User Guide

Version 3.0

Table of contents

1. CellworXs™

1.1 Introduction

In today's business environment it is critical that you understand what your underlying business drivers are. You need to manage significant amount of data made available by ERP systems, operational transaction systems or data warehouse applications. The most frequent complaint by business analysts is that: *"I spend most of the time to sort out my base data to fit it into the right format. Once I have the data in a way I want it, I have hardly time left to perform my business analysis! Very often also my manager asks the information slightly tweaked to our standard reporting based on a specific business purpose; this makes it even more frustrating. We cannot blame IT as they have to concentrate on the core business support and not hire an armada of report writers that produce the reports we need on a finger tip."*

An understandable complaint, but now there is an easy solution to it. CellworXs™ has been especially designed to *reduce "data manipulation time"* and to *add value making analysis time*. CellworXs™ is an add-in for Microsoft Excel that provides additional powerful functionality to your spreadsheet applications and analysis.

The main features of CellworXs™ are:

- Multi dimensional data analysis and customisable hierarchical data rollup (hierarchy builder): **"Structurizer"**
  CellworXs™ has added functionality that makes it very complementary to data warehouse and OLAP applications. Often users of these applications retrieve data into their Excel spreadsheets to further analyse. CellworXs™ helps to make the "last mile" significantly easier.
  If you do not use data warehouse and OLAP applications to analyse your data, CellworXs™ can even more help to put hierarchical structures on your "Excel Data Warehouse". You can easily model these hierarchical structures and apply them in any of your spreadsheets to analyse data through a large range of formulas covering:
  - Functions to retrieve a consolidated multidimensional item total from the data analysis table ("Excel database").
  - Functions to get information about the dimension item(s) and reporting on your hierarchy structures (i.e. parents-child relations, etc). Especially Hyperion Enterprise users might find this feature useful to create more dynamic Excel retrieve analysis worksheets.
  - Special predefined dimensional libraries that allow to easily report on different periods within the calendar or fiscal year (like YTD, QTD,….).
  - WHERE and LIKE statements that in addition to the dimensional hierarchies can be used as look-up conditions.
  - Functions to test/assure the data integrity of the loaded dimension items versus the data analysis table.
  - Error trapping as formula result: Rather than showing just "#Value" as formula result when certain formula parameter are not correct, an error message like (sample) "\[LIBARY OR DIMENSION NOT DEFINED OR LOADED]" is returned. This message helps to correct the initial formula parameter to get quickly to the desired correct result.

- Formula to value conversion (with original formula restore possibility): **"ValueMaker"**
  This functionality allows you to replace any defined spreadsheet formula including user defined functions with values and to optionally restore the original functionality also after the workbook had been saved. This functionality is very useful when a user has generated a report with user defined functions and he likes to share the report with other users not having the UDFs available.

- **Additional useful worksheet functions**, that facilitate analytical worksheet applications. These functions are especially to:

  - analyse the number of un-duped items in lists and to be able to return un-duped items via Excel formulas
  - a "text-to-formula" function, which is able to convert a text string into a formula, which then returns the formulas result
  - and more to discover…

- **No-dupe-copy:** a fast way to generate from a list with duplicate item occurrence just by "copy/paste" a list without duplicate items.

1.2 Release notes CellworXs™ version 2.x

1.2.1 What's new in Version 2.0:

Version 2.0 has been significantly enlarged in its functionality and renamed from ValueMaker to CellworXs™. All prior functionality is of course available and functioning complementary to the new added features. The following is new in CellworXs™ Version 2.0:

- Multi dimensional data analysis with CellworXs™'s "*STRUCTURIZER*".
- A list of useful data analysis worksheet functions has been introduced to CellworXs™.
- Multi user logon to one installation of CellworXs™: In prior versions of ValueMaker the add-in needed to be installed on the users computer or a by user unique network place. With CellworXs™ you can now place the add-in in the add-in library of Excel.

1.2.2 What's new in Version 2.1:

With the Version 2.1 release we improved performance speed and added functionality, which users have requested.

1.2.2.1 What's new in ValueMaker's *functionality?*

For *ValueMaker* we have added the possibility to **replace multiple formula types or cell references at the same time,** which will certainly be a time saver for users that need to run multiple type of formula replacements. This option provides the user with the possibility to do very specific cell formula with value replacement and original formula restore possibility.
Users have the possibility to save with their personal settings two sets of multiple replacement options each for formulas/function replacement and for cell reference replacement.

1.2.2.2 What's new in STRUCTURIZER *functionality?*

Performance:

The calculation speed of *STRUCTURIZER* VMSDxxx (VMSDVAL,VMSDMAX,…) formulas have been significantly increased. These functions are now performing significantly faster (especially on large Excel database tables) than comparable function constructions with Excel's standard functions (e.g.: DSUM, array functions, lookup functions). With this enhancement STRUCTURIZER not only gains significant time on designing Excel based analysis templates, but becomes also leading edge on the analysis execution time.

In addition the loading speed of large hierarchies structures has been significantly increased. There is virtually nearly no more waiting time even for the load of hierarchies containing 5000+ hierarchy nodes.

Functionality:

<u>The following further enhancements have been done for Structurizer's VMSDxxx functions:</u>

- The prior limit of 15 dimensions parameter included VMSDxxx() function has been increased to 27 dimension parameter in one function. Only VMSD12PVAL() can be used with up to 25 dimension parameter in one function.
- The WHERE statement and its comparison parameters (<,>,=,<>,>=,<=) can now also be executed for string comparisons, e.g. "WHERE.CUSTOMERCITY.=Berlin"
- The [BLANK] dimension item can now also be used for the WHERE statement, e.g. "WHERE.CUSTOMERCITY.<>[BLANK]"

- When using a WHERE, the comparisons will interpret the data in the dimension range as either numbers, dates or strings. It is the value being passed on in the WHERE that will determine if the data in the dimension is handled as a number, a date or a string. E.g. "WHERE.columnName.<12345" will handle the data in column <columnName> as numbers, because 12345 can be interpreted as a number. For a dimension range being handled as a number, an empty cell we be considered equal to 0. For a dimension range being handled as a date, an empty cell will be considered equal to 30 Dec 1899.
- Mapping [MISSING] in a dimensional hierarchy could slow in the prior version 2.0 calculation time down. This has been resolved in version 2.1, where it performs as any other dimension item.
- Important Note: The SumRange can not be outside of the DimensionRange any more. This was allowed in previous versions. E.g. DimensionRange is columns D:H, SumRange is column B.
- The SumRange does accept as of this version named ranges.
- If you call VMSD12PVAL with more than 12 columns in the parameter TwelvePeriodCol, you now get the following text back: "[TwelvePeriodCol needs to be a range of 12 columns wide]"
- If you call VMSD12PVAL and you specify a date outside of the year, for example Q.YTD with period = 5, you get back 0.

When using incomplete Dimension items in a format like “SalesReport.Customercity.”, an error is now returned rather than interpreting the dimension item as “SalesReport.Customercity.[BLANK]”.

For Excel versions earlier or equal to Excel 2002: when using Excel’s “Manual Calculation” mode and entering a VMSxxx function, this may return in a “non-refreshed” value based on an older version of the dimension range or an older version of the library dimension. As such make sure the workbook is recalculated.

1.2.2.3 What’s new in the “additional useful worksheet functions” functionality?

In this section we have concentrated our effort to increase the calculation performance speed of the VMUIC() and VMUIR() functions, which are useful to query Excel data bases for their unique items. These functions now significantly outperform comparable formula constructs with standard Excel functions.

1.3 Release notes CellworXs™ version 3.x

CellworXs version 3.0 also runs under Excel 2010.
CellworXs is now provided on a free of charge basis. There is no longer a need for activation keys. 30-day trial versions cease to exist.

2. *STRUCTURIZER* – Multidimensional and hierarchical analysis

2.1 Conceptual introduction

CellworXs™'s *STRUCTURIZER* functionality allows to easily build a structured data consolidation hierarchy on your different dimensions (typically columns) of your Excel database. The following explanations in this chapter are build on a data transaction extract of the well known Microsoft Access sample database "NORTHWIND Traders", which has been exported to our StructurizerQuickGuide.xls tutorial file (see sheet "Transaction extract").

With that sample tutorial we would like to show how easy transactional data can be analysed by:

- Easily building hierarchies on the different dimensions (i.e. customer, products,…),
- without any special Excel formula knowledge like combined look-up formulas etc.
- combining up to 15 different dimensions for your data analysis

Before we continue here a quick introduction into terminology we use. This will help you to more easily understand the following explanations:

2.1.1 Analysis data table

This is the Excel table that holds the data you want to analyse. Typically this data comes from:

- operational transaction systems (ERP systems, account systems,….) or
- OLAP and data warehousing systems.

Usually a data analysis table counts a considerable amount of transaction lines, which makes it difficult to perform ad-hoc analysis without creating complicated look-up formulas. Especially when you need to combine several "look-up" criteria, to calculate a specific value. Very often the creation of the look-up formulas need to be repeated as soon as a new analysis needs to be made.

****

Sample Case: Imagine based on the NorthWind Trader tutorial you need to calculate the sales revenue for a specific period, that has been achieved by a specific sales team for your key accounts in Western Europe.
In a traditional Excel analysis environment you would first create tables to establish your different dimension hierarchies structures:

- your sales team hierarchy as you base data is by sales person
- you key account hierarchies as your base data is by customer without customer type indication
- your country hierarchy as your customer data is by customer city or country only

With a little bit of luck you have done already an analysis like that and you can copy your hierarchy tables from the prior analysis.

Then you would typically establish in help columns next to the data analysis table look-up formulas to assign your different dimension items (for example a customer name) to your grouping hierarchy (for example a key account group for your customers). You would copy down your formulas all the long way next to your transactional data analysis table.

In a last step you would then establish your final look-up formulas (like DSUM, array formulas, combined SUMIF, …). This is the part you probably most "enjoy" as you might now finally get the number you want to calculate. But now you manager comes and tells you that we need to group the key account groups to some further hierarchical consolidation and if you could look at the data not only from a calendar period point of view, but also consider that the fiscal year is slightly offset (and starts in March rather than in January). Of course you will manage it as every time you get asked.

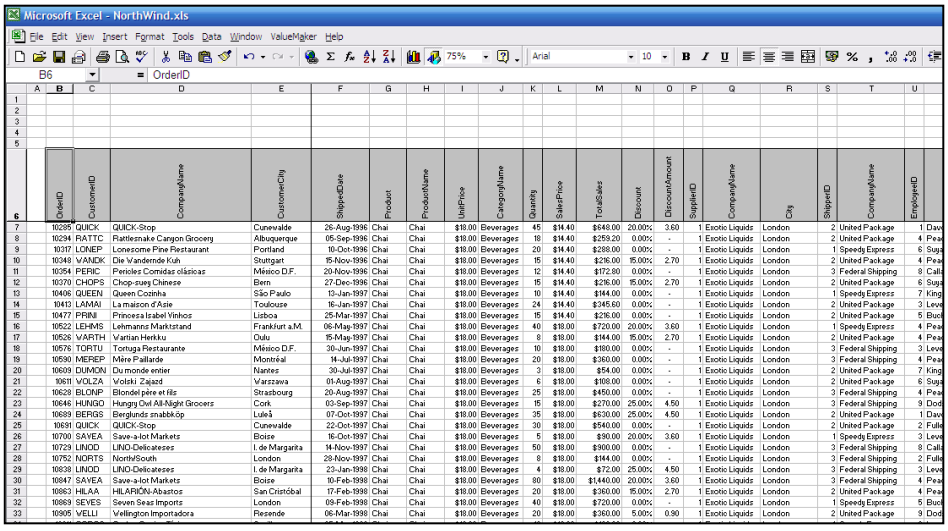
What a job! This sample case happens everyday and everywhere. More time is spend on crunching the data rather analysing it. There is a solution to it to inverse the equation!

****

With *STRUCTURIZER* you will be able to:

- Quickly establish grouping hierarchies for your different "dimensions" (columns) of your data analysis table.
- Re-use once established hierarchies in your different analysis.
- Share your hierarchies with your colleagues
- Allow for more analysis time, rather than data manipulation time
- Retrieve the result you are looking for with only one simple formula. No need to create additional help formulas. And you can investigate up to 15 (!) different dimensions criteria easily in your single formula.

Picture: Sample Data Analysis Table based on our "NorthWind Trader tutorial"



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1																					
2																					
3																					
4																					
5																					
6		OrderID	CustomerID	CompanyName	CustomerCity	ShippedDate	Product	ProductName	UnitPrice	CategoryName	Quantity	SalesPrice	TotalSales	Discount	DiscountAmount	SupplierID	CompanyName	City	ShipperID	CompanyName	EmployeeID
7		10295	QUICK	QUICK-Stop	Cunevalde	26-Aug-1996	Chai	Chai	$18.00	Beverages	45	$14.40	$648.00	20.00%	3.60	1	Exotic Liquids	London	2	United Package	1
8		10294	RATTC	Rattlesnake Canyon Grocery	Albuquerque	05-Sep-1996	Chai	Chai	$18.00	Beverages	18	$14.40	$259.20	0.00%	-	1	Exotic Liquids	London	2	United Package	4
9		10317	LONEP	Lonesome Pine Restaurant	Portland	10-Oct-1996	Chai	Chai	$18.00	Beverages	20	$14.40	$288.00	0.00%	-	1	Exotic Liquids	London	1	Speedy Express	6
10		10348	WANDK	Die Wandernde Kuh	Stuttgart	15-Nov-1996	Chai	Chai	$18.00	Beverages	15	$14.40	$216.00	15.00%	2.70	1	Exotic Liquids	London	2	United Package	4
11		10354	PERIC	Pericles Comidas clásicas	México D.F.	20-Nov-1996	Chai	Chai	$18.00	Beverages	12	$14.40	$172.80	0.00%	-	1	Exotic Liquids	London	3	Federal Shipping	8
12		10370	CHOPS	Chop-suey Chinese	Bern	27-Dec-1996	Chai	Chai	$18.00	Beverages	15	$14.40	$216.00	15.00%	2.70	1	Exotic Liquids	London	2	United Package	6
13		10406	QUEEN	Queen Cozinha	São Paulo	13-Jan-1997	Chai	Chai	$18.00	Beverages	10	$14.40	$144.00	0.00%	-	1	Exotic Liquids	London	1	Speedy Express	7
14		10413	LAMAI	La maison d'Asie	Toulouse	16-Jan-1997	Chai	Chai	$18.00	Beverages	24	$14.40	$345.60	0.00%	-	1	Exotic Liquids	London	2	United Package	3
15		10477	PRIMI	Princesa Isabel Vinhos	Lisboa	25-Mar-1997	Chai	Chai	$18.00	Beverages	15	$14.40	$216.00	0.00%	-	1	Exotic Liquids	London	2	United Package	5
16		10522	LEHMS	Lehmanns Marktstand	Frankfurt a.M.	06-May-1997	Chai	Chai	$18.00	Beverages	40	$18.00	$720.00	20.00%	3.60	1	Exotic Liquids	London	1	Speedy Express	4
17		10526	VARTH	Vartian Herkku	Oulu	15-May-1997	Chai	Chai	$18.00	Beverages	8	$18.00	$144.00	15.00%	2.70	1	Exotic Liquids	London	2	United Package	4
18		10576	TORTU	Tortuga Restaurante	México D.F.	30-Jun-1997	Chai	Chai	$18.00	Beverages	10	$18.00	$180.00	0.00%	-	1	Exotic Liquids	London	3	Federal Shipping	3
19		10590	MEREP	Mère Paillarde	Montréal	14-Jul-1997	Chai	Chai	$18.00	Beverages	20	$18.00	$360.00	0.00%	-	1	Exotic Liquids	London	3	Federal Shipping	4
20		10609	DUMON	Du monde entier	Nantes	30-Jul-1997	Chai	Chai	$18.00	Beverages	3	$18.00	$54.00	0.00%	-	1	Exotic Liquids	London	2	United Package	7
21		10611	WOLZA	Wolski Zajazd	Warszawa	01-Aug-1997	Chai	Chai	$18.00	Beverages	6	$18.00	$108.00	0.00%	-	1	Exotic Liquids	London	2	United Package	6
22		10628	BLONP	Blondel père et fils	Strasbourg	20-Aug-1997	Chai	Chai	$18.00	Beverages	25	$18.00	$450.00	0.00%	-	1	Exotic Liquids	London	3	Federal Shipping	4
23		10646	HUNGO	Hungry Owl All-Night Grocers	Cork	03-Sep-1997	Chai	Chai	$18.00	Beverages	15	$18.00	$270.00	25.00%	4.50	1	Exotic Liquids	London	3	Federal Shipping	9
24		10689	BERGS	Berglunds snabbköp	Luleå	07-Oct-1997	Chai	Chai	$18.00	Beverages	35	$18.00	$630.00	25.00%	4.50	1	Exotic Liquids	London	2	United Package	1
25		10691	QUICK	QUICK-Stop	Cunevalde	22-Oct-1997	Chai	Chai	$18.00	Beverages	30	$18.00	$540.00	0.00%	-	1	Exotic Liquids	London	2	United Package	2
26		10700	SAVEA	Save-a-lot Markets	Boise	16-Oct-1997	Chai	Chai	$18.00	Beverages	5	$18.00	$90.00	20.00%	3.60	1	Exotic Liquids	London	1	Speedy Express	3
27		10729	LINOD	LINO-Delicateses	I. de Margarita	14-Nov-1997	Chai	Chai	$18.00	Beverages	50	$18.00	$900.00	0.00%	-	1	Exotic Liquids	London	3	Federal Shipping	8
28		10752	NORTS	North/South	London	28-Nov-1997	Chai	Chai	$18.00	Beverages	8	$18.00	$144.00	0.00%	-	1	Exotic Liquids	London	3	Federal Shipping	2
29		10838	LINOD	LINO-Delicateses	I. de Margarita	23-Jan-1998	Chai	Chai	$18.00	Beverages	4	$18.00	$72.00	25.00%	4.50	1	Exotic Liquids	London	3	Federal Shipping	3
30		10847	SAVEA	Save-a-lot Markets	Boise	10-Feb-1998	Chai	Chai	$18.00	Beverages	80	$18.00	$1,440.00	20.00%	3.60	1	Exotic Liquids	London	3	Federal Shipping	4
31		10863	HILAA	HILARIÓN-Abastos	San Cristóbal	17-Feb-1998	Chai	Chai	$18.00	Beverages	20	$18.00	$360.00	15.00%	2.70	1	Exotic Liquids	London	2	United Package	4
32		10869	SEVES	Seven Seas Imports	London	09-Feb-1998	Chai	Chai	$18.00	Beverages	40	$18.00	$720.00	0.00%	-	1	Exotic Liquids	London	1	Speedy Express	5
33		10905	WELLI	Wellington Importadora	Resende	06-Mar-1998	Chai	Chai	$18.00	Beverages	20	$18.00	$360.00	5.00%	0.90	1	Exotic Liquids	London	2	United Package	9

2.1.2 Dimensions and dimension items

2.1.2.1 Dimensions

Dimensions are the categorical description of your transactional data. For a sales related transaction the "CustomerName" represents a dimension. Dimensions are typically organised in columns in your data analysis table. Other examples for dimensions are product codes or names, sales revenue, customer country, cost centre, P&L account code,….

The dimensions put the "story" about each transaction line of your data analysis table. You could read your transaction line for example like that: "The 5th of January 2007 customer Miller bought 500 pieces of our product "Uncle Bob's Organic Dried Pears" for a unit price of $1,- each and he received a 15% discount, resulting in a sales revenue of $425,-….."

