Agilent EZChrom *Elite*

Advanced Reports Reference

Agilent Technologies

Notices

Edition

January, 2006

Document Revision 3.1Ga

Printed in USA

Agilent Technologies, Inc.
6612 Owens Dr.
Pleasanton, CA 94588-3334

Warranty

The material contained in this document is provided “as is,” and is subject to being changed, without notice, in future editions. Further, to the maximum extent permitted by applicable law, Agilent disclaims all warranties, either exclick or implied, with regard to this manual and any information contained herein, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. Agilent shall not be liable for errors or for incidental or consequential damages in connection with the furnishing, use, or performance of this doc-ument or of any information contained herein. Should Agilent and the user have a separate written agreement with warranty terms covering the material in this document that conflict with these terms, the warranty terms in the separate agreement shall control.

Technology Licenses

The hardware and/or software described in this document are furnished under a license and may be used or copied only in accordance with such license.

Restricted Rights Legend

If software is for use in the performance of a U.S. Government prime contract or subcontract, Software is delivered and licensed as "Commercial computer software" as defined in DFAR 252.227-7014 (June 1995), or as a "commercial item" as defined in FAR 2.101(a) or as "Restricted computer software" as defined in FAR 52.227-19 (June 1987) or any equivalent agency regulation or contract clause. Use, duplication or disclosure of Software is subject to Agilent Technologies standard commercial license terms, and non-DOD Departments and Agencies of the U.S. Government will receive no greater than Restricted Rights as defined in FAR 52.227-19(c)(1-2) (June 1987). U.S. Government users will receive no greater than Limited Rights as defined in FAR 52.227-14 (June 1987) or DFAR 252.227-7015 (b)(2) (November 1995), as applicable in any technical data.

Contents

1 Using This Guide

Introduction

This guide describes how use the Advanced Reports feature of Agilent EZChrom *Elite* data system, through tutorials and examples. It also provides a complete list of functions used in Advanced Reports.

Who Should Read This Guide?

This document is designed for advanced users of the Agilent EZChrom *Elite* data system

How This Guide is Organized

The following table presents a brief description of each chapter in this guide.

In this chapter	Youll find
1—Using This Guide	Information about using this guide.
2—Introduction	Overview of this guide.
3—Tutorial #1	How to create an advanced report for sequence summary and statistics.
4—Tutorial #2	How to set up a BTU-type report.
5—Functional Reference	Listing of advanced report functions.
6—Parameter Notes	Notes on passing parameters.
6—Advanced Reporting Formulas	Listing of advanced report formulas.

Documentation Conventions

The following conventions are used in this guide.

Convention	Description
Bold	Database names, table names, column names, menus, commands, dialog box options, and text that must be typed exactly as shown.
Italic	Placeholders for information you must provide. For example, if you are instructed to type *ServerName*, then you must type the actual name of the server instead of the italicized term.
`Monospace`	Programming code samples and display text.
ALL CAPITALS	The keys you click on the keyboard. If combined with a plus sign (+), click and hold the first key while you click the remaining key(s). For example, click SHIFT+TAB.

The following notes may appear in this guide.

⚠	**Caution!**	A **CAUTION** notice denotes a hazard. It calls attention to an operating procedure, practice, or the like that, if not correctly performed or adhered to, could result in damage to the product or loss of important data. Do not proceed beyond a **CAUTION** notice until the indicated conditions are fully understood and met.
📌	**Note**	Notes contain special information and alert you to effects of an action.
💡	**Tip**	Tips provider additional information or an alternate method for completing a task.

2 Introduction

This manual contains details on using the Advanced Reporting feature of Agilent EZChrom *Elite*.

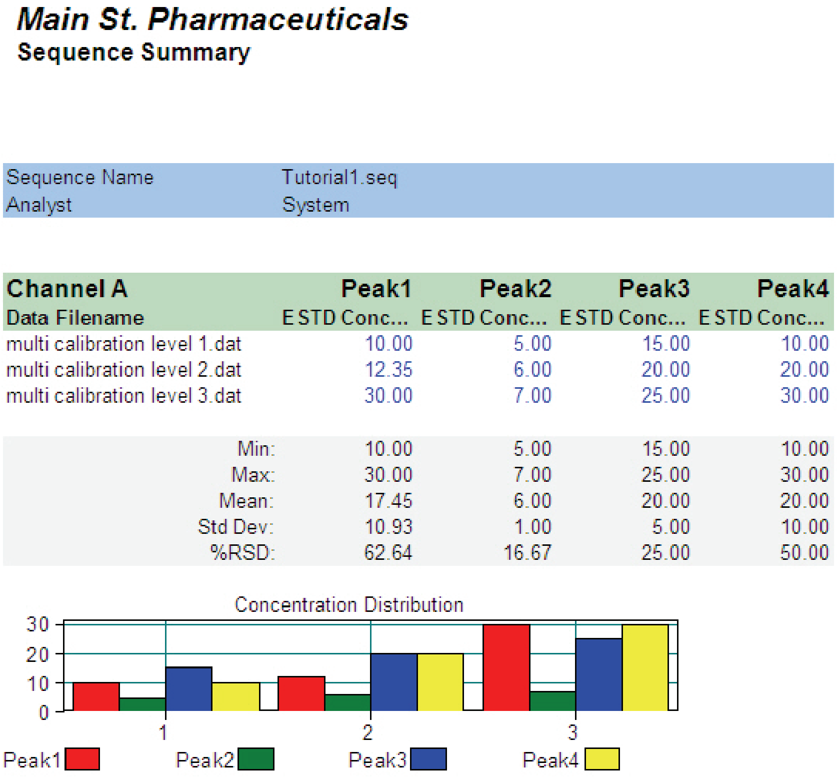
The Tutorial contains 2 separate tutorials to help familiarize the user with the Advanced Template Reporting feature of EZChrom *Elite*. The first tutorial takes the user through the steps of creating a sequence summary report. The second tutorial takes the reader through the steps of creating a BTU style report.

3 Tutorial #1 - Creating a Sequence Summary Report

This tutorial will walk you through creating a Sequence Summary Report. In addition, the report will have statistics that summarize certain data.

The final report will show peak concentrations for all named peaks, for each run in a sequence. The peak names will be reported across the page, and the sequence runs will be reported down the page. In addition, the report will contain statistics for the concentrations, and a chart that graphs the concentration distributions.

Example:



Main St. Pharmaceuticals

Sequence Summary

Sequence Name	Tutorial1.seq
Analyst	System

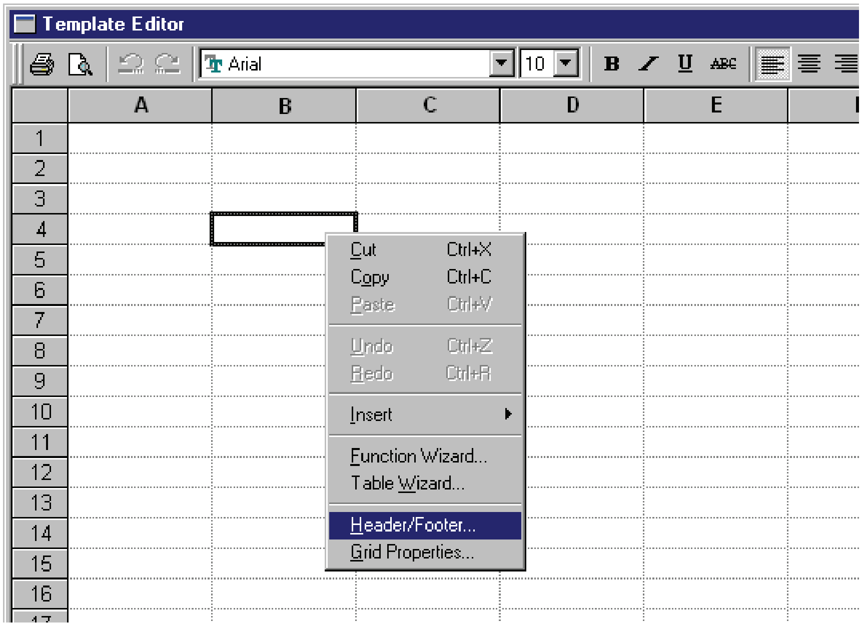
Channel A	**Peak1**	**Peak2**	**Peak3**	**Peak4**
Data Filename	E STD Conc...	E STD Conc...	E STD Conc...	E STD Conc...
multi calibration level 1.dat	10.00	5.00	15.00	10.00
multi calibration level 2.dat	12.35	6.00	20.00	20.00
multi calibration level 3.dat	30.00	7.00	25.00	30.00
Min:	10.00	5.00	15.00	10.00
Max:	30.00	7.00	25.00	30.00
Mean:	17.45	6.00	20.00	20.00
Std Dev:	10.93	1.00	5.00	10.00
%RSD:	62.64	16.67	25.00	50.00

Step 1: (Create a new report template)

1. If an instrument window is not already open, then launch an instrument either online or offline.
2. Choose the **Print Setup** command from the **File** menu and select a valid printer. Set the paper orientation to Landscape. Click OK on the Print Setup dialog.
3. Choose **New** in **Advanced Reports** from the **File** menu to create a new template report window.

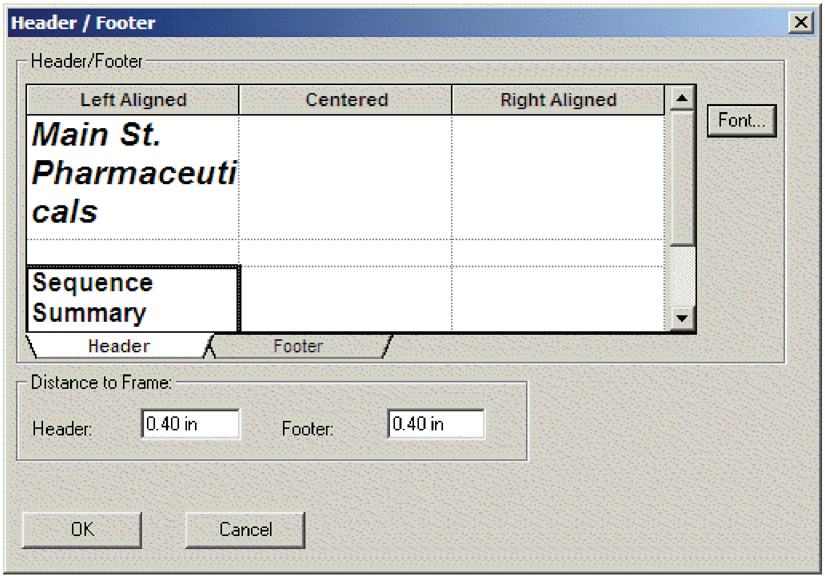
Step 2: (Create the report header and footer)

1. Right-click on any cell in the template grid, and choose **Header/Footer**.



2. Click on the cell in the first row of the Left Aligned column.
3. Type Main St Pharmaceuticals. into this cell.

4. Click **Font** and choose Arial, Bold Italic, and 16 for the font attributes and then click OK.
5. Click on the cell in the second row of the Left Aligned column.
6. Type EZChrom *Elite* Client/Server into this cell.
7. Click **Font** and choose Arial, Bold, and 12 for the font attributes and then click OK.
8. Click on the cell in the third row of the Left Aligned column.
9. Type Sequence Summary Report into this cell.
10. Click **Font** and choose Arial, Bold, and 12 for the font attributes and then click OK.



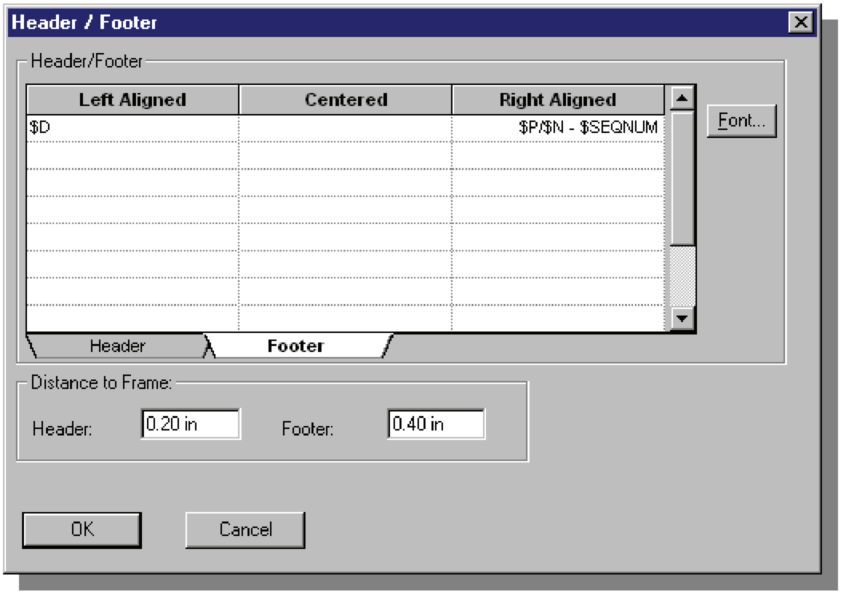
Left Aligned	Centered	Right Aligned
Main St. Pharmaceuticals		
Sequence Summary		

Header: 0.40 in

Footer: 0.40 in

11. Click on the Footer tab.
12. Click on the cell in the first row of the Left Aligned column.
13. Type **$D** into this cell. This indicates that the current date and time should be displayed.
14. Click **Font** and choose Arial, Regular, and 8 for the font attributes and then click OK.
15. Click on the cell in the first row of the Right Aligned column.

