

Pentaho Analyzer Cookbook: Calculations and Multidimensional Expressions (MDX)

HITACHI

Inspire the Next

Change log (if you want to use it):

Date	Version	Author	Changes

Contents

Overview	1
Core Recipes	2
Learning MDX	2
Getting Started with Designing Calculated Measures	5
Getting All SQL Statements for a Report	7
Sharing a User Defined Calculated Measure Across Multiple Reports	8
Basic Recipes	10
Showing Rows with No Data	10
Creating a Subtotal on Distinct Count Measures	11
Comparing Top 10 Customers Versus All Customers	12
Creating a Pareto Chart	13
Creating a Moving Average Calculation	16
Creating a Relative Date Filter (Current Year, Next N Years, N Years Ago)	17
Aggregate Total Type	18
Creating a Running Sum or Cumulative Calculation	19
Percent Change Year Ago Calculation	20
Advanced Recipes	22
Pinning a Measure to Specific Dimension Measures	22
Pinning a Measure to a Specific Hierarchy Level	24
Creating Semi-Additive Measures	26
Creating a Conditional Measure	29
Comparing YTD for Current and Previous Year	32
Generating Trend Lines Using Linear Regression	35
Conditionally Highlighting or Formatting Cells	38
Filtering on a Member Property	41
Adding Target or Goal Measures to a Report	42
Calculating a Weighted Average	44
How Solve Orders Affect Calculating Measures	45
Test Your MDX Knowledge	48
Related Information	49

This page intentionally left blank.

Overview

This Analyzer Cookbook was written to assist busy people with solving various analysis use cases encountered in real life customer scenarios. Most of these examples in this document came from requests from pre-sales, engineering service requests (ESRs), and talking to customers.

These examples focus on calculations and making use of the power of multidimensional expressions (MDX). Analyzer was built on top of Online Analytical Processing (OLAP) technology, and you can use those capabilities rather than just using operational reporting technology and relational structured query language (SQL).

Because all examples in this book work on the vanilla `SteelWheels` catalog (unless otherwise noted), they should be very easy to replicate and study.

The cookbook is divided up into three sections:

- [Core Recipes](#): Items that we recommend you read through, because they come in handy for most analysis scenarios.
- [Basic Recipes](#): Use cases that Analyzer supports out of the box with no MDX experience needed.
- [Advanced Recipes](#): Use cases that require you to understand how MDX expressions are evaluated.

The intention of this document is to speak about topics generally; however, these are the specific versions covered here:

Software	Version(s)
Pentaho	6.x, 7.x, 8.x
Mondrian	3.x

The [Components Reference](#) in Pentaho Documentation has a complete list of supported software and hardware.

Core Recipes

This section contains information that is applicable for most analysis scenarios. Topics discussed in this section are as follows:

- [Learning MDX](#)
- [Getting Started with Designing Calculated Measures](#)
- [Getting All SQL Statements for a Report](#)
- [Sharing a User Defined Calculated Measure Across Multiple Reports](#)

Learning MDX

MDX is the query language for querying and manipulating multidimensional data stored in OLAP cubes. Analyzer generates MDX statements based on report definitions and then executes them against Mondrian.

Learning MDX begins with browsing through the [Mondrian documentation](#), and then practicing with a few simple MDX queries.

First, try out this tutorial based on SteelWheels:

1. Create a new **Analysis Report** using the SteelWheels data source.
2. Click the settings wheel, then Administration, and open the **MDX** editor:

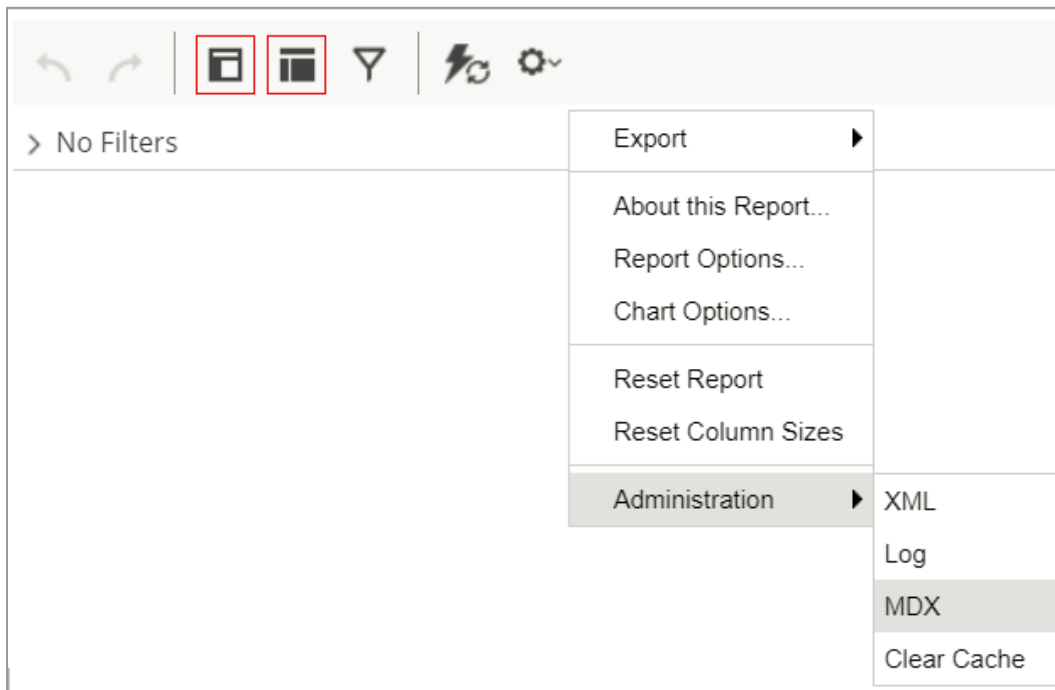


Figure 1: Opening the MDX Editor

3. Enter this MDX statement and select **Execute MDX**:

```
SELECT
    {[Measures].[Sales]} ON COLUMNS,
    {[Markets].[Territory].MEMBERS} ON ROWS
FROM [SteelWheelsSales]
```

	[Measures].[Sales]
[Markets].[#null]	
[Markets].[APAC]	1,281,706
[Markets].[EMEA]	5,008,224
[Markets].[Japan]	503,958
[Markets].[NA]	3,852,061

Figure 2: Creating a Cube

This simple query places the measure `Sales` on the columns axis, and all members of the `Markets` dimension on the row axis.

4. Cross check your `steelwheels.mondrian.xml` file to make sure you are comfortable with the dimensions and measures defined in the `SteelWheelsSales` cube.
5. Now, use this MDX statement to show `NA` (North America) sales and drill down into the next level of sales (`Country`).

```
SELECT
    {[Measures].[Sales]} ON COLUMNS,
    UNION({[Markets].[NA]},[Markets].[NA].Children) ON ROWS
FROM [SteelWheelsSales]
```

	[Measures].[Sales]
[Markets].[NA]	3,852,061
[Markets].[NA].[Canada]	224,079
[Markets].[NA].[USA]	3,627,983

Figure 3: Specifying Markets

This query uses the `UNION` MDX function to combine two sets. The first set contains a single `Markets` dimension member `[Markets].[NA]`. `NA` is just a member that comes from the `Territory` level. The second set calls the `Children` MDX function on the `NA` member, which returns all children of `NA`.

It is simple to get `NA` and child `Canada/USA` sales using a basic MDX expression. If you tried to do this with SQL, it would have been much more complicated and involved several `group-bys`.

6. Finally, calculate each country's percentage (%) contribution to the parent:

```

WITH
    MEMBER [Measures].[% of Total] AS
        '[Measures].[Sales]/([Measures].[Sales],[Markets].CurrentMember.Parent)',
        FORMAT_STRING = '###0.00%'
    SELECT
        {[Measures].[Sales], [Measures].[% of Total]} ON COLUMNS,
        UNION({[Markets].[NA]},[Markets].[NA].Children) ON ROWS
    FROM [SteelWheelsSales]

```

	[Measures].[Sales]	[Measures].[% of Total]
[Markets].[NA]	3,852,061	36.18%
[Markets].[NA].[Canada]	224,079	5.82%
[Markets].[NA].[USA]	3,627,983	94.18%

Figure 4: Calculating Contribution

In this example, we created a new calculated member in the `Measures` dimension called `% of Total`. When this calculated measure is evaluated, the numerator refers to the `Sales` for the current `Market` dimension member coming from the row axis, and the denominator refers to the `Sales` for the parent member of the current `Market` dimension member on the row axis.

This means that whatever `Market` dimension member is on the row axis, the `% of Total` calculates the corresponding contribution correctly. Try this with other territories, such as `EMEA`, to verify.

For `[Markets].[NA]`, the `% of Total` equals 36.18% because when the current member on the row axis is `NA`, then the current member's parent refers to the `All` member. The `All` member is a special member that exists in every hierarchy. It is the topmost level, and normally you don't have to include it explicitly when you come up with member unique names. For example, these two are equivalent:

- `[Markets].[All Markets].[NA]`
- `[Markets].[NA]`

Test your MDX knowledge: If you wanted the `% of Total` for the `[Markets].[NA]` member to total 100%, but not to use the `All` member in the denominator, how would you modify the `% of Total` calculated measure? The answer appears [near the end of the cookbook](#). Your expected output is:

	[Measures].[Sales]	[Measures].[% of Total]
[Markets].[NA]	3,852,061	100.00%
[Markets].[NA].[Canada]	224,079	5.82%
[Markets].[NA].[USA]	3,627,983	94.18%

Figure 5: % of Total is 100%

Getting Started with Designing Calculated Measures

You should always start designing and testing your MDX expressions in Analyzer by creating a user defined calculated measure. This allows you to:

- Get used to the MDX syntax.
- Try out different MDX functions.
- Rapidly see MDX parser errors.
- Validate the calculation results.

Create Calculated Measure

Invalid formula definition. Details: Mondrian Error:MDX object '[Sales]' not found in cube 'SteelWheelsSales'

Display Name:

Format: Decimal Places:

Type:

Quantity
Sales

[Sales] / [Quantity]

Subtotal Calculation
☐ Calculate subtotals using measure formula

+ - * / () Clear

OK Cancel

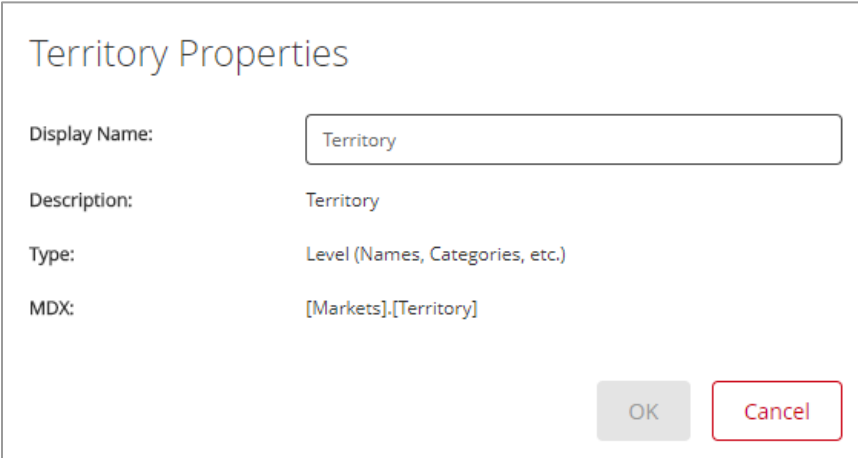
Figure 6: Checking Expressions for Errors

Since Analyzer is based on Mondrian, you should familiarize yourself with [what is available in Mondrian](#).

When you try to use a new MDX function, search online for it since the [Microsoft Software Developer Network \(MSDN\)](#) provides good descriptions of and examples for each function.

When you design MDX expressions, you will need to know dimension, hierarchy, level, and unique member names. You can get some of this from the Mondrian schema file. Alternatively, they are also exposed in Analyzer.