As you can see from above example, dimensions can be:

- **measurable**
  Measurable dimensions (like sales revenue, sales units,….) can be added up. They could also be conditioned in their query (Example: sale revenue > $1000).
- **non-measurable**
  Non-measurable dimensions (like customer name, product type) can not be added up but they can be grouped to structured hierarchies (Example: "Sasquatch Ale" belongs to Product Group Beverages). They could also be conditioned in the query (Example: CustomerName like Mill*).
- **a date**
  Time is a specific dimension as usually you want to retrieve information over a specific time period (Example: April 26th year-to-date). *STRUCTURIZER* provides different functionality that eases the handling of the dimension time. It allows also to easily handle if your fiscal year is offset from the calendar year and has an importance to your analysis.

Important learning item to understand how STRUCTURIZER works: The column headers of your data analysis table represent the dimension name, which is used by *STRUCTURIZER* when you use one of its spreadsheet functions, to refer to data in that column of the data analysis table.

2.1.2.2 Dimension Items

Dimension items are the individual items belonging to a dimension. For example each individual customer is a dimension item of the dimension "Customer". The individual dimension item in the analysis data table are called base items. To extract a list of base items without dupes from your analysis data table, you can use either CellworXs™ "Unique Item copy/paste" functionality or use CellworXs™'s spreadsheet functions (I.e. *VMUIR()* function) to retrieve a list of base items without dupes.

We said that typically each column presents a potential dimension, but sometimes the data in one dimension (column) is actually combining two or more dimensions. An example could be:

Dimension Item Code	Meaning of the code
EEG	European Export Goods, exported from our country *
EIG	European Import Goods, imported to our country
ETG	European Transit Goods (shipped between countries in Europe, but not touching our country)
OEG	Overseas Export Goods, exported from our country
OIG	Overseas Import Goods, imported to our country
OTG	Overseas Transit Goods (shipped between countries in Europe, but not touching our country)

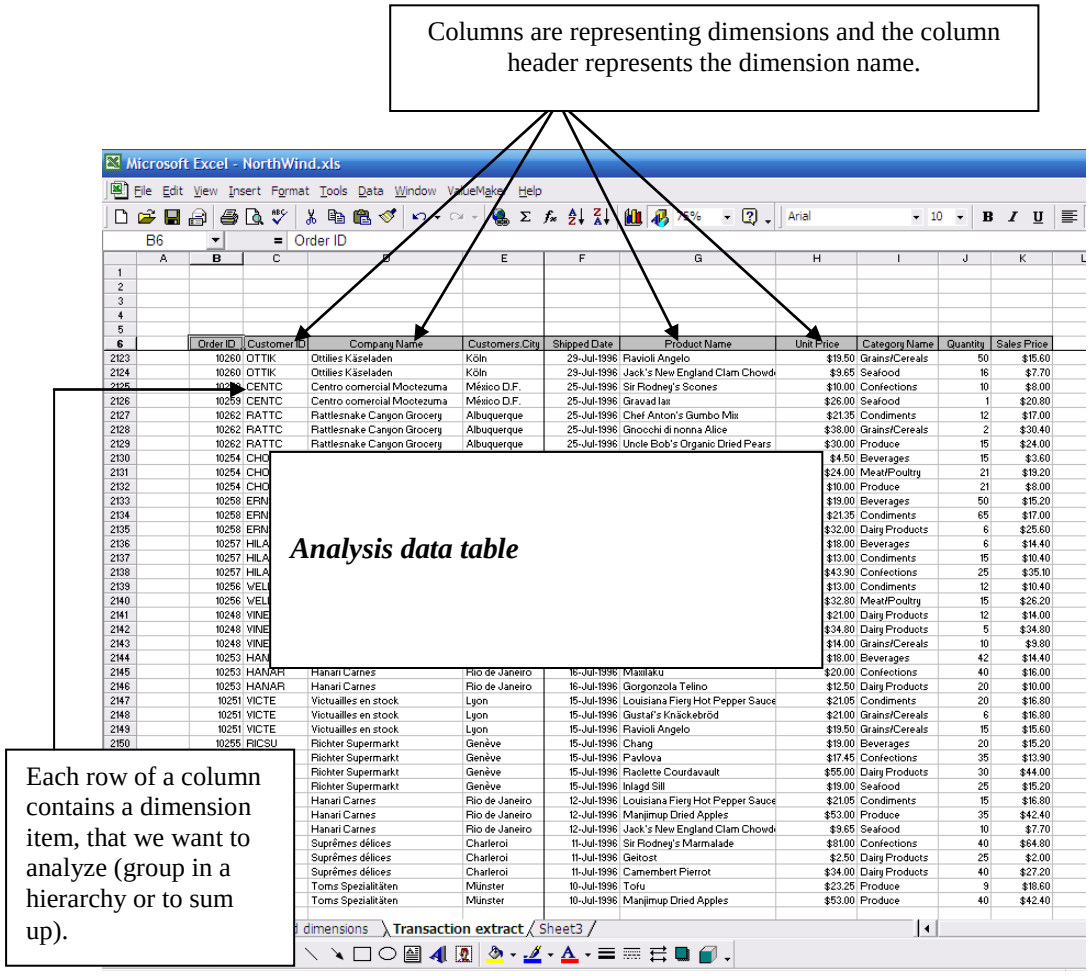
** "our country" in this example is located in Europe*

In above example the information of the flow of goods (import, export and transit) as well as the destination (Europe, Overseas) is combined in one dimension.

***STRUCTURIZER* will allow you – if needed – to report on each dimension individually even though they might be combined in one field!**

The following picture shows a sample data analysis table, with indications to the above discussed topics:

Picture: Data Analysis Table



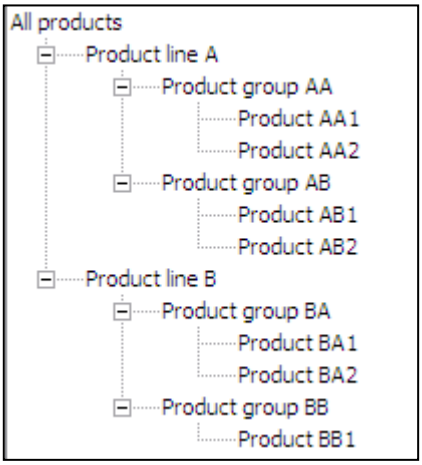
2.1.3 Hierarchical dimension structures

STRUCTURIZER will allow you to group - according to your criteria - your dimensions into "hierarchy trees". You can group i.e. individual products to product groups, then to product lines and let those roll into "all products". Imagine you have the following products as base items in the "product" dimension of your analysis data table:

Product AA1
Product AA2
Product AB1
Product AB2
Product BA1
Product BA2
Product BB1

With *STRUCTURIZER* you can easily create the following reporting hierarchy that allows you to easily retrieve information related to any of the hierarchy levels (i.e. get the total sales data by "Product line A" or "Product Group BA" ,…) . The next picture shows a parent-child hierarchy.

Picture: Sample parent-child product hierarchy



"Product AA1" is the base item that you will find back in your analysis data table. It is a child of the parent "Product group AA". Each item in the hierarchy (base children, children and parents) is unique and occurs only once.

2.1.4 Libraries to organise your dimensional hierarchies

With *STRUCTURIZER* you can principally create as many dimensional hierarchies as you want (eventually limited by hardware memory). In order to organize those you should consider to classify them in "virtual libraries". You might have various different reports (data analysis tables) and you could group them according to those. Or just to classify them according to your personal "classification rules".

The following shows an example, how you could classify your dimensions in libraries:

Library Name:	**Dimension Name**
SalesReport	CustomerName
SalesReport	Product
SalesReport	SalesPersonel

CostCentreReport	CostCenterCode
CostCentreReport	PLAccount
CostCentreReport	SupplierName

2.2 From "concept" to Excel: How to transform your Excel to an easy to use multidimensional analysis tool.
The next steps with which simple steps you can transform your Excel to a great performing multidimensional analysis tool based on your defined "virtual hierarchies".

2.2.1 STRUCTURIZER's "logic link" between the data analysis table and the custom dimensional hierarchies

With STRUCTURIZER you define your dimensional hierarchies in named Excel ranges. These ranges will need to be named according to *STRUCTURIZERS* syntax rules (see next chapter point). The ranges can be loaded into *STRUCTURIZER* and as soon as they are loaded, they are available in all of your different spreadsheet files for analysis purpose – even though the initial source file might be closed by now.
If you change the hierarchy definition in the initial source file's named range , i.e. for adding an additional item, you need to re-load the hierarchy and "refresh" the VMS*() formulas, to reflect the change in your formula results.

All hierarchies loaded in *STRUCTURIZER* can also be exported in different formats:

- either in hierarchy upload range format (option to include range names) or
- as "pick list format" in order to be used i.e. in data validation list

2.2.2 How refers *STRUCTURIZER* to different hierarchies and its items in the VMS*() formulas?

When you refer to a dimension item or its hierarchical parent in one of *STRUCTURIZER's* spreadsheet functions, you will always refer to it with the following syntax:
LibraryName.DimensionName.ItemName

Example: Imagine above "Product" hierarchy from chapter 2.1.3) is stored in the library "MyFirstLibrary". The dimension name (= column header of the analysis data table column) is "Product". In order to retrieve the sales revenue for "Product Line B" you would refer to it as **"MyFirstLibrary.Product.Product Line B"**

Having the possibility of creating libraries also gives the advantage that you can create alternative hierarchical dimension roll-ups on the same dimension. Remember the example earlier discussed, where one Excel dimensions (column) combines two or more dimensions (in that example the flow of goods and the destination of the goods were combined in one code).

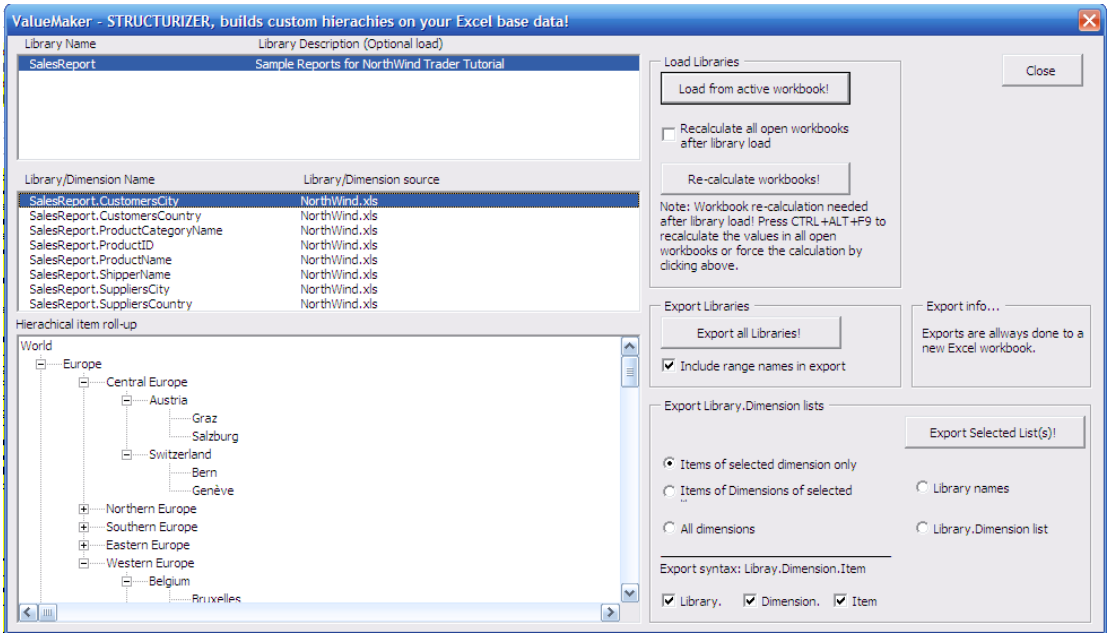
Note: When you refer to a dimension item in a VMS*() formula, the dimension item is not case sensitive and ignores leading and trailing spaces for the comparison with the data in the data analysis table. This presents a main advantage when "non clean" data analysis tables are used, which is often the case when data is extracted from operational databases.

2.2.3 Defining dimensional hierarchy "source" definitions in "named Excel ranges"

The following explains how to define your dimensional hierarchy source in named Excel ranges.
Imagine we would like to know the sales revenue of our customers based on their geographic location. Currently our NorthWind sample sales data transaction extract shows the city where our customer is located (and we could also include the country where our customer is located). But now we would like to know the sales of Western European Countries or the whole of Europe?
With the STRUCTURIZER functionality you can easily build in any Excel sheet a hierarchical "parent-child" dimension roll-up. The following example shows how to easily generate a geographical hierarchy for your customers, even though your transaction database contains only city names. You will then be able to report easily on your custom defined hierarchies like "Western Europe" or any

custom hierarchy level you defined. Once you have defined the dimension hierarchy, you upload it and you can visualize the hierarchical parent-child relations. As soon as a hierarchy is uploaded it is also available for reporting – in any of your Excel files. The next picture shows an example loaded into *STRUCTURIZER*:

Picture: Loaded libraries, dimensions and their hierarchical roll-up(s)



2.2.3.1 Set-up a hierarchy range (table)

1. Step: Create a two column table list describing your hierarchical "parent-child relations"

Anywhere in your Excel document you can define a hierarchy table. Once that table is loaded it will be available in all your Excel documents. The following example shows a hierarchical roll-up for countries into regions and finally into the dimension item "World".

Picture: Set-up of a simple "parent-child" relation ship table:

This is the table column containing the "**parent**": i.e. "World" is parent of "Europe".

Parent – Child relation table

This is the column containing the "**child**": i.e. "Europe" is child of "World"

	A	B	C
3		World	Europe
4		World	America
5		Europe	Central Europe
6		Europe	Northern Europe
7		Europe	Southern Europe
8		Europe	Eastern Europe
9		Europe	Western Europe
10		America	North America
11		America	South America
12		Central Europe	Austria
13		Western Europe	Belgium
14		Western Europe	France
15		Western Europe	Germany
16		Western Europe	Ireland
17		Central Europe	Switzerland
18		Western Europe	UK
19		Southern Europe	Italy
20		Southern Europe	Portugal
21		Southern Europe	Spain
22		Northern Europe	Sweden
23		Eastern Europe	Poland
24		Northern Europe	Denmark
25		Northern Europe	Finland
26		Northern Europe	Norway
27		South America	Argentina
28		South America	Brazil
29		South America	Venezuela
30		South America	Mexico
31		North America	Canada
32		North America	USA

Start the table with the top parent and define its related "child-children" at a following row below. Each parent in column B has to reoccur in a row above as child of a higher level parent. Only the final parent (in our case "World") will not reoccur in column C.

A loading error would occur, if you define in the first table rows "Europe" and it's children ("Central Europe", "Northern Europe",…) and below in a following row "World" with "Europe" as direct child.

Note: STRUCTURIZER will generate an error log file (Excel file) that states those wrong hierarchical definitions. This helps to quickly "debug" the table.

Important rules when setting up the hierarchy table:

- a hierarchy table contains two columns and as many rows as needed to define your hierarchical dimensions.
- The left column contains the parent and the right columns contains the child i.e.

World	Europe

  Meaning that "World" is parent of "Europe"
- The top of the table has to starts with the highest level hierarchy items and details down as you add rows, i.e.

World	Europe
World	America
World	Asia & Australia
Europe	Central Europe
Europe	Northern Europe
Europe	Southern Europe
Europe	Eastern Europe
Europe	Western Europe
America	North America
America	South America
Asia & Australia	Japan
Asia & Australia	Singapore
Asia & Australia	Australia

- An item without a assigned child is called a "base item" or "base child". In above example all items in the right column down from Central Europe to Australia would be base children. Base children are the items that you want to lookup in your Excel database for the hierarchical consolidation. The spelling of those has to be identical with the ones created in your hierarchy table.
- Each item has to have a parent – except the hierarchy "root" item. "World" is in the above example the hierarchy root item.
- Parent's parents have to be located above the parent item in the table. In above example "World" is parent of parent item "Europe" and their parent child relation has to be defined in a row above the last usage of "Europe" as parent, i.e. at least one row above the row describing the parent child relation "Europe – Central Europe".
- A dimension item can be mentioned only once as child (all items in the right column are unique) but several times as parent. *STRUCTURIZER* does not allow alternative roll-ups of one child into two different parents.
- The spelling of dimension item and parent has to be equal, i.e. if your item is called "America" as parent, in the higher relation it has to be called as well "America" and not "Ameri**k**a"
- The spelling of dimension items is:
  - not case sensitive
  - leading or ending spaces will be ignored / truncated when loaded
  - can contain any character including special characters.

STRUCTURIZER does test the integrity when you load the dimension and will not load violating relations. It generates an error log if a rule is violated that easily points the issue. For alternative hierarchical roll-ups of the same dimension you can create another "library".

In our "NorthWind" sample case we even further go from country level down to Country-City parent child relations (However we made sure there are no duplicate city names between countries)..

To define a library table just type a two column table, considering the following simple rules:

- The left column contains the "parent" dimension item and the right column contains the "child" dimension item.

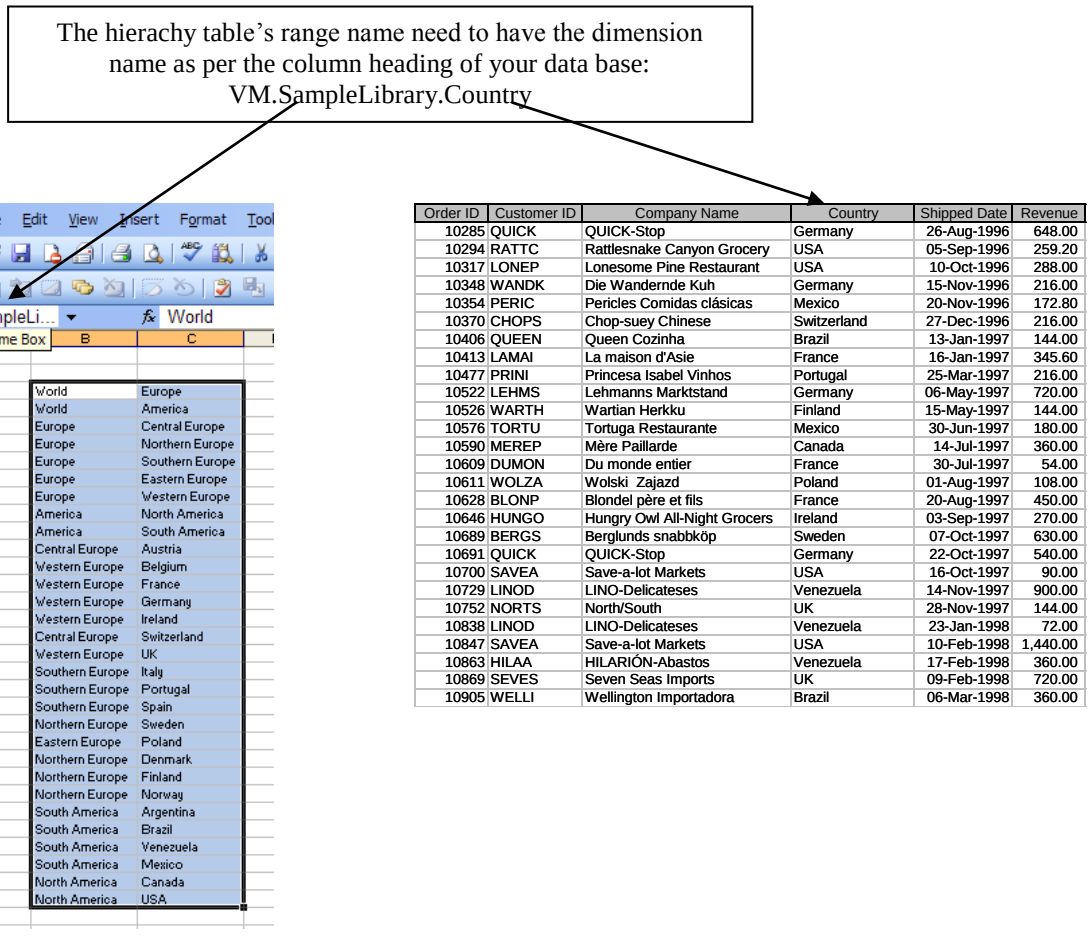
- You start from the top level relation (highest parent-child relation) and you detail down as you add table rows.