16. Type **$P/$N - $SEQNUM** into this cell. This indicates that page numbers like 1/5 - 1 should be displayed.
17. Click **Font** and choose Arial, Regular, and 8 for the font attributes and then click OK.



Left Aligned	Centered	Right Aligned
$D		$P/$N - $SEQNUM

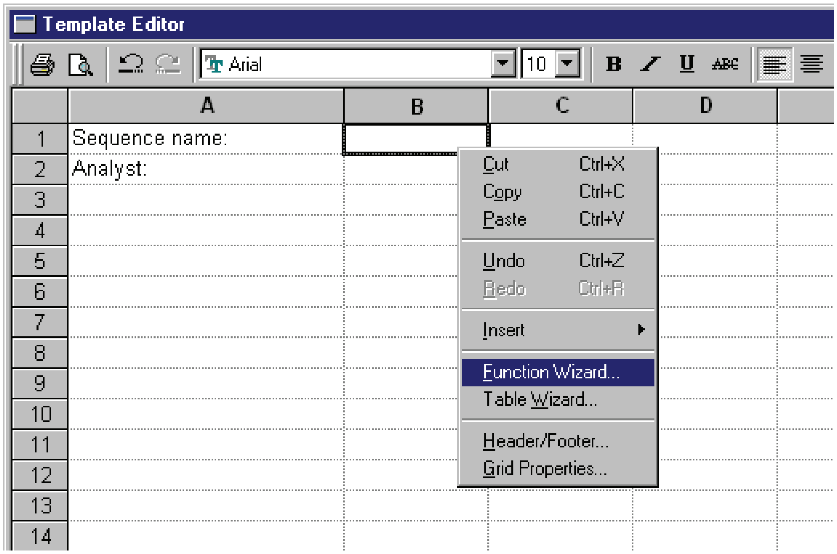
Header: 0.20 in

Footer: 0.40 in

18. Type **0.2** into the Header field for the Distance to Frame.
19. Type **0.4** into the Footer field for the Distance to Frame.
20. Click **OK**.

Step 3: (Create the page header)

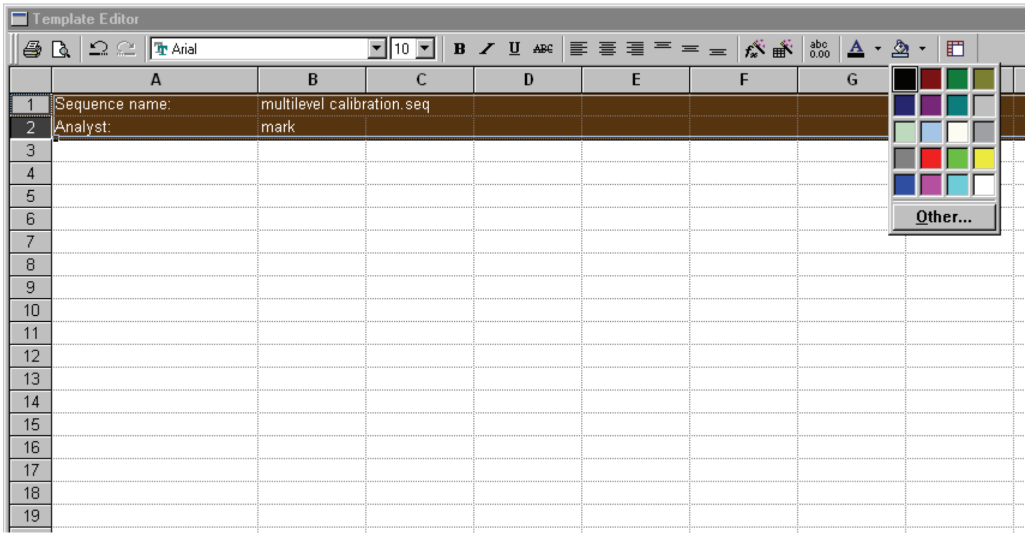
1. Click on cell A1 and type **Sequence name**:
2. Click on cell A2 and type **Analyst**:
3. Move the mouse over the column header area of the template grid, between column A and column B. The mouse will change to an icon with double arrows indicating that the column widths can be changed. Click and drag to approximately double the size of column A.
4. Right-click on cell B1 and choose **Function Wizard**...



	A	B	C	D
1	Sequence name:			
2	Analyst:			
3				
4				
5				
6				
7				
8				
9				
10				
11				
12				
13				
14				

5. Choose **Sequence file** as the data source. Choose **Sequence** from the left-hand list box. Choose **Filename** from the right-hand list box. Make sure the Repeating formula box is unchecked, and click **Finish**.
6. Right-click on cell B2 and choose **Function Wizard**...
7. Choose **Sequence file** as the **data source**. Choose **Instrument** from the left-hand list box. Choose **User Name** from the right-hand list box. Make sure the Repeating formula box is unchecked, and click Finish.
8. Use the mouse to highlight cells A1 and A2. Click the toolbar button with the B on it to make the text of these cells have a bold style.

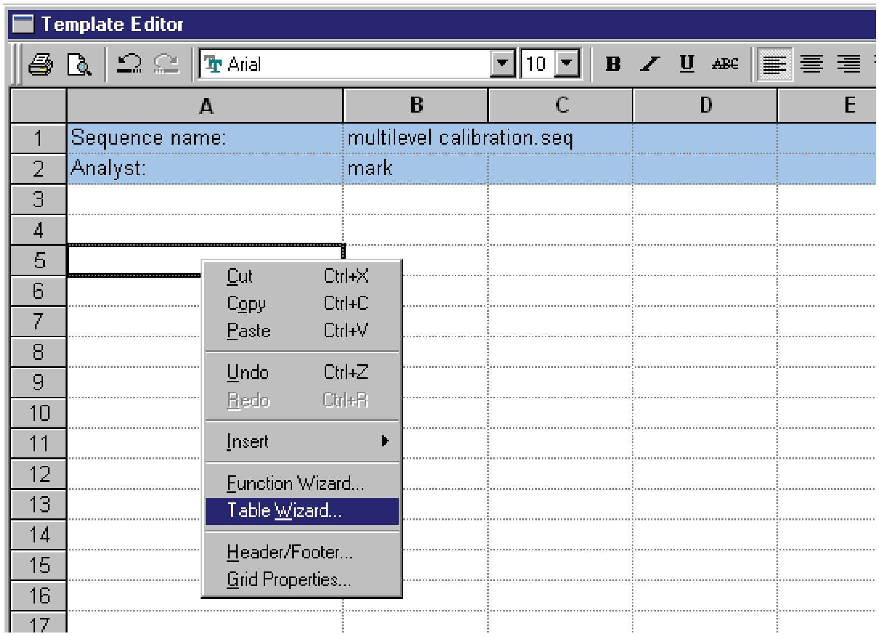
   Use the mouse to highlight rows 1 and 2. This can be done by clicking on the 1 in the header area for the first row. Now hold the Shift key down and click on the 2 in the header area for the second row. Now click the down arrow on the toolbar button with the bucket on it to set the background color. Choose the pale blue color near the center of the drop-down color menu.



	A	B
1	Sequence name:	multilevel calibration.seq
2	Analyst:	mark

Step 4: (Create the summary table)

1. Right-click on cell A5 and choose **Table Wizard...**



	A	B	C	D	E
1	Sequence name:	multilevel calibration.seq			
2	Analyst:	mark			

2. Highlight Sequence Summary Table and click **Next**.
3. Highlight ESTD Concentration in the left-hand list box, and click the green arrow to move it to the right-hand list box. Click **Next**.

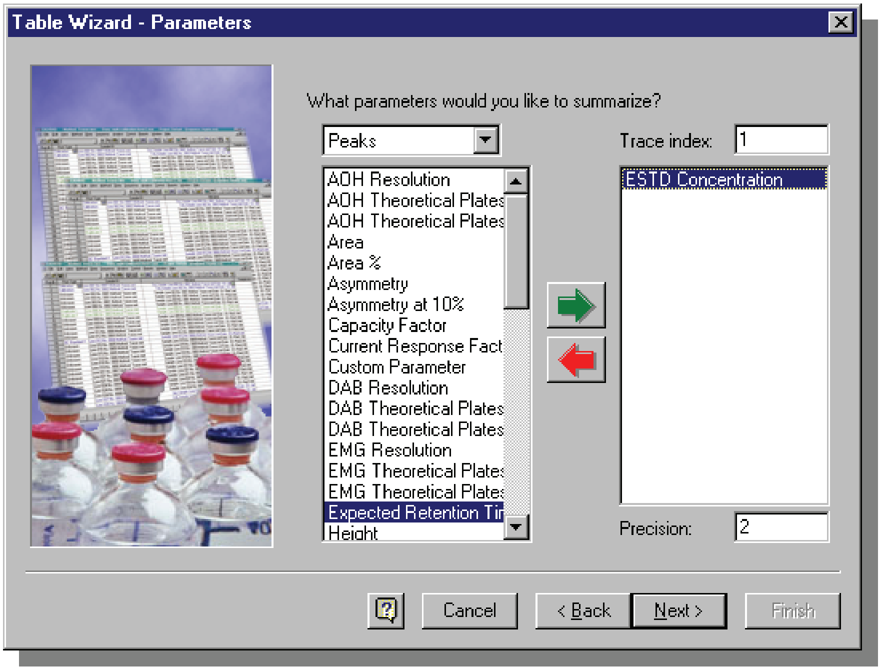


Table Wizard - Parameters

What parameters would you like to summarize?

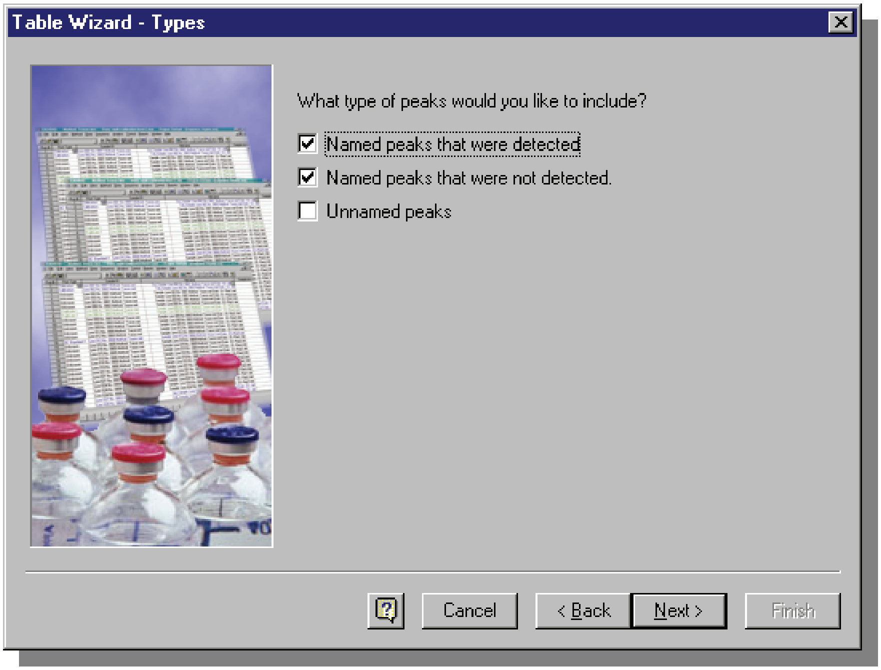
Peaks

Trace index: 1

ESTD Concentration

Precision: 2

4. In the “Types” dialog box, leave the first two checkboxes checked, and make sure that the last checkbox is unchecked. Click **Next**.



What type of peaks would you like to include?

☑ Named peaks that were detected

☑ Named peaks that were not detected.

□ Unnamed peaks

5. Highlight Data Filename in the left-hand list box, and click the green arrow to move it to the right-hand list box. Click Next.