Here are the dimension, hierarchy, and level name for Territory:



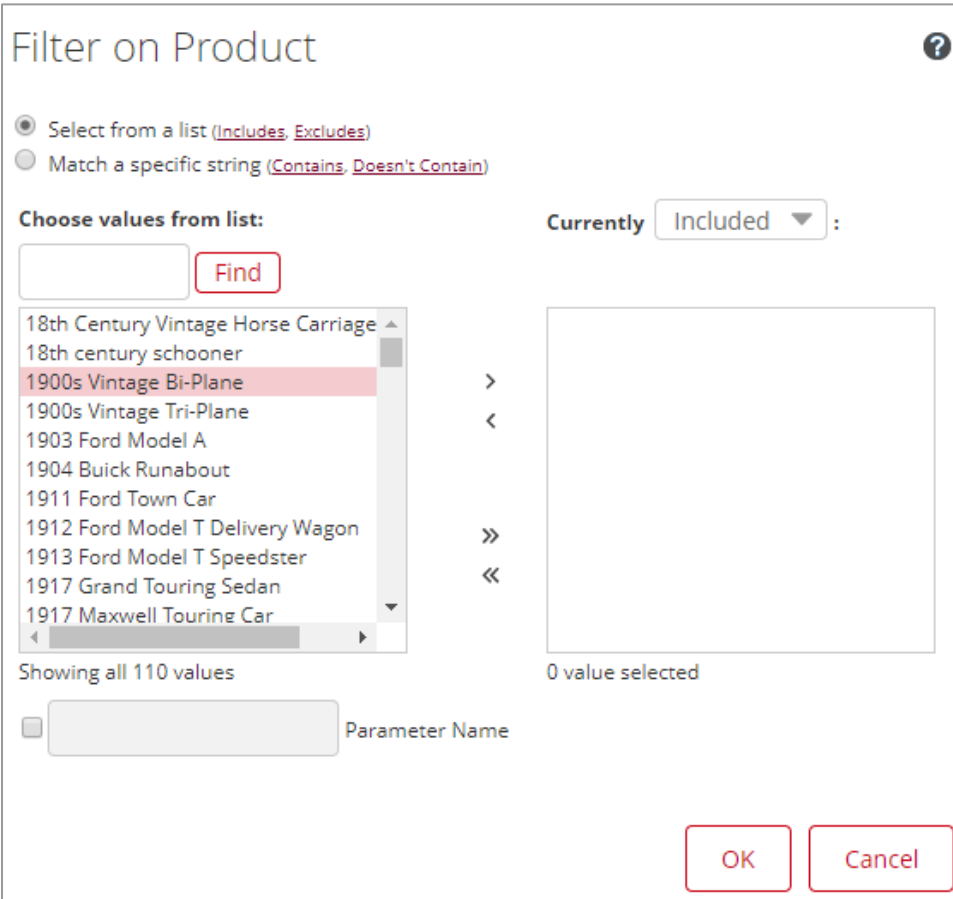
The 'Territory Properties' dialog box contains the following information:

Display Name:	Territory
Description:	Territory
Type:	Level (Names, Categories, etc.)
MDX:	[Markets].[Territory]

Buttons: OK, Cancel

Figure 7: Dimension, Hierarchy, and Level Name

The following figure shows how you can get the 1900s Vintage Bi-Plane member unique name by looking at the tooltip in the Product filter dialogue:



The 'Filter on Product' dialog box shows the following configuration:

- Radio buttons: ☒ Select from a list (Includes, Excludes); ☐ Match a specific string (Contains, Doesn't Contain)
- Choose values from list: [Input field] [Find]
- Currently: Included
- Product list (Showing all 110 values):
 - 18th Century Vintage Horse Carriage
 - 18th century schooner
 - 1900s Vintage Bi-Plane** (highlighted)
 - 1900s Vintage Tri-Plane
 - 1903 Ford Model A
 - 1904 Buick Runabout
 - 1911 Ford Town Car
 - 1912 Ford Model T Delivery Wagon
 - 1913 Ford Model T Speedster
 - 1917 Grand Touring Sedan
 - 1917 Maxwell Touring Car
- 0 value selected
- Parameter Name: [Input field]

Buttons: OK, Cancel

Figure 8: Member Unique Name

You can also design and test MDX expressions directly in Analyzer's **MDX editor**. You must be an Administrator to access this functionality.



Usually, the most effective way to test and debug with the MDX editor is first to build the report, then grab the MDX from the query log, and finally start tweaking the MDX in the editor.

Getting All SQL Statements for a Report

A single Analyzer report may generate one or more MDX statements, and each MDX statement will usually result in multiple SQL statements. Whether SQL statements are executed will depend on what has already been cached in Mondrian. Therefore, it is useful to be able to get all SQL statements that will be generated for a single report, by clearing the cache.

If you are logged in as an Administrator, you will have access to the following admin functionality:

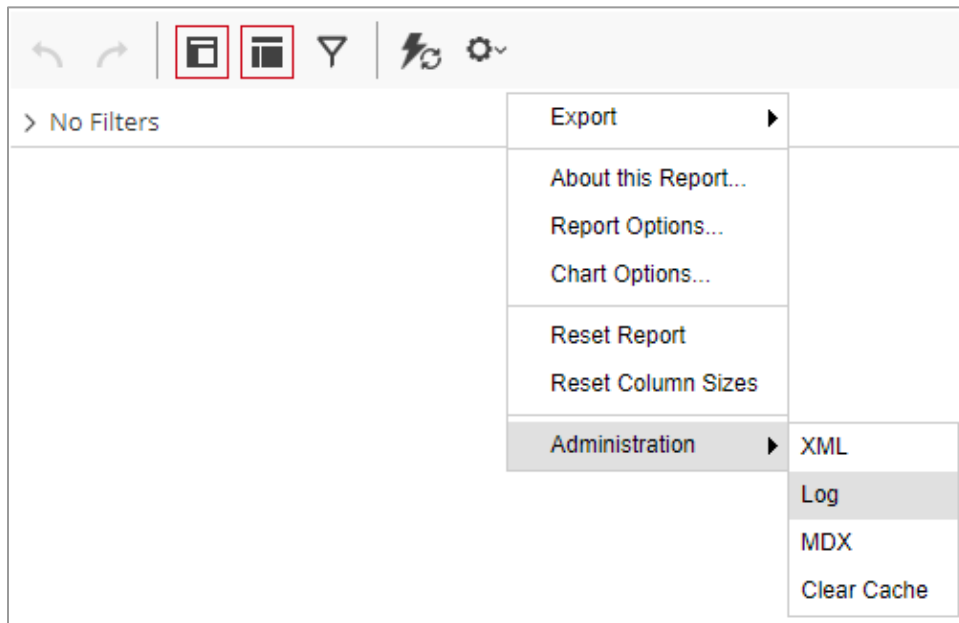


Figure 9: Admin Log

To get all SQL statements for a report:

1. Build the report.
2. Select **Clear Cache**.
3. Select **XML** and then **OK**.
4. Select **Log**.

A new browser window will open with all MDX and SQL statements executed for the report. SQL statements will show up in pairs. The first statement (`exec`) refers to when the first row is returned. The second statement (`exec+fetch`) is when all rows have been read from the query. The last statement breaks down the report total execution time.

Sharing a User Defined Calculated Measure Across Multiple Reports

To share a user defined calculated measure across multiple reports, you will need to **define the measure in the Mondrian schema**. Define the measure within a cube, and it will show up as a calculated measure in the `Measures` dimensions.

Here is an example of a user defined calculated measure, and how it would appear in the Mondrian schema:

```
<CalculatedMember name="Average Sales Price" dimension="Measures">
  <Formula>[Measures].[Sales]/[Measures].[Quantity]</Formula>
  <CalculatedMemberProperty name="FORMAT_STRING" value="$#,##0.00" />
  <CalculatedMemberProperty name="SOLVE_ORDER"
value="200"></CalculatedMemberProperty>
</CalculatedMember>
</Cube>
```

Create Calculated Measure

Display Name:

Format: Decimal Places:

Type:

Measures List: Quantity, Sales, Quantity (selected), Sales

Formula:

Subtotal Calculation: ☒ Calculate subtotals using measure formula, ☐ Apply to data source

Calculator: +, -, *, /, (,), Clear

Buttons: OK, Cancel

Figure 10: Calculated Measure

Here are a few things to keep in mind when converting the MDX expression used in the report to the one that will be used in the schema:

- References to measures such as [Sales] or [Quantity] need to be prefixed with the dimension, so update them to [Measures].[Sales] or [Measures].[Quantity].
- Make sure your MDX expression does not refer to user defined numbers that were defined on the report, as they will obviously not exist in the schema.
- Set the SOLVE_ORDER using a CalculatedMemberProperty.
 - If you are running Analyzer 5.0 or later, and you checked the **Calculate subtotals using formula** box, then set SOLVE_ORDER to 200. If you did not check the box, do not set this property.
 - Set the FORMAT_STRING using a CalculatedMemberProperty. For static format strings, you will set the value attribute in the CalculatedMemberProperty. For dynamic format strings based on MDX expressions, you will set the expression attribute on the CalculatedMemberProperty. Format strings are mapped as follows:

Table 1: Format String Mapping

Format String Attribute	Formatting
General Number	#,##0.0
Currency	\$#,##0.00
Percentage	###0.00%

Here is an example of a dynamic format string:

```

<CalculatedMember name="Style Unit Sales" dimension="Measures">
  <Formula>[Measures].[Unit Sales]</Formula>
  <CalculatedMemberProperty name="FORMAT_STRING" expression="
    IIf([Measures].[Unit Sales] > 100000,
      '|#,###|style=green',
      IIf([Measures].[Unit Sales] > 50000,
        '|#,###|style=yellow',
        '|#,###|style=red'))" />
</CalculatedMember>

```

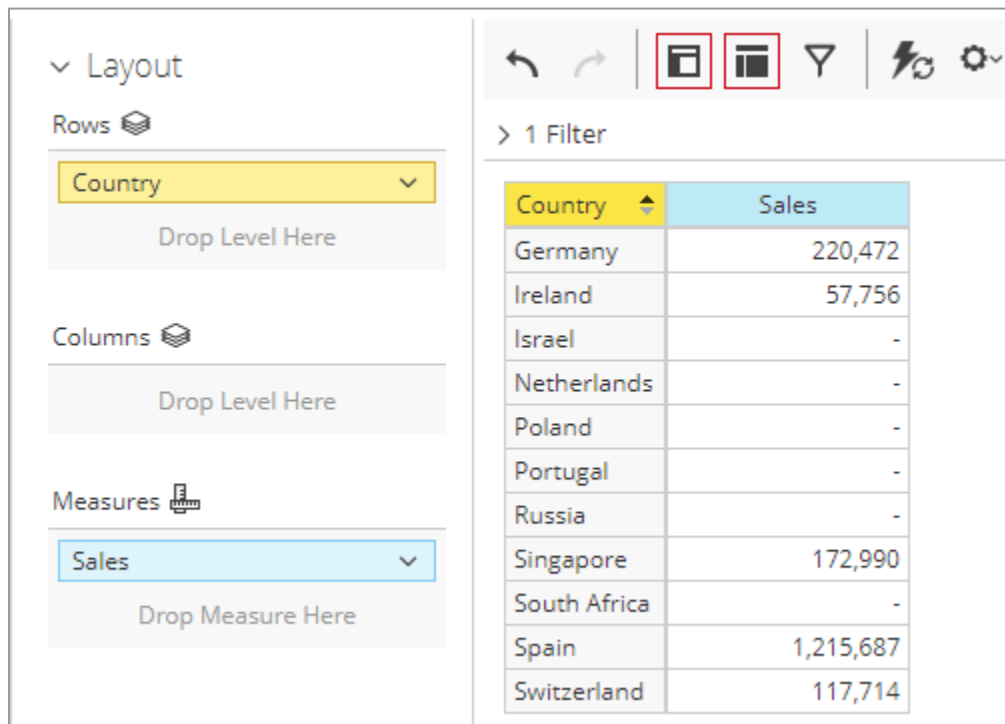
Basic Recipes

Here, you will find information on out of the box examples supported by Analyzer, where no experience is necessary. Topics covered in this section are as follows:

- [Showing Rows with No Data](#)
- [Creating a Subtotal on Distinct Count Measures](#)
- [Comparing Top 10 Customers Versus All Customers](#)
- [Creating a Pareto Chart](#)
- [Creating a Moving Average Calculation](#)
- [Creating a Relative Date Filter](#)
- [Aggregate Total Type](#)
- [Creating a Running Sum or Cumulative Calculation](#)
- [Percent Change Year Ago Calculation](#)

Showing Rows with No Data

Normally, Analyzer will filter out combinations of dimension members that have no data. No data means that either the measure value is null or that there is no such tuple in the fact table. However, you can show these empty cells by changing the report option **Show rows or columns with** to **Show all even blank measures**.



The screenshot shows the Pentaho Analyzer interface. On the left, the 'Layout' pane has three sections: 'Rows' with 'Country' selected, 'Columns' with 'Drop Level Here', and 'Measures' with 'Sales' selected. On the right, the table view shows a table with two columns: 'Country' and 'Sales'. The table contains data for 11 countries, with some countries having a dash '-' in the 'Sales' column, indicating no data.

Country	Sales
Germany	220,472
Ireland	57,756
Israel	-
Netherlands	-
Poland	-
Portugal	-
Russia	-
Singapore	172,990
South Africa	-
Spain	1,215,687
Switzerland	117,714

Figure 11: Showing Rows with No Data



Be careful with performance when using this feature. When you enable this option, all dimension members from each dimension will be cross-joined together. So, if you had three dimensions with 100 members each, your report would contain $100 \times 100 \times 100 = 1,000,000$ rows.

Creating a Subtotal on Distinct Count Measures

Remember to create this at the schema level as described in [Sharing a User Defined Calculated Measure Across Multiple Reports](#).

A distinct count measure is a measure defined in the Mondrian schema with:

```
aggregator = "distinct-count"
```

When showing the subtotal for a distinct count measure, you may want a straight sum of all the details rows, or you may want to compute the distinct count of transactions records in that total.

In this example, based on Foodmart, **Grand Total** will double count customers who have bought more than one product, whereas the **Grand Aggregate** will only count each customer once.

Product Family	Customer Count
Drink	1,753
Food	2,735
Non-Consumable	2,206
Grand Total	6,694
Grand Aggregate	2,755

Figure 12: Customer Count Table

You can show **Grand Total** and **Grand Aggregate** totals by right-clicking **Customer Count** and selecting the **Subtotals (Sums, Averages, etc.)** menu option:

Average, Max, Min, etc. ?

For **Customer Count**, show these **types of totals**:

- ☒ Sum
- ☒ Aggregate
- ☐ Average
- ☐ Max
- ☐ Min

OK
Cancel

Figure 13: Showing Total Types

Comparing Top 10 Customers Versus All Customers

To compare sales for the top 10 customers versus all customers, you need to compare the two groups side by side. You can do this with a `Top 10` filter. Click on report options at the bottom and choose **Totals with filtered values**.