2. Step: Assign a "named range" to your table range

When you load your dimensional hierarchy definitions, **CellworXs™ will look for range names defined in the active workbook that apply the following syntax:**
VM.[LibraryName].[DimensionName]

The following picture shows how to define a range name. For the naming you need to consider the following:

- You can choose the library name as per your convenience (respecting Excel's rules of naming worksheet ranges).
- The dimension name has to be the spelling of the column heading of your data analysis table (but also respecting Excel's rules of naming worksheet ranges).
- The "." (dot) is used to separate library and dimension name. There should be no mare than two dots used in the library name (i.e. not allowed syntax: VM.MyLibarary.DimensionName.1)



Order ID	Customer ID	Company Name	Country	Shipped Date	Revenue
10285	QUICK	QUICK-Stop	Germany	26-Aug-1996	648.00
10294	RATTC	Rattlesnake Canyon Grocery	USA	05-Sep-1996	259.20
10317	LONEP	Lonesome Pine Restaurant	USA	10-Oct-1996	288.00
10348	WANDK	Die Wandernde Kuh	Germany	15-Nov-1996	216.00
10354	PERIC	Pericles Comidas clásicas	Mexico	20-Nov-1996	172.80
10370	CHOPS	Chop-suey Chinese	Switzerland	27-Dec-1996	216.00
10406	QUEEN	Queen Cozinha	Brazil	13-Jan-1997	144.00
10413	LAMAI	La maison d'Asie	France	16-Jan-1997	345.60
10477	PRINI	Princesa Isabel Vinhos	Portugal	25-Mar-1997	216.00
10522	LEHMS	Lehmanns Marktstand	Germany	06-May-1997	720.00
10526	WARTH	Wartian Herkku	Finland	15-May-1997	144.00
10576	TORTU	Tortuga Restaurante	Mexico	30-Jun-1997	180.00
10590	MEREP	Mère Paillarde	Canada	14-Jul-1997	360.00
10609	DUMON	Du monde entier	France	30-Jul-1997	54.00
10611	WOLZA	Wolski Zajazd	Poland	01-Aug-1997	108.00
10628	BLONP	Blondel père et fils	France	20-Aug-1997	450.00
10646	HUNGO	Hungry Owl All-Night Grocers	Ireland	03-Sep-1997	270.00
10689	BERGS	Berglunds snabbköp	Sweden	07-Oct-1997	630.00
10691	QUICK	QUICK-Stop	Germany	22-Oct-1997	540.00
10700	SAVEA	Save-a-lot Markets	USA	16-Oct-1997	90.00
10729	LINOD	LINO-Delicateses	Venezuela	14-Nov-1997	900.00
10752	NORTS	North/South	UK	28-Nov-1997	144.00
10838	LINOD	LINO-Delicateses	Venezuela	23-Jan-1998	72.00
10847	SAVEA	Save-a-lot Markets	USA	10-Feb-1998	1,440.00
10863	HILAA	HILARIÓN-Abastos	Venezuela	17-Feb-1998	360.00
10869	SEVES	Seven Seas Imports	UK	09-Feb-1998	720.00
10905	WELLI	Wellington Importadora	Brazil	06-Mar-1998	360.00

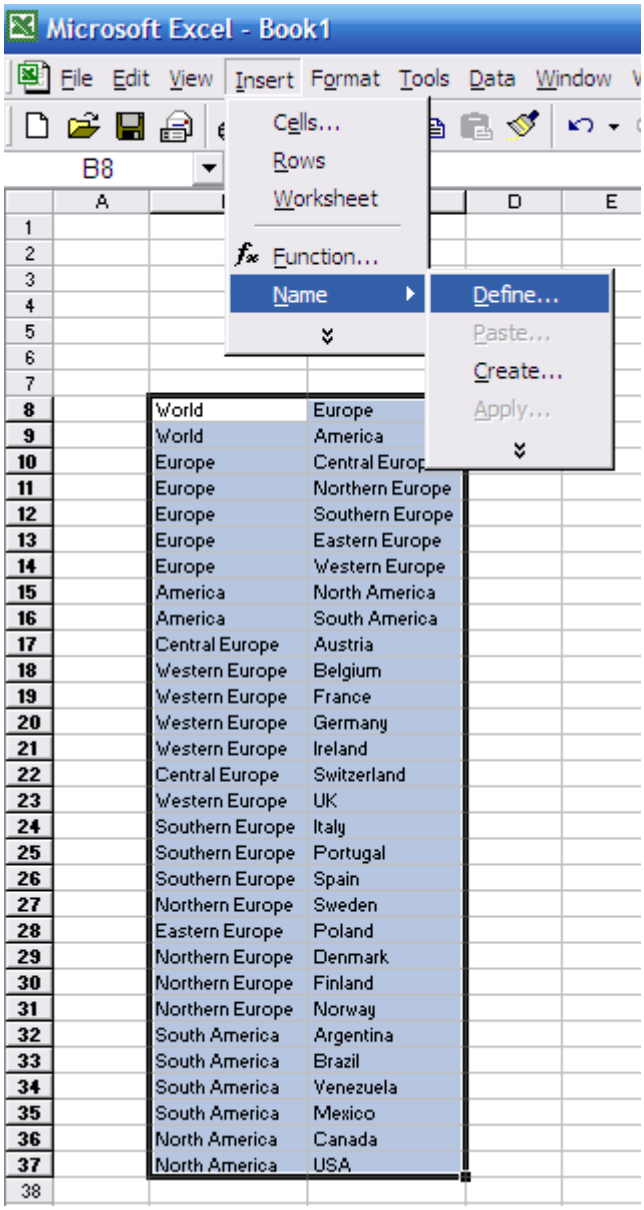
There are two ways to enter a named range in the Excel sheet:

1. Direct edit in the menu's name box (see above picture). This is fast and useful when a range is established for the first time. To do so, select the entire range you want to define as hierarchy range and then type directly in the name box the library/dimension name according to *STRUCTURIZER's* syntax (VM.Mylibrary.MyDimensionName)
2. Edit via the Excel menu "Insert/Name/Define…": This option is useful to edit/modify already entered ranges.

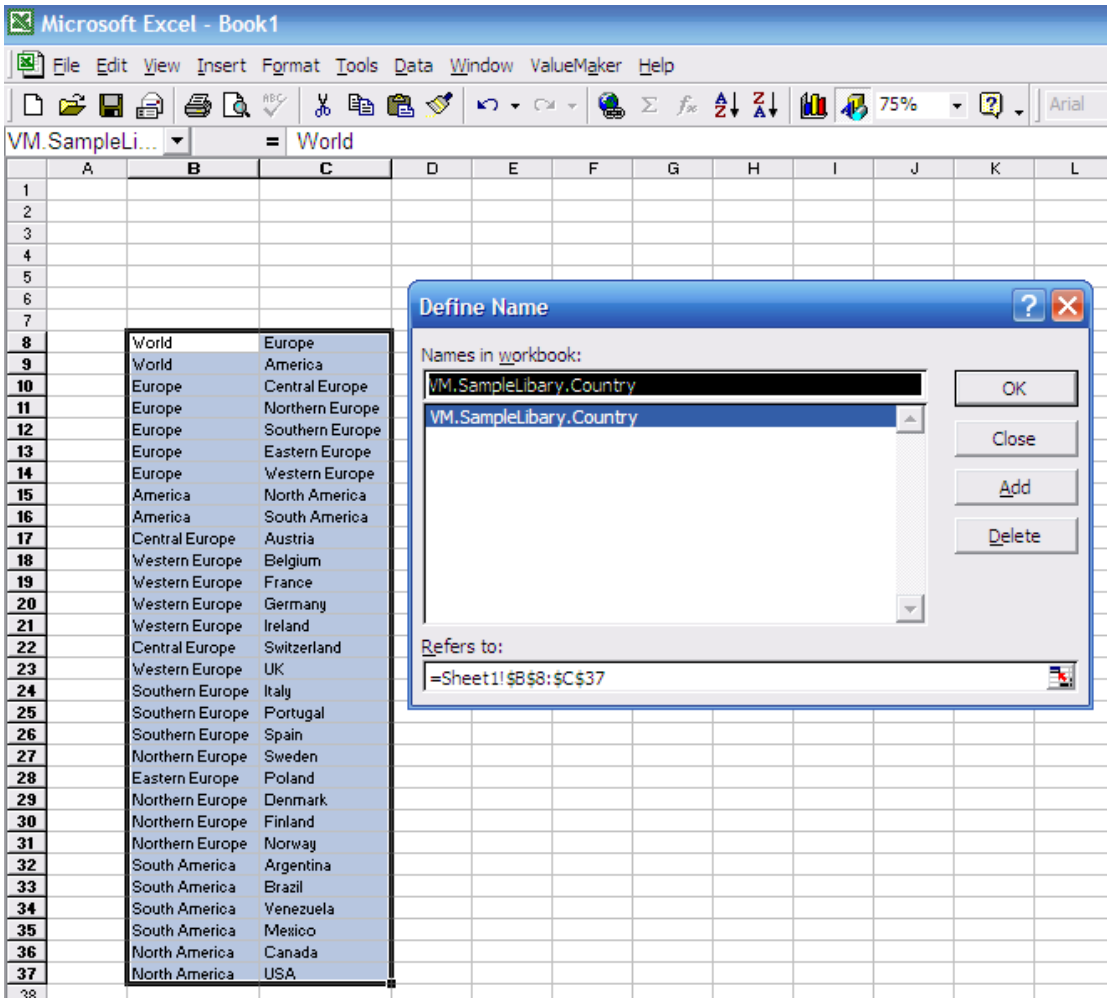
Important note:

MULTIPLE NAMES RANGES: It is possible to enter several names for the same range. This is useful if you have different data analysis tables that have the same content in a column but different column headings (remember the column heading has to be equal to last part (after the second dot) of your range name.

Below picture shows how to establish a range name in Excel via the menu bar. As per above sample table, high light your hierarchy definition table and select from the Excel menu "Insert/Name/Define…"



Enter the range name with the syntax: VM.[LibaryName].[DimensionName] for example: "VM.SampleLibrary.Country"



8	World	Europe
9	World	America
10	Europe	Central Europe
11	Europe	Northern Europe
12	Europe	Southern Europe
13	Europe	Eastern Europe
14	Europe	Western Europe
15	America	North America
16	America	South America
17	Central Europe	Austria
18	Western Europe	Belgium
19	Western Europe	France
20	Western Europe	Germany
21	Western Europe	Ireland
22	Central Europe	Switzerland
23	Western Europe	UK
24	Southern Europe	Italy
25	Southern Europe	Portugal
26	Southern Europe	Spain
27	Northern Europe	Sweden
28	Eastern Europe	Poland
29	Northern Europe	Denmark
30	Northern Europe	Finland
31	Northern Europe	Norway
32	South America	Argentina
33	South America	Brazil
34	South America	Venezuela
35	South America	Mexico
36	North America	Canada
37	North America	USA

IMPORTANT NOTE: Given the Excel based naming conventions for named cell ranges, you cannot use special characters in the range name definition except "." (dot) and "_" (underscore). See above comments about the usage of the "dot" when defining a named hierarchy range.

Important Rules:

- **The dimension name has to be the spelling of the column header of your Excel database. In our above example the Excel database holding your transaction data need to have a column called "Country".**
- **The column headings in your database should be unique. If duplication in column heading exists, *STRUCTURIZER* will take the first column matching the dimension name.**
- **When you define range names, library names and dimension names cannot contain a dot (".") . The dot is only allowed as separator for the prefix "VM", the library name and dimension name (I.e.: VM.MyLibrary.MyDimension)**
- **You cannot use reserved library names, which are:**

- **"CDate": used for calendar date roll-up (see date period hierarchies)**
- **"FDate": used for fiscal date roll-up (see date period hierarchies), which are offset from the normal calendar year**
- **"Where": used for comparing operator statements (<, >, <>, <=, >=) on numeric, date or string data type dimensions.**
  The following formats are accepted:
  Where.MyNumberDimension.>=10.5 (for decimal separation use your decimal separator as per your local computer settings, i.e. "," or ".")
  Where.MyDateDimension.>=39084 (date as number format)
  Where.MyDateDimension.>=2 January 2007
  Recognizes date formats according to the locale setting of your system. Exceptions: The correct order of day, month, and year may not be determined if it is provided in a format other than one of the recognized date settings. In addition, a long date format is not recognized if it also contains the day-of-the-week string.
  The date format <u>should not be</u> "." (dot) delimited in the WHERE parameter (i.e. "WHERE.SalesDate.>01.02.2007")
- **"Like": used for textual comparisons (non case sensitive) on your dimension data (i.e. Like.MyDimension.Orang*)**

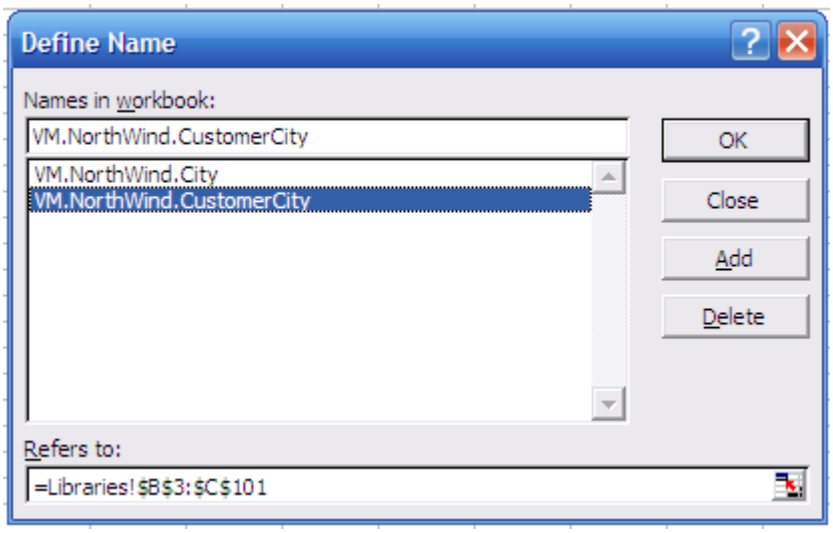
- **Dimension items are <u>not case sensitive and ignore leading and trailing spaces</u> when you load them into the hierarchy, as well as for the comparison with the data in the data analysis table. This presents a main advantage when "non clean" data analysis tables are used, which is often the case when data is extracted from operational databases.**

Tip 1: *You can create a dedicated library Excel file that stores all your libraries you commonly use. You can define in the options of STRUCTURIZER that this file is automatically loaded when you start-up Excel. Immediately all libraries from that file will be available for use in your spreadsheets. If you work in teams, this dedicated library file also gives the possibility, that the libraries need to be maintained only once.*

Tip 2*: You can define in the options of STRUCTURIZER to automatically search and load dimension libraries when an Excel file is opened. In case a dimension hierarchy has been loaded already with the same naming convention (library/dimension name) the new library will be loaded and "overwrite" the prior loaded dimension hierarchy.*

Tip 3*: Try to align your different Excel database's column headers to standard naming, so you can reuse dimensions you have defined. Example one report customer report calls the customer's city in the column heading "City" another sales report calls the customer's city in the column heading "Customer City". Standardize the reports column headings in both reports to the meaning full name "Customer City". You will not only be able to use your dimensional hierarchy for "Customer City" on both reports, but also when you use a STRUCTURIZER spreadsheet retrieve function on your analysis data table, the function will be much easier to read.*
Example: =VMDVAL($E$8:$P$5193,$J$8:$J$5193,"CustomerLibrary.Customer City.Europe")

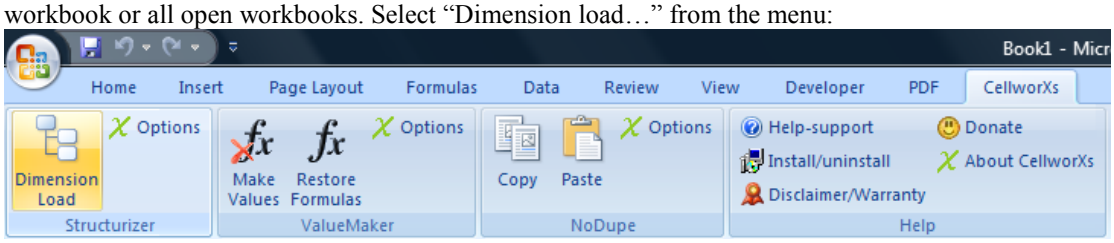
Tip 4*: In case you cannot as per Tip 3 standardize your analysis data table column headings of your different reports, you can define for the same spreadsheet range two (or more) different range names, reflecting your different column headings of your different reports. This way you would only need to maintain the hierarchy dimension once, but STRUCTURIZER would load them twice (or more often) with their different dimension names. The following picture shows two dimension libraries that refer to the same Excel worksheet range:*



Tip 5: *How to add a description to children (i.e. base items, but also at any other hierarchy level) which you can retrieve in your spreadsheet?*
When you define the child in you named table range, just map the child to it's description as parent. Then you map the description to the real parent. You can then retrieve via the VMSItemParent(child name) function the item description.
If you work with descriptions in your hierarchy you can use the function VMSItem2ndParent(child name) to retrieve the second level parent.

2.2.3.2 Loading dimensional hierarchies

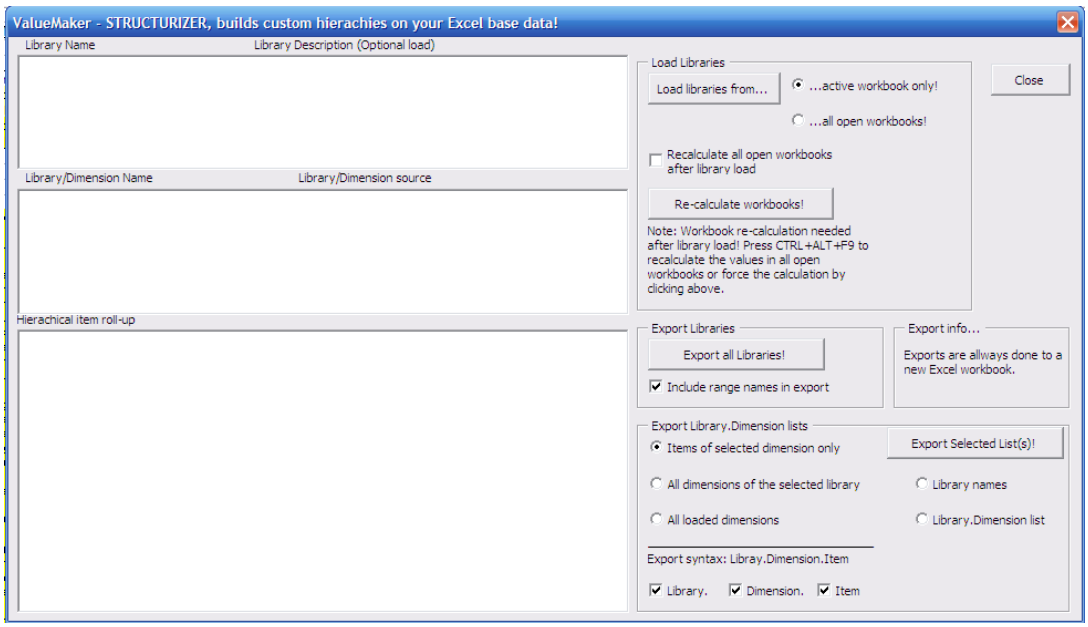
Once you have defined named ranges in *STRUCTURIZER's* format, they can be loaded from the active workbook or all open workbooks. Select "Dimension load…" from the menu:



(In Excel 2003 or earlier, this function sits under the CellworXs menu.)

