summarizes the measures in the cell ranges. (cells defaults to sum) |
| **Count** | Counts the number of cells in the cell range having any measure value. (normally used for counting dimension values) |
| **Allcount** | Counts the number of all cells in the cell range regardless of measure values. |
| **Max** | Returns the maximum value for measures in the cell range. |
| **Min** | Returns the minimum value for measures in the cell range. |
| **Stdev** | Returns standard deviation for measures in the cell range. |
| **Avg** | Returns average for measures in the cell range. |

As explained above, the general usage is :

**Function**(**cell range x**, **cell range y**, **measure list** (m1;m2;m3… etc).

**Cell range** can be just a single value like d1 or a range like d1:d3 (1st to 3rd)

**Measure list** can be a single measure like m1 or a list of measures like m1;m2. (1st and 2nd).

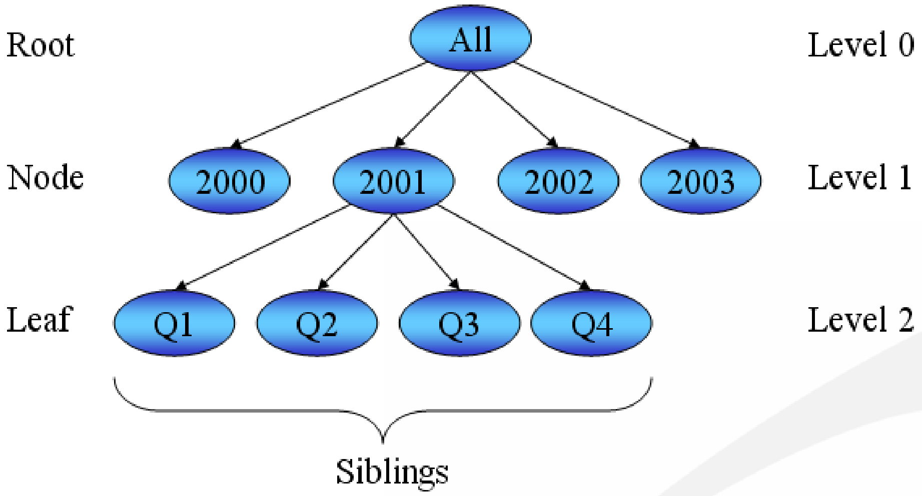
Here is a couple of examples based on the crosstab shown above.

| Calculation | Formula |
|---|---|
| Average revenue in 2003 – Q2 | Avg(d-1,d1:d-1,m1) |
| Profit in 2003 – Q2 for each product. | Sum(d-1,0,m1)-Sum(d-1,0,m2) |
| Accumulated average for Cost in 2003 – Q1 | Avg(d-2,d1:0,m2) |
| Lets assume that all rows where **m1 < average** for the last column is hidden by a visibility agent. We want to calculate the visible rows as a percent of all rows for that column. | Sum(d-1,d1:d-1 (v),m1) % sum(d-1,d1:d-1,m1) |

## Result modifiers

| Function | Description |
|---|---|
| Abs(x) | Returns the absolute value of x<br>Examples:<br>Abs(-3) =3<br>Abs(0) = 0<br>Abs(3) = 3 |
| Ceil(x) | Returns the smallest integer not less than x<br>Examples:<br>Ceil(3.01) = 4<br>Ceil(3.5) = 4<br>Ceil (3.99) = 4 |
| Div(x;y) | Returns the result of x/y with the fractional part discarded. (Integer division)<br>Examples:<br>Div(10;3) = 3<br>Div(-10;3) = -3<br>Div (-10;-3) = 3 |
| Floor(x) | Returns the largest integer not greater than x<br>Examples:<br>Floor(3.01) = 3<br>Floor(3.5) =3<br>Floor(3.99) =3 |
| Int(x) | Returns the integer part of x.<br>Examples:<br>Int(3.25) =3<br>Int(-3.25) = -3 |
| Mod(x;y) | Returns the remainder of x/y<br>Examples:<br>Mod(7;3) = 1<br>Mod(-7;3) = -1 |
| Round(x;y) | Rounds the number x with y numbers of decimals (if y is negative the integer part of a number is rounded)<br>Examples:<br>Round(7.21;1) = 7.2<br>Round(7.125;2) = 7.13<br>Round(133;-1) = 130 |

### Dimension structure and naming




Children: only include children of the current element.