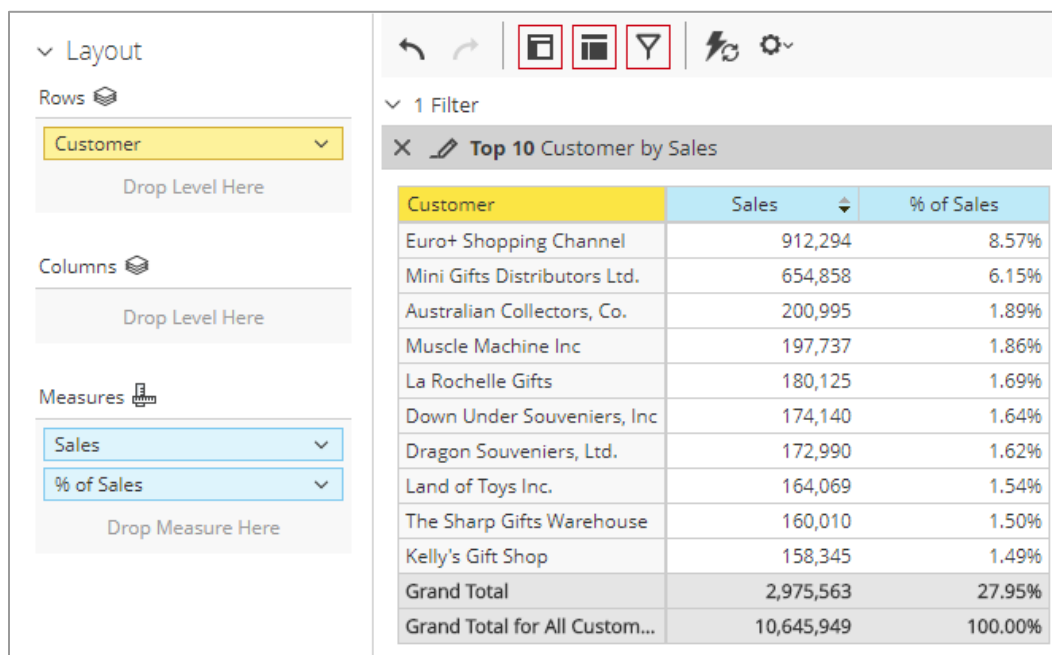


Figure 14: Top 10 Customers Sales

Creating a Pareto Chart

You can use a Pareto chart to highlight the most important of a large set of factors. For example, this chart shows that the top three countries result in 55.93% of total sales:

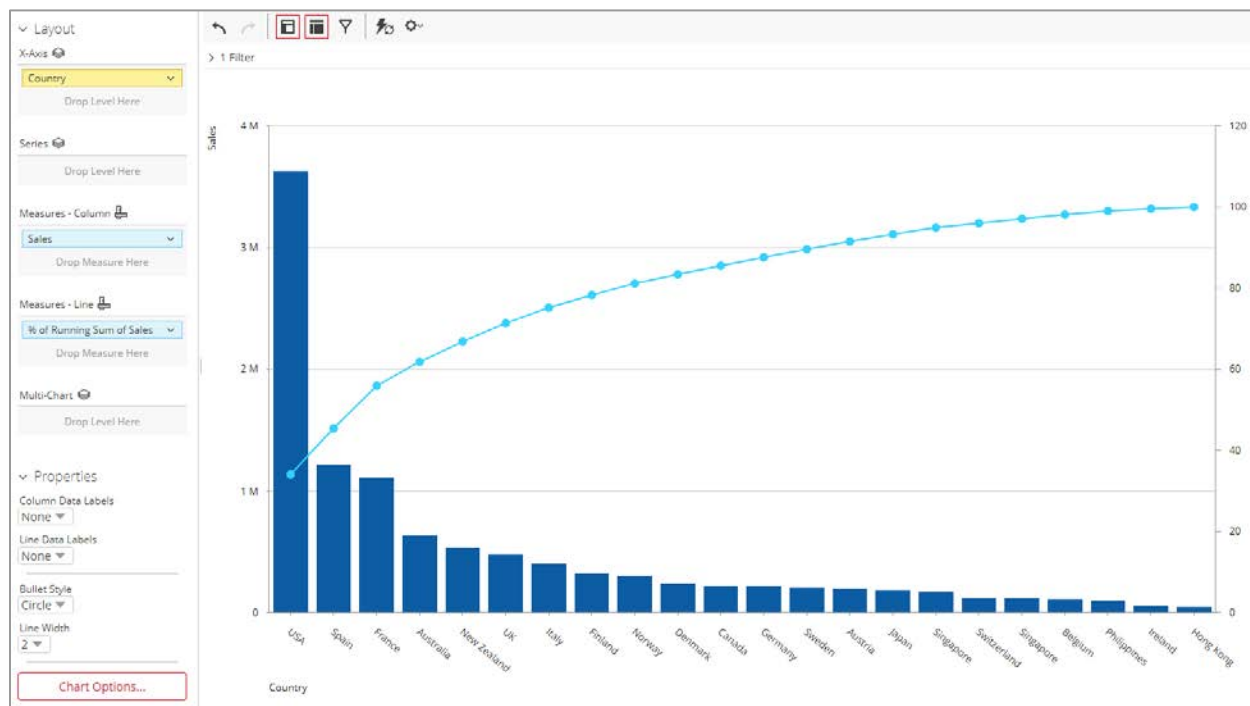


Figure 15: Pareto Chart

Create this Pareto chart by using a Column-Line Combo chart where the Measures – Column contains the Sales and the Measures – Line contains a % Running Sum of Sales (that is, a % Cumulative Sales). To do this, using the example above:

1. In the PUC, create a new Analysis Report and choose your data source.
2. Once the report has opened, on the top right, change the view to Chart by clicking here:

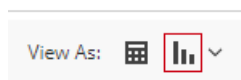
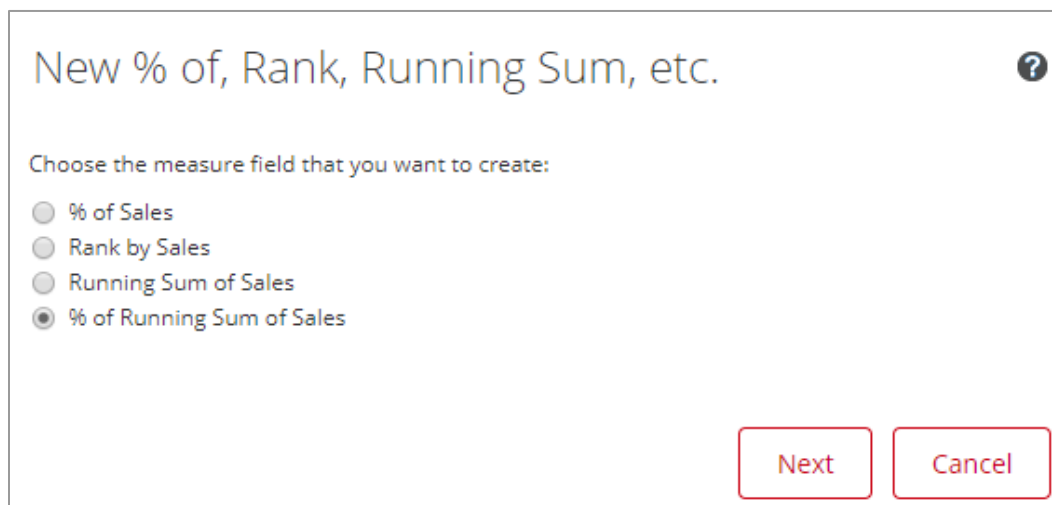


Figure 16: View As Chart

3. Click the drop-down next to the chart view icon, and change the chart type to **Column-Line Combo**.
4. Drag the **Country** market to the **X-Axis** of the chart, and the **Sales** measure to the **Measures - Column** section in **Layout**.

5. Create the % Running Sum of Sales with a user defined % of Running Sum measure:
 - a. Click the drop-down by **Sales** and choose **User Defined Measure > % of, Rank, Running Sum**.
 - b. Choose the measure field **% of Running Sum of Sales** and click **Next**:



New % of, Rank, Running Sum, etc. ?

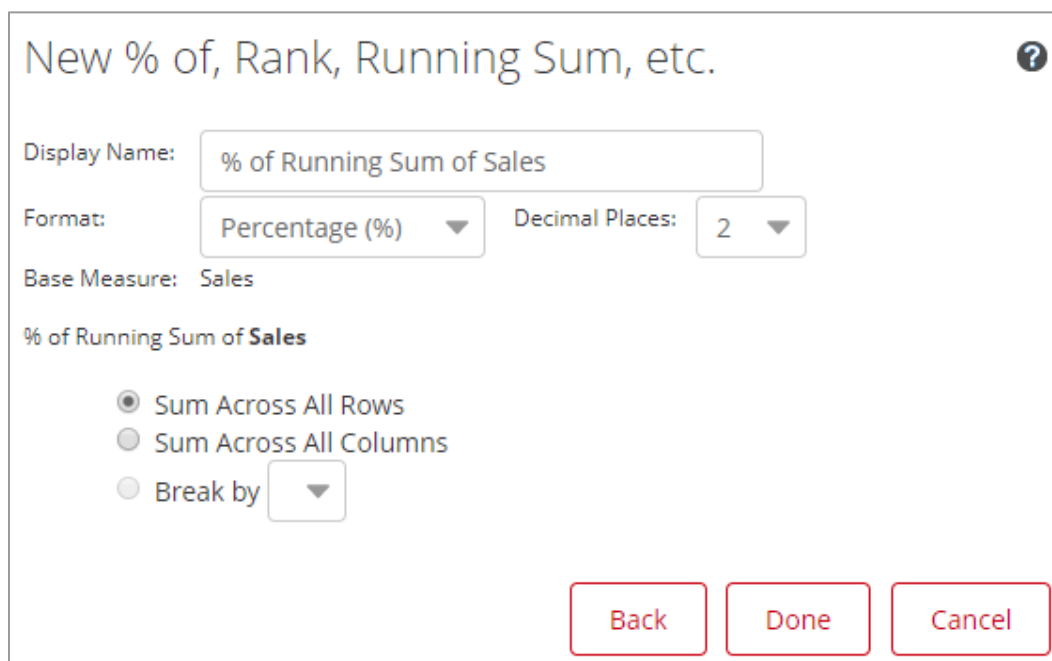
Choose the measure field that you want to create:

☐ % of Sales
☐ Rank by Sales
☐ Running Sum of Sales
☒ % of Running Sum of Sales

Next Cancel

Figure 17: Creating New Measure

6. Configure the measure as follows and click **Done**:



New % of, Rank, Running Sum, etc. ?

Display Name: % of Running Sum of Sales

Format: Percentage (%) Decimal Places: 2

Base Measure: Sales

% of Running Sum of Sales

☒ Sum Across All Rows
☐ Sum Across All Columns
☐ Break by ▼

Back Done Cancel

Figure 18: Configuring % Running Sum of Sales Measure

- Finally, order the chart values by clicking the drop-down by **Sales** and choosing **Sort Values High -> Low**:

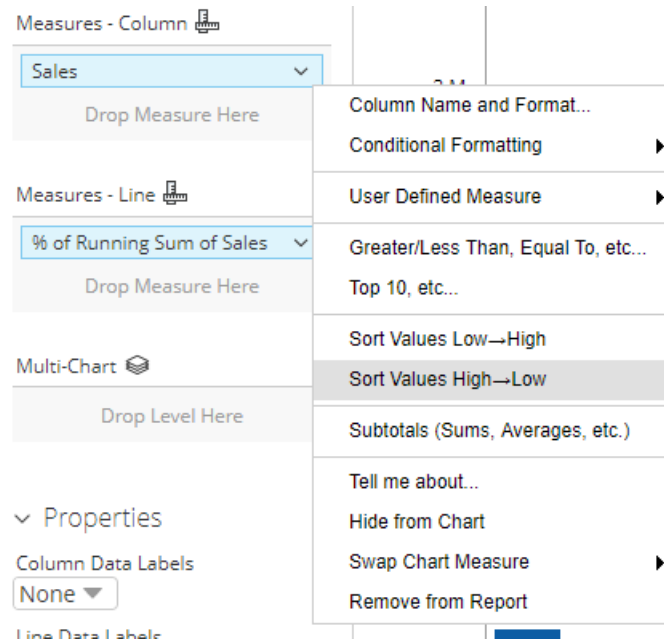


Figure 19: Sort Values

Creating a Moving Average Calculation

A moving average calculation calculates the current period by taking the average of N number of previous periods. This is useful for smoothing out a time series, as shown:

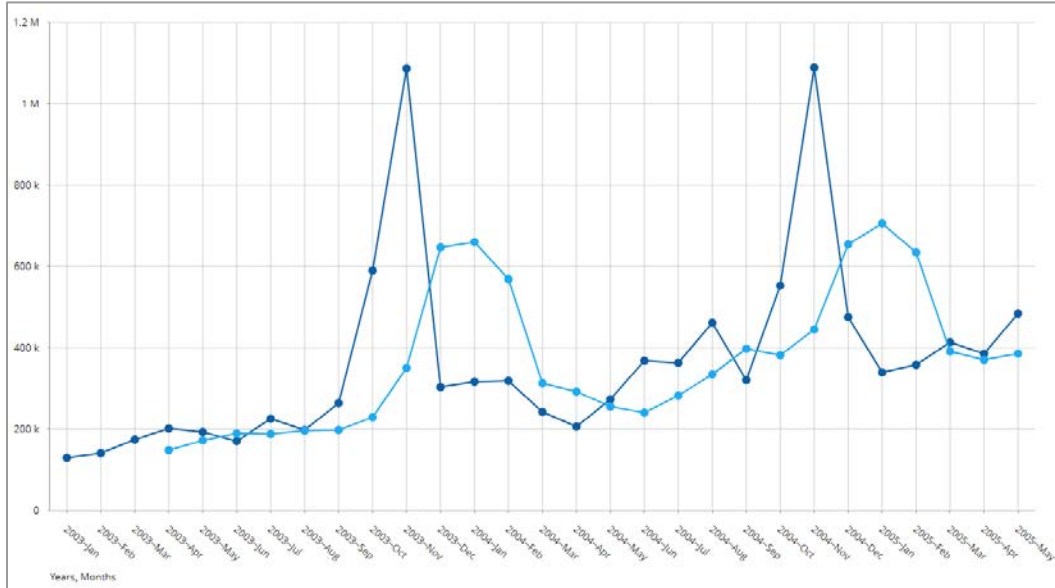


Figure 20: Moving Average Calculation

Create the moving average calculation by clicking **Sales** and choosing **User Defined Measure** and **Trend Measure**.