Click the "Load from active workbook!", to load all dimensional hierarchies that are in the recognized STRUCTURIZER formatted named ranges. (Note: Alternatively you can also load libraries from the "STRUCTURIZER Building Options…")

Picture: STRUCTURIZER's dimensional hierarchy load and management menu



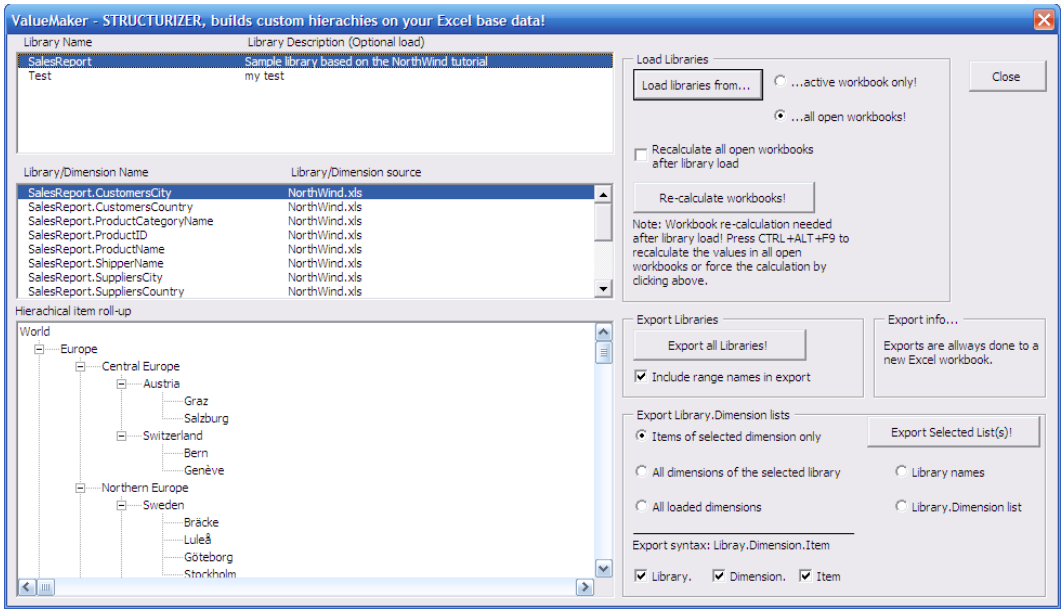
The load of the defined STRUCTURIZER hierarchies will also include hierarchy ranges that are defined on protected or hidden worksheets of the active workbook.

Some simple loading rules to remember:

- Whenever you change your hierarchy in your Excel range, you need to reload the hierarchy from the workbook into *STRUCTURIZER's* memory.
- You can reload as often as you want.
- You can close the file that contains your hierarchy definition after the load.
- Hierarchies stay loaded in *STRUCTURIZER's* memory and are available for reporting until Excel is closed.
- After a the library load you need to recalculate/refresh the workbooks containing formulas referring to any of the changed hierarchies. Press CTRL+ALT+F9 to recalculate the values in all open workbooks or force the calculation by clicking the "Re-calculate workbooks!" button. Pressing only F9 will not recalculate the VMS* formulas!
- An error log is generated if any loading error occurs (see following chapter).

- If you load the same "library/dimension name" from two different files, the first version will be replaced by the second version (last file loaded).
- *STRUCTURIZER* shows from which file a library has been loaded.
- It is possible to load library descriptions (see following chapter) for informative purposes.
- You can fore hold empty lines anywhere in your dimension ranges. STRUCTURIZER will ignore empty lines in the defined named ranges.

The following picture shows loaded hierarchies based on the "NorthWind trader" example.



Selecting a library name in the library list box will display all the dimensional hierarchies which are defined under that library, in the "library/dimension" list box.
Selecting a dimension name of the list box will display the hierarchy tree that you have defined. You can expand or collapse the nodes by clicking the +/- sign.

2.2.3.3 Loading library descriptions

You can optionally load library descriptions to further document your libraries. To load library descriptions define a Excel named range with the following name: VM.Libraries
The range need to consist of two columns, where the cell in the first column contains the library name and the second cell contains the library description. The following shows an example:

SalesReport	Sample library based on the NorthWind tutorial
AnotherReport	Another description ….

Libraries names that are mentioned in the library description table but that are not yet loaded in STRUCTURIZER are ignored from the load, as well as blank cells in the "VM.Libraries" range.
The library description range needs to be located in the file from which the libraries are loaded.
If for above example the "SalesReport" libraries and the "AnotherReport" libraries are located in different files, you need to generate in each file a VM.Libraries range in order to load the descriptions.

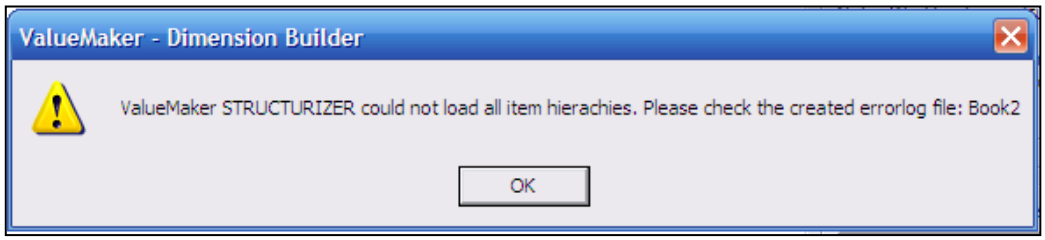
2.2.3.4 Error trapping when loading hierarchies

When you load dimensional hierarchies and parent-child relations are not defined according to the rules like, duplication of children, missing parents, etc. STRUCTURIZER will generate an Excel error log file that will help you to identify the root cause.

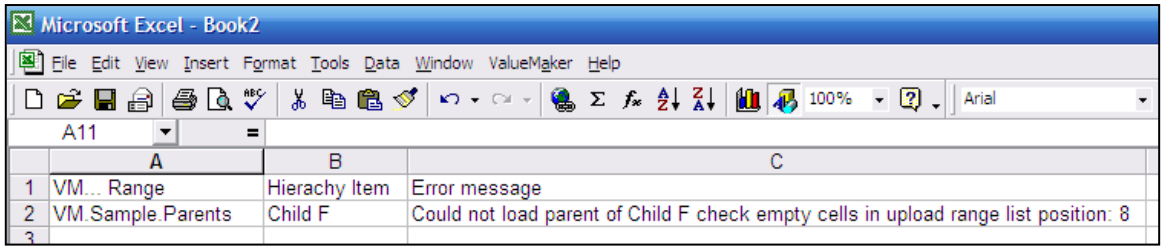
The following show a dimension hierarchy range that does not contain a parent for "Child F" :

All Parents	Parent of ABC
All Parents	Parent of DEF
Parent of ABC	Child A
Parent of ABC	Child B
Parent of ABC	Child C
Parent of DEF	Child D
Parent of DEF	Child E
	Child F

When loading the above range you will get the following error message, indicating that not all dimensions could be loaded and the file name of the generated error log:



The error log file will contain information about the issue that needs to be resolved as per the following example:



2.2.3.5 Exporting dimensional hierarchies or list item content

You can export your loaded library definition ranges or dimensional item lists (Menu: CellworXs / Structurizer Dimension load….).
This is useful:

- When you have generated several libraries in different Excel file and you would like to generate a summary library document.
- When you want to generate "data validation" lists as per Excel's menu, to build user applications that allow easy multidimensional data drill.

All exports are made to a new Excel workbook.

2.3 How to manage consistent base analysis data

This chapter explains how to best manage your analysis data integrity caused by:

- **Missing hierarchy items** that are present in your base analysis data table, but not defined in your hierarchy definition.
- The **roll-up of blank cells** in your base analysis data table.

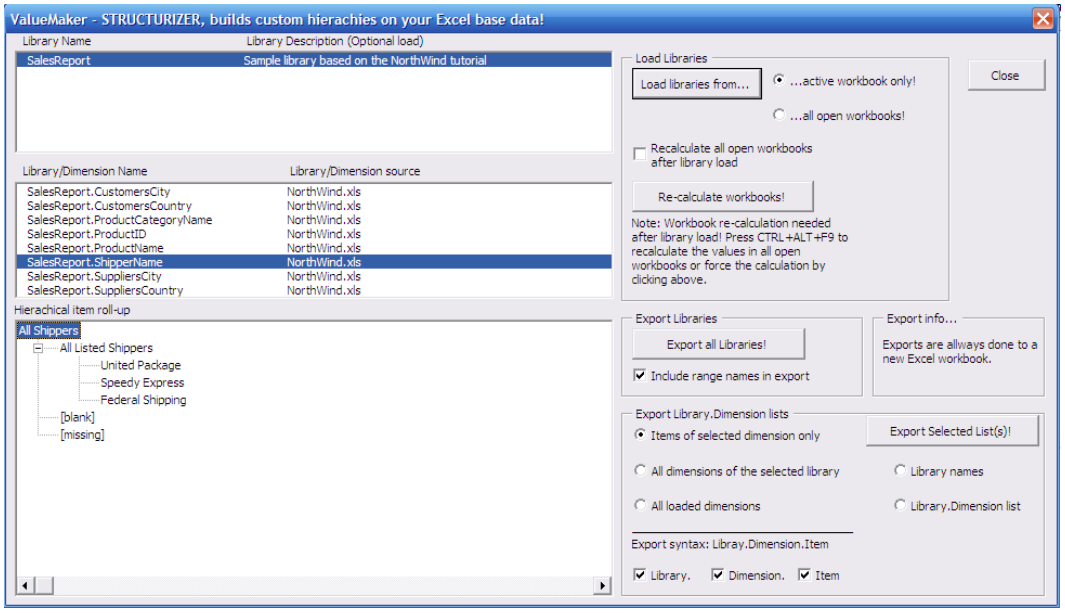
Missing and blank items can be mapped as any other base dimension item into your hierarchical dimension definition. The reserved key words are (non-case sensitive):

- [Missing]
- [Blank]

The following shows a hierarchy definition to map blank and missing items based on our NorthWind trader example as Excel table….

All Shippers	All Listed Shippers
All Shippers	[blank]
All Shippers	[missing]
All Listed Shippers	United Package
All Listed Shippers	Speedy Express
All Listed Shippers	Federal Shipping

…and loaded into *STRUCTURIZER*….



<u>In addition there are several worksheet functions defined that allow you to test your data integrity especially for missing dimension items.</u> Please see for further information the chapter of *STRUCTURIZER's* worksheet functions.

2.3.1 How to manage missing items not defined in your hierarchy

It can be on purpose or by neglecting that one or several dimensional items are not defined and loaded in its belonging dimensional hierarchy. To assure analysis data integrity when missing items occur you can manage as follows:

- If needed map [MISSING] just below the top level parent. This allows to assure all items are included in the top level parent and you can also retrieve the value for missing items.
- Use the VMSMissingItems() function, the VMSMissingItem() function or the VMSMissingItemsSlashList() function on a standalone base to check the integrity of your database with your loaded hierarchies. Correct and reload your hierarchies accordingly.
- In the VMSDVAL() function you can use the [MISSING] as stand alone without being mapped in the hierarchy. The Syntax to use is: Mylibary.MyDimension.[Missing]
  It will perform and return all values related to [MISSING] dimension items.
  Note: Any other dimension item that is NOT defined and loaded in the library/dimension, but is used in the VMSDVAL() function would return the following error message:
  [DIMENSION ITEM: MYLIBRARY.DIMENSION.[item name] IS NOT DEFINED OR LOADED]

2.3.2 How to manage blank cell roll-up

Often data dimensions are characterized by optional data, meaning that your data analysis table column contains values but also blank cells. As prior mentioned, blank cells can be rolled up in a hierarchy or they can be stand alone queried without being mapped into the hierarchy of the dimension.

In the VMSDVAL() function you would reference the [BLANK] cells as follows:
Mylibary.MyDimension.[BLANK]
It will include all [BLANK] dimension items (empty cells) as matching condition in the VMSDVAL() function.

2.4 How to manage hierarchical roll-up of date values / periods

Periodical data can be organized in different formats in your base analysis data table as:

- Date data type in a table column like 01/01/2007
- Non-date data type but representing a date per its definition. An example would be if your data table contains separate columns for the year value, month value,…

The following table shows an example for both cases:

TransactionDate	FiscalYear	FiscalMonth
1/1/2007	2007	1
1/2/2007	2007	1
1/3/2007	2007	1
1/4/2007	2007	1

You might also come across permutations where the month (or quarters, trimesters, weeks…) are represented in columns as follows:

TransactionDate	FiscalYear	1	2	3	4	5	6	7	8	9	10	11	12
1/1/2007	2007	0.29	0.84	0.56	0.44	0.04	0.31	0.95	0.43	0.03	0.20	0.21	0.08
1/2/2007	2007	0.13	0.91	0.84	0.62	0.48	0.53	0.07	0.21	0.68	0.24	0.40	0.55
1/3/2007	2007	0.73	0.27	0.40	0.24	0.34	0.03	0.30	0.30	0.64	0.09	0.62	0.92
1/4/2007	2007	0.42	0.70	0.08	0.57	0.32	0.83	0.87	0.23	0.82	0.88	0.01	0.63

With STRUCTURIZER you can principally manage to report on all different analysis base table formats.

You can do a custom mapping of dates like with any other hierarchy dimension; however STRUCTURIZER provides build-in hierarchical roll-up functionality for date data type dimensions as mentioned in the following chapters:

2.4.1 "CDate" Library – Calendar year date library

The "CDate" library refers to the selected calendar year and allows the hierarchical grouping within a calendar year.
In a STRUCTURIZER worksheet function you would refer to a year-to-date "CDate" hierarchy as follows: CDate.MyDimension.YTD*yourdatevalue*

The "CDate" library accepts all date values in date formats that Excel recognizes. Some examples would be:
Date recognisable spelling: CDate.MyDimension.YTDFebruary 12, 2007
Date value formats: CDate.MyDimension.YTD39447
Date and time value formats: CDate.MyDimension.YTD 39106.8872820602

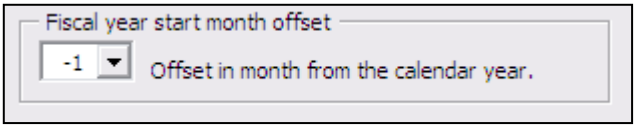
Type:	**Description**	1st day of the period where the date belongs to:	Period hierarchy example: CDate.MyDimension.[Type]August 15, 2007
YTD	Year-to-date	1st of January	1/1/2007 – 15/8/2007
STD	Semester-to-date	1st day of semester (1/1 or 1/6)	1/6/2007 – 15/8/2007
TTD	Trimester-to-date	1st day of trimester (1/1, 1/5, 1/9)	1/5/2007 – 15/8/2007
QTD	Quarter-to-date	1st day of quarter (1/1, 1/4, 1/7, 1/10)	1/7/2007 – 15/8/2007
MTD	Month-to-date	1st day of the month	1/8/2007 – 15/8/2007
WTD	Week-to-date	1st day of the week (Monday)	13/8/2007 – 15/8/2007
DTD	Day-to-date	The selected day	15/8/2007 – 15/8/2007

2.4.2 "FDate" Library – Fiscal year date library

The FDate library gives the possibility to offset the fiscal year from the calendar year, by the defined number of month. The offset is defined in the user preference. The offset can range from -11 to 11.
Example:

- An offset of the calendar year 2007 by -1 would let the fiscal year 2007 start on December 1st 2006 and end on November 30th 2007. Respectively all period breakdowns covering more than a month (YTD, STD, TTD, QTD) will be offset by one month. I.e. the second quarter of fiscal 2007 starts on March 1st.
- An offset of the calendar year 2007 by -1 would let the fiscal year 2007 start on December 1st 2006 and end on November 30th 2007.

Picture: Defining the fiscal year offset in the user preferences:



The FDate library follows the same definition rules as the CDate library. In a *STRUCTURIZER* worksheet function you would refer to a year-to-date "FDate" hierarchy as follows: FDate.MyDimension.YTD*yourdatevalue*

2.4.3 Period data organised in 12 monthly columns

Often the monthly data is organized in 12 "monthly" columns in the data analysis table, where eventually the year is mentioned in an additional column. *STRUCTURIZER* provides the VMSD12PVAL() function in order to allow consolidated hierarchical reporting. The function follows the same rules like the VMSDVAL(), but required in addition the following parameter:

- Period: month number from 1 to 12
- Frequency: to define a year-to-date or any other available view like: YTD, STD, TTD, QTD and MTD
- Period range: The 12 period column range

For details on the VMSD12PVAL() function please see the chapter about the *STRUCTURIZER*'s function descriptions.

2.5 Other reserved library names for analysis purposes

2.5.1 "WHERE" library

The "WHERE" library name is reserved for analytical purposes for the VMSVAL() function, to query the analysis base data table on columns containing numerical and textual data. When using a WHERE, the comparisons will interpret the data in the dimension range as either numbers, dates or strings. It is the value being passed on in the WHERE that will determine if the data in the dimension is handled as a number, a date or a string. E.g. "WHERE.columnName.<12345" will handle the data in column <columnName> as numbers, because 12345 can be interpreted as a number. For a dimension range being handled as a number, an empty cell we be considered equal to 0. For a dimension range being handled as a date, an empty cell will be considered equal to 30 Dec 1899.

The following shows the syntax to reference i.e. the "Discount" column in your data analysis table you would use the following syntax in your VMSVAL() function as parameter option:

WHERE.Discount.>0.20

Note: Enter the decimal separator according to you local country settings, which could be a comma (",") or dot (".").

The following comparison operator can be used: =, >, <, <>, >=, <= in the "WHERE" statement.

Filtering by using "WHERE" is not case sensitive and ignores leading and trailing spaces.

2.5.2 "LIKE" library

The "LIKE" library name is reserved for analytical purposes for the VMSVAL() function, to query the analysis base data table on columns containing textual data. The "LIKE" statement performs with a slower calculation time than the "WHERE" statement, however the "LIKE" library allows "*" and "?" place holders for the textual comparison.

The following shows the syntax to reference i.e. the "SupplierName" column in your data analysis table you would use the following syntax in your VMSVAL() function as parameter option:

LIKE.SupplierName.Leka Trading

Filtering by using "LIKE" is not case sensitive and ignores leading and trailing spaces.

2.6 STRUCTURIZER's worksheet functions overview

STRUCTURIZER provides a large number of worksheet functions to:

- get information about the dimension item(s) and their hierarchical organisation
- test the data integrity of defined hierarchies versus the different dimension items of the data analysis table
- retrieve a consolidated multidimensional item total from the data analysis table

In addition STRUCTURIZER provides many descriptive error return values in case non valid function parameter are entered.