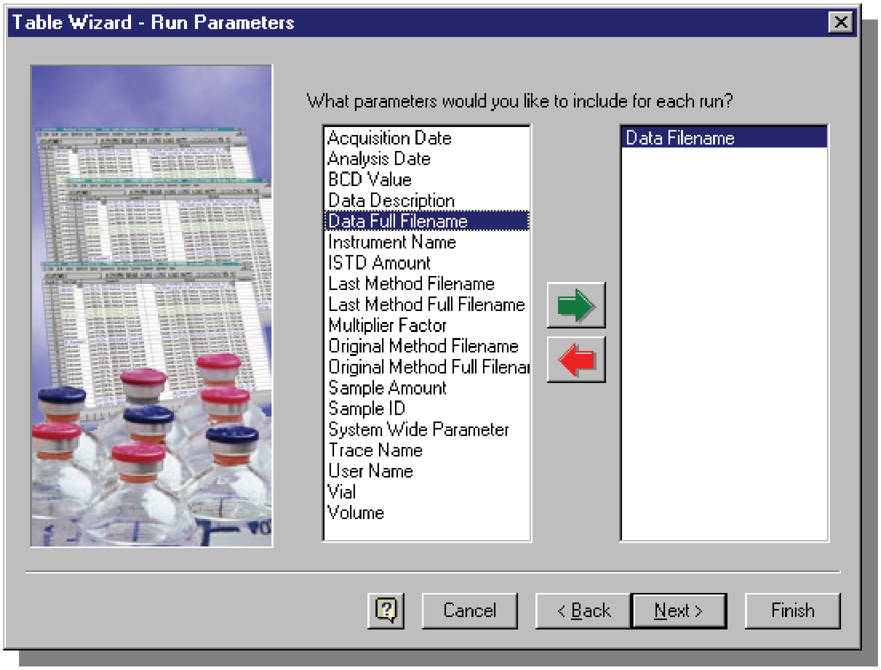


Table Wizard - Run Parameters

What parameters would you like to include for each run?

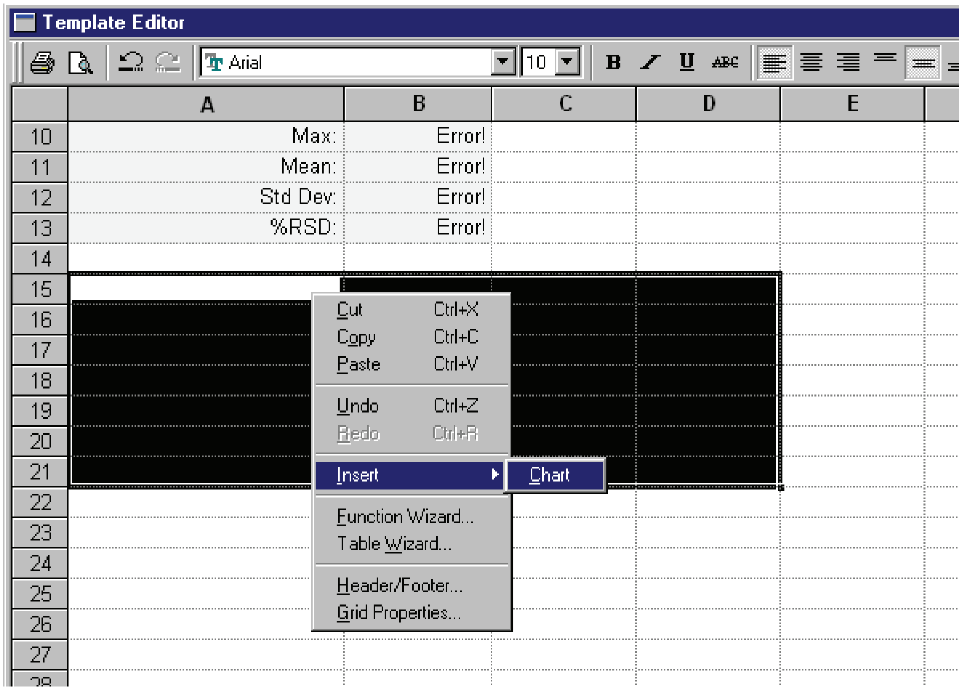
- Acquisition Date
- Analysis Date
- BCD Value
- Data Description
- Data Full Filename
- Instrument Name
- ISTD Amount
- Last Method Filename
- Last Method Full Filename
- Multiplier Factor
- Original Method Filename
- Original Method Full Filenar
- Sample Amount
- Sample ID
- System Wide Parameter
- Trace Name
- User Name
- Vial
- Volume

Data Filename

6. Select **Down** to generate the runs down the report. Click **Next**.
7. Select **Yes** to include statistics with the report. Click **Finish**.

Step 5: (Create the data chart)

1. Use the mouse to highlight cells A15 to D21.
2. Right-click on a cell in the highlighted area and choose **Insert** followed by **Chart** from the popup menu.



	A	B	C	D	E
10	Max:	Error!			
11	Mean:	Error!			
12	Std Dev:	Error!			
13	%RSD:	Error!			

3. Type **Concentration Distribution** into the Chart Title field.
4. Choose **Vertical Bar Chart** for the chart style.
5. Click on the cell in the first row of the Initial Cell column, and type B7. This is the cell that will contain the first ESTD Concentration.
6. Click on the cell in the first row of the Data Set column, and choose Vertical. This is the direction that a group of data is repeated. We want to make a graph of the peak concentrations across all runs in a sequence, so the data set will be vertical.
7. Click on the cell in the first row of the Groups column, and choose Multiple. This indicates that multiple groups of data will be charted. We want to chart all of the available peaks, so we must set this value to Multiple.
8. Click on the cell in the first row of the Group Titles column, and type B5. This is the cell that will contain the name of the first peak. This cell can either contain text, or a cell reference. If a cell reference is specified, such as B5, then the titles for the groups of data will be pulled from the data in the template grid.

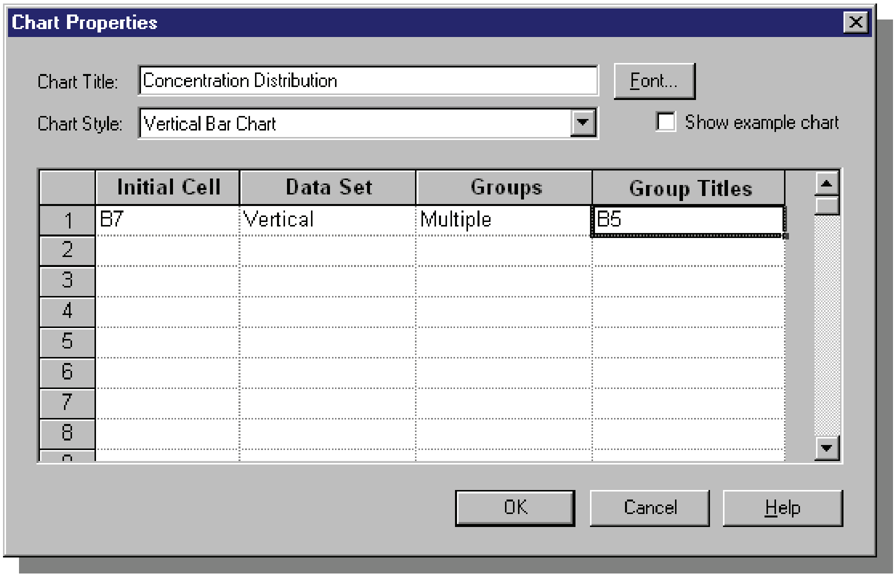


Chart Properties

Chart Title: Concentration Distribution

Chart Style: Vertical Bar Chart

☐ Show example chart

	Initial Cell	Data Set	Groups	Group Titles
1	B7	Vertical	Multiple	B5
2				
3				
4				
5				
6				
7				
8				

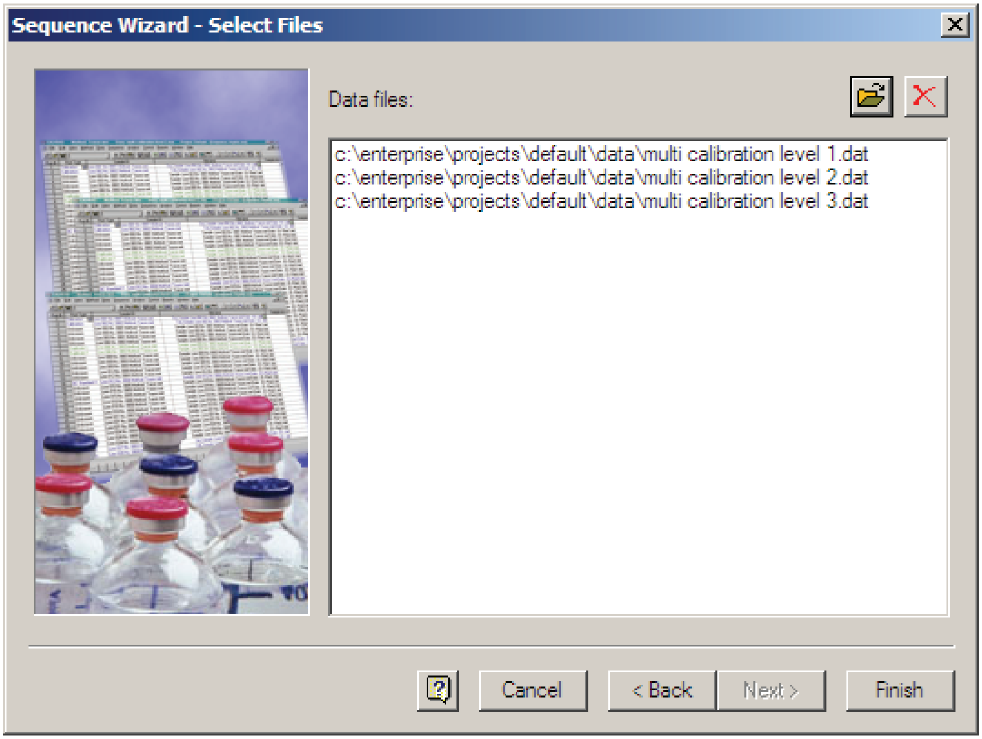
9. Click **OK**.

Step 6: (Save the template)

1. Choose **Save As...** in **Advanced Reports** from the **File** menu.
2. Type Tutorial 1.tpl into the File name field and click **Save**.
3. Close the template report window, by clicking on the [X] in the upper right-hand corner of the window.

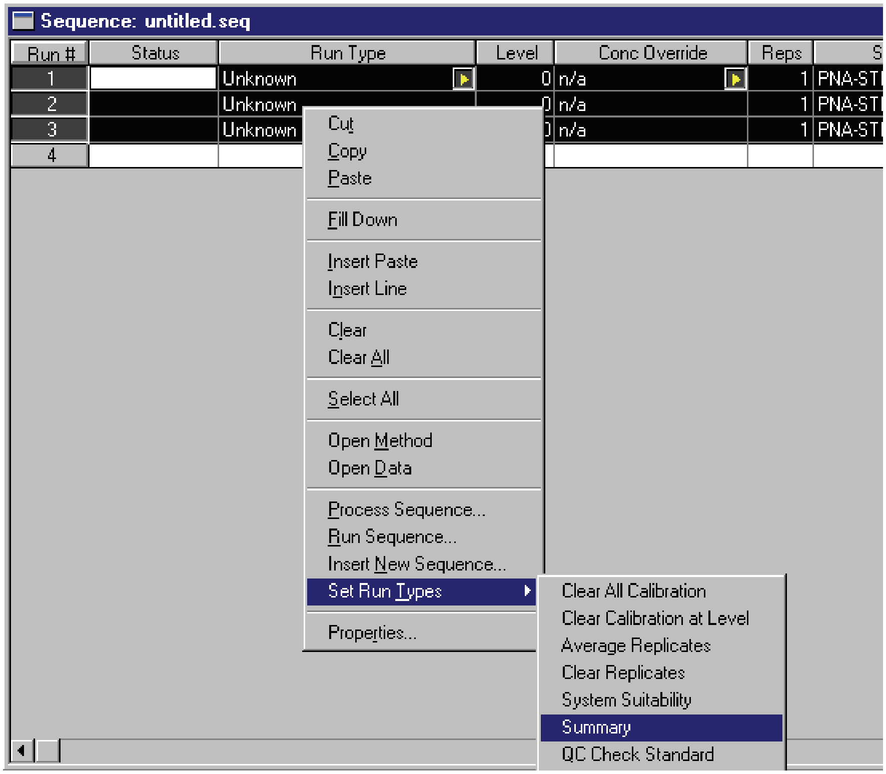
Step 7: (Create a reprocessing sequence)

1. From the **File** menu, select **Sequence** and then **Sequence Wizard.**
2. Click the Browse button for the method field to locate and select the **multilevel calibration.met** file provided with the installation of EZChrom *Elite*.
3. Set the **Data File Type** to **From existing data files**.
4. Click the **Next** button.
5. Click the Browse button for the data files field to locate and selected the **multi calibration level 1.dat**, **multi calibration level 2.dat,** and **multi calibration level 3.dat** files provided with the installation of EZChrom *Elite*.



c:\enterprise\projects\default\data\multi calibration level 1.dat
c:\enterprise\projects\default\data\multi calibration level 2.dat
c:\enterprise\projects\default\data\multi calibration level 3.dat

6. Click the **Finish** button.
7. Use the mouse to highlight the three lines in the sequence table.
8. Right-click on the sequence table and choose **Set Run Type** followed by **Summary** from the popup menu.