New Trend Measure

Display Name:

Trended Measure: Sales

Period type:
Months

Number of periods:

Show trend as:
Average of previous periods

Decimal Places:

Note: To trend, use time attributes only (e.g. Year, Quarter, Month).
To select a time attribute, you need to include one in the report.

Figure 21: Creating Trend Measure

Creating a Relative Date Filter (Current Year, Next N Years, N Years Ago)

When you are filtering on dates, you may want to use relative date filters instead of absolute date filters. Analyzer provides many filtering options.

These “commonly used time periods” are available only if you have set the `levelType` of time in the Mondrian schema, and `AnalyzerDateFormat` annotation.

Filter on Years

☒ Choose a commonly used time period
☐ Select from a list ([Includes](#), [Excludes](#))
☐ Select a range ([Between](#), [After](#), [Before](#))

☐ Current Years
☐ Previous Years
☐ Next Years
☐ Previous Years
☐ Next Years
☐ Years ago
☐ Years ahead

☐ Parameter Name

OK Cancel

Figure 22: Filter on Years

Further information on setting up relative date filters is available at [Enable Relative Date Filters in Mondrian](#) in the Pentaho documentation.

Aggregate Total Type

The aggregate total type is a special total type which aggregates the report details rows using the underlying relational measure aggregator. In the Mondrian schema file, relational measures can have the following aggregators: `sum`, `count`, `min`, `max`, `avg`, and `aggregate`.

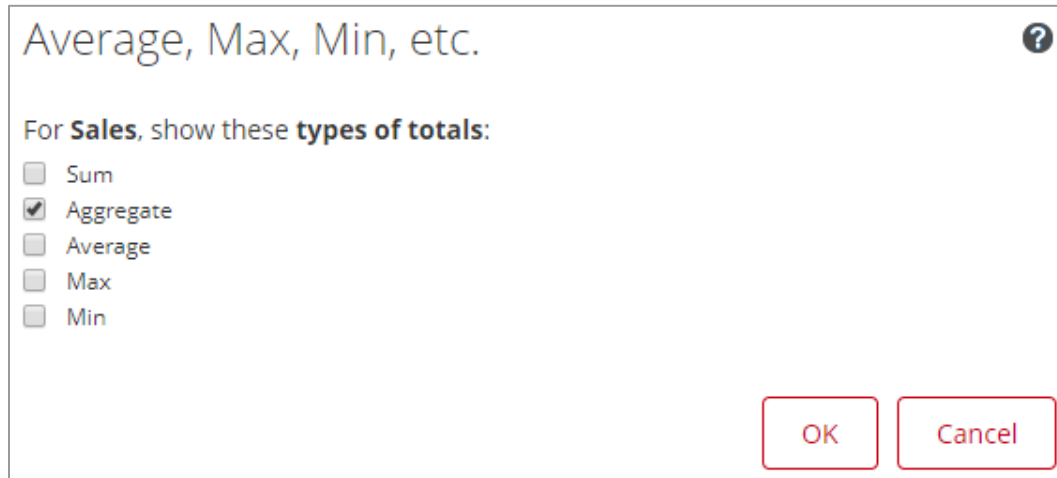


Figure 23: Average, Max, Min, etc.

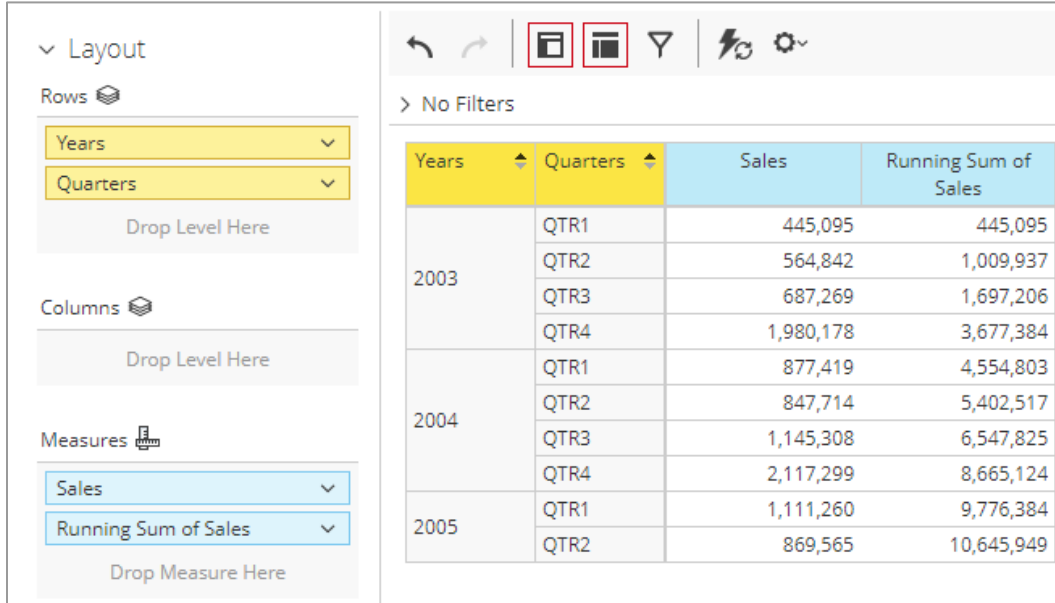
The aggregate total type is particularly useful for distinct-count subtotals, because the total needs to remove duplicates from report details rows.

Keep these things in mind when you are using calculated measures with aggregate totals:

- The solve order of the calculated measure must be greater than the aggregate total calculated member. The aggregate total calculated member always has a -100 solve order.
- When a calculated measure contains a ratio, the total will calculate a ratio of sums (assuming the numerator and denominator are both based on relational measure with an aggregator `sum`).
- A calculated measure like `[Sales]/100` will divide the total sales by 100 instead of summing up all details sales divided by 100.
- Be careful if your calculated measure MDX expression refers to dimension members coming from the same dimension as the total. The current evaluation context will most likely then be the `All` member instead of the members in the total.

Creating a Running Sum or Cumulative Calculation

A cumulative calculation is like a running sum, useful for continuously adding up a measure over time. This could be used for a Year-to-Date (YTD) calculation:

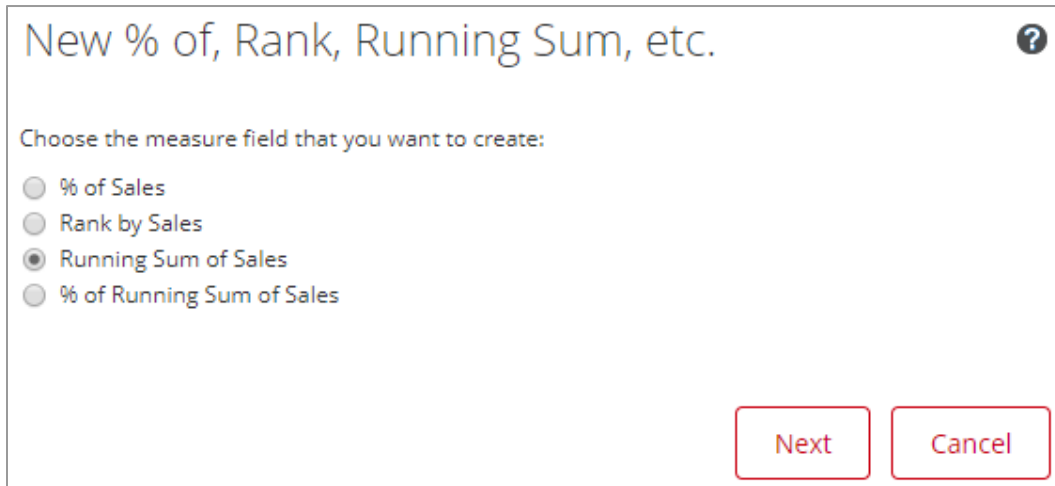


The screenshot shows the Pentaho Analyzer interface. On the left, the 'Layout' pane has 'Rows' with 'Years' and 'Quarters' selected, and 'Measures' with 'Sales' and 'Running Sum of Sales' selected. The main table displays data for 2003, 2004, and 2005, with columns for Years, Quarters, Sales, and Running Sum of Sales. The 'Running Sum of Sales' column shows cumulative values for each quarter.

Years	Quarters	Sales	Running Sum of Sales
2003	QTR1	445,095	445,095
	QTR2	564,842	1,009,937
	QTR3	687,269	1,697,206
	QTR4	1,980,178	3,677,384
2004	QTR1	877,419	4,554,803
	QTR2	847,714	5,402,517
	QTR3	1,145,308	6,547,825
	QTR4	2,117,299	8,665,124
2005	QTR1	1,111,260	9,776,384
	QTR2	869,565	10,645,949

Figure 24: Cumulative Calculation

The Running Sum of Sales was created by clicking **Sales** and selecting **User Defined Measure**, then **% of, Rank, Running Sum**.



The dialog box titled 'New % of, Rank, Running Sum, etc.' contains a section 'Choose the measure field that you want to create:' with four radio button options. The 'Running Sum of Sales' option is selected. At the bottom right, there are 'Next' and 'Cancel' buttons.

Choose the measure field that you want to create:

- ☐ % of Sales
- ☐ Rank by Sales
- ☒ Running Sum of Sales
- ☐ % of Running Sum of Sales

Next Cancel

Figure 25: New % of, Rank, Running Sum

New % of, Rank, Running Sum, etc. ?

Display Name:

Format: Decimal Places:

Base Measure: Sales

Running Sum of **Sales**

☒ Sum Across All Rows
☐ Sum Across All Columns
☐ Break by

Figure 26: Running Sum of Sales

If you want to reset the running sum for each year, choose the **Break by Years** radio button.

Percent Change Year Ago Calculation

A percent change year ago calculation will tell you the percent change between this period and the same period a year ago (YAGO).

This calculation will work regardless of whether you use quarters, months, weeks, or days, as long as the position of the period is the same in this year and the previous year.

Here is an example:

The screenshot shows the Pentaho Analyzer interface. On the left, the 'Layout' pane has three sections: 'Rows' with 'Years' and 'Months' selected, 'Columns' with 'Drop Level Here', and 'Measures' with 'Sales' and '% Change YAGO' selected. The main area shows a table with the following data:

Years	Months	Sales	% Change YAGO
2005	Jan	339,543	7.3%
	Feb	358,186	12.4%
	Mar	413,531	70.8%
	Apr	385,529	87.0%
	May	484,036	77.0%

Figure 27: % Change YAGO Calculation

For example, February is calculated as $\left(\frac{\text{Sales in February 2005}}{\text{Sales in February 2004}}\right) - 1$.

Create this calculation by clicking **Sales**, and choosing **User Defined Measure** and **Trend Measure**.

The 'New Trend Measure' dialog box is shown with the following fields and values:

- Display Name: % Change YAGO
- Trended Measure: Sales
- Period type: Years
- Number of periods: 1
- Show trend as: % of change from previous period
- Decimal Places: 1

Note: To trend, use time attributes only (e.g. Year, Quarter, Month). To select a time attribute, you need to include one in the report.

Buttons: OK, Cancel

Figure 28: Creating % Change YAGO Trend Measure

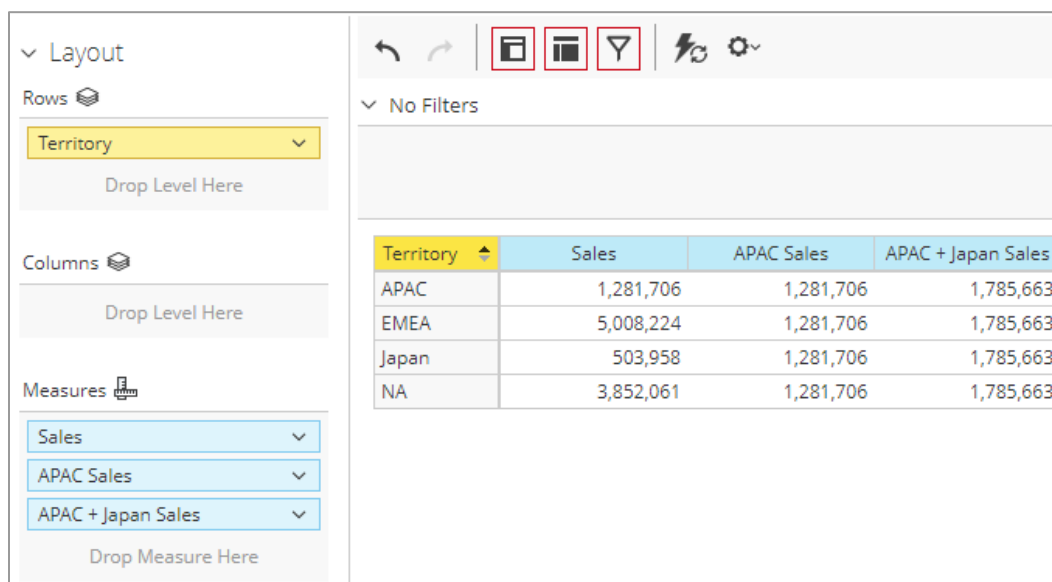
Advanced Recipes

In this section, you will find information on use cases that require an understanding of MDX expressions and how they are evaluated. This section covers topics such as:

- [Pinning a Measure to Specific Dimension Measures](#)
- [Pinning a Measure to a Specific Hierarchy Level](#)
- [Creating Semi-Additive Measures](#)
- [Creating a Conditional Measure](#)
- [Comparing YTD for Current and Previous Year](#)
- [Generating Trend Lines Using Linear Regression](#)
- [Conditionally Highlighting of Formatting Cells](#)
- [Filtering on a Member Property](#)
- [Adding Target or Goal Measures to a Report](#)
- [Calculating a Weighted Average](#)
- [How Solve Orders Affect Calculated Measures](#)

Pinning a Measure to Specific Dimension Measures

A measure that is pinned to specific dimension measures will always be evaluated by those pinned members, regardless of what is on the rows or columns. For example, this report below has APAC Sales and APAC + Japan Sales pinned to their respective countries:



The screenshot shows the Pentaho Analyzer interface. On the left, the 'Layout' pane has 'Territory' in the Rows section and 'Sales', 'APAC Sales', and 'APAC + Japan Sales' in the Measures section. The main report area shows a table with the following data:

Territory	Sales	APAC Sales	APAC + Japan Sales
APAC	1,281,706	1,281,706	1,785,663
EMEA	5,008,224	1,281,706	1,785,663
Japan	503,958	1,281,706	1,785,663
NA	3,852,061	1,281,706	1,785,663

Figure 29: APAC Sales and APAC + Japan Sales

APAC Sales is pinned to APAC:

APAC Sales Properties

Display Name:

Format: Decimal Places:

Type:

MDX:

Available Measures:

- Sales
- APAC Sales
- APAC + Japan Sales
- APAC Sales
- Quantity
- Sales

MDX Formula:

Subtotal Calculation

☐ Calculate subtotals using measure formula

Buttons: +, -, *, /, (,), Clear

Buttons: OK, Cancel

Figure 30: Pinning APAC Sales to APAC

For APAC + Japan Sales, use the Sum function, since there is more than one territory:

```
Sum( {[Markets].[APAC],[Markets].[Japan]}, [Measures].[Sales])
```

Pinning a Measure to a Specific Hierarchy Level

When you pin a measure to a specific hierarchy level, that measure will always roll up to that level, regardless of what levels are on the report. This type of measure is sometimes known as a level based measure.