All functions are also documented in the "StructurizerQuickGuide.xls" tutorial file. In below samples the function parameter input are yellow and the formula result green.

2.6.1 Functions to get information about the dimension items and their hierarchical organisation

2.6.1.1 VMSDirectChilds()

Function to get the number of direct children of a dimension item, which are just mapped below the selected item in the hierarchy.
Syntax: VMSDirectChilds(DimensionItem)
DimensionItem: A reference to a dimension item or a hard coded string referring to a loaded dimension item.

Sample as per the StructurizerQuickGuide.xls tutorial file:

The dimension item:	has the following number of direct children:
SalesReport.CustomersCity.Europe	**5**

2.6.1.2 VMSDirectChild()

Function to return a specific child dimension item name of a selected parent dimension item.
Syntax: VMSDirectChild(DimensionItem, ChildNumber)
DimensionItem: A reference to a dimension item or a hard coded string referring to a loaded dimension item.
ChildNumber: A number from 1 to the total number of children below the selected dimension item. If the ChildNumber entered is above the actual number of children the following error message will appear (sample): [MAX DIRECT CHILDS BELOW ITEM Europe: 5]

Sample as per the StructurizerQuickGuide.xls tutorial file:

The direct child number:	of the dimension item:	is the following dimension item:
3	SalesReport.CustomersCity.Europe	**Southern Europe**

ATTENTION: this function may result in different return values for the same ChildNumber depending on the order how the dimensions are defined when the ranges are loaded and if additional items were added afterwards.

2.6.1.3 VMSBaseChilds()

Function to retrieve the number of base items below the specified dimension item. Base items are the lowest items (with no children) in the dimensional hierarchy. They are as well the items that are in the different dimensions of the data analysis table.
Syntax: VMSBaseChilds(DimensionItem)
DimensionItem: A reference to a dimension item or a hard coded string referring to a loaded dimension item.

Sample as per the StructurizerQuickGuide.xls tutorial file:

The dimension item:	has the following number of base dimension item children:
SalesReport.CustomersCity.Europe	**59**

2.6.1.4 VMSBaseChildsSlashList()

Function to retrieve a slash separated list of the base items below the specified dimension item.
Syntax: VMSBaseChildsSlashList(DimensionItem)
DimensionItem: A reference to a dimension item or a hard coded string referring to a loaded dimension item.

Sample as per the StructurizerQuickGuide.xls tutorial file:

The dimension item:	has the following number of base dimension item children:
SalesReport.CustomersCity.Spain	**\Barcelona\Madrid\Sevilla\Oviedo**

2.6.1.5 VMSBaseChild()

Function to get a specific base item name of a selected parent dimension item.
Syntax: VMSBaseChild(DimensionItem, ChildNumber)
DimensionItem: A reference to a dimension item or a hard coded string referring to a loaded dimension item.
ChildNumber: A number from 1 to the total number of base children below the selected dimension item. If the ChildNumber entered is above the actual number of base children the following error message will appear (sample): [MAX DIRECT CHILDS BELOW ITEM Europe: 59]

Sample as per the StructurizerQuickGuide.xls tutorial file:

The base child number:	of the dimension item:	is the following dimension item:
3	SalesReport.CustomersCity.Europe	**Bruxelles**

ATTENTION: this function may result in different return values for the same ChildNumber depending on the order how the dimensions are defined when the ranges are loaded and if additional items were added afterwards.

2.6.1.6 VMSDirectParent()

Function to get the parent dimension item of a selected child dimension item.
Syntax: VMSDirectParent(DimensionItem, ChildNumber)
DimensionItem: A reference to a dimension item or a hard coded string referring to a loaded dimension item.

Sample as per the StructurizerQuickGuide.xls tutorial file:

The dimension item:	has the following parent:
SalesReport.CustomersCity.Europe	**World**

2.6.1.7 VMSItem2ndParent()

Function to get the 2nd level parent ("parent's parent") of the selected dimension item.
Syntax: VMSItem2ndParent(DimensionItem)
DimensionItem: A reference to a dimension item or a hard coded string referring to a loaded dimension item.

Sample as per the StructurizerQuickGuide.xls tutorial file:

The dimension item:	has the following 2nd level parent:
SalesReport.CustomersCity.Western Europe	**World**

2.6.1.8 VMSItemRollUp()

Function to get the entire hierarchy roll-up path (as slash delimited list) of a selected dimension item.
Syntax: VMSItemRollUp(DimensionItem)
DimensionItem: A reference to a dimension item or a hard coded string referring to a loaded dimension item.

Sample as per the StructurizerQuickGuide.xls tutorial file:

The dimension item:	has the following roll-up path:
SalesReport.CustomersCity.Paris	**World\Europe\Western Europe\France\Paris**

2.6.2 Functions to test the data integrity of hierarchies and data analysis table

The following explains functions that test the data integrity of loaded hierarchies and the data analysis table. The examples are based on the hierarchies as defined in the *StructurizerQuickGuide.xls* tutorial file and the below data analysis table:

Sample as per the StructurizerQuickGuide.xls tutorial file:

Sample data analysis table:

CustomerName	CustomersCity	TotalSales1996
Island Trading	Cowes	901.20
North/South	London	-
Rattlesnake Canyon Grocery	Albuquerque	10,861.60
Old World Delicatessen	Anchorage	5,091.50
Save-a-lot Markets	Boise	6,155.90
The Cracker Box	Butte	-
Hungry Coyote Import Store	Elgin	338.00
Great Lakes Food Market	Eugene	-
Trail's Head Gourmet Provisioners	Kirkland	-
Split Rail Beer & Ale	Lander	8,400.20
Lonesome Pine Restaurant	Portland	712.00
Let's Stop N Shop	San Francisco	-
White Clover Markets	Seattle	3,532.00
Lazy K Kountry Store	Walla Walla	-
Island Trading 2	Hamburg City	4,560.00
New "The Cracker Box" Branch	Miami	8,352.00

The above data analysis table contains two dimension items in the "CustomerCity" dimension that are not defined and loaded StructurizerQuickGuide.xls tutorial file:

1. Hamburg City
2. Miami

The following formula examples will show how to spot the data integrity issue, to assure you always show coherent data analysis.

2.6.2.1 VMSMissingItems ()

Function to test how many items of a specific dimension contained in the data analysis table are missing in the loaded hierarchy.
Syntax: VMSMissingItems(DimensionRange, LibraryDimension)
DimensionRange: A reference to a range that contains the dimension. A column header of the range has to be of the same spelling as the defined "LibraryDimension"
LibraryDimension: A reference to a dimension or a hard coded string referring to a loaded dimension.

Sample as per the StructurizerQuickGuide.xls tutorial file:

The dimension:	has the following number of missing items not defined in the loaded hierachy:
SalesReport.CustomersCity	**2**

Note: If [BLANK] is not defined as dimension item in the library/dimension and exists in the appropriate column of the data analysis table, it will be handled like a missing item and increase the missing item counter by one.

2.6.2.2 VMSMissingItem ()

Returns the item name that is missing in the loaded dimension hierarchy but which is part of the data analysis table dimension.
Syntax: VMSMissingItem(DimensionRange, LibraryDimension, ItemNumber)
DimensionRange: A reference to a range that contains the dimension. A column header of the range has to be of the same spelling as the defined "LibraryDimension"
LibraryDimension: A reference to a dimension or a hard coded string referring to a loaded dimension.
ItemNumber: A number from 1 to the total number of missing items not declared. If ItemNumber is taller than the total number of missing items, the return value of the function is "blank".

Sample as per the StructurizerQuickGuide.xls tutorial file:

The missing item number:	of the dimension:	has the following name:
1	SalesReport.CustomersCity	**Hamburg City**

2.6.2.3 VMSMissingItemsSlashList ()

Function to get a "slashed" delimited list of all missing dimension items of the selected library/dimension.
Syntax: VMSMissingItemsSlashList(DimensionRange, LibraryDimension)
DimensionRange: A reference to a range that contains the dimension. A column header of the range has to be of the same spelling as the defined "LibraryDimension"
LibraryDimension: A reference to a dimension or a hard coded string referring to a loaded dimension.

Sample as per the StructurizerQuickGuide.xls tutorial file:

The dimension:	has the following number of missing items in the list, but not defined in the loaded hierachy:
SalesReport.CustomersCity	**\Hamburg City\Miami**

2.6.2.4 VMSMissingListItems ()

Function to test how many items of a list are missing in the specified loaded dimension hierarchy.
Syntax: VMSMissingListItems(DimensionListRange, LibraryDimension)
DimensionListRange: A reference to a single column range that contains the dimension. There is no column header required that has the same spelling as the defined "LibraryDimension". The function will test the selected range against the defined loaded library dimension.
LibraryDimension: A reference to a dimension or a hard coded string referring to a loaded dimension.

Sample as per the StructurizerQuickGuide.xls tutorial file:

Sample list containing "Hamburg City" and "Miami" which are not defined / loaded in the dimension "SalesReport.CustomerCity":

Cowes
London
Albuquerque
Anchorage
Boise
Butte
Elgin
Eugene
Kirkland
Lander
Portland
San Francisco
Seattle
Walla Walla
Hamburg City
Miami

The dimension:	has the following number of missing items in the list, but not defined in the loaded hierachy:
SalesReport.CustomersCity	**2**

2.6.2.5 VMSMissingListItem()

Returns the item name that is missing in the loaded dimension hierarchy but that is part of the data analysis table dimension.
Syntax: VMSMissingListItem(DimensionListRange, LibraryDimension, ItemNumber)
DimensionListRange: A reference to a single column range that contains the dimension. There is no column header required that has the same spelling as the defined "LibraryDimension". The function will test the selected range against the defined loaded library dimension.
LibraryDimension: A reference to a dimension or a hard coded string referring to a loaded dimension.
ItemNumber: A number from 1 to the total number of missing items not declared. If ItemNumber is taller than the total number of missing items, the return value of the function is "blank".

The following sample is referring to the list mentioned in the prior formula description.

Sample as per the StructurizerQuickGuide.xls tutorial file:

The missing item number:	of the dimension:	has the following name ion the defined list:
1	SalesReport.CustomersCity	**Hamburg City** (this formula refers to the table above)

2.6.3 Functions to retrieve a consolidated multidimensional item total or other statistical information from the data analysis table

STRUCTURIZER provides several multidimensional data retrieve functions:

- **Sum** all values where the dimension criteria are matching
- **Count** the occurrences where the dimension criteria are matching
- Establish a **maximum** and **minimum** value where the dimension criteria are matching
- Establish an **average** value where the dimension criteria are matching

In addition STRUCTURIZER provides for data analysis tables that have their value data (i.e. monthly sales) organized in 12 monthly columns, a function to dynamically select the retrieve period, like year-to-date, quarter-to-date, month-to-date, etc.

2.6.3.1 VMSDVAL()

VMSDVAL is a powerful function to retrieve a total sum of a range under the condition that all conditional parameter are matched. Up to 27 dimensional criteria can be defined and combined:

- Hierarchical as per your loaded hierarchies, i.e. Salesreport.CustomerName.Europe
- Text comparison parameter (incl. wildcards), i.e. LIKE.CustomerName.Mill*
- Numeric comparison, i.e. WHERE.SalesDiscount.>0.2
- Periodic calendar year reporting, i.e. CDate.TransactionDate.QTD31 March 2007
- Periodic fiscal year reporting in case of an offset from the fiscal year versus the calendar year, i.e. FDate.TransactionDate.YTD31 March 2007

Data cleansing tip: You can ignore blank lines by having a WHERE.CUSTOMERCITY.<>[BLANK] statement.
For more information regarding the comparison parameter see chapter 2.4 and 2.5

Syntax: VMSDVAL(DimensionRange, SumRange, DimItem1, Optional DimItem2,…. Optional DimItem27)
DimensionRange: A reference to a range that contains the dimension. A column header of the range has to be of the same spelling as the defined "LibraryDimension". The SumRange can be part of the dimension range.
SumRange: Is the range you want to total up. You can enter either:

- a valid cell range which should include the table header line.
  Example: VMSDVAL('Transactions'!$B$6:$Z$2165,'Transactions'!$P$6:$P$2165,E2,E3,E4)
  The sum range can be one to multiple columns (next to each other). I.e. in case your sum range contains 3 columns and all dimension criteria are matched, VMSDVAL will take the total of the 3 cells of that row.

- the column header description of the data analysis table. If the header description is used, the SumRange has to be within the DimensionRange.
  Example: VMSDVAL('Transactions'!$B$6:$Z$2165,"TotalSales",E2,E3,E4)
  If the SumRange is referenced as column header name, the sum range will be by default one column only.

DimItem1 (to optionally DimItem27): A reference to a dimension item or a hard coded string referring to a loaded dimension item. Dimension items are referenced with the following syntax:

- Hierarchical items: MyLibrary.MyDimension.MyItem
- Text comparison items: LIKE.MyDimension.Comparetext (including wildcards: *, ?)
- Numeric comparison: WHERE.MyDimension.ComparingOperator&Value
  The comparing operator can be: >, <, <>, =, >=, <=
  Value: any numeric value. For "percent" comparison use a decimal value,
- Date hierarchies for calendar date or fiscal date (offset from calendar year by +/- number of month) reporting: CDate.MyDimension.PeriodRange&Date
  The PeriodRange can be: YTD, STD, TTD, QTD, MTD, WTD, DTD
  The Date can be any Excel recognizable date format (also numeric).

"MyDimension" has to be a loaded hierarchical dimension and a valid column header name within the DimensionRange.

2.6.3.2 VMSDCOUNT()

Counts the number of row occurrences matching the up to 27 dimensional criteria.

Syntax: VMSDCOUNT(DimensionRange, DimItem1, Optional DimItem2,…. Optional DimItem27)

It operates the same way as the VMSDVAL() function, except that no SumRange is entered.

2.6.3.3 VMSDAVG()

Returns the average value of a sum range divided by its row occurrences based on up to 27 dimensional criteria.

Syntax: VMSDAVG(DimensionRange, SumRange, DimItem1, Optional DimItem2,…. Optional DimItem27)

It operates the same way as the VMSDVAL() function.

2.6.3.4 VMSDMAX()

Returns the maximum value of a sum range based on its row occurrences based on up to 27 dimensional criteria.

Syntax: VMSDMAX(DimensionRange, SumRange, DimItem1, Optional DimItem2,…. Optional DimItem27)

It operates the same way as the VMSDVAL() function.

2.6.3.5 VMSDMIN()

Returns the minimum value of a sum range based on its row occurrences based on up to 27 dimensional criteria.

Syntax: VMSDMIN(DimensionRange, SumRange, DimItem1, Optional DimItem2,…. Optional DimItem27)

It operates the same way as the VMSDVAL() function.

2.6.3.6 VMSD12PVAL()

Returns the total of a sum range organized in 12 (monthly) period columns based on up to 25 dimensional criteria. Depending on the period and frequency selection the appropriate columns will be included in the returned total. This functionality allows a highly dynamic browsing through different period ranges.

Syntax: VMSD12PVAL(DimensionRange, TwelvePeriodCol, Period, Frequency, DimItem1, Optional DimItem2,…. Optional DimItem25)

It operates the same way as the VMSDVAL() function, for the "DimensionRange" and "DimItem".

TwelvePeriodCol: Is the range of 12 columns that is taken into account for the evaluation to total up (the headers of the data analysis table needs to be included in the range).
Period: Is a number from:

- 1 to 12 if the frequency is "M" (monthly) i.e. "M.YTD"
- 1 to 4 if the frequency is "Q" (quarterly) i.e. "Q.YTD"
- 1 to 3 if the frequency is "T" (trimester) i.e. "T.YTD"
- 1 to 2 if the frequency is "S" (semester) i.e. "S.YTD"
- 1 if the frequency is "Y" i.e. "Y.YTD"

The period is used as reference for the period range from which the total will be returned if all the other conditional parameter (DimItem) are matching. Depending of the selected frequency, either the month, quarter-to-date, trimester-to-date, semester-to-date or year-to-date value is returned.

Frequency: This is a combination of frequency and data view and needs to be entered in the following format sample: M.YTD (meaning "monthly year-to-date"). The following frequencies and data views can be defined/combined:

Combination grid:	MTD	QTD	TTD	STD	YTD
M	M.MTD	M.QTD	M.TTD	M.STD	M.YTD
Q		Q.QTD		Q.STD	Q.YTD
T			T.TTD		T.YTD
S				S.STD	S.YTD
Y					Y.YTD

The frequency for M, Q, T, S and Y will automatically select the correct number of columns from the 12 period columns. Example: if frequency is "Q", STRUCTURIZER recognizes that the selection will be 3 columns. In combination with the period, it will know if columns 1-3, 4-6, 7-9 or 10-12 need to be taken as sum range.

2.7 Error return values of worksheet functions
STRUCTURIZER provides a large number of error return values when functional parameter are not correctly entered or hierarchical dimensions are referenced in a formula, but not loaded into STRUCTURIZERS memory.

The following gives some examples for error return values:

If a dimension item is referenced in a formula but not loaded in the STRUCTURIZER memory, the following error value will be returned:
[DIMENSION ITEM: SALESREPORT.CUSTOMERSCITY.EUROPES IS NOT DEFINED OR LOADED]

If a dimension is referenced in a formula but not part of the data analysis tables dimension range, the following error value will be returned:
[Dimension: CUSTOMERSCITI is not a column header of the DimensionRange]

If a dimension is referenced in a formula but not loaded in the STRUCTURIZER memory, the following error value will be returned:
[LIBRARY OR DIMENSION NOT DEFINED, CORRECT SPELLED OR LOADED]

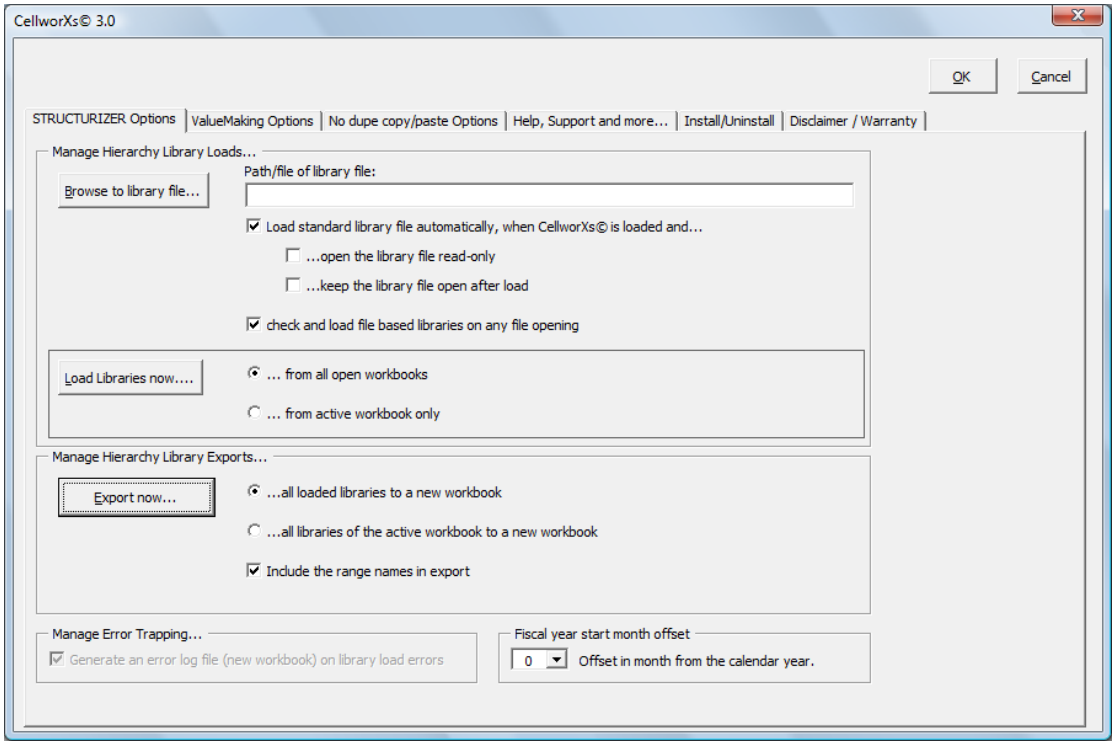
We have given our best to capture as many as possible error return values that will help you to trouble shoot your functions. If you believe some additional error return values are needed, please send us your case and we will investigate to add those.