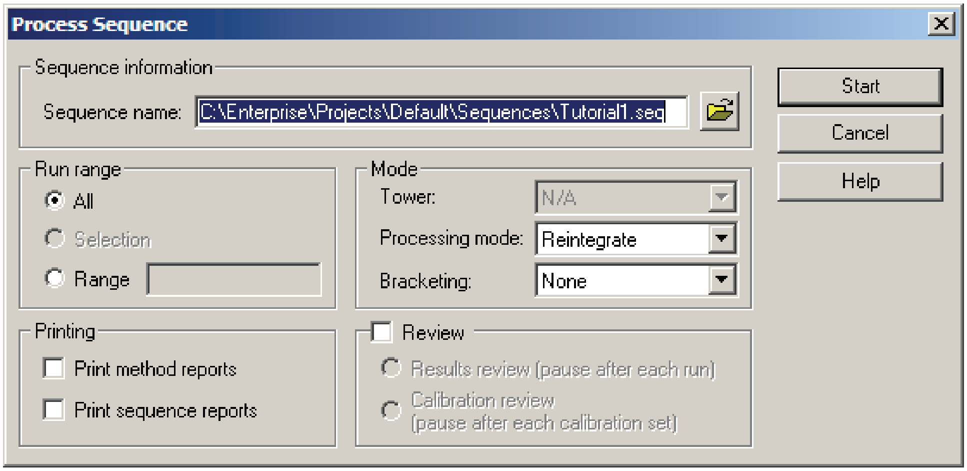


Run #	Status	Run Type	Level	Conc Override	Reps	S
1		Unknown	0	n/a	1	PNA-ST
2		Unknown	0	n/a	1	PNA-ST
3		Unknown	0	n/a	1	PNA-ST
4						

9. Click on the cell containing the **Summary Begin** run type. Click on the blue button in the run type cell to activate the run type dialog.
10. Click **Begin Summary**, and use the browse button to specify the **Tutorial 1.tpl** for the report template. Click **OK**.
11. Choose **Sequence** followed by **Save As..** from the **File** menu [to save the sequence. Specify **Tutorial 1.seq** as the file name for the sequence .

Step 8: (View the report)

1. Choose **Process...** from the **Sequence** menu to process the current sequence file.
2. Leave all of the default choices and click **Start** on the **Process Sequence dialog**.


Process Sequence

Sequence information

Sequence name: C:\Enterprise\Projects\Default\Sequences\Tutorial1.seq

Run range

● All

○ Selection

○ Range

Mode

Tower: N/A

Processing mode: Reintegrate

Bracketing: None

Printing

□ Print method reports

□ Print sequence reports

□ Review

○ Results review (pause after each run)

○ Calibration review (pause after each calibration set)

3. Choose **View** followed by **Sequence Custom Report** from the **Reports** menu.
4. Highlight the **Sequence Summary** report and click the **View** button.
5. The completed sequence summary report should now be displayed in a window.
6. Click the **Close** button to return to editing mode on the report. It is now possible to change the report, and click the **Print Preview** button in the toolbar to see the changes using the real data from the sequence summary.

Step 9: (Add a new statistic to the report)

This final step is an advanced step that shows how to directly edit the report, by modifying the formulas in the cells to create a new statistic on the summary data. In this example, we will simply create a new statistic that is the sum of the ESTD Concentration for each of the peaks across all of the runs.

1. If the report is still in preview mode from step 8, exit preview mode by clicking the **Close** button.
2. Highlight row 23 by clicking on the 23 in the row header.
3. Click and drag row 23 up to row 14. A red line is displayed indicating where the row will be placed when the mouse is released.

4. Click on cell A13 and click <Ctrl-C>. Click on cell A14 and click <Ctrl-V>. This will copy the text and background colors of the A13 cell. Now click on cell A14 and type Sum:
5. Click on cell B13 and click <Ctrl-C>. Click on cell B14 and click <Ctrl-V>. This will copy the text and background colors of the B13 cell. Now click on cell B14 and click F2. Clicking F2 on a cell with a formula will toggle the editing mode of the formula in the cell.
6. The formula should currently be: =EX.R(B13/B12*100,"R1","T1","PA;3;0;0")
7. Change the formula to: =EX.R(SUM(EX.D(B7,2)),"R1","T1","PA;3;0;0")
8. Choose [File][Advanced Reports][Save] to save the modified report.
9. Click on the **print preview** button (second button from the left) on the toolbar to see the completed report.

Analysis of the new formula:

=EX.R(SUM(EX.D(B7,2)),"R1","T1","PA;3;0;0")

This formula contains 3 parts:

Part 1

=EX.R(SUM(EX.D(B7,2)),"R1","T1","PA;3;0;0")

This first part is a function that will be used to repeat the enclosed function or set of functions over cells in the template grid. In this case, the formula SUM(EX.D(B7,2)) will be repeated. The parameters to the EX.R() function specify how the enclosed formula should be repeated. In this case the parameters "R1","T1","PA;3;0;0" specify that the enclosed formula should be repeated ***across*** the grid for ***all named peaks*** of the ***first trace*** of the ***first run*** in a sequence.

Part 2

=EX.R(SUM(EX.D(B7,2)),"R1","T1","PA;3;0;0")

This second part is a function that will be used to sum the data over a range of cells in the template grid. The SUM() function takes a parameter that specifies the range of cells to compute the sum for. This could be a static range of cells - for example, B7..B20. This could also be a dynamic range of cells, for which the EX.D() function is normally used.

Part 3

=EX.R(SUM(EX.D(B7,2)),"R1","T1","PA;3;0;0")

This third part is a function that will be used to create a dynamic cell range for the SUM() function. In this case the function pulls the dynamic range information from the cell B7. In our report, cell B7 contains the formula =EX.R(PEAK.ESTDCONCENTRATION(),"RA;1;0","T1","PA;3;0;0"). This formula creates a table of ESTD Concentrations. The table is laid out like the following:

Peak 1 Conc. from Sequence Run 1	Peak 2 Conc. from Sequence Run 1	Peak 3 Conc. from Sequence Run 1
Peak 1 Conc. from Sequence Run 2	Peak 2 Conc. from Sequence Run 2	Peak 3 Conc. from Sequence Run 2
Peak 1 Conc. from Sequence Run 3	Peak 2 Conc. from Sequence Run 3	Peak 3 Conc. from Sequence Run 3

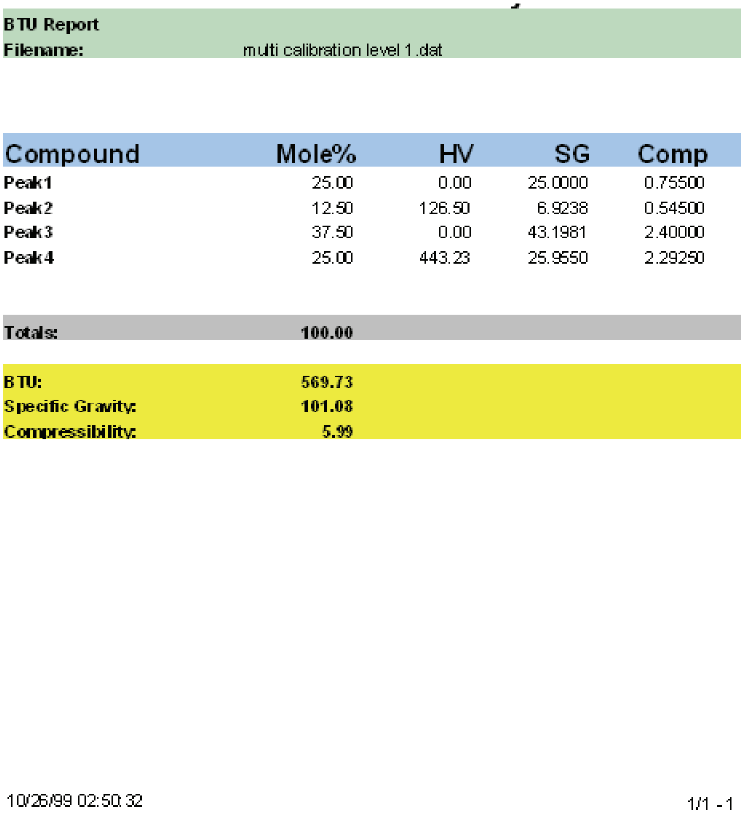
In this example, we are interested in summing the data from the ***columns*** of the table. That is what the "2" is for in the EX.D(B7,2) function. If a "1" was specified, as in EX.D(B7,1), then the cell range returned would be for the ***rows*** of the above table.

4 Tutorial #2 - Creating a BTU Style Report

This tutorial will walk you through creating a BTU Style Report.

The final report will show peak concentrations for specific named peaks, along with calculations for their specific gravities and heating values. In addition, the report will contain calculations at the end to summarize the results.

Example:



BTU Report
Filename: multi calibration level 1.dat

Compound	Mole%	HV	SG	Comp
Peak1	25.00	0.00	25.0000	0.75500
Peak2	12.50	126.50	6.9238	0.54500
Peak3	37.50	0.00	43.1981	2.40000
Peak4	25.00	443.23	25.9550	2.29250

Totals:	**100.00**
BTU:	**569.73**
Specific Gravity:	**101.08**
Compressibility:	**5.99**

10/26/99 02:50:32 1/1 - 1

Step 1: (Create a new report template)

1. If an instrument window is not already open, then launch an instrument either online or offline.

2. Choose **Print Setup…** from the **File** menu and select a valid printer. Set the paper orientation to **Portrait**. Click **OK** on the **Print Setup** dialog.
3. Choose **Advanced Reports** followed by **New…** from the **File** menu to create a new template report window.

Step 2: (Create the page header)

1. Click on cell A1 and type **EZChrom *Elite* Client/Server System**
2. Click the down arrow on the toolbar button with the A on it to set the text color. Choose the red color near the center of the drop-down color menu.
3. Use the toolbar to set the font size to 20.
4. Click the toolbar button with the B on it to make the text have a bold style.
5. Move the mouse over the column header area of the template grid, between column A and column B. The mouse will change to an icon with double arrows indicating that the column widths can be changed. Click and drag to resize the width of column A approximately twice the width as column B.
6. Click on cell A2 and type **BTU Report**
7. Click on cell A3 and type **Filename**:
8. Right-click on cell B3 and choose **Function Wizard…**
9. Choose **Current data file** as the data source. Choose **DATA** from the left-hand list box. Choose **Data Filename** from the right-hand list box. Make sure the **Repeating formula** box is unchecked, and click Finish.
10. Use the mouse to highlight cells A2 and A3. Click the toolbar button with the B on it to make the text of these cells have a bold style.
11. Use the mouse to highlight rows 2 and 3. This can be done by clicking on the 2 in the header area for the first row. Now hold the Shift key down and click on the 3 in the header area for the second row. Now click the down arrow on the toolbar button with the bucket on it to set the background color. Choose the pale green color near the center of the drop-down color menu.

Step 3: (Create the table header)

1. Click on cell A7 and type **Compound**
2. Click on cell B7 and type **Mole %**

3. Click on cell C7 and type **HV**
4. Click on cell D7 and type **SG**
5. Click on cell E7 and type **Comp**
6. Use the mouse to highlight row 7.
7. Use the toolbar to set the font size to 20, and the font style to Bold.
8. Now click the down arrow on the toolbar button with the bucket on it to set the background color. Choose the pale blue color near the center of the drop-down color menu.
9. Use the mouse to highlight cells B7 through E7. Click the **Right Justified** button in the toolbar to right justify the text in the selected cells.

Step 4: (Create the table data)

1. Right-click on cell A8 and choose **Function Wizard...**
2. Choose **Current data file** as the data source. Choose **PEAK** from the left-hand list box. Choose **Name** from the right-hand list box. Make sure the **Repeating formula** box is unchecked, and click **Next**.
3. Set the Trace Index to 1.
4. Select the Peak ID radio button and set the **Peak ID** to 1.
5. Click **Finish**
6. Repeat steps 1-5 three more times specifying peak ids of 2, 3, and 4. These formulas will be placed into cells A9, A10, and A11.
7. Use the mouse to highlight cells A8 through A11.
8. Use the toolbar to set the font style to **Bold**.
9. Right-click on cell B8 and choose **Function Wizard...**
10. Choose **Current data file** as the data source. Choose **PEAK** from the left-hand list box. Choose **NORM Concentration** from the right-hand list box. Make sure the **Repeating formula box** is unchecked, and click Next.
11. Set the **Trace Index** to 1.
12. Select the **Peak ID** radio button and set the **Peak ID** to 1.
13. Click **Finish**
14. Repeat steps 9-13 three more times specifying peak ids of 2, 3, and 4. These formulas will be placed into cells B9, B10, and B11.
15. Use the mouse to highlight cells B8 through B11.
16. Click the **Cell Style** button in the toolbar. This is the button with the abc and 0.00 on it, near the right end of the toolbar.