For example, in SteelWheels, the Markets dimension has a hierarchy with Territory → Country → State → City levels. This example shows a measure called Country Sales which is pinned to the Country level:

Layout

Rows

Territory

Country

State Province

Drop Level Here

Columns

Drop Level Here

Measures

Sales

Country Sales

Drop Measure Here

> 1 Filter

Territory	Country	State Prov...	Sales	Country Sales
NA	Canada	BC	149,874	224,079
		Québec	74,205	224,079
	Canada Total		224,079	224,079
	USA	CA	1,505,542	3,627,983
		CT	238,661	3,627,983
		MA	666,444	3,627,983
		NH	131,685	3,627,983
		NJ	83,228	3,627,983
		NV	82,751	3,627,983
		NY	646,344	3,627,983
PA		273,327	3,627,983	
USA Total		3,627,983	3,627,983	

Figure 31: Country Sales Measure

The Country Sales measure is defined as follows:

Create Calculated Measure

Display Name: **Country Sales**

Format: **Default** Decimal Places: **2**

Type: **Calculated Measure**

Sales
Quantity
Sales

>

```

([Measures].[Sales], Ancestor([Markets].CurrentMember,
[Markets].[Country]))

```

Subtotal Calculation

☒ Calculate subtotals using measure formula

☐ Apply to data source

Figure 32: Creating the Country Sales Measure

If your report contained only Territory, then you would get no data, because there is no ancestor above any Territory member that is at the Country level.

If, however, you just wanted to return `Sales` for that `Territory`, you could use the expression in the following figure:

Create Calculated Measure

Display Name: **Country Sales**

Format: **Default** Decimal Places: **2**

Type: **Calculated Measure**

Sales

 Quantity
 Sales

([Measures].[Sales],
 If([Markets].CurrentMember.Level.Ordinal > 2,
 Ancestor([Markets].CurrentMember,[Markets].
 [Country]), [Markets].CurrentMember))

Subtotal Calculation
☐ Calculate subtotals using measure formula
☐ Apply to data source

+ - * / () Clear

OK Cancel

Figure 33: Calculating Sales for a Territory

In this expression, you can check the `Markets` current member's level ordinal to determine what level is on the report, and then conditionally pick the `Country` ancestor or the current member.



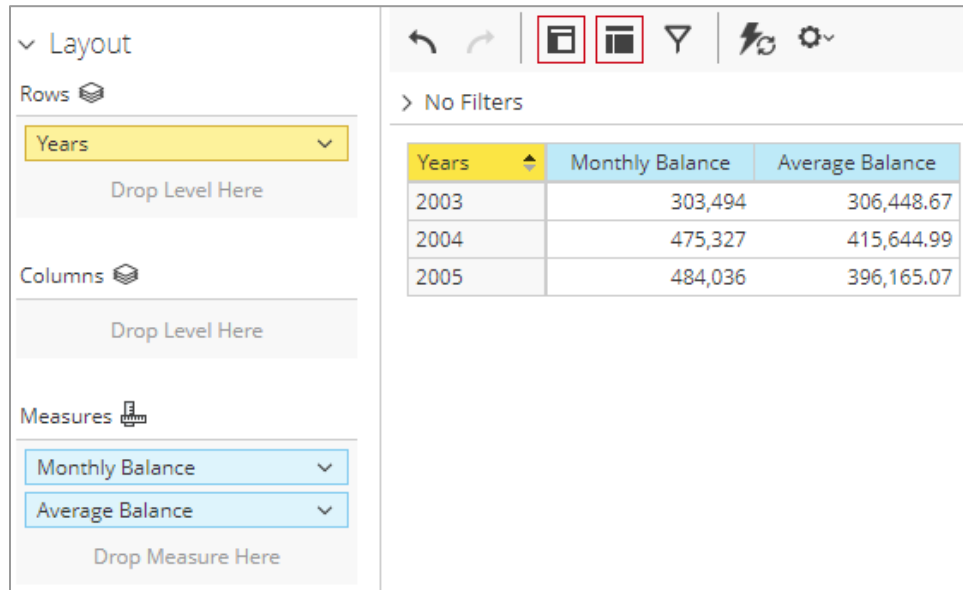
Remember that ordinal 0 is the `All` level, ordinal 1 is the `Territory` level, ordinal 2 is the `Country` level, and so forth.

Creating Semi-Additive Measures

A semi-additive measure is a measure which does not uniformly aggregate across all dimensions within a cube. A common example would be a snapshot of account balances over time, such as bank account balances or inventory level balances. It would not make sense to add up all monthly bank account balances within a year. Instead, you would define whether you wanted the last account balance or the average account balance in the year.

There are many different types of semi-additive functionality, including `first child`, `last child`, `count`, `min`, `max`, and `average`.

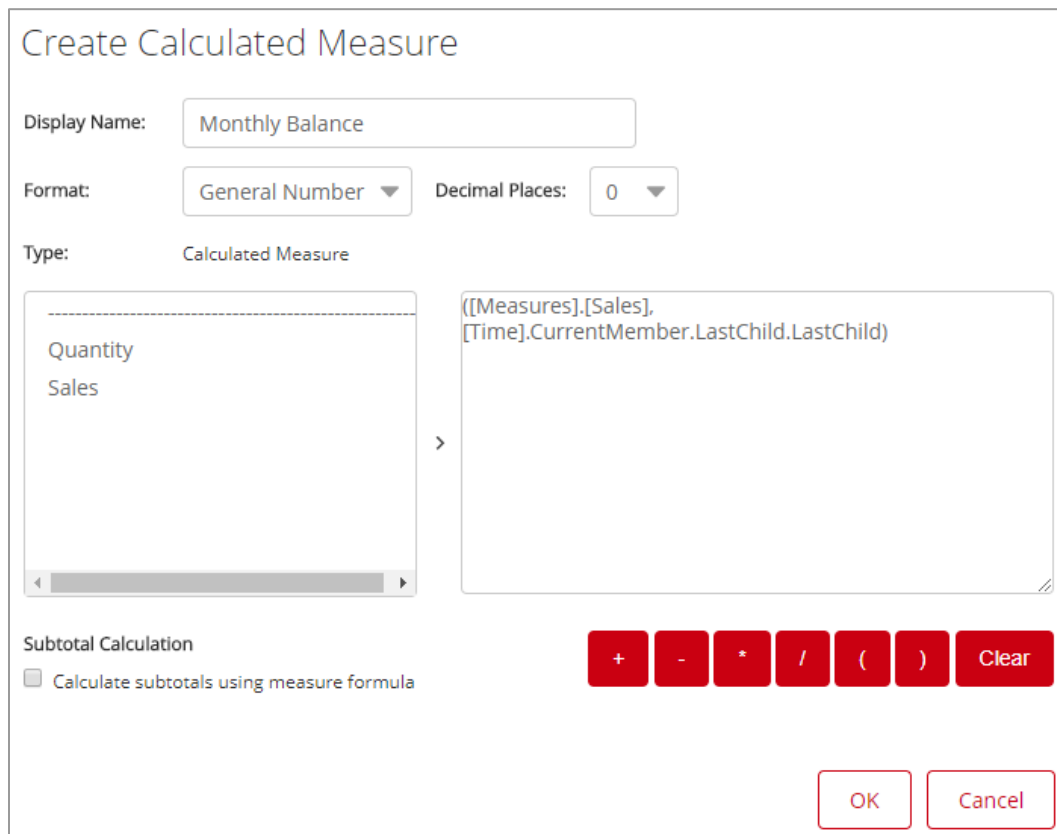
In this example, use these monthly snapshots of an account balance:



Years	Monthly Balance	Average Balance
2003	303,494	306,448.67
2004	475,327	415,644.99
2005	484,036	396,165.07

Figure 34: Monthly Account Balances

Monthly Balance shows the last month for the year, while Average Balance shows the average balance for the year. Create the user defined measure Monthly Balance this way:



Create Calculated Measure

Display Name:

Format: Decimal Places:

Type:

Quantity
Sales

>

([Measures].[Sales],
[Time].CurrentMember.LastChild.LastChild)

Subtotal Calculation
☐ Calculate subtotals using measure formula

+ - * / () Clear

OK Cancel

Figure 35: Creating Monthly Balance Measure

This MDX expression assumes that the current time member is at the `Year` level, and that we can directly ask for the last child twice to go from `Year` → `Quarter` → `Month`.

A more robust way to get the last month balance is with this MDX expression:

```
([Measures].[Sales], Tail(Descendants([Time].CurrentMember, [Time].[Months]), 1).Item(0))
```

This expression will work regardless of whether you have `Years` or `Quarters` on the report, because it will use the `Dscendants` function to get all `Month` level descendants for the current `Year` or `Quarter`. It uses the `Tail` function to get the last item in the set.

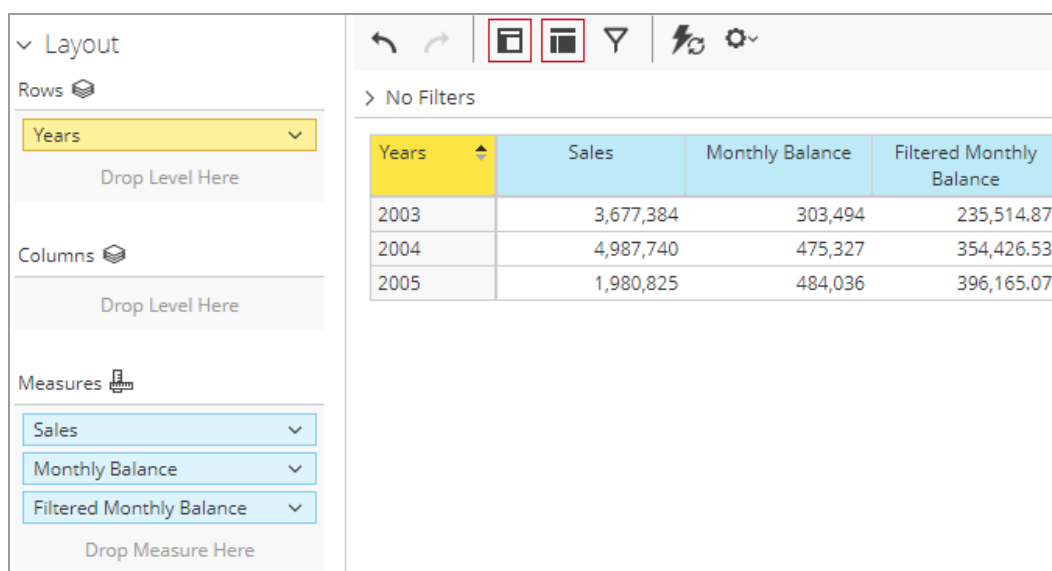
Create the `Average Balance` as follows:

Figure 36: Creating `Average Balance Measure`

You can easily replace the `Avg` with `Min`, `Max`, `Sum`, or `Count` to get a different semi-additive behavior.

Sometimes, you might need to filter the set that goes into the semi-additive calculation. For example, if you added `Month` onto the above report, you would notice that November had much higher sales than the other months. If this is an outlier and you want to exclude that month by considering only months with balances less than 1,000,000, you can achieve that with the `Filter` MDX function by creating a calculated measure using this code:

```
Avg(Filter(Descendants([Time].CurrentMember, [Time].[Months]), [Measures].[Sales] < 1000000), [Measures].[Sales])
```



The screenshot shows the Pentaho Analyzer interface. On the left, the 'Layout' pane has 'Years' in the Rows section and 'Sales', 'Monthly Balance', and 'Filtered Monthly Balance' in the Measures section. The main table area shows a table with 4 columns: 'Years', 'Sales', 'Monthly Balance', and 'Filtered Monthly Balance'. The table has 3 rows of data for the years 2003, 2004, and 2005. The 'Filtered Monthly Balance' column shows values for each year, indicating that the filter is applied across the entire year.