2.8 STRUCTURIZER user options

STRUCTURIZER provides a series of user options that allow to:

- manage hierarchy library loads
- manage hierarchy library exports
- define the fiscal year offset from the calendar year

All user options are individual options kept in the users preference file.

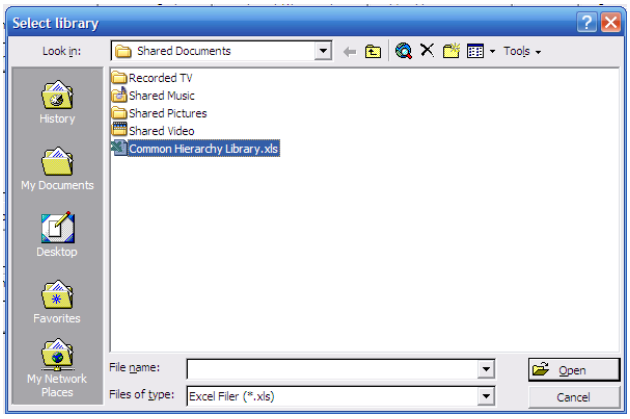


2.8.1 Manage and automate hierarchy library loads

You have several options to manage and automate the loading of your hierarchy ranges:

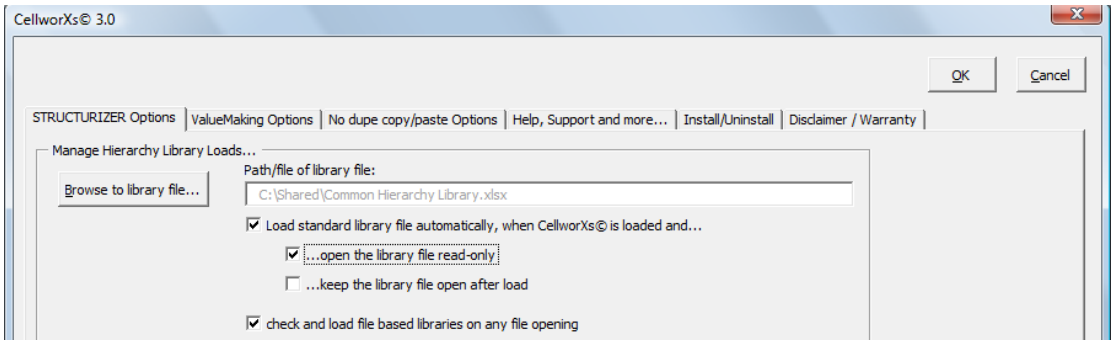
- **Defining a standard library file**: STRUCTURIZER provides the possibility to point to a standard library file **and automatically load the libraries from that file** when you open Excel. This is recommended when you want to re-use dimensional hierarchy table definitions in different Excel reports. You define and maintain the hierarchy once in one location and you can share it with other users. To facilitate multi-user sharing you can define to open the standard file "read-only" and if the file should be kept open after the load. In order to define the path/file of the standard library file, click the "browse to library file…" and select the Excel file in your file browser window and click open. The file name and path will be taken into your user options. See also next picture.
- **Automatically load hierarchical libraries when a file is opened**: This option will automatically check for libraries and load them if you open an Excel file. This is useful, as you do not need to remember each time to manually load the libraries that are defined in a file each time you open a file.
- **Loading of libraries**: You have the option to load hierarchy libraries from all open files or from the active workbook only. Library load is also possible from the menu "Dimension load…".

Selecting a standard library file:



Note: the browser window will not show the selected file in the "File name" drop down box. This is to ignore, just select the file and click "Open" to point to your standard library file. The file will now been shown with its full path in the user options:

Defining a common library file:



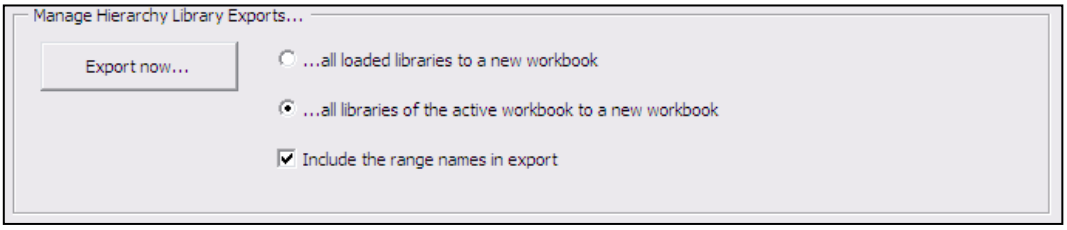
2.8.2 Export of loaded or file based hierarchy libraries

You have the possibility to export from the STRUCTURIZER user options:

- all loaded libraries
- all libraries defined in the active workbook (file based).

Exports are always done to a new workbook. Optionally you can define if the range names should be included in the export. The export is also possible from the menu "Dimension load…" with some more features.

Loaded library export options:



The libraries are exported with the following rules for loaded libraries:

- the first row contains the full range / library name (VM.[LibraryName].[DimensionName])
- The first export starts in column 'C"
- Between each exported library there is a blank column

Export of loaded library:

	A	B	C	D	E	F	G
1	First row for information only: VM.[LibaryName].[DimensionName]		VM.SALESREPORT.CUSTOMERSCITY			VM.SALESREPORT.PRODUCTCATEGORYNAME	
2			World	Europe		All Products	All Listed Products
3			World	America		All Products	Products to Investigate
4			World	Asia & Australia		All Listed Products	Beverages
5			World	[blank]		All Listed Products	Condiments
6			World	[missing]		All Listed Products	Confections
7			Europe	Central Europe		All Listed Products	Dairy Products
8			Europe	Northern Europe		All Listed Products	Grains/Cereals
9			Europe	Southern Europe		All Listed Products	Meat/Poultry
10			Europe	Eastern Europe		All Listed Products	Produce
11			Europe	Western Europe		All Listed Products	Seafood
12			America	North America		Products to Investigate	[MISSING]
13			America	South America		Products to Investigate	[Blank]
14			Asia & Australia	Japan			
15			Asia & Australia	Singapore			
16			Asia & Australia	Australia			
17			Central Europe	Austria			
18			Western Europe	Belgium			

File based library export is grouping all VM.* named ranges in a new workbook, in the same format as the loaded libraries but also adding the source file.

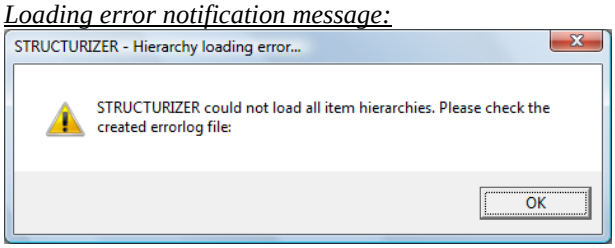
Export of file based library:

	A	B	C	D	E	F	G
1			VM library: VM.SalesReport.CustomersCity			VM library: VM.SalesReport.ProductCategoryName	
2			Source: NorthWind.xls			Source: NorthWind.xls	
3							
4			World	Europe		All Products	All Listed Products
5			World	America		All Products	Products to Investigate
6			World	Asia & Australia		All Listed Products	Beverages
7			World	[blank]		All Listed Products	Condiments
8			World	[missing]		All Listed Products	Confections
9			Europe	Central Europe		All Listed Products	Dairy Products
10			Europe	Northern Europe		All Listed Products	Grains/Cereals
11			Europe	Southern Europe		All Listed Products	Meat/Poultry
12			Europe	Eastern Europe		All Listed Products	Produce
13			Europe	Western Europe		All Listed Products	Seafood
14			America	North America		Products to Investigate	[MISSING]
15			America	South America		Products to Investigate	[Blank]
16			Asia & Australia	Japan			
17			Asia & Australia	Singapore			
18			Asia & Australia	Australia			

2.8.3 Error trapping

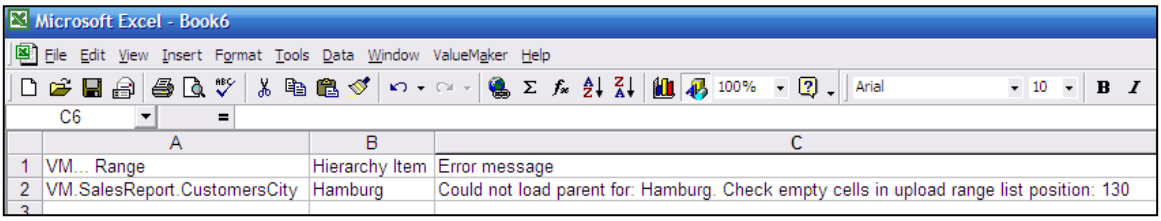
This option can currently not be disabled. The error trapping is used to point to errors when libraries are loaded in order to quickly identify the potential loading issue. When one or more loading errors occur, a new Excel workbook is generated, containing a detailed error description. At the end of the loading process an error message will alert you and point to the generated error log file.

Loading error notification message:



The error file will mention the range name, hierarchy item and error message as per the next sample:

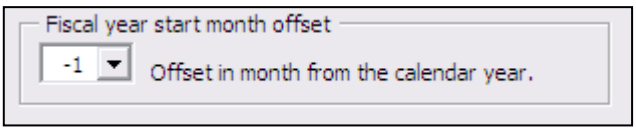
Sample error log file:



2.8.4 Fiscal year offset definition

The fiscal year offset is used for the "FDate" Library (Fiscal year date library), to report on periods which are offset from the calendar year. You can define a –11 month to +11 month fiscal year offset.

Fiscal year offset definition:



What is the difference between a –11 and +1 offset, as in both cases the fiscal year would start in February? The difference will show in the fiscal year as per below example of a sample date, with a in February starting fiscal year:

Offset in month	Sample date	Fiscal year
-11	1st of May 2007	2008
+1	1st of May 2007	2007

2.9 Case inspiration: Building dynamic retrieve reporting applications (i.e. for Hyperion Enterprise)

If your local consolidation or accounting system provides the possibility to extract data based their proprietary Excel retrieve add-in, you can use STRUCTURIZER's functionality to build hierarchies and to return the relation ships of dimension items to their parents, children and roll-up path to build dynamic reporting.

An example would be the Hyperion Enterprises Excel retrieve module, which allows to return values from their consolidation database via the "HPVAL()" function and to return for dimension items like entities or accounts their description via the "HPFUL()" or "HPHEA()" function.

You can use STRUCTURIZER to load the hierarchical entity structure. You would be able to quickly establish in Excel with the functions to elaborate the loaded STRUCTURIZER hierarchies (i.e. VMSDirectChilds(),VMSDirectChild(), VSDirectParent,…) a dynamic Excel retrieve application that let's you browse the hierarchy structure.

3. ValueMaker - Formula conversion & restore

3.1 Functional overview and conceptual introduction

CellworXs™ *'s ValueMaker* replaces Excel cell formulas based on a user-defined formula syntax / string with their calculated cell values.
Example: If your replacement definition is "VMS", because you want to share a report that you have generated with *STRUCTURIZER™* 's hierarchy reporting capabilities, ValueMaker will replace all cells that contain formulas with "VMS" as part of their formula string with values.
Once the cell is valued any other user not having the CellworXs™ add-in installed will be able to see the numbers and it avoids the '#VALUE!' error in the report's spreadsheet cells if the other user refreshes / recalculates the file.

All other formulas in cells not matching the defined replacement criteria are not impacted. This way you still keep a formula audit trail of the remaining cells. Prior valued formulas can be restored even after having saved the workbook.
Therefore ValueMaker solves especially the problem of viewing / sharing Excel retrieve files and reports with other users who do not have, or should not have, access to:

- the detailed source data; i.e. in case of Excel retrieve database functions from products like Hyperion Enterprise, Essbase, Outlooksoft, TM1, or various other consolidation systems, BI and cube systems and accounting packages like SUN Accounting etc.
- add-ins that contain user defined functions

ValueMaker is also useful for remote off-data-source working or for the creation of data input templates.

Typical users are financial or business analysts using user defined functions that may not be available to the "audience" of their reports. Some examples for user defined functions are:

- Accounting packages (SUN Accounting / Vision for Excel …)
- Financial consolidation and OLAP packages (Hyperion Enterprise, HFM or Essbase, Outlooksoft, ...)
- Custom user defined VBA functions incorporated in your workbook.
- Custom functions add-in like CellworXs™ providing custom worksheet functions.

For the different type of replacement possibilities please refer to the chapter about ValueMaker's replacement options.

Once cells are replaced with values based on the defined criteria and the restore option has been selected, ValueMaker generates a hidden formula recovery sheet in the treated file. This sheet called "HPValueMakerRecovery" contains the original formula location, the original formula as text string format and the replacement option that triggered the replacement of the cell formula with the value. The recovery sheet is optionally protected with a user-unknown password or not protected and can as such been edited (see value making user options). Editing the restore sheet should only be done by experienced users. You can always make a copy of the sheet. After the formula restore of prior valued cells, the "HPValueMakerRecovery" sheet will be removed from the workbook.

Important note: After valuing cells with the restore option, there should be no structural changes made on the workbook that could impact the original formula restore. *The* ValueMaker *"replacement with restore possibility" generates a" static snapshot" of your formulas content and its positions in the workbook. In order to not impact the restore functionality and its formula results,* ***do not:***

- Rename worksheets that contain "valued cells" or where the valued formula refers to that worksheet.
- Insert cells, rows or columns that would impact the position of the "valued cells" or where the valued formula refers to that worksheet cell that would move position by inserting rows, columns or cells.

Here are some inspirational examples for *ValueMaker* business use:
1.) Sharing of Hyperion, Outlooksoft or any other "proprietary" retrieve Excel reports or data collection files with non-retrieve users (i.e. with colleagues in your company who are information user or information provider, the external auditors, edit-exchange of files...)
2.) Off line work on Excel retrieve workbooks without database connection or the retrieve add-in.
3.) Keep non-retrieve formulas audit trail: Only cells with above mentioned retrieve formulas will be replaced by values, all other formulas stay untouched. So you still have an audit trail on the other formulas. In addition it is possible to restore the formulas. This is a main advantage versus an entire "copy/paste values", which is used often as workaround.
4.) Finding or replacing links to external files, to avoid users updating those links.
5.) Generate a workbooks formula audit trail.

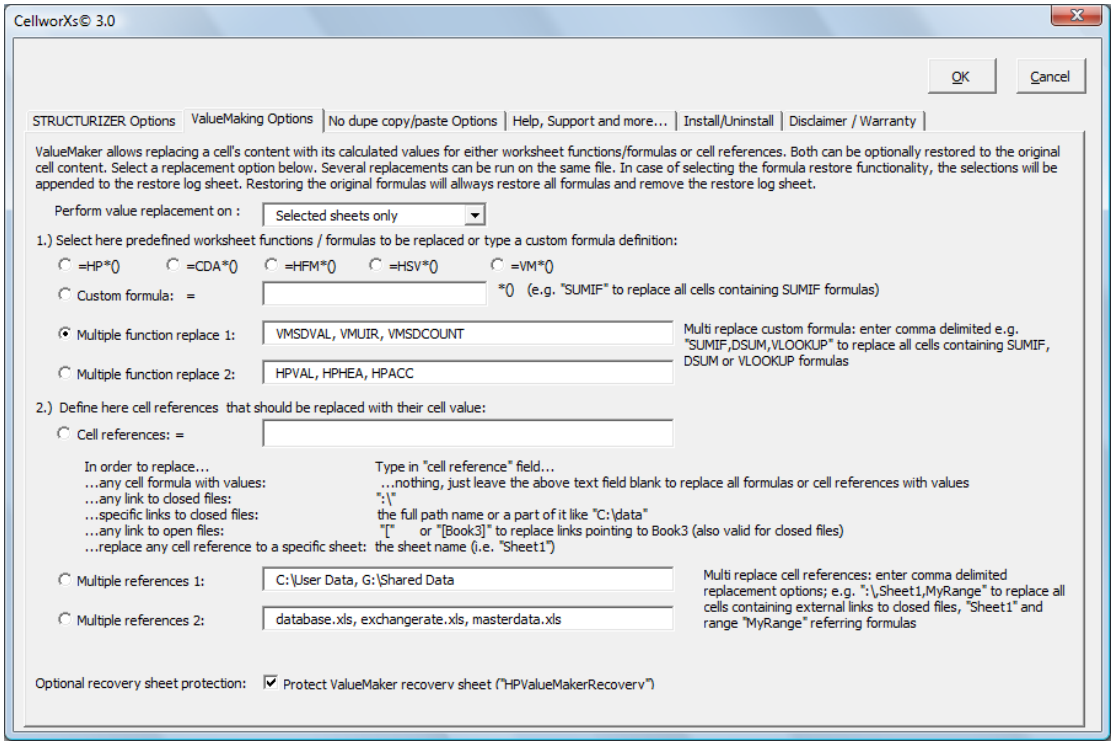
3.2 "ValueMaking" options

ValueMaker allows two options of cell content replacement with values, which both can be optionally restored to the original cell content:

- Replacement of worksheet functions / formulas like:
  =HPVAL($C32,$D$6,E$24,$D$4,$D$2,$D$1)
- Replacement of cell references and links like:
  ='C:\Documents and Settings\Sample User\My Documents\[Book1.xls]Sheet1'!$C$6

For the initial search of the replacement option text (i.e. "HP") *ValueMaker* uses Excel's "Find engine" to search for cells that potentially needed to be replaced with values. The next two chapters explains the replacement logic for above two cases and the possibilities that are offered.

Picture: ValueMaking user options



It can be defined in the user options if the replacement process should occur on:

- All sheets (incl. hidden sheets)
- Selected sheets only
- All visible sheets only
- Hidden sheets only

When “Make values now!” is executed with the formula restore option, a hidden formula recovery sheet (“HPValueMakerRecovery”) is attached to the file. It can be defined in the user options if that formula recovery sheet is password protected, to avoid user modifications.

3.2.1 Worksheet function / formula replacement

This option tests during the "ValueMaking" process if the cell content matches:

- the replacement option text (i.e. “HP”) and
- the formula characteristics of starting with an equal sign "=" and
- an opening bracket "(" in the cell content.

Important to understand: If you intend to replace Hyperion retrieve formulas that have the syntax HPVAL(…) and you choose as replacement option “HP”, any cell that contains “HP” in their formula will be replaced with a value. If for example your cell formula is defined as: ”=C12+SUM(HO8:HP8)”, this cell will be replaced under the “HP” replacement option. In case of ambiguous interpretation, you should define a less ambiguous replacement option like “HPV”.