Ragged: include elements that have a given number of children (part of a ragged hierarchy)

Leaves: The special “leaves” filter (short: “rl”) is the exact same as ragged on level zero, “r0”

## TARGIT formula syntax

### Arithmetic operators

Please note that operators listed first are 'stronger' than the later ones, e.g. multiplication is stronger than addition.

After all, 2 + 3 * 4 equals 14; if addition had been stronger (or equally strong) it would have been 20.

| Operator | Description |
|---|---|
| - | Unary minus, negates the expression, e.g. -5. |
| ^ | Power, e.g. 5 ^ 2 = 25, and 25 ^ 0.5 = 5. |
| *, /, % | Multiplication, division and a new division operator, which simply divides and then multiplies the result by 100. |
| +, - | Addition and subtraction |

### Boolean operators

All Boolean operators return 1 ('nonzero') if the condition is met and 0 if it is not. The operator 'not' can be used to negate an expression, but remember to use parenthesis, as 'not' is stronger than all other operators: 'not 1 > -1' is nonzero, while 'not (1 > -1)' is zero. The other Boolean operators are all weaker than the arithmetic operators.

| Operator | Description |
|---|---|
| Not | Nonzero if the expression after 'not' is zero, otherwise zero. |
| <, <=, >, >=, =, <> | Value comparison operators. |
| And | Nonzero if the expressions on either side of 'and' are both nonzero. |
| Or | Nonzero if one or both of the expressions on either side of 'or' is nonzero. |

**Other operators**

| Operator | Description |
|---|---|
| [label:] (  ) | Use parenthesis to group expressions, e.g. to make '(2+3)*4' equal 20.<br><br>Also, by supplying a label, the expression can be used more than once in a statement without having to copy it textually.  A complex expression that you want to use several times is easier to only have to adjust in one place; -or if, in spite of the added Boolean operators, you have to use the same expression in two different branches of an 'if-then-else' expression.<br><br>The label name can contain the letters A-Z, underscore ('_') and 0-9.  The first letter of the label can only be A-Z or underscore.<br><br>E.g: 'if AccumAvg:(avg(d-1,d1:0,m1)) <> 0 then<br><br>sum(d-1,0,m1) % AccumAvg else 100' |
| if [A] then [B] else [C] | Evaluates to B if A is nonzero, or to C if A is zero. |

**Aggregation functions**

| Function | Description |
|---|---|
| sum( [element sets] ) | Simple sum of elements |
| count( [element sets] ) | Number of non-empty elements |
| allcount( [element sets] ) | Number of elements, both empty and non-empty. |
| stdev( [element sets] ) | Standard deviation of elements |
| avg( [element sets] ) | Average of elements |
| max( [element sets] ) | Maximal value found in elements |
| min( [element sets] ) | Minimal value found in elements |

**Element reference modifiers**

The element references can be filtered by appending a list of modifier names to the element range.

E.g. 'sum(d-1, d1:0 (visible, siblings), m1)' gives the accumulated sum of the first measure of the last column, but includes only the visible elements, and only the elements that are siblings to the current row.

As a shorthand, the abbreviation letter in the table below can be supplied instead of the entire name.

| Visibility modifier | Short | Description |
|---|---|---|
| all | a | Both visible and hidden elements are included.<br>This is the default visibility filter. |
| visible | v | Only visible elements are included. |
| hidden | h | Only hidden elements are included. |
| Hierarchy modifier | Short | Description |
| level | l | Only elements on the same hierarchy levels in the dimensions as the current element are included.<br>This is the default hierarchy filter.<br>If an integer >= 0 immediately follows this filter name, e.g. 'l2', then only the values in level 2 are included, with level zero being the grand total. |
| siblings | s | Only elements with the same hierarchy parents in the dimensions as the current element are included. |

| | | |
|---|---|---|
| children | c | Only elements that are nested inside the current element are included. Please note that only data references can be used with this filter.<br><br>Example: This filter can be used with an if-then-else operator for making subtotals that only include the visible elements, when some elements have been hidden by a Visibility Agent:<br><br>if allcount(0,d1:d-1(c),m1) > 0 then sum(0,d1:d-1(v,c),m1) else sum(0,0,m1) |