17. Select **Fixed** from the category list on the left. Type 2 in for the decimal places. Click the OK button to close the dialog box.
18. Click the **Right Justified** button in the toolbar to right justify the text in the selected cells.
19. Click on cell C8 and type =B8*F8/100. Click the F2 key to take the cell out of editing mode.
20. Place the mouse over the small square in the lower right-hand corner of the highlighted cell C8. The cursor will change shapes to a cross to indicate correct placement. Click and drag the mouse down to cell C11. When the mouse is released, the formula in cell C8 will be repeated in the highlighted cells. In addition, the formula will be adjusted to contain the correct calculations. For example, the formula in cell C9 will be =B9*F9/100.
21. Use the toolbar to right justify the selected cells.
22. Use the toolbar to set the cell style of the selected cells to Fixed with 2 decimal places of precision.
23. Click on cell D8 and type =B8*G8. Click the F2 key to take the cell out of editing mode.
24. Place the mouse over the small square in the lower right-hand corner of the highlighted cell D8. The cursor will change shapes to a cross to indicate correct placement. Click and drag the mouse down to cell D11.
25. Use the toolbar to right justify the selected cells.
26. Use the toolbar to set the cell style of the selected cells to Fixed with 4 decimal places of precision.
27. Click on cell E8 and type =B8*H8. Click the F2 key to take the cell out of editing mode.
28. Place the mouse over the small square in the lower right-hand corner of the highlighted cell E8. The cursor will change shapes to a cross to indicate correct placement. Click and drag the mouse down to cell E11.
29. Use the toolbar to right justify the selected cells.
30. Use the toolbar to set the cell style of the selected cells to **Fixed** with 5 decimal places of precision.
31. Click on cell F8 and type **0**
32. Click on cell F9 and type **1012**
33. Click on cell F10 and type **0**
34. Click on cell F11 and type **1772.9**
35. Click on cell G8 and type **1**

36. Click on cell G9 and type **0.5539**
37. Click on cell G10 and type **1.15195**
38. Click on cell G11 and type **1.0382**
39. Click on cell H8 and type **0.0302**
40. Click on cell H9 and type **0.0436**
41. Click on cell H10 and type **0.064**
42. Click on cell H11 and type **0.0917**
43. Use the mouse to highlight cells F8 through H11.
44. Click the **Cell Style** button in the toolbar.
45. Select **Hidden** from the category list on the left. Click the **OK** button to close the dialog box.
46. Use the mouse to resize columns F, G, and H. These columns should be very narrow. Resize them to be only as wide as their label in the header.

Step 5: (Create the total line)

1. Click on cell A14 and type **Totals**:
2. Click on cell B14 and type **=SUM(B8..B11)**
3. Set the cell style of cell B14 to Fixed with 2 decimal places of precision. Also make this cell right justified.
4. Use the mouse to highlight row 14.
5. Use the toolbar to set the text to **bold**.
6. Now click the down arrow on the toolbar button with the bucket on it to set the background color. Choose the light gray color on the right-hand side of the drop-down color menu.

Step 6: (Create the calculations)

1. Click on cell A16 and type **BTU:**
2. Click on cell B16 and type **=SUM(C8..C11)**
3. Click on cell A17 and type **Specific Gravity:**
4. Click on cell B17 and type **=SUM(D8..D11)**
5. Click on cell A18 and type **Compressibility:**
6. Click on cell B18 and type **=SUM(E8..E11)**
7. Highlight cells B16 through B18 and set the cell style to Fixed with 2 decimal places of precision. Also make these cells right justified.

8. Use the mouse to highlight rows 16 through 18.
9. Use the toolbar to set the text to **bold**.
10. Now click the down arrow on the toolbar button with the bucket on it to set the background color. Choose the yellow color on the right-hand side of the drop-down color menu.

Step 7: (Change the report margins)

1. Right-click on cell and choose **Grid Properties…** from the popup menu.
2. Set the margins to 0.75 for all four sides.
3. Click **OK** to close the **Grid Properties** dialog box.

Step 8: (Save the template)

1. Choose **Advanced Reports** followed by **Save As…** from the **File** menu.
2. Type **Tutorial 2.tpl** into the **File name** field and click **Save**.
3. Close the template report window, by clicking on the [X] in the upper right-hand corner of the window.

Step 9: (View the report)

1. Choose **Open…** followed by **Data** from the **File** menu. Locate and select the multi **calibration level 1.dat** file provided with the installation of EZChrom *Elite*.
2. Click the **Print Preview** button in the toolbar of the template report window to see the completed BTU report.

5 Introduction

This section describes the functionality available for creating advanced reports with the template editor.

6 Functional Reference

The following functions are available when creating advanced reports using the template editor.

Syntax Notes:

All of the functions described here are placed in cells in the template reporting spreadsheet, and must begin with an = sign. For example, if a function were described as Custom.Func(["Param A"]), then the actual function would look something like

=Custom.Func("Param A")

[] These brackets indicate optional parameters. The brackets themselves are not included in the actual parameters. For example, if a function were described as Custom.Func(["Param A"]), then the actual function would look something like

=Custom.Func("Param A")

<> These brackets indicate required parameters. The brackets themselves are not included in the actual parameters. For example, if a function were described as Custom.Func(<"Param A">), then the actual function would look something like

=Custom.Func("Param A")

"" Quotation marks shown are required. The quortation marks are included in the actual parameters. For example, if a function were described as Custom.Func(<"Param A">), then the actual function would look something like

=Custom.Func("Param A")

Datafile Functions

These functions return information about data files that have been collected and analyzed.

Data.AcquisitionDate

Returns date and time of acquisition for the specified data file.

Syntax

=Data.AcquisitionDate(<Run Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

Date/Time

Data.AnalysisDate

Returns date and time of the last analysis for the specified data file.

Syntax

=Data.AnalysisDate(<Run Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

Date/Time

Data.BCDValue

Returns BCD value of the specified data file.

Syntax

=Data.BCDValue(<Run Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Data.Description

Returns the description of the specified data file.

Syntax

=Data.Description(<Run Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

String

Data.Filename

Returns file name of the specified data file. Only the file name is returned the path information is not returned.

Syntax

=Data.Filename(<Run Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

String

Data.FullFilename

Returns full file name of the specified data file. The file name and path information is returned.

Syntax

=Data.FullFilename(<Run Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

String

Data.InstrumentName

Returns the name of the instrument that was used to acquire the specified data file.

Syntax

=Data.InstrumentName(<Run Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

String

Data.ISTDAmount

Returns ISTD amount of the specified data file.

Syntax

=Data.ISTDAmount(<Run Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Data.LastMethodFilename

Returns the name of the last method file that was used to analyze the specified data file.

Syntax

=Data.LastMethodFilename(<Run Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

String

Data.LastMethodFullFilename

Returns the full name and path of the last method file that was used to analyze the specified data file.

Syntax

=Data.LastMethodFullFileName(<Run Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

String

Data.MultiplierFactor

Returns multiplier factor of the specified data file.

Syntax

=Data.MultiplierFactor(<Run Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Data.OriginalMethodFilename

Returns the name of the method file that was used to acquire the specified data file.

Syntax

=Data.OriginalMethodFilename(<Run Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

String

Data.OriginalMethodFullFilename

Returns the full name and path of the method file that was used to acquire the specified data file.

Syntax

=Data.OriginalMethodFullFileName(<Run Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

String

Data.SampleAmount

Returns sample amount of the specified data file.

Syntax

=Data.SampleAmount(<Run Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Data.SampleID

Returns sample id of the specified data file.

Syntax

=Data.SampleID(<Run Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

String

Data.SystemWideParam

Returns a custom system wide result from the specified data file.

Syntax

=Data.SystemWideParam (<Param ID>, <Run Info>)

Parameters

<Param ID> A numeric identifier of the requested system wide custom parameter.

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

String / Number

Data.SystemWideParamByName

Returns a custom system wide result from the specified data file.

Syntax

=Data.SystemWideParamByName (<Param Name>, <Run Info>)

Parameters

<Param ID> The name of the identifier of the requested system wide custom parameter.

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

String / Number

Data.TraceName

Returns trace name for the specified index and data file.

Syntax

=Data.TraceName(<Trace Index>, <Run Info>)

Parameters

<Trace Index> A numeric index of the requested trace.

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

String

Data.UserName

Returns the name of the user that acquired the specified data file.

Syntax

=Data.UserName(<Run Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

String

Data.Vial

Returns vial of the specified data file.

Syntax

=Data.Vial(<Run Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Data.Volume

Returns volume of the specified data file.

Syntax

=Data.Volume(<Run Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Extended Helper Functions

These functions are provided to make the user of other features in the spreadsheet easier to user.

Ex.D

Returns a cell range for a dynamic set of data. If data in a cell will be expanded dynamically, then this function can be used to create a reference to the cells that the data expands into.

Syntax

=Ex.D (<Cell>, [Range Direction])

Parameters

<Cell> Contains a reference to a cell that will be expanded for a dynamic data range. This is in the form of B5, C12, etc. There are no enclosing quotes on the cell reference.

[Range Direction] This is an optional numeric parameter that specifies the direction of the dynamic expansion to use. If data is being expanded in only one direction, then this parameter is not necessary. If data is being expanded both across and down, then this parameter can be used to control the range that will be used. The values of this parameter are as follows:

Not Used or 0 ....Use any dynamic range that is available. If the data expands both across and down, then a range will be generated that contains the entire expansion.

1 ....Only generate a range for dynamic data that expands across the spreadsheet.

2 ....Only generate a range for dynamic data that expands down the spreadsheet.

Return Type

Cell Range

Ex.R

This function can be used to repeat an enclosed spreadsheet formula over a series of cells, based on a dynamic data set. For example, this function could be used to produce a total field showing the sum of a set of peak areas for all peaks in a data file.

When using this function, the enclosed function must not have any run, trace, or peak information. For example, the formula to show the peak area for the first named peak from the current data file using the first trace would be:

```
=Peak.Area("RC", "T1", "P1;3")
```

However, when repeating the formula with the EX.R function to show the peak area for all named peaks from all runs of a sequence using the first trace, the formula would look as follows:

```
=Ex.R(Peak.Area(), "RA;1;0", "T1", "PA;3;0;0")
```

Syntax

=Ex.R(<Spreadsheet Formula>, <Dynamic Run Info>, [Trace Info], [Dynamic Peak Info])

or

=Ex.R(<Spreadsheet Formula>, <Dynamic Run Info>, [Trace Info], [Dynamic Group Info])

Parameters

<Spreadsheet Formula> Contains any valid spreadsheet formula that will be expanded for a dynamic data range.

[Range Direction] This is an optional numeric parameter that specifies the direction to repeat the formula. If the referenced cell is being repeated in only one direction, then this parameter is not necessary. If the referenced cell is being repeated both across and down, then this parameter can be used to control the direction that will be used. The values of this parameter are as follows:

Not Used or 0 ....Repeat exactly like the referenced cell. If the referenced cell repeats both across and down, then this formula will be repeated both across and down.

1 ....Only repeat the formula across the spreadsheet as the referenced cell does.

2 ....Only repeat the formula down the spreadsheet as the referenced cell does.

<Dynamic Run Info> Used to determine the dynamic range to expand the formula over.

[Trace Info] This is an optional parameter that is used to determine the dynamic range to expand the formula over. See the appendix for a description of this parameter.

[Dynamic Peak Info] This is an optional parameter that is used to determine the dynamic range to expand the formula over.

Return Type

None

Dynamic Run Info may be one of the following:

"RC"	The current run or currently loaded data file.
"R<x>"	The sequence run specified by run <x>.
"R<x-y>; <direction>; <separation>"	The sequence runs specified by <x-y>. The runs will be repeated in the

	direction specified by <direction>, and will be separated by <separation> rows or columns.
"RA; <direction>; <separation>"	All sequence runs. The runs will be repeated in the direction specified by <direction>, and will be separated by <separation> rows or columns.

Dynamic Peak Info may be one of the following:

"P<x>; <peak type>"	The peak with an index of <x> having the given peak type.
"P<x-y>; <peak type>; <direction>; <separation>"	The peaks with an index in the range of <x-y> having the given peak type. The peaks will be repeated in the direction specified by <direction>, and will be separated by <separation> rows or columns.
"PA; <peak type>; <direction>; <separation>"	All peaks of the given peak type. The peaks will be repeated in the direction specified by <direction>, and will be separated by <separation> rows or columns.

Dynamic Group Info may be one of the following:

"G<x>; <group type>"	The group with an index of <x> having the given group type.

"G<x-y>; <group type>; <direction>; <separation>"	The groups with an index in the range of <x-y> having the given group type. The groups will be repeated in the direction specified by <direction>, and will be separated by <separation> rows or columns.
"GA; <group type>; <direction>; <separation>"	All groups of the given group type. The groups will be repeated in the direction specified by <direction>, and will be separated by <separation> rows or columns.

Direction may be one of the following:

0	The data will be repeated across the spreadsheet.
1	The data will be repeated down the spreadsheet.

Peak Type may be any combination of the following:

1	Report named peaks that were detected.
2	Report named peaks that were not detected.
4	Report unnamed peaks.

Group Type may be one of the following:

0	Report calibrated range groups that calculate concentrations for unnamed peaks in this group.
1	Report calibrated range groups that do not calculate concentrations for unnamed peaks in this group.
2	Report named peak groups.

Group Functions

These functions return information about groups.

Group.Area

Returns the area for the requested group(s).

Syntax

=Group.Area(<Run Info>, <Trace Info>, <Group Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Group Info> Describes the group(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Group.AreaPercent

Returns the area percent for the requested group(s).