ValueMaker offers a couple of predefined formula options:

- “VM” containing formulas, to replace CellworXs™ ‘s add-in functions
- Hyperion Enterprise, Essbase and HFM retrieve formulas (“HP”, “CDA”, “HFM”, “HSV”)

Select predefined options (VM, HP, CDA,...) or select your "Custom formula" option and enter your replacement option text. Examples:

- If your replacement option text is “VM”, *ValueMaker* will search all cells with formulas that contain “VM”.
- For custom formula replacement proceed as follows: Assume the formula syntax is XXXYYY(****), where XXX is the characteristic syntax for the retrieve formula group (i.e. each formula starts with XXX) and YYY is the specific formula (i.e. retrieving a value from a database) and (***) are the formula parameters. To replace all type of retrieve formulas enter XXX in the custom formula text box. To replace only one specific formula, enter the entire formula syntax i.e. XXXYYY in the custom formula text box.

ValueMaker allows the replacement of multiple different formulas in one session. This applies for functions as well as cell references. For both types the user can define two replacement sets in the ValueMaker options menu.

3.2.2 Cell range reference replacement

This option tests during the "ValueMaking" process if the cell content matches:

- your replacement option text (i.e. "Sheet1" to replace all cell formulas referring to "Sheet1") and
- the formula characteristics of equal sign "=".

In order to replace:	**Type in the “cell references” field:**
Any cell formula with values	Leave the field blank
Any link to closed files	:\
Specific links to closed files	the full path/file name or a part of it like "C:\data"
Any link to open files	[
A link to a specific open file (i.e. Book1.xls)	[Book1.xls]
Replace any cell reference to a specific sheet (i.e. Sheet1	Sheet1

3.2.3 Formula recovery sheet

When "Make values now!" is executed with the formula restore option, a hidden formula recovery sheet ("HPValueMakerRecovery") is attached to the file. This sheet can be protected (default system option) or not-protected.
The key concept is that the recovery sheet stores the formula information that has been replaced as a "hard coded" text. The recovery sheet uses four columns to store the information:

- worksheet name, from which the formula was replaced
- cell address of the worksheet where the formula was replaced
- the replaced formula as text string
- the replacement option that triggered the replacement

Picture: "HPValueMakerRecovery" sheet example:

H7

	A	B	C	D
1	STRUCTURIZER FunctionQuickGuide	$I$160	=VMSMissingListItems(M161:M176,E160)	VM
2	STRUCTURIZER FunctionQuickGuide	$J$182	=VMSMissingListItem($M$161:$M$176,G182,E182)	VM
3	STRUCTURIZER FunctionQuickGuide	$J$204	=vmsdval('Transaction extract'!$B$6:$Z$2165,'Transaction extract'!$P$6:$P$2165,E205,E206	VM
4	STRUCTURIZER FunctionQuickGuide	$J$256	=vmsdval('Transaction extract'!$B$6:$Z$2165,'Transaction extract'!$P$6:$P$2165,D256)	VM
5	STRUCTURIZER FunctionQuickGuide	$J$259	=vmsdval('Transaction extract'!$B$6:$Z$2165,'Transaction extract'!$P$6:$P$2165,D259)	VM
6	STRUCTURIZER FunctionQuickGuide	$J$262	=vmsdval('Transaction extract'!$B$6:$Z$2165,'Transaction extract'!$P$6:$P$2165,D262)	VM
7	STRUCTURIZER FunctionQuickGuide	$J$266	=vmsdval('Transaction extract'!$B$6:$Z$2165,'Transaction extract'!$P$6:$P$2165,D266)	VM
8	STRUCTURIZER FunctionQuickGuide	$J$18	=vmsdval('Transaction extract'!$B$6:$Z$2161,'Transaction extract'!$P$6:$P$2161,"SalesRep(	VM
9	STRUCTURIZER FunctionQuickGuide	$J$19	=vmsdval('Transaction extract'!$B$6:$Z$2161,'Transaction extract'!$P$6:$P$2161,K19&M19&	VM
10	STRUCTURIZER FunctionQuickGuide	$J$24	=vmsdval('Transaction extract'!$B$6:$Z$2161,'Transaction extract'!$P$6:$P$2161,"SalesRep(	VM
11	STRUCTURIZER FunctionQuickGuide	$K$46	=vmsdval('Transaction extract'!$B$6:$Z$2165,'Transaction extract'!$P$6:$P$2165,$E$46)	VM
12	STRUCTURIZER FunctionQuickGuide	$K$47	=vmsdval('Transaction extract'!$B$6:$Z$2165,'Transaction extract'!$P$6:$P$2165,$E$46)	VM
13	STRUCTURIZER FunctionQuickGuide	$K$58	=vmsdval('Transaction extract'!$B$6:$Z$2165,'Transaction extract'!$P$6:$P$2165,E58)	VM
14	STRUCTURIZER FunctionQuickGuide	$J$66	=vmsdirectchilds(G66)	VM
15	STRUCTURIZER FunctionQuickGuide	$J$72	=vmsdirectchild(G72,E72)	VM
16	STRUCTURIZER FunctionQuickGuide	$J$80	=VMSBaseChilds(E80)	VM
17	STRUCTURIZER FunctionQuickGuide	$J$87	=VMSBaseChildsShlashList(E87)	VM
18	STRUCTURIZER FunctionQuickGuide	$J$93	=VMSBaseChild(G93,E93)	VM
19	STRUCTURIZER FunctionQuickGuide	$J$101	=VMSItemParent(E101)	VM

If the replacement exceeds 65536 cells or a multiple of it (driven by row limit of Excel up to version 2003) the replacement will be recorded in the next free 4 columns.

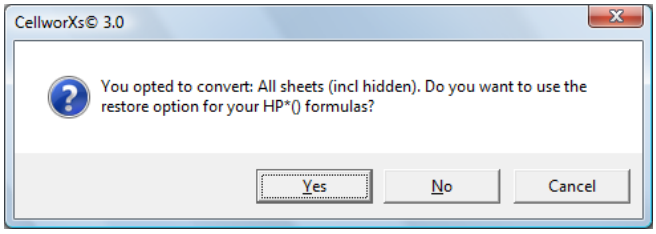
Important notes if you edit the recovery sheet:

- The table with replacement information needs to be continuously filled, i.e. do not leave a blank line. Otherwise the following cells will not be able to be restored anymore.
- Assure correct spelling of your changes in order to have formulas working properly after the formula restore.
- We recommend to make a copy of the "HPValueMakerRecovery" sheet before you edit the sheet content. This allows you to always restore the original formula version of your file, by renaming the copy of the sheet to "HPValueMakerRecovery" and launching the "Restore formulas now!" action.

If you run macros that protect all worksheets in your workbook, you need to unselect the automatic recovery sheet protection. Otherwise your macros will stop at that sheet (except if you have programmed an error handling).
However we recommend in all other cases to use a protection in order to assure users do not modify the content of the record track worksheet and eventually disable through manipulation the possibility to restore the formulas correctly.

3.3 "Make values now!" functionality

As discussed above, "Make values now!" will replace the cell content with values of all cells in defined sheets (see 'ValueMaking Options') of the active workbook as per selected ValueMaking options. Calling "Make values now!" will prompt an option to have the possibility to restore all replaced ("valued") cells. If this option is chosen, up to 4.1 million prior valued cells can be restored (even after saving & closing the file!). If you do not chose this option you will not be able to restore the original cell content (except if you close the workbook without saving)!
The following picture shows the value conversion confirmation requested from the user:



Considerations before running "Make values now!":

- Before running 'Make Values Now!' assure the workbook and the defined worksheets in the file are not protected!
- Even though it does not negatively impact the 'value making' or 'restore formulas' functionality, assure all cells you want to replace with values are correctly retrieved/calculated (i.e. no "#Values" or "#Name?").
- Calling 'Make Values Now!' will not save the workbook. Closing the workbook without saving will keep the status of the last saved version.
- Multiple sequential "ValueMaking" with different replacement criteria is possible. Running the "Make values now!" with the recovery option, will optionally append the new session to any existing recovery record from prior replacements. Other options are to erase the prior recovery record (with no more restore possibility of the prior valued formulas!) or abort the ValueMaking. You might consider the option to run first the 'Restore formulas' function, if you want to restore prior valued formulas.
- Hidden rows or columns (also hidden via grouping) are included in the value making as well as restore process, EXCEPT for rows hidden via the auto filter. As such set the auto filter to "Show all" before running the value making process.

The following table shows a sample replacement, before the value making, after the value making and after the value making restore:

	Cell content	Cell Display
Before	=(HPVAL($B$3,$B$7,$B$5,E$5,$B$6,$B$4)/1000)+F5+Sheet1!B7	1560
After	1560	1560
Restored	=(HPVAL($B$3,$B$7,$B$5,E$5,$B$6,$B$4)/1000)+F5+Sheet1!B7	1560

Important to read, if you select the formula recovery option:
After valuing cells with the restore option, there should be no structural changes made on the workbook that could impact the original formula restore. *The ValueMaker "replacement with restore possibility" generates a" static snapshot" of your formulas content and its positions in the workbook.* *In order to not impact the restore functionality and its formula results*, ***do not:***

- Rename worksheets that contain "valued cells" or where the valued formula refers to that worksheet.
- Insert cells, rows or columns that would impact the position of the "valued cells" or where the valued formula refers to that worksheet cell that would move position by inserting rows, columns or cells. This applies also for valued references to named ranges (i.e. do not rename the range once the formula has been replaced by a value).

- Removing or renaming the hidden record track worksheet ("HPValueMakerRecovery") generated by ValueMaker. See also the chapter about the “Formula recovery sheet”

If replaced formulas contain links/references to other workbooks, we recommended to close those files before running "Make values now!' with the restore option. This is to assure that the full correct file path is captured in the recovery record. However some environments are independent to this recommendation (please test your environment).

3.4 "Restore formulas now!" functionality

"Restore formulas now!" will restore the previously valued cell content with the original formulas, if at the times of running the 'Make Values Now!' functionality the restore option had been selected. Before running 'Restore formulas now!' assure the workbook and the worksheets (at least those sheets which will be affected by the restore) in the file are not protected!
A faster restore performance will be achieved if you unprotect all sheets as (excl. "HPValueMakerRecovery"). Otherwise the sheet protection has to be tested for each restore record.

Important to read:
Valuing your retrieve formulas with the restore option adds a hidden record sheet ("HPValueMakerRecovery") to your workbook, which captures your formulas and its original location. It is important that after running the 'Make Values Now!' with restore option the workbook structure does not get changed. Please see the prior chapter for more details.

Experienced users can edit the record sheet, if the setting "Protect ValueMaker recovery sheet ("HPValueMakerRecovery")" has not been opted for. The record sheet is organized in columns of 4 (source sheet, source cell, replaced cell formula/reference, replace option) from left to right in a continuous running order without empty lines.
Watch out: The first empty line will indicate the end of the record sheet. Entering an empty line will consequently only restore the formulas until the empty line and ignore any other entries in rows further down or column sets further to the right. After running "Restore formulas now!" the record sheet will be removed from the file.

4. f(x) - Additional analytical worksheet functions

CellworXs™ provides a couple of worksheet functions that add analytical value. All formulas are documented in the FunctionSample.xls file available on the web: www.cellworXs.com .

4.1 VMOPIR - "Outer Position in Range" - Worksheet Lookup Function

Returns the first or last used non-empty cell (optional cell data type: text, value or any data type) in a defined range.

Syntax: VMOPIR (RngInput, LookFromLeft, Position, CellValueType)
RngInput: A single range to be evaluated.
LookFromLeft (Optional): The direction from where the look-up starts.

If LookFromLeft = TRUE (default if omitted) the range look-up starts from the left-to-right of a horizontal range or the top-to-bottom of a vertical range.
If LookFromLeft = FALSE the range look-up starts from the right-to-left of a horizontal range or the bottom-to-top of a vertical range.

Position (Optional): Returns the index position of the returned cell from the left/top, right/bottom or the cell value (default if omitted)

If Position is equal 1 (default if omitted): The cell value is returned.
If Position is equal 2: the index position from the left-to-right of a horizontal range or the top-to-bottom of a vertical range is returned
If Position is equal 3: the index position from the right-to-left of a horizontal range or the bottom-to-top of a vertical range is returned.

CellValueType (Optional): Defines if the first non-empty cell should be a "numeric" data type, a "text" data type or any data type.

CellValueType= 1: returns the first numeric value
CellValueType= 2: returns the first text value
CellValueType= 3 (default if omitted): returns any value from the first non-empty cell in the range.

Sample as per the VMFunctionsSamples.xls:

Current Customer Pricelist

Product	Price	Prices valid as of:
Product 1	1.35	4-Mar-07
Product 2	2.29	5-Apr-07
Product 3	2.9	12-Apr-07
Product 4	0.67	1-Jan-07
Product 5	4.15	8-Feb-07
Product 6	2.6	1-Jan-07
Product 7	Product not continued	4-Mar-07

Internal tracking of product price changes

Date of price change:	1-Jan-07	25-Jan-07	8-Feb-07	4-Mar-07	5-Apr-07	12-Apr-07
Product 1	1.15			1.35		
Product 2		2.02		2.13	2.29	
Product 3						2.9
Product 4	0.67					
Product 5			4.15			
Product 6	2.6					
Product 7		3.2	Product not continued			

4.2 VMUIC - "Unique Items Count" - Worksheet Lookup Function

Counts the number of unique items in a specific range by eliminating duplicated elements. The element count can also be limited to the following data types: blank cell, text cells, numeric cells.

Syntax: VMUIC (RngInput, TakeDataType)
RngInput: A single range to be evaluated.
TakeDataType (Optional): Counts any (default if omitted) element or only text, numeric a an eventual blank cell content in the range.

If TakeDataType = 0 (default if omitted) any element in the range is counted (incl. blank cell) to enumerate the total number of unique elements.
If TakeDataType = 1 only elements that are of text data type are counted to return the unique element number.
If TakeDataType = 2 only elements that are of numeric data type are counted to return the unique element number.
If TakeDataType = 3 returns 1 if one or more blank cells are in the range.

Sample as per the VMFunctionsSamples.xls:

Example: The product extract report needs to be evaluated on how many different product codes exist.

Number of all unique elements in the product codes range (incl. Blank, text and numeric values):	10
Number of unique elements in the product codes range which are of text data type:	8
Number of unique elements in the product codes range which are of number data type:	1
Evaluates if a blanc cell is included:	1

Tip:
Use formula combinations to define i.e. how many text elements excluding the blank cell element are in the range.

# of text and numeric items	9

Product/Country Extract Report

Product Code	Country	Revenue
A1673	Germany	439
C7895	Germany	15
C4589	Germany	1,000
A7522	Germany	113
C4589	Belgium	74
A4568	Belgium	389
C7895	Belgium	466
A7522	Belgium	576
C1673	UK	602
A6879	UK	692
C7895	UK	441
A7522	UK	176
C4589	Italy	911
A4568	Italy	88
A6879	Italy	485
C7895	Italy	237
A7522	Italy	375
27/Dec/06		
C1673	France	666
C1589	France	370
C4589	France	890
A4568	France	216
A6879	France	108
C1673	Netherlands	51
C1589	Netherlands	306
A6879	Netherlands	721
C7895	Netherlands	174
A7522	Netherlands	489

4.3 VMUIR - "Unique Item Retrieve" - Worksheet Lookup Function

Returns the unique item based on its relative position in the (optionally sorted) list of unique items. The element count can also be limited to the following data types: text cells and numeric cells. Blank cells are ignored.

Syntax: VMUIR (RngInput, ItemNumber, TakeDataType, SortedElements)
RngInput: A single range (of multiple cells) to be evaluated.
ItemNumber: The position in the (optionally sorted) unique list from where the item should be retrieved.
TakeDataType (Optional): Counts any (default if omitted) element or only text, numeric a an eventual blank cell content in the range.

If TakeDataType = 0 (default if omitted) any element in the range (excl. blank cell) is taken to establish the unique element list.
If TakeDataType = 1 only text data type elements in the range are taken to establish the unique element list.
If TakeDataType = 2 only numeric data type elements in the range are taken to establish the unique element list.

SortedElements: Defines if the list of unique elements should be non-sorted (default), ascending or descending sorted.

If SortedElements = 0 (default if omitted) the list of unique items is established by adding each new item of the evaluated range to the end of the unique item list.
If SortedElements = 1 the final list of unique items is sorted ascending before the element is retrieved from its index position in the list.
If SortedElements = 2 the final list of unique items is sorted descending before the element is retrieved from its index position in the list

Sample as per the VMFunctionsSamples.xls:

Example: get a list of unique product codes from the "Product/Country Extract Report"

Number of text type product codes in the "Product/Country Extract Report": 8 (see VMUIC function)

# Item	Non-Sorted List	Sorted List (ascending)
1	A1673	A1673
2	C7895	A4568
3	C4589	A6879
4	A7522	A7522
5	A4568	C1589
6	C1673	C1673
7	A6879	C4589
8	C1589	C7895
0		
0		
0		
0		
0		
0		
0		
0		
0		
0		
0		
0		
0		
0		

Note: above example hides the "#Value!" result via conditiona formatting

Product/Country Extract Report

Product Code	Country	Revenue
A1673	Germany	439
C7895	Germany	15
C4589	Germany	1,000
A7522	Germany	113
C4589	Belgium	74
A4568	Belgium	389
C7895	Belgium	466
A7522	Belgium	576
C1673	UK	602
A6879	UK	692
C7895	UK	441
A7522	UK	176
C4589	Italy	911
A4568	Italy	88
A6879	Italy	485
C7895	Italy	237
A7522	Italy	375
27/Dec/06		
C1673	France	666
C1589	France	370
C4589	France	890
A4568	France	216
A6879	France	108
C1673	Netherlan	51
C1589	Netherlan	306
A6879	Netherlan	721
C7895	Netherlan	174
A7522	Netherlan	489

4.4 VMTTF - "Text to Formula" - Worksheet Function

The VMTTF(InputText) function, converts a text string into an Excel formula and returns its evaluated result. It also converts array formulas defined as text string (without the usual "shift ctrl enter" input).

With this “simple” function you have the possibility to create highly dynamic data analysis applications based on for example array formulas. You can construct your formula text string based on the user input and then return via the VMTTF function the calculated result. In the sample file “VMFunctionsSamples.xls” we are showing an example that dynamically sizes your array formula based on the users selection.

Syntax: VMTTF (InputText)
InputText: A single cell or text string to be evaluated. The text string formula can be entered with or without the equal sign in the beginning. If an equal sign is entered in the beginning the cell needs to be formatted as text or a hyphen needs to be placed before the equal sign (‘=).

Sample as per the VMFunctionsSamples.xls:

Examples:

Sample data table:

Dim 1	Dim 2	Dim 3	Value
a	y	s	10
c	y	t	15
a	y	t	5
b	x	s	20

Example for a text formula conversion:

The text formula: F18+F19 Results via VMTTF to: 25

The text formula: SUM(F19:F22) Results via VMTTF to: 50

Example for a dynamic array formula creation:

	Select:	Array formula component:
Dim 1	a	(C19:C22=D31)*
Dim 2	y	(D19:D22=D32)*
Dim 3	t	(E19:E22=D33)*
	Sum range:	(F19:F22)

Combined array formula as text string: Sum((C19:C22=D31)*(D19:D22=D32)*(E19:E22=D33)*(F19:F22))

Array formula result via VMTTF of above string: 5

4.5 VMVAL - "Period Value" - Worksheet Lookup Function
Returns a subtotal of a range depending on the period and frequency selection. The VMVAL function allows quickly specific periodic data views (subtotals) on data tables that contain either 12 monthly or 4 quarterly columns. The VMVAL function is a powerful function to generate flexible reports on "financial spreadsheet databases".