<table>
<tr><td>ragged</td><td>r</td><td>This filter is useful e.g. for making an accumulation which follows the current expansion of single elements:<br><br>
<table>
<tr><th colspan="2">Time</th><th>Turnover</th><th>Accumulated</th></tr>
<tr><td>Total</td><td></td><td>750</td><td>750</td></tr>
<tr><td>2000</td><td></td><td>500</td><td>500</td></tr>
<tr><td rowspan="5">2001</td><td>Total</td><td>-400</td><td>-400</td></tr>
<tr><td>Q1</td><td>250</td><td>750</td></tr>
<tr><td>Q2</td><td>300</td><td>1050</td></tr>
<tr><td>Q3</td><td>-500</td><td>550</td></tr>
<tr><td>Q4</td><td>-450</td><td>100</td></tr>
<tr><td>2002</td><td></td><td>250</td><td>350</td></tr>
<tr><td>2003</td><td></td><td>400</td><td>750</td></tr>
</table>
<br>Like with the ‘level’ filter, an integer >= 0 can follow this filter name, e.g. ‘r0’. In ragged filter, however, this value is somewhat more complicated to explain:<br><br>Level zero are the leaf elements, i.e. the elements that have no children. Level one are their parents. In the example above, only 2001 is on level one. Level two are the elements that have level 1 children (grand total above), etc.</td></tr>
<tr><td>leaves</td><td>rl</td><td>Only elements with no children are included. This is the exact same as ‘r0’.</td></tr>
<tr><th>Order modifier</th><th>Short</th><th>Description</th></tr>
<tr><td>sorted</td><td>o</td><td>Relative and data references are indexed according to the current sorting of the grid.<br><br>This is the default order filter.</td></tr>
</table>

| | | |
|---|---|---|
| unsorted | u | Relative and data references are indexed according to the order of the dimension values in the cube. |

## Template metadata

If you want to reuse a calculation, it may be a good idea to put some flexibility into your calculation, so that e.g. the choice of which value should be used as index 100 can be changed easily, from the smartpad instead of having to edit the formula manually.  To tell the SmartCalculations editor that there is such a customizable reference in the formula, you must supply the metadata (name, default value etc.) for the ranges in question.

The metadata for a range are given in square brackets after the range, just as range filters are given in parenthesis.

| Metadata element | Syntax example | Description |
|---|---|---|
| Parameter name | `d1 ["A"]` | Name, identifies the parameter.  Must always be present.  All ranges with the same name are synchronized. |
| Allow range | `d1 ["A":range]` | Allow the user to select both starting and ending point. |
| Editable filters | `d1 ["A":filter(v,h,o)]` | Allow the user to edit the given filter types.<br>Filters: v = visibility, h = hierarcy, o = order.<br>If the parenthesis are missing, all filters can be edited. |

| Metadata element | Syntax example | Description |
|---|---|---|
| Default value | `d1 ["A"=0]` | When a template is used to add a new calculation, the default reference can be specified here.<br><br>If a relative reference is given, like it is in this example, it is modified unless the calculation is added 'for all dimension values', i.e. as a custom measure.<br><br>If it is greater than zero, it is modified to a left-to-right data reference, e.g. 2 => d2.<br><br>If it is less than or equal to zero, it is modified to a right-to-left data reference, e.g. 0 => d-1 and -1 => d-2.<br><br>In this way, if the template is designed as a custom measure, it can reasonably easily be added as calculated columns and rows too. |
| Description | `d1 ["A","Source data"]` | Short explanation to be shown in the list of parameters and on the edit page of this parameter. |
| `d1(v) ["Base":filter=d1,"Index 100"]` | | |

The metadata for measures are like the metadata for ranges, except the ':range' and ':filter' elements are not available.