Syntax

=Group.AreaPercent(<Run Info>, <Trace Info>, <Group Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Group Info> Describes the group(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Group.ESTDConcentration

Returns the ESTD concentration for the requested group(s).

Syntax

=Group.ESTDConcentration(<Run Info>, <Trace Info>, <Group Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Group Info> Describes the group(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Group.Height

Returns the height for the requested group(s).

Syntax

=Group.Height(<Run Info>, <Trace Info>, <Group Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Group Info> Describes the group(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Group.HeightPercent

Returns the height percent for the requested group(s).

Syntax

=Group.HeightPercent(<Run Info>, <Trace Info>, <Group Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Group Info> Describes the group(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Group.ISTDConcentration

Returns the ISTD concentration for the requested group(s).

Syntax

=Group.ISTDConcentration(<Run Info>, <Trace Info>, <Group Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Group Info> Describes the group(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Group.Name

Returns the group name for the requested group(s).

Syntax

=Group.Name(<Run Info>, <Trace Info>, <Group Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Group Info> Describes the group(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

String

Group.NORMConcentration

Returns the NORM concentration for the requested group(s).

Syntax

=Group.NORMConcentration(<Run Info>, <Trace Info>, <Group Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Group Info> Describes the group(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Group.Number

Returns the group number for the requested group(s).

Syntax

=Group.Number(<Run Info>, <Trace Info>, <Group Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Group Info> Describes the group(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Group.Quantitation

Returns the group quantitation for the requested group(s). This will return Area, Height, or Counts.

Syntax

=Group.Quantitation (<Run Info>, <Trace Info>, <Group Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Group Info> Describes the group(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

String

Group.ResponseFactor

Returns the response factor for the requested group(s).

Syntax

=Group.ResponseFactor(<Run Info>, <Trace Info>, <Group Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Group Info> Describes the group(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Group.Units

Returns the units for the requested group(s).

Syntax

=Group.Units(<Run Info>, <Trace Info>, <Group Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Group Info> Describes the group(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

String

Instrument Functions

These functions return information about the current instrument.

Instrument.ID

Returns the internal instrument id of the current instrument.

Syntax

=Instrument.ID()

Parameters

None

Return Type

Number

Instrument.Name

Returns the instrument name of the current instrument.

Syntax

=Instrument.Name()

Parameters

None

Return Type

String

Instrument.UserName

Returns the name of the user logged into the current instrument.

Syntax

```
=Instrument.UserName()
```

Parameters

None

Return Type

String

Peak Functions

These functions return information about detected and named peaks.

Peak.AOHResolution

Returns the AOH resolution for the requested peak(s).

Syntax

=Peak.AOHResolution(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.AOHTheoreticalPlates

Returns the AOH theoretical plates for the requested peak(s).

Syntax

=Peak.AOHTheoreticalPlates(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.AOHTheoreticalPlatesPerMeter

Returns the AOH theoretical plates per meter for the requested peak(s).

Syntax

=Peak.AOHTheoreticalPlatesPerMeter(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.Area

Returns the area for the requested peak(s).

Syntax

=Peak.Area(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.AreaPercent

Returns the area percent for the requested peak(s).

Syntax

=Peak.AreaPercent(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.Asymmetry

Returns the asymmetry for the requested peak(s).

Syntax

=Peak.Asymmetry(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.AsymmetryTenPercent

Returns the asymmetry at 10% for the requested peak(s).

Syntax

=Peak.AsymmetryTenPercent(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.CapacityFactor

Returns the capacity factor for the requested peak(s).

Syntax

=Peak.CapacityFactor(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.CurrentResponseFactor

Returns the current response factor for the requested peak(s).

Syntax

=Peak.CurrentResponseFactor(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.CustomParam

Returns a custom peak result for the requested peaks.

Syntax

=Peak.CustomParam(<Param ID>, <Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Param ID> A numeric identifier of the requested peak custom parameter.

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

String / Number

Peak.CustomParamByName

Returns a custom peak result for the requested peaks.

Syntax

=Peak.CustomParamByName(<Param Name>, <Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Param ID> The name of the requested peak custom parameter.

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value. See the appendix for a description of the above parameter(s).

Return Type

String / Number

Peak.DABResolution

Returns the DAB resolution for the requested peak(s).

Syntax

=Peak.DABResolution(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.DABTheoreticalPlates

Returns the DAB theoretical plates for the requested peak(s).

Syntax

=Peak.DABTheoreticalPlates(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.DABTheoreticalPlatesPerMeter

Returns the DAB theoretical plates per meter for the requested peak(s).

Syntax

=Peak.DABTheoreticalPlatesPerMeter(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.EMGResolution

Returns the EMG resolution for the requested peak(s).

Syntax

=Peak.EMGResolution(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.EMGTheoreticalPlates

Returns the EMG theoretical plates for the requested peak(s).

Syntax

=Peak.EMGTheoreticalPlates(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.EMGTheoreticalPlatesPerMeter

Returns the EMG theoretical plates per meter for the requested peak(s).

Syntax

=Peak.EMGTheoreticalPlatesPerMeter(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.ESTDConcentration

Returns the ESTD concentration for the requested peak(s).

Syntax

=Peak.ESTDConcentration(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.ExpectedRetentionTime

Returns the expected retention time for the requested peak(s).

Syntax

=Peak.ExpectedRetentionTime(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.Height

Returns the height for the requested peak(s).

Syntax

=Peak.Height(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.HeightPercent

Returns the height percent for the requested peak(s).

Syntax

=Peak.HeightPercent(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.Index

Returns the peak index information for the requested named peak, based on its peak id. The returned information can be used in place of <Peak Info> for another function. For example to find the peak name of a named peak with a peak id of 2 in the current data file, use the following formula: =Peak.Name("RC", "T1", Peak.Index(2, "RC", "T1"))

Syntax

=Peak.Index(<Peak ID>, <Run Info>, <Trace Info>)

Parameters

<Peak ID> A numeric identifier of the requested named peak. This number comes from the peak table.

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

String

Peak.IntegrationCodes

Returns the integration codes for the requested peak(s).

Syntax

=Peak.IntegrationCodes(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

String

Peak.ISTDConcentration

Returns the ISTD concentration for the requested peak(s).

Syntax

=Peak.ISTDConcentration(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.JPResolution

Returns the JP resolution for the requested peak(s).

Syntax

=Peak.JPResolution(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.JPTheoreticalPlates

Returns the JP theoretical plates for the requested peak(s).

Syntax

=Peak.JPTheoreticalPlates(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.JPTheoreticalPlatesPerMeter

Returns the JP theoretical plates per meter for the requested peak(s).

Syntax

=Peak.JPTheoreticalPlatesPerMeter(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.Name

Returns the peak name for the requested peak(s).

Syntax

=Peak.Name(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

String

Peak.NORMConcentration

Returns the NORM concentration for the requested peak(s).

Syntax

=Peak.NORMConcentration(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.Number

Returns the detected peak number for the requested peak(s).

Syntax

=Peak.Number(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.Quantitation

Returns the peak quantitation for the requested peak(s). This will return Area, Height, or Counts.

Syntax

=Peak.Quantitation (<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

String

Peak.RelativeRetentionTime

Returns the relative retention time for the requested peak(s).

Syntax

=Peak.RelativeRetentionTime(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.Resolution

Returns the resolution for the requested peak(s).

Syntax

=Peak.Resolution(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.ResolutionID

Returns the resolution ID for the requested peak(s).

Syntax

=Peak.ResolutionID(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.ResponseFactor

Returns the response factor for the requested peak(s).

Syntax

=Peak.ResponseFactor(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.RetentionTime

Returns the retention time for the requested peak(s).

Syntax

=Peak.RetentionTime(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.StartTime

Returns the start time for the requested peak(s).

Syntax

=Peak.StartTime(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.StopTime

Returns the stop time for the requested peak(s).

Syntax

=Peak.StopTime(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.TheoreticalPlates

Returns the theoretical plates for the requested peak(s).

Syntax

=Peak.TheoreticalPlates(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.TheoreticalPlatesPerMeter

Returns the theoretical plates per meter for the requested peak(s).

Syntax

=Peak.TheoreticalPlatesPerMeter(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.Units

Returns the concentration units for the requested peak(s).

Syntax

=Peak.Units(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

String

Peak.USPResolution

Returns the USP resolution for the requested peak(s).

Syntax

=Peak.USPResolution(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.USPTheoreticalPlates

Returns the USP theoretical plates for the requested peak(s).

Syntax

=Peak.USPTheoreticalPlates(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.USPTheoreticalPlatesPerMeter

Returns the USP theoretical plates per meter for the requested peak(s).

Syntax

=Peak.USPTheoreticalPlatesPerMeter(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.USPWidth

Returns the USP width for the requested peak(s).

Syntax

=Peak.USPWidth(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.Width

Returns the width for the requested peak(s).

Syntax

=Peak.Width(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.WidthFiftyPercent

Returns the width at 50% for the requested peak(s).

Syntax

=Peak.WidthFiftyPercent(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.WidthFivePercent

Returns the width at 5% for the requested peak(s).

Syntax

=Peak.WidthFivePercent(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Peak.WidthTenPercent

Returns the width at 10% for the requested peak(s).

Syntax

=Peak.WidthTenPercent(<Run Info>, <Trace Info>, <Peak Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

<Trace Info> Describes the trace that will be used to extract the value.

<Peak Info> Describes the peak(s) to use for the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

Project Functions

These functions return information about the current project.

Project.DataPath

Returns the default path used to store data files in the current project.

Syntax

=Project.DataPath()

Parameters

None.

Return Type

String

Project.Description

Returns the description for the current project.

Syntax

=Project.Description()

Parameters

None.

Return Type

String

Project.MethodPath

Returns the default path used to store method files in the current project.

Syntax

=Project.MethodPath()

Parameters

None.

Return Type

String

Project.Name

Returns the name of the current project.

Syntax

=Project.Name()

Parameters

None.

Return Type

String

Project.RootPath

Returns the default root path for the current project.

Syntax

=Project.RootPath()

Parameters

None.

Return Type

String

Project.SequencePath

Returns the default path used to store sequence files in the current project.

Syntax

=Project.SequencePath()

Parameters

None.

Return Type

String

Project.TemplatePath

Returns the default path used to store report template files in the current project.

Syntax

=Project.TemplatePath()

Parameters

None.

Return Type

String

Sequence Functions

These functions return information about the sequence file that will be used for reporting purposes.

Sequence.Filename

Returns file name of the sequence file that will be used for reporting. Only the file name is returned the path information is not returned.

Syntax

=Sequence.Filename()

Parameters

None.

Return Type

String

Sequence.FullFilename

Returns full file name of the sequence file that will be used for reporting. The file name and path information is returned.

Syntax

=Sequence.FullFilename()

Parameters

None.

Return Type

String

Sequence.RunNumber

Returns the run number of the specified sequence run. This function can be used in conjunction with the EX.R() formula to generate the run number of the runs in a sequence. For example, the following formula would generate run numbers for all runs of a sequence going down:
=EX.R(SEQUENCE.RUNNUMBER(),"RA;1;0")

Syntax

=Sequence.RunNumber(<Run Info>)

Parameters

<Run Info> Describes the data file(s) that will be used to extract the value.

See the appendix for a description of the above parameter(s).

Return Type

Number

7 Appendix

The following describes the parameters that may be passed to template functions to describe the requested data file, peak, and group information.

Run Info may be one of the following:

"RC"	The current run or currently loaded data file.
"R<x>"	The sequence run specified by run <x>.

Trace Info may be one of the following:

"T<x>"	The trace specified by index <x>.

Peak Info may be one of the following:

"P<x>; <peak type>"	The peak with an index of <x> having the given peak type.

Group Info may be one of the following:

"G<x>; <group type>"	The group with an index of <x> having the given group type.

Direction may be one of the following:

0	The data will be repeated across the spreadsheet.
1	The data will be repeated down the spreadsheet.

Peak Type may be any combination of the following:

1	Report named peaks that were detected.
2	Report named peaks that were not detected.
4	Report unnamed peaks.

Group Type may be one of the following:

0	Report calibrated range groups that calculate concentrations for unnamed peaks in this group.
1	Report calibrated range groups that do not calculate concentrations for unnamed peaks in this group.
2	Report named peak groups.

8 Advanced Reporting Formulas

This document gives details on formulas available in the spreadsheet engine used in the Advanced Reports of EZChrom *Elite* Client/Server.

1. Spreadsheet Formulas

Formulas are the backbone of the spreadsheet, establishing and calculating mathematical relationships between elements of the spreadsheet. Whereas numeric entries remain the same until you change them, cells defined by formulas are automatically changed to reflect changes in referenced cells - even where there are complex interdependencies among cells.

Spreadsheet formulas can calculate with numbers, text, logical values, cell references, and other formulas. For example, you can easily calculate the sum of a series of cells, the total of values in a column, a minimum or maximum value within a range, the rounded result of another formula, or the absolute value of a cell entry. Formulas can exclick complex interdependencies among cells, and they can define constraints on the calculation, such as limits on acceptable values or specific conditions under which a calculation should take place.