Syntax: VMVAL (RngInput, Period, Frequency)
RngInput: A range to be evaluated. The range can be organized horizontally or vertically and can consist of 12 ("monthly data organization"), 4 ("quarterly data organization") cells.
Period: Defines the selected period of the range of which the subtotal should be returned.
Frequency: Defines if the data should be subtotalled with the following frequency: monthly, quarterly, trimester, semester or yearly. The data view can be "to-date": year-to-date, semester-to-date, trimester-to-date, quarter-to-date or month-to-date. The functions "Frequency" is a combination of frequency and data view, for example: M.YTD

The following table shows the possible combinations of frequency, data view and valid periods, which are available if the range is organised in 12 monthly columns (or rows):

Frequency	Function periods	Columns (**)	Data View				
			MTD	QTD	TTD	STD	YTD
M	1-12	12	M.MTD	M.QTD	M.TTD	M.STD	M.YTD
Q	1-4	12 or 4	Q.MTD	Q.QTD *		Q.STD *	Q.YTD *
T	1-3	12			T.TTD		T.YTD
S	1-2	12 or 4				S.STD *	S.YTD *
Y	1	12 or 4					Y.YTD *

(*): these marked combinations are also available if the data is organized in 4 (quarterly) columns or rows. (**): Number of columns required in Excel database table.

The following two examples visualize how the VMVAL() function sums the different columns based on the selected period and frequency. The first examples shows a monthly data organization and the second examples shows a quarterly data organization.

Sample as per the VMFunctionsSamples.xls:

Example 1: Data organized in 12 month columns.

Dataview:

Yearly						Year							
Semester					Semester 1				Semester 2				
Trimester				Trimester 1			Trimester 2			Trimester 3			
Quater			Quater 1			Quater 2			Quater 3			Quater 4	
Month		1	2	3	4	5	6	7	8	9	10	11	12
Monthly sales result:		**100**	**50**	**60**	**120**	**110**	**200**	**40**	**70**	**130**	**170**	**90**	**120**
Period	**Frequency**	**Result:**											
6	M.MTD	**200**	100	50	60	120	110	200	40	70	130	170	90	120
8	M.MTD	**70**	100	50	60	120	110	200	40	70	130	170	90	120
6	M.QTD	**430**	100	50	60	120	110	200	40	70	130	170	90	120
8	M.QTD	**110**	100	50	60	120	110	200	40	70	130	170	90	120
6	M.TTD	**310**	100	50	60	120	110	200	40	70	130	170	90	120
8	M.TTD	**420**	100	50	60	120	110	200	40	70	130	170	90	120
6	M.STD	**640**	100	50	60	120	110	200	40	70	130	170	90	120
8	M.STD	**110**	100	50	60	120	110	200	40	70	130	170	90	120
6	M.YTD	**640**	100	50	60	120	110	200	40	70	130	170	90	120
8	M.YTD	**750**	100	50	60	120	110	200	40	70	130	170	90	120
2	Q.QTD	**430**	100	50	60	120	110	200	40	70	130	170	90	120
3	Q.STD	**240**	100	50	60	120	110	200	40	70	130	170	90	120
3	Q.YTD	**880**	100	50	60	120	110	200	40	70	130	170	90	120
2	T.TTD	**420**	100	50	60	120	110	200	40	70	130	170	90	120
2	T.YTD	**750**	100	50	60	120	110	200	40	70	130	170	90	120
2	S.STD	**620**	100	50	60	120	110	200	40	70	130	170	90	120
2	S.YTD	**1260**	100	50	60	120	110	200	40	70	130	170	90	120
1	Y.YTD	**1260**	100	50	60	120	110	200	40	70	130	170	90	120

Example 2: Data organized in 4 columns representing quaters.

Dataview:

Year		Year			
Semester		Semester 1		Semester 2	
Quater		1	2	3	4
Monthly sales result:		**330**	**430**	**240**	**380**
Period	**Frequency**	**Result:**			
3	Q.QTD	**240**	330	430	240	380
3	Q.YTD	**1000**	330	430	240	380
3	Q.STD	**240**	330	430	240	380
4	Q.STD	**620**	330	430	240	380
2	S.STD	**620**	330	430	240	380
2	S.YTD	**1380**	330	430	240	380
1	Y.YTD	**1380**	330	430	240	380

4.6 VMFPSD - "Fiscal Period Start Date" Function

The VMFPSD function returns a period start date based on the defined end date, data view and month offset. The start date will be returned as first date of the selected data view that belongs to the referenced "AnyDate".
Example: With the data view "QTD" (quarter-to-date), the "AnyDate" May 15th 2007 will return April 1st 2007 as start date. May 15th is part of the second quarter, which starts April 1st.

The VMFPSD function is a powerful function to quickly define a fiscal date range.

Syntax: VMFPSD(AnyDate, DataView, MonthOffset)

AnyDate: The date indicating the end of the period. Accepts any date format or date as valid serial number.
DataView: Defines the selected period of the range of which the subtotal should be returned. The following parameter can be entered as "DataView":

YTD: Year-to-date view
STD: Semester-to-date view
TTD: Trimester-to-date view
QTD: Quarter-to-date view
MTD: Month-to-date view
WTD: Week-to-date view (always refers to the first Monday before the "AnyDate")
DTD: Day-to-date view

MonthOffset (Optional): The number of month the fiscal year is offset from the calendar year. The MonthOffset will have no impact for MTD, WTD and DTD data view, as those views would not be influenced by a fiscal year that is offset from a calendar year.

If MonthOffset = 0 (default if omitted): the start date is calculated based on a normal calendar year.
If MonthOffset = -1 to –11: the start date is calculated based on a fiscal year that is negatively offset by the number of specified month (year shift to left).
Example: MonthOffset = -1: the fiscal year starts on December 1st rather than January 1st.

If MonthOffset = 1 to 11 the start date is calculated based on a fiscal year that is positively offset by the number of specified month (year shift to right).
Example: MonthOffset = 3, the fiscal year starts on April 1st rather than January 1st.

Sample as per the VMFunctionsSamples.xls:

AnyDate	DataView	MonthOffset	Calculated Period Start Date
15-Aug-07	YTD	0	1-Jan-07
15-Aug-07	STD	0	1-Jul-07
15-Aug-07	TTD	0	1-May-07
15-Aug-07	QTD	0	1-Jul-07
15-Aug-07	MTD	0	1-Aug-07
15-Aug-07	WTD	0	13-Aug-07
15-Aug-07	DTD	0	15-Aug-07
15-Aug-07	YTD	-2	1-Nov-06
15-Aug-07	STD	-2	1-May-07
15-Aug-07	TTD	-2	1-Jul-07
15-Aug-07	QTD	-2	1-Aug-07
15-Aug-07	MTD	-2	1-Aug-07
15-Aug-07	WTD	-2	13-Aug-07
15-Aug-07	DTD	-2	15-Aug-07
15-Aug-07	YTD	2	1-Mar-07
15-Aug-07	STD	2	1-Mar-07
15-Aug-07	TTD	2	1-Jul-07
15-Aug-07	QTD	2	1-Jun-07
15-Aug-07	MTD	2	1-Aug-07
15-Aug-07	WTD	2	13-Aug-07
15-Aug-07	DTD	2	15-Aug-07

4.7 VMFYSD - "Fiscal Year Start Date" Function

The VMFYSD function returns the first day of the fiscal year to which the defined "AnyDay" belongs. It takes into consideration a possible month offset of the fiscal year vs the calendar year.
Example: With the "AnyDate" May 15th 2007 and a MonthOffset of -2 the returned value will be November 1st 2006.

Syntax: VMFYSD(AnyDate, MonthOffset)
AnyDate: The date indicating any date of the period you want to query. Accepts any date format or date as valid serial number.

MonthOffset (Optional): The number of month the fiscal year is offset from the calendar year. Any full number between -11 and 11.

If MonthOffset = 0 (default if omitted): the start date is calculated based on a normal calendar year.
If MonthOffset = -1 to –11: the start date is calculated based on a fiscal year that is negatively offset by the number of specified month (year shift to left).
Example: MonthOffset = -1: the fiscal year starts on December 1st rather than January 1st.

If MonthOffset = 1 to 11 the start date is calculated based on a fiscal year that is positively offset by the number of specified month (year shift to right).
Example: MonthOffset = 3: the fiscal year starts on April 1st rather than January 1st.

Sample as per the VMFunctionsSamples.xls:

AnyDate	MonthOffset		Fiscal Year Start
1-Aug-07 with month offset	-11	belongs to fiscal year:	1-Feb-07
1-Aug-07 with month offset	-10	belongs to fiscal year:	1-Mar-07
1-Aug-07 with month offset	-9	belongs to fiscal year:	1-Apr-07
1-Aug-07 with month offset	-8	belongs to fiscal year:	1-May-07
1-Aug-07 with month offset	-7	belongs to fiscal year:	1-Jun-07
1-Aug-07 with month offset	-6	belongs to fiscal year:	1-Jul-07
1-Aug-07 with month offset	-5	belongs to fiscal year:	1-Aug-07
1-Aug-07 with month offset	-4	belongs to fiscal year:	1-Sep-06
1-Aug-07 with month offset	-3	belongs to fiscal year:	1-Oct-06
1-Aug-07 with month offset	-2	belongs to fiscal year:	1-Nov-06
1-Aug-07 with month offset	-1	belongs to fiscal year:	1-Dec-06
1-Aug-07 with month offset	0	belongs to fiscal year:	1-Jan-07
1-Aug-07 with month offset	1	belongs to fiscal year:	1-Feb-07
1-Aug-07 with month offset	2	belongs to fiscal year:	1-Mar-07
1-Aug-07 with month offset	3	belongs to fiscal year:	1-Apr-07
1-Aug-07 with month offset	4	belongs to fiscal year:	1-May-07
1-Aug-07 with month offset	5	belongs to fiscal year:	1-Jun-07
1-Aug-07 with month offset	6	belongs to fiscal year:	1-Jul-07
1-Aug-07 with month offset	7	belongs to fiscal year:	1-Aug-07
1-Aug-07 with month offset	8	belongs to fiscal year:	1-Sep-06
1-Aug-07 with month offset	9	belongs to fiscal year:	1-Oct-06
1-Aug-07 with month offset	10	belongs to fiscal year:	1-Nov-06
1-Aug-07 with month offset	11	belongs to fiscal year:	1-Dec-06

4.8 VMFYED - "Fiscal Year End Date" Function

The VMFYED function returns the last day of the fiscal year to which the defined "AnyDay" belongs. It takes into consideration a possible month offset of the fiscal year vs the calendar year.
Example: With the "AnyDate" May 15th 2007 and a MonthOffset of -2 the returned value will be Oct 31st 2007.

Syntax: VMFYED(AnyDate, MonthOffset)
AnyDate: The date indicating any date of the period you want to query. Accepts any date format or date as valid serial number.

MonthOffset (Optional): The number of month the fiscal year is offset from the calendar year. Any full number between -11 and 11.

If MonthOffset = 0 (default if omitted): the start date is calculated based on a normal calendar year.
If MonthOffset = -1 to –11: the start date is calculated based on a fiscal year that is negatively offset by the number of specified month (year shift to left).

Example: MonthOffset = -1 the fiscal year starts on December 1st rather than January 1st.

If MonthOffset = 1 to 11: the start date is calculated based on a fiscal year that is positively offset by the number of specified month (year shift to right).
Example: MonthOffset = 3, the fiscal year starts on April 1st rather than January 1st.

Sample as per the VMFunctionsSamples.xls:

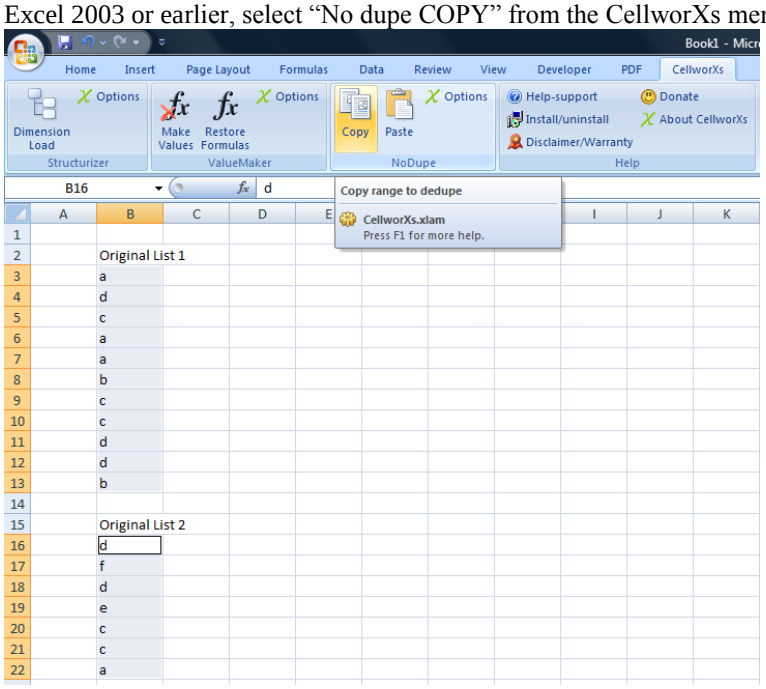
AnyDate	MonthOffset		Fiscal Year Start	Fiscal Year End
1-Aug-07 with month offset	-11	belongs to fiscal year:	1-Feb-07	31-Jan-08
1-Aug-07 with month offset	-10	belongs to fiscal year:	1-Mar-07	29-Feb-08
1-Aug-07 with month offset	-9	belongs to fiscal year:	1-Apr-07	31-Mar-08
1-Aug-07 with month offset	-8	belongs to fiscal year:	1-May-07	30-Apr-08
1-Aug-07 with month offset	-7	belongs to fiscal year:	1-Jun-07	31-May-08
1-Aug-07 with month offset	-6	belongs to fiscal year:	1-Jul-07	30-Jun-08
1-Aug-07 with month offset	-5	belongs to fiscal year:	1-Aug-07	31-Jul-08
1-Aug-07 with month offset	-4	belongs to fiscal year:	1-Sep-06	31-Aug-07
1-Aug-07 with month offset	-3	belongs to fiscal year:	1-Oct-06	30-Sep-07
1-Aug-07 with month offset	-2	belongs to fiscal year:	1-Nov-06	31-Oct-07
1-Aug-07 with month offset	-1	belongs to fiscal year:	1-Dec-06	30-Nov-07
1-Aug-07 with month offset	0	belongs to fiscal year:	1-Jan-07	31-Dec-07
1-Aug-07 with month offset	1	belongs to fiscal year:	1-Feb-07	31-Jan-08
1-Aug-07 with month offset	2	belongs to fiscal year:	1-Mar-07	29-Feb-08
1-Aug-07 with month offset	3	belongs to fiscal year:	1-Apr-07	31-Mar-08
1-Aug-07 with month offset	4	belongs to fiscal year:	1-May-07	30-Apr-08
1-Aug-07 with month offset	5	belongs to fiscal year:	1-Jun-07	31-May-08
1-Aug-07 with month offset	6	belongs to fiscal year:	1-Jul-07	30-Jun-08
1-Aug-07 with month offset	7	belongs to fiscal year:	1-Aug-07	31-Jul-08
1-Aug-07 with month offset	8	belongs to fiscal year:	1-Sep-06	31-Aug-07
1-Aug-07 with month offset	9	belongs to fiscal year:	1-Oct-06	30-Sep-07
1-Aug-07 with month offset	10	belongs to fiscal year:	1-Nov-06	31-Oct-07
1-Aug-07 with month offset	11	belongs to fiscal year:	1-Dec-06	30-Nov-07

5. Unique item copy (No dupe copy)

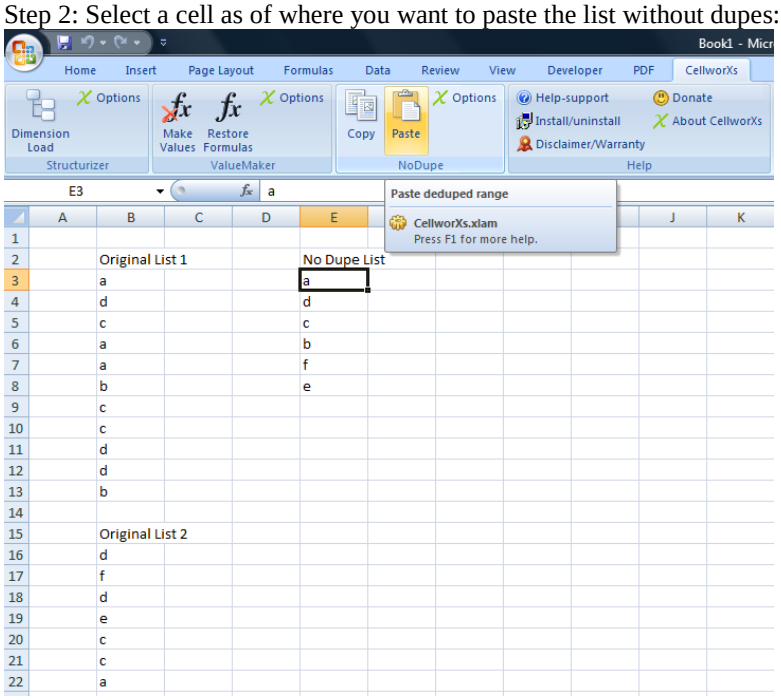
CellworXs™ provides a functionality to copy from (multiple selection) ranges containing lists with duplicate items a list of un–duped items. The pasted list can be optionally sorted at the out put. In case of large copy lists, we recommend to use the non-sorted copy option and to sort the pasted list afterwards via Excel's menu.

5.1 Using no dupe copy / paste

Step 1: Select a single or multiple range and select "Copy" in the NoDupe section of the ribbon. (In Excel 2003 or earlier, select "No dupe COPY" from the CellworXs menu.)



Step 2: Select a cell as of where you want to paste the list without dupes:



5.2 No dupe copy / paste options

You can define in the options if the output list is sorted or in the order as the items are read from the list. The sorted output takes slightly longer, so you might consider to sort via Excel menu the list that has been returned by the "no dupe paste".

6. Technical requirements / installation

6.1 General remarks

CellworXs™ is an Excel add-in. The current version 3.0 is designed to run under Excel 97 through Excel 2010.
The add-in is multi-user installation enabled. User specific information is hold in the "CellworXsUserName.ini" file, which is generated automatically in the directory of the add-in installation. The user ini file contains the individual preferred user settings. It needs to be assured that the user has directory "write rights" to the add-in installation directory.

Depending on the installation the workload of CellworXs™ will be performed on the local machine (local installation or through normal network installation) or on the server (i.e. CITRIX, thin client,….) CellworXs™ can be installed under CITRIX environments.

6.2 Installation

Please refer to the CellworXs Installation Guide for installation instructions.

6.3 Support

We provide email and phone support. Please contact us at info@CellworXs.com or via the "contact" page at our web site www.CellworXs.com .

7. Recommend CellworXs™ : Our commitment - your benefit!

CellworXs™ has been build based on analysts feedback and functionality requests. If you think CellworXs™ can be useful for your direct colleagues or peers in other companies, please don't hesitate to send them the link to our corporate web www.CellworXs.com .

Our commitment:

- We seriously evaluate and take into consideration your suggested product enhancements.
- We will try to include possible small enhancements that are valid for other users within minor releases.
- We will try to include major functional enhancements and additions with new major releases.
- We will continue to have an ear for your feedback!

Your benefit:

- You will benefit from functionality enhancements, which you might not have thought about.
- You will have regular new functionality available that will further enhance the analytical capability.

WE THANK YOU FOR YOUR TRUST IN CellworXs™ !