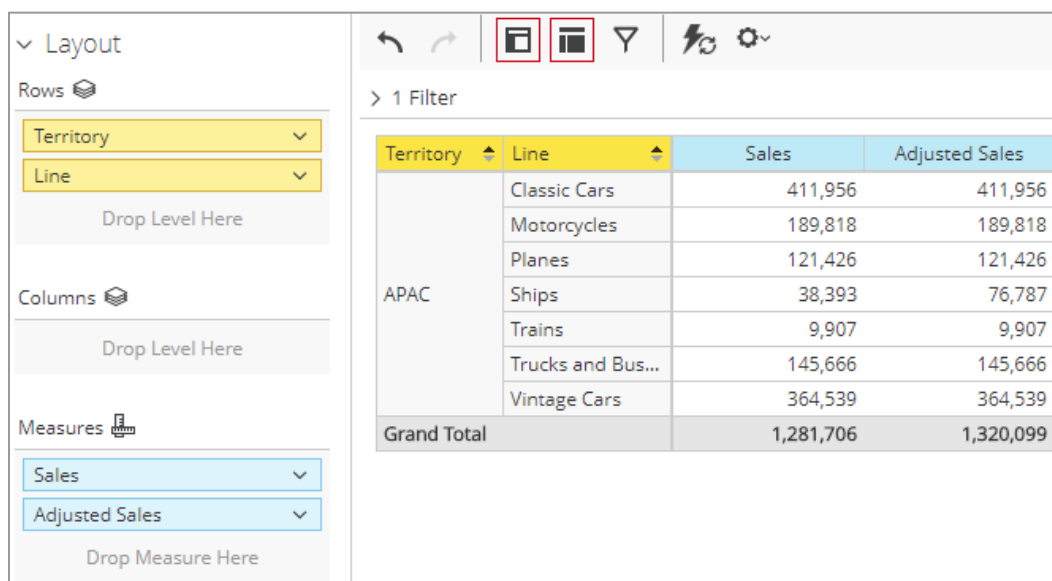
Years	Sales	Monthly Balance	Filtered Monthly Balance
2003	3,677,384	303,494	235,514.87
2004	4,987,740	475,327	354,426.53
2005	1,980,825	484,036	396,165.07

Figure 37: Monthly Balances (Filtered)

Another example of something you could do with the `Filter` MDX function is to find the first month that does have a balance, if the last month of the year does not have a balance.

Creating a Conditional Measure

A conditional measure is any measure which has `if-else` logic, like `case` statements in SQL. In this example, a user defined measure increases only `Ships` sales by 100%.



The screenshot shows the Pentaho Analyzer interface. On the left, the 'Layout' pane has 'Territory' and 'Line' in the Rows section and 'Sales' and 'Adjusted Sales' in the Measures section. The main table area shows a table with 4 columns: 'Territory', 'Line', 'Sales', and 'Adjusted Sales'. The table has 7 rows of data for the APAC territory, including 'Classic Cars', 'Motorcycles', 'Planes', 'Ships', 'Trains', 'Trucks and Bus...', and 'Vintage Cars'. The 'Adjusted Sales' column shows values for each line item, indicating that the conditional measure is applied to the 'Ships' line item.

Territory	Line	Sales	Adjusted Sales
APAC	Classic Cars	411,956	411,956
	Motorcycles	189,818	189,818
	Planes	121,426	121,426
	Ships	38,393	76,787
	Trains	9,907	9,907
	Trucks and Bus...	145,666	145,666
	Vintage Cars	364,539	364,539
Grand Total		1,281,706	1,320,099

Figure 38: Conditional Measure

Adjusted Sales is defined as follows:

Create Calculated Measure

Display Name:

Format: Decimal Places:

Type:

>

```
IIf(
[Product].CurrentMember.Name = 'Ships',
[Measures].[Sales]*2,
[Measures].[Sales])
```

Subtotal Calculation

☐ Calculate subtotals using measure formula

☐ Apply to data source

Figure 39: Adjusted Sales Measure

You can use AND/OR logic as well as nest the IIF function to create more interesting conditions.

In the above example, Product Line was on the report, so you could just conditionally check the current Product Line member.

However, if `Product Line` was not on the report, you could obtain the same results by explicitly iterating over all `Product Line` members and then applying your conditions. Create the arrows by setting the **Trend Arrow** option:

Layout
Rows
Territory
Drop Level Here
Columns
Drop Level Here
Measures
Sales
Adjusted Sales
Change
Drop Measure Here

No Filters

Territory	Sales	Adjusted Sales	Change
APAC	1,281,706	1,320,099	3.00% ↗
EMEA	5,008,224	5,435,489	8.53% ↗
Japan	503,958	536,973	6.55% ↗
NA	3,852,061	4,102,059	6.49% ↗
Grand Total	10,645,949	11,394,620	7.03%

Figure 40: Trend Arrows

Adjusted Sales will iterate over all Product Line members and check to see if the current Product Line is Ships. If it is, Adjusted Sales will increase sales by 100%:

Create Calculated Measure

Display Name:

Format: Decimal Places:

Type:

Selected Measures: Sales

Formula:

```
Sum([Product].[Line].Members,
If(
[Product].CurrentMember.Name = 'Ships',
[Measures].[Sales]*2,
[Measures].[Sales])
)
```

Subtotal Calculation

☐ Calculate subtotals using measure formula

☐ Apply to data source

Buttons: +, -, *, /, (,), Clear, OK, Cancel

Figure 41: Adjusted Sales Measure Iterating over all Product Lines

The Change measure is just another user defined measure:

$$\left(\frac{\text{Adjusted Sales}}{\text{Sales}} \right) - 1$$

Be sure to check the **Calculate subtotals using measure formula** so that you get a ratio of sums, and not a sum of ratios.

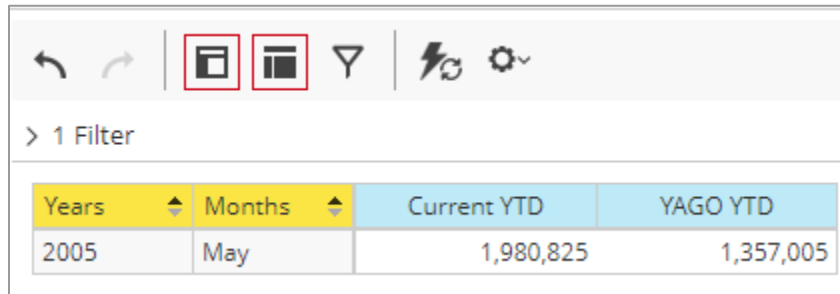
Comparing YTD for Current and Previous Year

When looking at YTD for the current year, you may want to compare it to the YTD for the previous year.



When calculating YTD for the previous year, include only the previous year's dates up to the current year's same date.

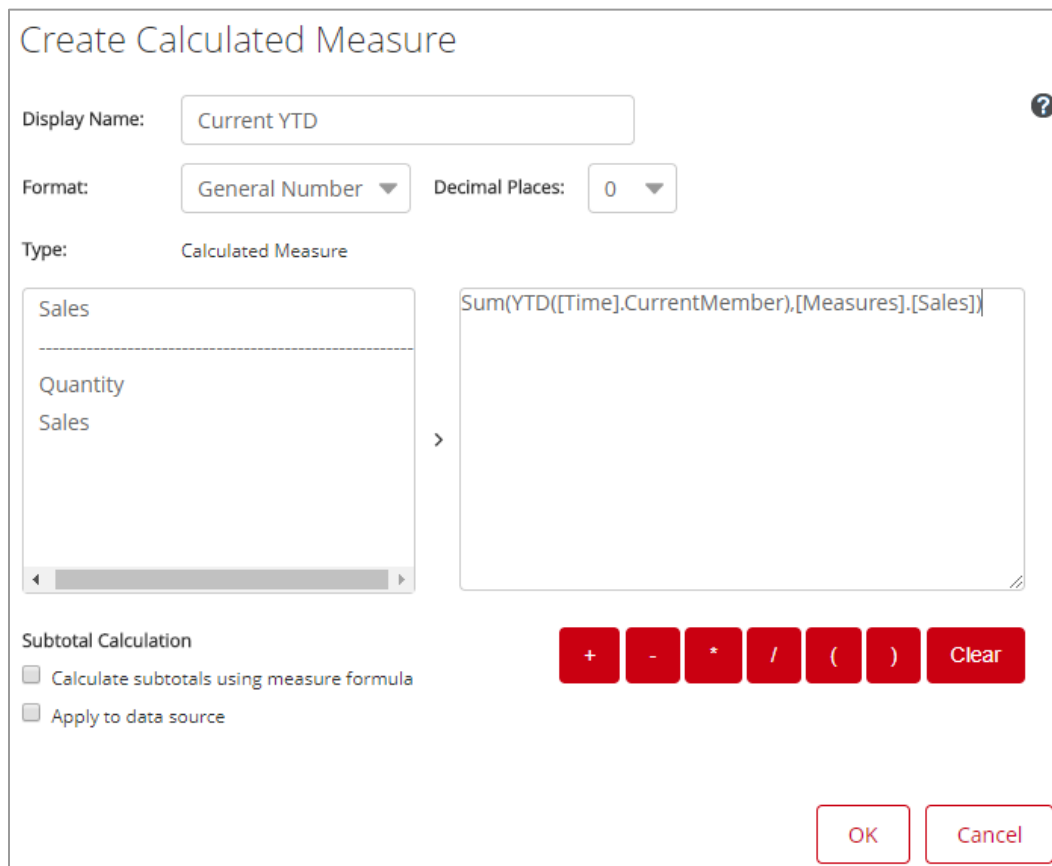
In this example, use May 2005 YTD sales. For the YAGO YTD, add up January 2004 to May 2004, so you can make a valid YTD comparison between the two years.



Years	Months	Current YTD	YAGO YTD
2005	May	1,980,825	1,357,005

Figure 42: YAGO YTD

Set up the Current YTD user defined measure:



Create Calculated Measure

Display Name:

Format: Decimal Places:

Type:

Sales

Quantity

Sales

Sum(YTD([Time].CurrentMember),[Measures].[Sales])

Subtotal Calculation

☐ Calculate subtotals using measure formula

☐ Apply to data source

OK Cancel

Figure 43: Current YTD Measure

For the YTD, MTD, QTD, and WTD functions to work, you must define the `LevelType` attribute on the time levels in the Mondrian schema.

You can also get the same date member by using the `PeriodsToDate` function. This expression is equivalent to the above YTD calculation:

```
Sum(PeriodsToDate([Time].[Years], [Time].CurrentMember), [Measures].[Sales])
```

The YAGO YTD calculation is just based on a trend measure:

New Trend Measure

Display Name:

Trended Measure:

Period type:

Number of periods:

Show trend as:

Decimal Places:

**Note: To trend, use time attributes only (e.g. Year, Quarter, Month).
To select a time attribute, you need to include one in the report.**

Figure 44: YAGO YTD Trend Measure

As an MDX exercise, you can also achieve the same YAGO YTD result with this MDX expression:

```
Sum(YTD(ParallelPeriod([Time].[Years],
1,[Time].CurrentMember)), [Measures].[Sales])
```

In this expression, you are using the `ParallelPeriod` function to get the same month a year ago. The `CurrentMember` refers to May 2005, and the same month one year ago would be May 2004.

Generating Trend Lines Using Linear Regression

You can add trend lines to a line chart by fitting a line to your data. For example, this chart shows the least squares regression of Sales on Months:

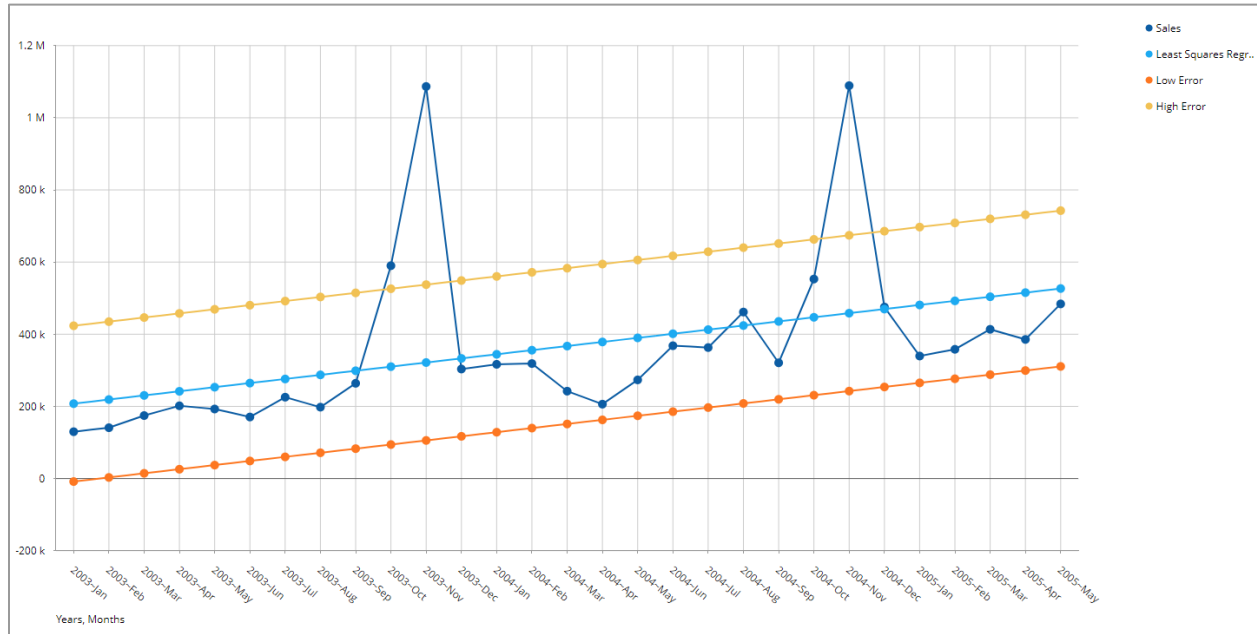


Figure 45: Least Squares Regression of Sales on Months

The Sales measure is the actual sales amount in each month. A least squares regression uses the `LinRegPoint` function to predict Sales, given the explanatory variable Month.

You can also calculate residuals and the standard error of regression, which is the standard deviation of the residuals. Because these residuals have a bell-shaped distribution around the fitted line, you can infer that two-thirds of the actual sales were within +/- one standard deviation of the fitted line. This +/- boundary is marked by the High and Low Error lines.