Once entered in a cell, formulas are hidden behind the scenes, perform their work in the background, and display only the result of their calculation. To view the formula in a cell, simply select the cell.

Spreadsheet also provides a wide array of functions that perform certain tasks. Functions can be used alone or in conjunction with formulas and other functions. Spreadsheet provides many specialized functions in addition to those that are found in typical financial spreadsheets.

1.1 Formula Syntax

The general form of an Spreadsheet formula is:

= exclickion ; constraint exclickion // comment

where exclickion defines the calculations needed to generate the cells value, constraint exclickion places limits on acceptable values or the circumstances under which the calculation should take place, and comment is any text you want to attach to the cell.

The exclickion part of Spreadsheet formulas looks just like an algebraic formula; it contains values and operators that define the relationships between values.

Spreadsheet uses the following conventions for formulas:

- A formula must begin with an equal (=) sign. When you begin typing into a cell, Spreadsheet automatically assumes that you are typing a formula if you start with one of the following characters:

0 1 2 3 4 5 6 7 8 9 . - @ =+

- Formulas can have as many as 511 characters. You can type spaces if you wish, but the spreadsheet automatically removes them.

1.2 Formula Values

Formulas can contain any or all of the following types of values:

- Numbers, such as 123, -123, 12.3.
- Addresses of single cells, such as A1, D5, Z100.
- Addresses of cell ranges such as B12..G29, A1..D5.
- Absolute cell references denoted with dollar signs before the fixed coordinate (A1, $A1, or A$1), which will not be updated when the referencing cell is moved or copied.
- Spreadsheet functions, such as @SUM or @RADIANS, with their arguments.
- Text surrounded by double quotation marks, such as "The sum is " or "Total".
- User-defined cell names or cell range names, such as **TOTALS** or **PROJECT1**

1.3 Formula Operators

Spreadsheet supports all the arithmetic, boolean and logical operators available in the C programming language. It does not support the C address operators or the operators that have side effects, such as ++. Spreadsheet provides two operators, exponentiation (**) and percent (%), that are not available in the C language.

Spreadsheet Formulas can contain the following operators to define relationship between values.

Operator	Precedence	Definition
%	14	Unary percent
**	13	Exponentiation
+	12	Unary plus
-	12	Unary minus
~	12	Bitwise complement (integer)
!	12	Logical not
*	11	Multiplication
/	11	Division
%	11	Remainder (integer)
+	10	Addition
-	10	Subtraction
<<	9	Shift left (integer)
>>	9	Shift right (integer)
<	8	Less Than
>	8	Greater Than
<=	8	Less Than or Equal
=	8	Greater Than or Equal
==	7	Equal

!=	7	Not Equal
&	6	Bitwise And, or String Concatenation
^	5	Bitwise Exclusive-Or (integer)
|	4	Bitwise Or
&&	3	Logical And
||	2	Logical Or
?:	1	Conditional

In formulas with more than one operator, Spreadsheet evaluates operators in the order of precedence presented above, with highest precedence first. That is, AND/OR/NOT operators are evaluated after inequality operators in a logical exclickion, and multiplication/division operations are performed before subtraction/addition operations in an arithmetic exclickion. Operators at the same precedence level are evaluated from left to right.

The precedence of operators can be overridden by using parentheses to explicitly specify the order of evaluation.

Here are some special notes about Spreadsheet operators:

- The operators marked ``(integer) on the table above automatically convert their operands to integers.

- The & operator performs double duty: as a bit-wise ``and if the operands are numbers or as a string concatenation operator joining two strings together if the operands are text.

- The % operator also performs double duty: as the ``percent operator when appended to a number or numeric exclickion, or as the C-style ``modulus operator when applied between two integer exclickions.

- Operators that define equality/inequality relationships (such as == and <) can be used to compare text strings lexically (alphabetically). In comparing mixed strings lexically, Spreadsheet considers string operands to be lower than numeric operands.

- The conditional operator returns its second operand if its first operand evaluates True (non-zero) and returns its third operand if it evaluates False, (zero).

- In formulas with conditional operators, the second and third operands may be any type the spreadsheet supports, including ranges. For example, the exclickion

=@SUM(A1 ? B1..C20 : C10..D15)

returns the sum of B1..C20 if A1 evaluates to non-zero; otherwise it returns the sum of C10..D15.

- Spreadsheet accepts most arithmetic operators used in other spreadsheets like MS Excel, but there are a few differences in syntax and precedence.

1.4 Referencing Other Cells in Formulas

The real power of Spreadsheet lies in its ability to calculate relationships among different cells in the spreadsheet by typing the row/column coordinates, or address, in the formula.

To reference a cell by address:

Type the row and column coordinates of the cell in the formula. For example, to reference Row 5 in Column D, type D5.

To reference a contiguous group of cells by address:

Type the row and column coordinates of two cells in opposite corners of the block to be referenced, with two periods (..) between the coordinates. For example, to reference the first five columns and the first five rows of the spreadsheet, type A1..E5.

1.5 Cell Referencing in Spreadsheet

Spreadsheet differentiates between relative, absolute, and indirect references. The latter is unique to Spreadsheet.

1.5.1 Relative Reference

Spreadsheet tracks the referenced cell by considering its position relative to the formula cell, not by its address. For example, if the formula in cell A1 references cell B2, Spreadsheet remembers that the referenced cell is one row down and one column right. If you copy the formula in cell A1 to another location (e.g., D17), the formula will reference the cell one row down and one column right of the new location (e.g., E18).

1.5.2 Absolute Reference

Absolute references remain the same, no matter where you move or copy the original formula. For example, if the formula in cell A1 references cell B2, and you copy the formula in cell A1 to another location (e.g. D17), the formula still references cell B2. To specify an absolute cell address, insert a dollar sign ($) before the address coordinate to be fixed, or before both coordinates if the row and column coordinates are to be fixed. For example: B2.

To specify all or part of a cell address to be absolute, insert a dollar sign ($) before the address coordinate to remain fixed For example:

- B$5 makes the complete address absolute.

- $B5 makes the column coordinate (B) absolute, the row coordinate (5) relative.

- B$5 makes the column coordinate (B) relative, the row coordinate (5) absolute.

Cell ranges are also relative, so when you move a cell range, references in formulas within that range are updated to reflect their new location.

To specify an absolute range reference, insert dollar signs ($) before the coordinates in the formula. For example, to make the range A1..D5 absolute, type the reference as A1..D5.

To specify part of a cell range to be absolute, insert dollar signs only before the coordinates to remain absolute. For example, $A1..$D5 will fix the column coordinates of cell references but adjust the row coordinates to reflect the new location.

1.5.3 To reference a cell or range by name:

Type the pre-assigned name of the cell or cell block into the formula.

To assign a name to a cell or range of cells, use the SetRangeName command.

1.5.4 Current Cell Reference

Certain exclickions within the context of Spreadsheet require a means to exclick the current cell.

Examples include the conditional statistical functions described in "Built-In Worksheet Functions", and constraint exclickions described in Section Constraint Exclickions.

The current cell is identified in any exclickion with a pound sign (#). References to cells in the neighborhood of the current cell are made with offset values enclosed in braces ({}) following the #.

The offsets tell Spreadsheet where to look, in relation to the current cell, for the cell being referenced.

The format is as follows:

#{column offset, row offset}

- If you include only one value in the offset, Spreadsheet assumes that it is a column offset. For example, the offset reference #{-1} tells Spreadsheet to look to the column just left of the current cell.

- The offset values may be constants or exclickions.

Examples:

- #{0,-1} refers to the cell above the current cell.

- #{-2} refers to the cell two columns left of the current cell.

- #{1} refers to the cell to the right of the current cell.

- #{0,1} refers to the cell below the current cell.

- @CSUM(C4..C100, #{-1} == "Joe") calculates the sum of all the values in the range C4..C100 for which the cell in the column to the left contains the string ``Joe.

- @CCOUNT(C4..C100, # #{0,-1}) counts all the cells in the range C4..C100 whose value is greater than the contents of the cell immediately above.

- @XVALUE("master.xs3", #) returns the value of the same cell reference in which this function is stored from the sheet indicated.

- /verb/#-1+2/ adds 2 to the cell value from the cell to the left.

1.6 Constraint Exclickions

Constraints are limitations or conditions placed on the variables in your spreadsheet. They are exclicked as algebraic statements appended to formulas. You can attach a constraint exclickion to any formula, by typing a semicolon (;) and the constraint conditions after the formula.

Constraint exclickions establish conditions under which a formula operates or boundaries for valid results of the formula. Constraint exclickions may be simple equality/inequality relationships, or they can be arbitrary formulas. Any valid Spreadsheet exclickion which returns a numeric value is also a valid constraint exclickion. However, unlike the exclickion that defines a cell value, a constraint exclickion can reference the cell in which it resides, using the symbol #.

For example, the formula

```
=A1 + A2 ; #2 && #<=B5 || #==C7
```

means, ``the value of the current cell is the sum of cells A1 and A2, and that value must be either greater than 2 and less than or equal to the value of cell B5, or equal to the value of cell C7.

Constraint exclickions are used for example in the conditional statistical functions.

The benefit of constraint exclickions is maximized when combine with current cell reference support

(#) as indicated in the above example.

1.7 Explicit Dependency

There may be instances where you need to force a recalculation when certain cell values change, when there is no implicit dependency in the formula that would trigger an automatic recalculation. This option is indicated by appending a backslash (\) to the end of the dependent formula. For example, the formula:

=@SUM(A1..A20)\D50

instructs Spreadsheet to recalculate @SUM(A1..A20) whenever the contents of D50 change. This feature is particularly important when you have a constraint exclickion containing an offset reference that produces a cell reference outside the cell range referenced in a dependent formula. Under these circumstances, Automatic Recalculation would not necessarily be triggered. Take for instance, the example from above:

@CCOUNT(C4..C100, # #{0,-1})

counts all the cells in the range C4..C100 whose value is greater than the contents of the cell immediately above. In order for C4 to be evaluated, it must be compared to C3 - which is not part of the explicit range, C4..C100. Without indicating an explicit dependency, C4 would never be evaluated properly. So, in this case, we would indicate the dependency as follows:

@CCOUNT(C4..C100, # #{0,-1})\C3..C99

which tells Spreadsheet to recalculate whenever any cell in the range C3..C99 changes.

2. Built-in Functions

Spreadsheet functions are predefined formulas supplied with the program. They offer a shortcut approach to accomplishing the work of long, complex formulas. Mathematical and statistical functions are often used to sum a column of numbers, compute an average, determine a minimum or maximum value, or round the results of a formula. Other functions are used for more specialized purposes such as computing the future value of an investment or the product of multiplying one cell range by another range. Some functions perform calculations that arithmetic operators cannot handle such as text-string manipulations.

Spreadsheet functions fall into the following categories:

- Mathematical
- Statistical
- Conditional Statistical
- String
- Logic

- Digital Logic
- Financial
- Date and Time
- Miscellaneous
- Embedded Tools

2.1 Mathematical Functions

Mathematical Functions perform calculations such as determining absolute value, finding the integer portion of a number, or establishing the value of a constant. Although you could accomplish these tasks with a formula, using a function saves time and trouble.

Spreadsheet also provides a full range of trigonometric functions including sine, cosine, tangent, arc sine, hyperbolic sine, hyperbolic arc sine as well as vector and matrix arithmetic and manipulation.

Mathematical functions perform calculations with numeric values as arguments, returning numeric values.

2.2 Statistical Functions

Statistical Functions perform aggregation operations such as calculating means, minimums, maximums, and averages.

Spreadsheet also provides more sophisticated statistical test functions that perform operations on a group of values exclicked as a list of arguments. These include the F-test, T-tests, correlation coefficient, deviations, and all common averages.

Statistical functions return numeric values.

2.3 Conditional Statistical Functions

Conditional Statistical Functions operate much like statistical aggregation functions, except that the last argument is a constraint exclickion that Spreadsheet evaluates for each cell in the argument list.

Only cells that meet constraint criteria are included in the calculation. The constraint exclickion may be any Spreadsheet exclickion that evaluates to a numeric result.

Conditional Statistical Functions return a numeric value.

2.4 String Functions

String Functions manipulate and evaluate character strings. For example, string functions can return the length of a string, find the first occurrence of a string in a range, change a string from upper to lower-case and vice versa, or replace one string with another.

String functions return strings or numeric values.

2.5 Logic Functions

Logic Functions return one value if an argument meets certain criteria, another value if it does not.

Logic functions are used as an adjunct to conditional statements.

Logic functions return the value 1, 0, or a value.

2.6 Digital Logic Functions

Digital Logic Functions perform digital logic operations such as AND, OR, NOT, etc.

Digital logic functions return the values 0, 1, or -1 (unknown). Any value whose integer portion is not equal to 0 or 1 is considered unknown. Unknown input values may cause unknown output values.

2.7 Financial Functions

Financial Functions perform common financial calculations, such as calculating the future value of an annuity at a given interest rate, straight-line depreciation, double-declining depreciation, or the payment term for a given investment. The financial functions in Spreadsheet cover annuities, cash flows, assets. bonds, and Treasury Bills.