Here is the underlying data for the chart:

Years	Months	Sales	Least Squares Regression	Residuals	Standard Error of Regression	Low Error	High Error
2003	Jan	129,754	207,541	-77,787	215,940	-8,399	423,481
	Feb	140,836	218,938	-78,102	215,940	2,998	434,879
	Mar	174,505	230,335	-55,831	215,940	14,395	446,276
	Apr	201,610	241,733	-40,123	215,940	25,792	457,673
	May	192,673	253,130	-60,457	215,940	37,190	469,070
	Jun	170,559	264,527	-93,968	215,940	48,587	480,467
	Jul	225,486	275,924	-50,438	215,940	59,984	491,864
	Aug	197,809	287,321	-89,512	215,940	71,381	503,262
	Sep	263,973	298,719	-34,745	215,940	82,778	514,659
	Oct	589,964	310,116	279,848	215,940	94,176	526,056
	Nov	1,086,720	321,513	765,207	215,940	105,573	537,453
	Dec	303,494	332,910	-29,416	215,940	116,970	548,850
2004	Jan	316,577	344,307	-27,730	215,940	128,367	560,248
	Feb	318,699	355,705	-37,006	215,940	139,764	571,645
	Mar	242,143	367,102	-124,959	215,940	151,161	583,042
	Apr	206,148	378,499	-172,351	215,940	162,559	594,439
	May	273,438	389,896	-116,458	215,940	173,956	605,836
	Jun	368,127	401,293	-33,166	215,940	185,353	617,233
	Jul	363,056	412,690	-49,635	215,940	196,750	628,631
	Aug	461,501	424,088	37,414	215,940	208,147	640,028
	Sep	320,751	435,485	-114,734	215,940	219,545	651,425
	Oct	552,924	446,882	106,042	215,940	230,942	662,822
	Nov	1,089,048	458,279	630,769	215,940	242,339	674,219
	Dec	475,327	469,676	5,650	215,940	253,736	685,617
2005	Jan	339,543	481,074	-141,530	215,940	265,133	697,014
	Feb	358,186	492,471	-134,285	215,940	276,530	708,411
	Mar	413,531	503,868	-90,337	215,940	287,928	719,808
	Apr	385,529	515,265	-129,736	215,940	299,325	731,205
	May	484,036	526,662	-42,626	215,940	310,722	742,603

Figure 46: Data for Least Squares Regression

Create Least Squares Regression as a user defined measure:

Create Calculated Measure

Display Name:

Format: Decimal Places:

Type:

Measures List: Sales, Quantity, Sales

MDX Expression: `LinRegPoint(Rank([Time].CurrentMember, [Time].CurrentMember.Level.Members), ([Time].CurrentMember.Level.Members), [Measures].[Sales], Rank([Time].CurrentMember, [Time].CurrentMember.Level.Members))`

Subtotal Calculation:

☐ Calculate subtotals using measure formula

☐ Apply to data source

Buttons: +, -, *, /, (,), Clear, OK, Cancel

Figure 47: Defining Least Squares Regression

Calculate Residuals with this MDX expression:

```
[Measures].[Sales]-[Least Squares Regression]
```

Calculate the Standard Error of Regression:

```
Power(
Sum([Time].CurrentMember.Level.Members, Power([Residuals], 2))
/ (Count([Time].CurrentMember.Level.Members)-2)
, .5)
```

Calculate Low Error with this MDX expression:

```
[Least Squares Regression]-[Standard Error of Regression]
```

Calculate High Error with this MDX expression:

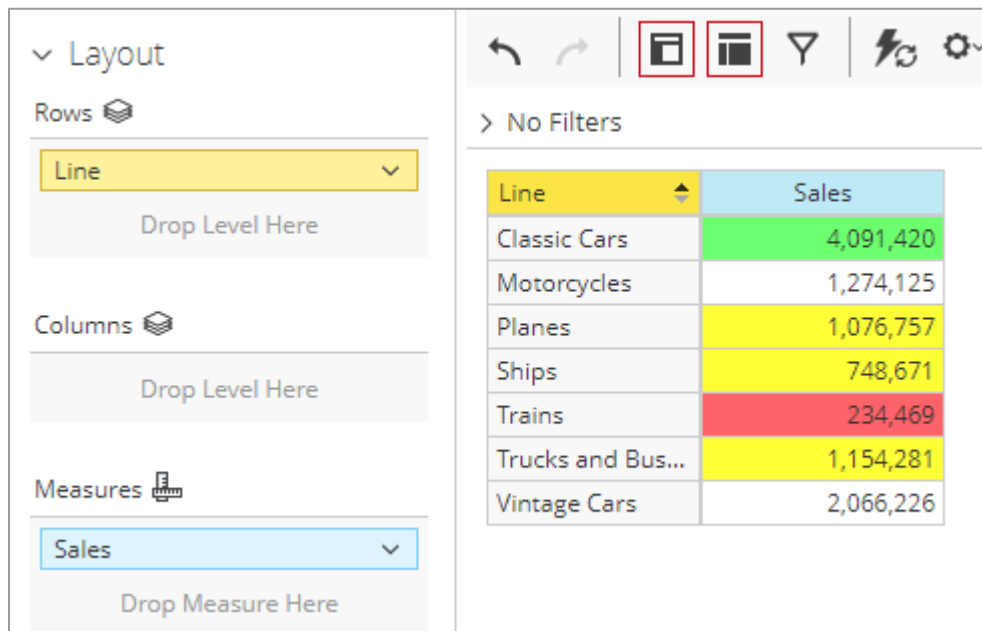
```
[Least Squares Regression]+[Standard Error of Regression]
```

Conditionally Highlighting or Formatting Cells

You can conditionally format cells based on the measure value or the current dimension members intersecting on the current cell, using an MDX format expression.

In the next report, color cells based on the following rules:

- Sales < 250,000 then color Red
- Sales < 1,250,000 then color Yellow
- Sales > 4,000,000 then color Green



Line	Sales
Classic Cars	4,091,420
Motorcycles	1,274,125
Planes	1,076,757
Ships	748,671
Trains	234,469
Trucks and Bus...	1,154,281
Vintage Cars	2,066,226

Figure 48: Line and Sales

Define the MDX expression as follows:

```

Case
When [Measures].CurrentMember < 250000
Then ' |#,##0|style=red'
When [Measures].CurrentMember < 1250000
Then ' |#,##0|style=yellow'
When [Measures].CurrentMember > 4000000
Then ' |#,##0|style=green'
Else ' |#,##0'
End

```

Edit Column ?

Display Name:

Original Name: Sales

Format: Expression ▼

MDX Format Expression:

```
Case
When [Measures].CurrentMember < 250000
Then '|#,##0|style=red'
When [Measures].CurrentMember < 1250000
Then '|#,##0|style=yellow'
When [Measures].CurrentMember > 4000000
Then '|#,##0|style=green'
End Case
```

OK Cancel

Figure 49: Defining Format Expression

The Case MDX expression must return a valid format string such as:

```
|#,##0|style=red
```

Here are some useful format strings:

- General Number: `#,##0.0`
- Currency: `$#,##0.00`
- Percentage: `###0.00%`

Analyzer supports the following additional formatting properties:

- `style=red|yellow|green` - Highlights the cell with the given color. Only these three colors are supported.
- `image=icon.gif` - Shows an icon in the cell next to the cell value. This icon must be placed in `system/analyzer/images/report/`.
- `arrow=up|down|none` - Shows an up arrow/down arrow next to the cell value.
- `link=http://www.pentaho.com` - Makes the cell value a hyperlink which opens the indicated URL.
- `color=#FFFFFF` - Sets the background color using an RGB color in Hex value.

Here is an example of how this additional formatting could look:

The screenshot shows the Pentaho Analyzer interface. On the left, the 'Layout' pane has 'Line' in the Rows section and 'Sales' in the Measures section. On the right, a table is displayed with the following data:

Line	Sales
Classic Cars	4,091,420
Motorcycles	1,274,125
Planes	1,076,757
Trains	234,469

The 'Planes' row is highlighted with a light blue background, and its 'Sales' value is a blue hyperlink. The 'Classic Cars' row has a green background for the 'Sales' cell. The 'Trains' row has a small bar chart icon next to its value.

Figure 50: Additional Formatting

By referencing the dimension members on the current evaluation context of the cell, you can also parameterize the format expressions.

For example, you could use an expression to create a link to search on the customer's address, which is exposed as a `Customer` member property.

The format MDX expression for `Sales` would then look like:

```
'|#,##0| "link=http://www.google.com/search?query=' ||
[Customers].CurrentMember.Properties("Address") || "'
```

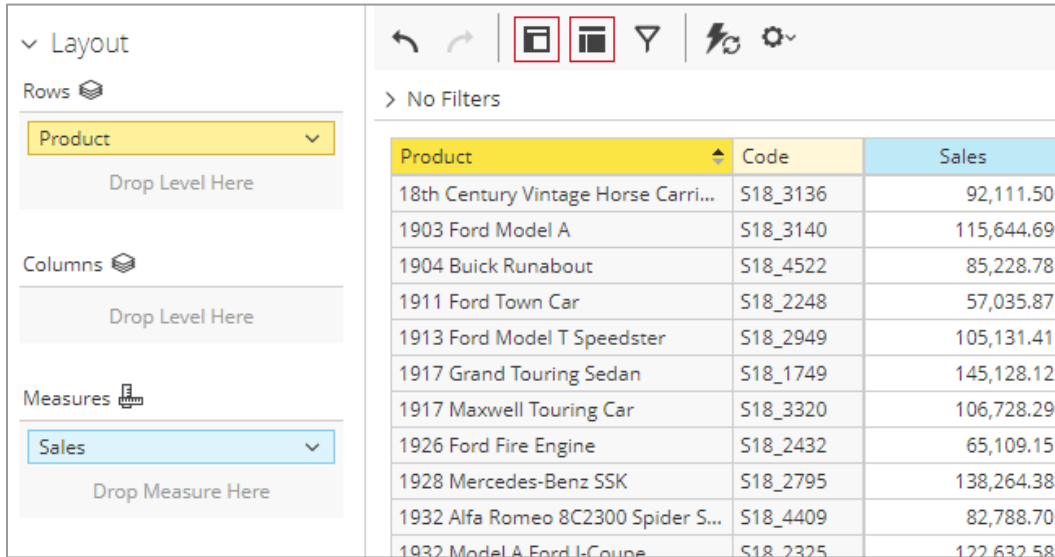
There are a few more things to note:

- The MDX format expression is first evaluated on the current cell which includes the measure and all dimension members that intersect on the cell.
- You can use `IIF` and `CASE` MDX functions to conditionally return a format string.
- The format string is used to format the cell value. If the formatted value takes the form of: `I<formatted value>|<property name>=<property value>`, additional formatting such as the background color or hyperlink is added to the formatted value.
- Use single quotes to escape string literals in the MDX expression.
- Use double quotes to escape string literals in the format string.
- A double quote is used within the single quoted string literal to escape the double quoted string literal within the format string. In the above example, this saves you the trouble of having to escape special formatting characters in the `Address` member property.
- `||` is used to concatenate strings in MDX expressions
- Refer to [Mondrian documentation](#) for more MDX functions.
- Refer to [MSDN documentation](#) for more information on constructing format strings.

Filtering on a Member Property

Members in a level can have additional member properties such as product codes or IDs. Normally, you cannot filter directly on these member properties, since these properties do not show up as fields. However, you can work around this by creating a user defined measure that uses the `Properties` MDX function to retrieve a member property and then filter on it.

In this example, filter on the `Product Code` member property to include only products with codes that start with S18:



Product	Code	Sales
18th Century Vintage Horse Carriage	S18_3136	92,111.50
1903 Ford Model A	S18_3140	115,644.69
1904 Buick Runabout	S18_4522	85,228.78
1911 Ford Town Car	S18_2248	57,035.87
1913 Ford Model T Speedster	S18_2949	105,131.41
1917 Grand Touring Sedan	S18_1749	145,128.12
1917 Maxwell Touring Car	S18_3320	106,728.29
1926 Ford Fire Engine	S18_2432	65,109.15
1928 Mercedes-Benz SSK	S18_2795	138,264.38
1932 Alfa Romeo 8C2300 Spider S...	S18_4409	82,788.70
1932 Model A Ford Coupe	S18_2325	122,632.58

Figure 51: Filtering on Product Code

Create the user defined measure `Sales` as follows:

Create Calculated Measure

Display Name:

Format: Decimal Places:

Type:

Selected Measures:

- Sales
- Quantity
- Sales

MDX Formula:

```
IIF(
[Product].CurrentMember.Properties('Code')
MATCHES 'S18_*',
[Measures].[Sales], null)
```

Subtotal Calculation

☐ Calculate subtotals using measure formula

☐ Apply to data source

Operators: +, -, *, /, (,), Clear

Buttons: OK, Cancel

Figure 52: Measure for Finding Specific Product Codes

The `MATCHES` MDX function is a Mondrian-specific function that uses a [Java regular expression](#). Notice that when the filter does not match, `null` is returned. Since the report does not show `null` cells, those products that don't meet this filter will not be shown on the report. Note also that the `Grand Total` will be computed correctly.

Adding Target or Goal Measures to a Report

Often, you want to compare actual versus previously set targets or goals. These goals may be stored directly in the report so that you can monitor the % Completion over time:

Territory	Sales	Goal	% Completion
APAC	1,281,706	2,000,000	64.1%
EMEA	5,008,224	6,000,000	83.5%
Japan	503,958	1,000,000	50.4%
NA	3,852,061	3,000,000	128.4%
Grand Total	10,645,949	12,000,000	-

Figure 53: % Completion Over Time

The Goal measure is just a user defined measure that checks each Territory member and assigns a goal value.

Create Calculated Measure

Display Name:

Format: Decimal Places:

Type:

> `CASE
 WHEN [Markets].CurrentMember IS [Markets].[APAC]
 THEN 2000000
 WHEN [Markets].CurrentMember IS [Markets].[EMEA]
 THEN 6000000
 WHEN [Markets].CurrentMember IS [Markets].[Japan]
 THEN 1000000
 WHEN [Markets].CurrentMember IS [Markets].[NA]
 THEN 3000000
 ELSE 0
 END`

Subtotal Calculation

☐ Calculate subtotals using measure formula

☐ Apply to data source

Figure 54: Goal Measure

The % Completion is another user defined measure with the following MDX expression:

```
[Measures].[Sales]/[Measures].[Goal]
```

You can achieve the colored highlighting with the following format MDX expression:

```
Case
When [Measures].CurrentMember < .6
Then '|#,##0.0%|style=red'
When [Measures].CurrentMember < .8
Then '|#,##0.0%|style=yellow'
When [Measures].CurrentMember > 1
Then '|#,##0.0%|style=green'
Else '|#,##0.0%'
End
```

Calculating a Weighted Average

A weighted average is useful in calculating an average unit price that considers the quantity of each transaction at a given price. In this next example, `Sales` is the unit price paid by each customer for the 1928 Mercedes-Benz SSK.

Product	Customer	Sales	Quantity	Weighted Average
1928 Mercedes-Benz S...	AV Stores, Co.	\$6,338.47	55	\$8,350.67
	Amica Models & Co.	\$4,455.00	22	\$8,350.67
	Australian Collectables, Ltd	\$3,675.32	22	\$8,350.67
	Australian Collectors, Co.	\$11,298.76	55	\$8,350.67
	Auto Associés & Cie.	\$3,577.60	20	\$8,350.67
	Blauer See Auto, Co.	\$3,773.38	26	\$8,350.67
	Corporate Gift Ideas Co.	\$7,209.12	48	\$8,350.67
	Corrida Auto Replicas, Ltd	\$9,534.50	50	\$8,350.67
	Euro+ Shopping Channel	\$14,433.52	104	\$8,350.67
	Extreme Desk Decorations, L...	\$5,315.80	35	\$8,350.67
	GiftsForHim.com	\$7,749.28	56	\$8,350.67
	Herkku Gifts	\$8,272.34	43	\$8,350.67
	Lyon Souvenirs	\$4,462.20	30	\$8,350.67
	Mini Caravy	\$3,726.00	24	\$8,350.67
	Mini Gifts Distributors Ltd.	\$14,393.59	107	\$8,350.67
	Online Diecast Creations Co.	\$5,797.44	36	\$8,350.67
	Rovelli Gifts	\$3,452.68	22	\$8,350.67
	Saveley & Henriot, Co.	\$5,579.02	29	\$8,350.67
	Signal Collectibles Ltd.	\$4,230.62	23	\$8,350.67
	Souvenirs And Things Co.	\$7,956.46	41	\$8,350.67
	Toys4GrownUps.com	\$3,033.28	32	\$8,350.67
1928 Mercedes-Benz SSK Sum		-	880	-
1928 Mercedes-Benz SSK Average		\$6,584.02	-	\$8,350.67

Figure 55: Weighted Average Results

The average price paid as shown in the `Sales` total is \$6,584.02. However, this is just a straight average that does not consider that some customers such as Euro+ Shopping Channel and Mini Gifts Distributors, Ltd. purchased higher quantities at much higher prices. The Weighted Average measure will weigh the unit sale price by the quantities purchased at each price.

Create Calculated Measure

Display Name:

Format: Decimal Places:

Type:

☐ Calculate subtotals using measure formula
☐ Apply to data source

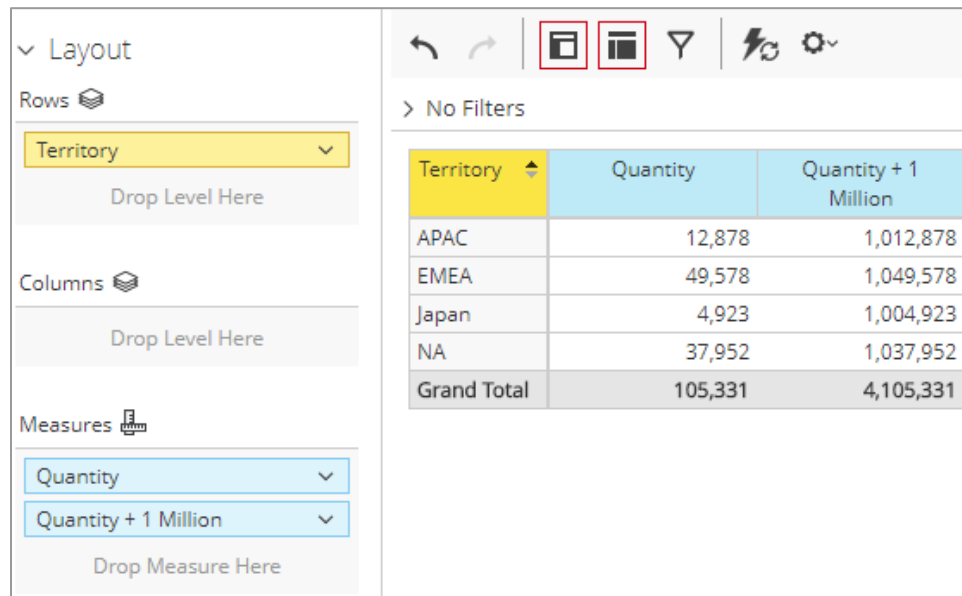
Figure 56: Weighted Average Measure

When dealing with many transaction records, it is more efficient for you to do the calculation in the database, rather than in Mondrian. Define a relational measure which returns the sum of `Sales * Quantity` and another relational measure which returns the sum of `Quantity`, then create a calculated measure to divide the two.

How Solve Orders Affect Calculating Measures

Solve orders determine the order of calculations when there are competing calculated members on a single cell. MSDN has a detailed explanation with a working example at [MDX Data Manipulation – Understanding Pass Order and Solve Order](#).

Here is another example to help you understand solve orders in Analyzer:

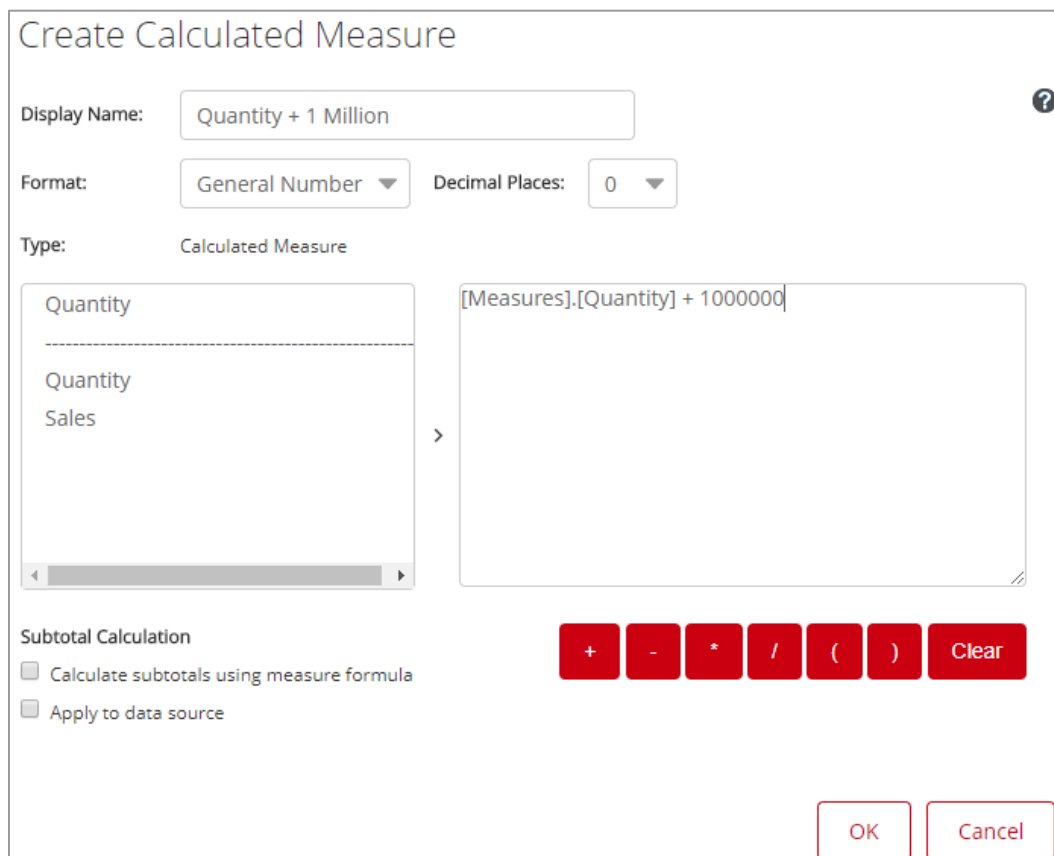


The screenshot shows the Pentaho Analyzer interface. On the left, the 'Layout' pane has 'Territory' in the Rows section and 'Quantity' and 'Quantity + 1 Million' in the Measures section. The main view displays a table with the following data:

Territory	Quantity	Quantity + 1 Million
APAC	12,878	1,012,878
EMEA	49,578	1,049,578
Japan	4,923	1,004,923
NA	37,952	1,037,952
Grand Total	105,331	4,105,331

Figure 57: *Quantity + 1 Million*

Create *Quantity + 1 Million* with a user defined measure:



The 'Create Calculated Measure' dialog box is shown. The 'Display Name' is 'Quantity + 1 Million'. The 'Format' is 'General Number' and 'Decimal Places' is '0'. The 'Type' is 'Calculated Measure'. The formula field contains '[Measures].[Quantity] + 1000000'. The 'Subtotal Calculation' section has two unchecked checkboxes: 'Calculate subtotals using measure formula' and 'Apply to data source'. The dialog includes a calculator keypad and 'OK' and 'Cancel' buttons.

Figure 58: *Quantity + 1 Million Calculated Measure*

The Grand Total in the Quantity + 1 Million column has two competing calculated members, listed below with their corresponding MDX as grabbed from the log file:

Quantity + 1 Million → Solve Order = 0

```
MDX: MEMBER [Measures].[*CALCULATED_MEASURE_0] AS
'[Measures].[Quantity]+1000000', FORMAT_STRING = '#,##0', SOLVE_ORDER=0
```

Grand Total → Solve Order = 100

```
MDX: MEMBER [Markets].[*TOTAL_MEMBER_SEL~SUM] AS 'SUM([*NATIVE_CJ_SET])',
SOLVE_ORDER=100
```

Since Grand Total has a higher solve order than Quantity + 1 Million, the bottom right cell will be calculated by iterating over each Territory and then summing up the Quantity + 1,000,000.

Now, if you go back to the same report and check the **Calculate subtotals using measure formula** box in the calculated measure, the Quantity + 1 Million's solve order will be changed from 0 to 200.

In this case, because Quantity + 1 Million has a higher solve order than Grand Total, the bottom right cell will now be calculated by adding 1,000,000 to the sum of each Territory's Quantity:

The screenshot shows the Pentaho Analyzer interface. On the left, the 'Layout' pane has 'Territory' in the Rows section and 'Quantity' and 'Quantity + 1 Million' in the Measures section. The main area displays a pivot table with the following data:

Territory	Quantity	Quantity + 1 Million
APAC	12,878	1,012,878
EMEA	49,578	1,049,578
Japan	4,923	1,004,923
NA	37,952	1,037,952
Grand Total	105,331	1,105,331

Figure 59: Calculating Quantity + 1 Million Using Measure Formula

By default, all schema defined measures will have a solve order of 0. You can override this for calculated measures by setting the CalculatedMemberProperty SOLVE_ORDER:

```

<CalculatedMember name="Average Sales Price" dimension="Measures">
  <Formula>[Measures].[Sales]/[Measures].[Quantity]</Formula>
  <CalculatedMemberProperty name="FORMAT_STRING" value="$#,##0.00" />
  <CalculatedMemberProperty name="SOLVE_ORDER"
value="200"></CalculatedMemberProperty>
</CalculatedMember>
</Cube>

```

For reference, here are the solve orders that Analyzer uses for different types of calculated members:

Table 2: Solve Orders for Analyzer

Calculated Member	Solve Order	Purpose
Formatting	500	Used to apply a format string
Numeric Filters	400	Used to apply context for Top N, Greater Than, and Similar filters
Min, Max, Count, and Average Totals	300	Used for subtotals
"Smart" Measures	200	Used when Calculate subtotal using formula is checked
Sum Totals	100	Used for subtotals
Aggregate Totals	-100	Used for subtotals
Trend Hangar	-200	Used to apply context for trend user defined measures
Slicer	-300	Used to apply slicer filters

Test Your MDX Knowledge

This is the answer to the quiz question in the Core Recipe Learning MDX:

```

WITH
MEMBER [Measures].[% of Total] AS
'[Measures].[Sales]/([Measures].[Sales],Ancestor([Markets].CurrentMember,
[Markets].[Territory]))', FORMAT_STRING = '###0.00%'
SELECT
{[Measures].[Sales], [Measures].[% of Total]} ON COLUMNS,
UNION({[Markets].[All Markets].[NA]},[Markets].[NA].Children) ON ROWS
FROM [SteelWheelsSales]

```

The answer is highlighted in red. Here, you used the `Ancestor` MDX function to find the ancestor member at the Territory level. For USA and Canada, this works fine and returns NA. However, for NA, since it is already at the Territory level, the `Ancestor` function just returns NA again. This type

of measure is also known as a level based measure, described in [Pinning a Measure to a Specific Hierarchy Level](#).

Related Information

Here are some links to information that you may find helpful while using this best practices document:

- [Java regular expressions](#)
- [Julian Hyde's blog entry on CurrentDateMember](#)
- [MDX Data Manipulation – Understanding Pass Order and Solve Order](#)
- [MSDN](#)
- [Pentaho: Enable Relative Date Filters in Mondrian](#)
- [Pentaho: Mondrian documentation](#)