Financial functions are most useful for solving cash flow calculations where you know all but one variable. For example, if you know the present value of an investment, interest rate, and periodic payment, you can use the @FV function to calculate the future value of the investment. If you know the future value and other variables, but need to know the present value, you can use the @PV function.

Many financial functions require specifying a Day Count Basis. A Day Count Basis indicates the way in which the days in a month and the days in a year are to be counted. Most of the financial functions in securities involve 4 different Day Count Basis: 30/360, actual/actual, actual/360 and actual/365. 30/360 Day Count Basis assumes 30-day months and 360-day years (12 months x 30 days). Spreadsheet also follows the ``End-of-Month rule which assumes that a security pays interest on the last day of the month and will always make its interest on the last day of the month. Special rules are followed when calculating the days between two dates on 30/360 Day Count Basis.

For example, let Start_Date = D1/M1/Y1, End_Date = D2/M2/Y2.

1. If D1=31, Spreadsheet uses 30 for D1.

2. If D2=31, Spreadsheet uses 31, unless D1=30 or D1=31. In this case, Spreadsheet uses 30.

3. If D1 is the last day of February (D1=28 or 29 in a leap year), Spreadsheet uses 30 for D1.

4. If D2 is the last day of February (D2=28 or 29 in a leap year) and D1 is also the last day of

February, Spreadsheet uses 30 for D2.

The special arguments used by Spreadsheet financial functions are defined in Table TODO:

Financial functions use the arguments defined in Table **interest rate** The interest rate to be used in the calculations. The rate may be specified as annual, monthly or quarterly, but it must agree with the increment you use for periods. By default the interest rate is an annual rate.

present value The present value of an investment, representingthe amount already received from or committed to an investment.

period The number of periods over which the loan, investment or depreciation is to be calculated. The periods may be defined in months, quarters or years, but must agree with the increment used to define interest rate.

future value The future value of an investment, given a certain present value, interest rate, and number of periods.

cost The original cost of a depreciable capital asset.

salvage value The remaining value of a capital asset after the depreciation period has expired.

allowable life The allowable life of a depreciable item.

yield The interest rate that will make the present value

of the expected future cash flows equal to the

price of the financial instrument.

price The present value of the expected future cash

flows where the discount rate is equal to the yield

of the financial instrument.

coupon rate The annual coupon rate of a security.

frequency The number of coupon payments in a year.

basis The day count basis to be used in calculation.

Functions related fixed income securities usually require special dates as arguments: issue date, settlement date, first coupon date, last coupon date, maturity date of a security. When specified, the following constraints should be followed:

issue settlement maturity

issue first coupon maturity

issue last coupon maturity

2.8 Date and Time Functions

Date and Time Functions return values corresponding to the specified date, month, year, hour, minute or second. You can also use date/time functions to enter the current system time and date in a cell.

These functions open up many possibilities for managing accounts receivable and calculating test times.

Spreadsheet internally stores date and time information using the same convention as other popular spreadsheet programs:

• Dates are represented as an integer equal to the number of days since December 31, 1899, so

January 1, 1900 equals 1.

• Times are represented as fractions of a day, starting at midnight. For example, 6:00 AM is stored

as 0.25 (a quarter of a 24-hour day).

Using this convention, date and time values may be used together. For example, the date/time value

1.25 corresponds to 6:00:00 AM, January 1, 1900.

2.9 Miscellaneous Functions

Miscellaneous Functions perform a variety of calculations, such as returning a reference to specific cells or ranges or returning the Nth argument from a list of arguments.

2.10 Embedded Tools

Embedded Tools are a powerful feature in Spreadsheet. Their power derives in part from their ability to return a set of data, not just a single value. This function makes non-scalar operations such as matrix multiplication and "live" recalculation as easy to use as an ordinary spreadsheet function.

Embedded tools store values in a group of adjacent cells. These adjacent cells are set to constant formulas with explicit dependencies on their neighboring cells. For example, an embedded tool in cell

B2 might produce the formula =1.3459\B2 in cell B3. This formula indicates that the cell currently contains the constant 1.3459 but that its value depends on the contents of cell B2 (the cell containing the embedded tool).

This notion of explicit dependencies is important for recalculation. It guarantees that any cell that references B3 will not be recalculated until after cell B2 is recalculated. This ensures that data generated by the embedded tool is always current.

Embedded tools look like normal functions, and they can be copied, moved and formatted just as any other formula in the spreadsheet. However, you must follow one important guideline: DO NOT combine embedded tools with other embedded tools in a single formula. For example, the formula

@INVERT(@MMUL(A1..C4,F1..I3))

is not allowed.

3. Quick-Reference Guide to Built-in Functions

3.1 Mathematical Functions

@ABS(*X*) -The absolute value of *X*.

@ACOS(*X*) -The arc cosine of *X*.

@ASIN(*X*) -The arc sine of *X*.

@ATAN(*X*) -The 2-quadrant arc tangent of *X*.

@ATAN2(*X, Y*) -The 4-quadrant arc tangent of *Y/X*.

@CEIL(*X*) -The smallest integer greater than or equal to *X*.

@COS(*X*) -The cosine of *X*.

@COSH(*X*) -The hyperbolic cosine of *X*.

@DEGREES(*X*) -Converts the angle exclicked in radians to degrees ( ).

@DET(*M*) -The determinant of the matrix range *M*, which must be a square matrix.

@DOT(*R1*, *R2*) -The dot product of the vectors *R1* and *R2*.

@EXP(*X*) -*e* raised to the *X* power.

@FACT(*N*) - The value of *N!*.

@FLOOR(*X*) -The largest integer less than or equal to *X*.

@FRAC(*X*) -The fractional portion of *X*.

@GAMMA(*X*) -The value of the gamma function evaluated at *X*.

@GRAND -A 12th-degree binomial approximation to a Gaussian random number with zero mean and unit variance.

@INT(*X*) -The integer portion of *X*.

@LN(*X*) -The natural log (base *e*) of *X*.

@LNGAMMA(*X*) -The log base *e* of the gamma function evaluated at *X*.

@LOG(*X*) -The log base 10 of *X*.

@LOG10(*X*) -The log base 10 of *X*.

@LOG2(*X*) -The log base 2 of *X*.

@MOD(*X, Y*) -The remainder of *X/Y*.

@MODULUS(*X, Y*) -The modulus of *X/Y*.

@PI -The value of p.

@POLY(*X*, ...) -The value of an *N*th-degree polynomial in *X*.

@PRODUCT(*X*, ...) -The product of all the numeric values in the argument list.

@RADIANS(*X*) -Converts the angle exclicked in degrees to radians ( ).

@RAND -A uniform random number on the interval [0,1).

@ROUND(*X, n*) -*X* rounded to n number of decimal places (0 to 15).

@SIGMOID(*X*) -The value of the sigmoid function .

@SIN(*X*) -The sine of *X*.

@SINH(*X*) -The hyperbolic sine of *X*.

@SQRT(*X*) -The positive square root of *X*.

@SUMPRODUCT(*R1, R2*) -The dot product of the vectors *R1* and *R2*, where *R1* and *R2* are of equal dimension.

@TAN(*X*) -The tangent of *X*.

@TANH(*X*) -The hyperbolic tangent of *X*.

@TRANSPOSE(*M*) -The transpose of matrix *M*.

@VECLEN(...) -The square root of the sum of squares of its arguments.

3.2 Statistical Functions

@AVG(...) -The average (arithmetic mean) of its arguments.

@CORR(*R1, R2*) -Pearsons product-moment correlation coefficient for the paired data in ranges *R1* and *R2*.

@COUNT(...) -A count of its non-blank arguments.

@F(*M, N, F*) -The integral of Snedecors *F*-distribution with *M* and *N* degrees of freedom from minus infinity to *F*.

@ERF(*L[, U]*) -Error function integrated between 0 and *L*; if *U* specified, between *L* and *U*.

@ERFC(*L*) -Complementary error function integrated between *L* and infinity.

@FORECAST(...) -Predicted *Y* values for given *X*.

@FTEST(*R1, R2*) -The significance level ( ) of the two-sided *F*-test on the variances of the data specified by ranges *R1* and *R2*.

@GMEAN(...) -The geometric mean of its arguments.

@HMEAN(...) -The harmonic mean of its arguments.

@LARGE(*R, N*) -The *N*th largest value in range *R*.

@MAX(...) -The maximum of its arguments.

@MEDIAN(...) -The median (middle value) of the range *R1*.

@MIN(...) -The minimum of its arguments.

@MODE(...) - The mode, or most frequently occurring value.

@MSQ(...) -The mean of the squares of its arguments.

@PERCENTILE(*R, N*) -The value from the range *R* which is at the *N*th percentile in *R*.

@PERCENTRANK(*R, N*) -The percentile rank of the number *N* among the values in range *R*.

@PERMUT(*S, T*) -The number of *T* objects that can be chosen from the set *S*, where order is significant.

@PTTEST(*R1, R2*) -The significance level ( ) of the two-sided *T*-test for the paired samples contained in ranges *R1* and *R2*.

@QUARTILE(*R, Q*) -The quartile *Q* of the data in range *R*.

@RANK(*E, R[, O]*) -The rank of a numeric argument *E* in the range *R*.

@RMS(...) -The root of the mean of squares of its arguments.

@SMALL(*R, N*) -The *N*th smallest number in range *R*.

@SSE(...) -The sum squared error of its arguments. It is equivalent to @VAR(...) @COUNT(...).

@SSQ(...) -The sum of squares of its arguments.

@STD(...) -The population standard deviation (N weighting) of its arguments.

@STDS(...) -The sample standard deviation (N-1 weighting) of its arguments.

@SUM(...) -The sum of its arguments.

@T(*N, T*) -The

integral of Students *T*-distribution with *N* degrees of freedom from minus infinity to *T*.

@TTEST(*R, X*) -The significance level (of the two-sided single population *T*-test for the population samples contained in range *R*.

@TTEST2EV(*R1, R2*) -The significance level ( ) of the two-sided dual population *T*-test for ranges *R1* and *R2*, where the population variances are equal.

@TTEST2UV(*R1, R2*) -The significance level ( ) of the two-sided dual population *T*-test for ranges *R1* and *R2*, where the population variances are not equal.

@VAR(...) -The sample variance (N weighting) of its arguments.

@VARS(...) -The sample variance (N-1 weighting) of its arguments.

@VSUM(...) -The ``visual sum of its arguments, using precision and rounding of formatted cell values.

3.3 Conditional Statistical Functions

@CAVG(..., C) – Conditional average.

@CCOUNT(..., C) – Conditional count.

@CMAX(..., C) – Conditional maximum.

@CMIN(..., C) – Conditional minimum.

@CSTD(..., C) – Conditional sample standard deviation (N weighting).

@CSTDS(..., C) – Conditional sample standard deviation (N-1 weighting).

@CSUM(..., C) – Conditional sum.

@CVAR(..., C) – Conditional population variance (N weighting).

@CVARS(..., C) – Conditional population variance (N-1 weighting).

3.4 String Functions

@CHAR(*N*) -The character represented by the code *N*.

@CLEAN(*S*) -The string formed by removing all non-printing characters from the string *S*.

@CODE(*S*) -The ASCII code for the first character in string *S*.

@EXACT(*S1, S2*) -Returns true (1) if string *S1* exactly matches string *S2*, otherwise returns 0.

@FIND(*S1, S2, N*) -The index of the first occurrence of *S1* in *S2*.

@HEXTONUM(*S*) -The numeric value for the hexadecimal interpretation of *S*.

@LEFT(*S, N*) -The string composed of the leftmost *N* characters of *S*.

@LENGTH(*S*) -The number of characters in *S*.

@LOWER(*S*) -*S* converted to lower case.

@MID(*S, N1, N2*) -The string of length *N2* that starts at position *N1* in *S*.

@NUMTOHEX(*X*) - The hexadecimal representation of the integer portion of *X*.

@PROPER(*S*) -The string *S* with the first letter of each word capitalized.

@REGEX(*S1, S2*) -Returns true (1) if string *S1* exactly matches string *S2*; otherwise returns false (0). Allows ``wildcard comparisons by interpreting *S1* as a regular exclickion.

@REPEAT(*S, N*) -The string *S* repeated *N* times.

@REPLACE(*S1, N1, N2, S2*) -The string formed by replacing the *N2* characters starting at position *N1* in *S1* with string *S2*.

@RIGHT(*S, N*) -The string composed of the rightmost *N* characters of *S*.

@STRCAT(...) -The concatenation of all its arguments.

@STRING(*X, N*) -The string representing the numeric value of *X*, to *N* decimal places.

@STRLEN(...) -The total length of all strings in its arguments.

@TRIM(*S*) -The string formed by removing spaces from the string *S*.

@UPPER(*S*) -The string *S* converted to upper case.

@VALUE(*S*) -The numeric value represented by the string *S*; otherwise 0 if *S* does not represent a number.

3.5 Logic Functions

@FALSE -The logical value 0.

@FILEEXISTS(*S*) -1 if file *S* can be opened for reading; otherwise 0.

@IF(*X, T, F*) -The value of *T* if *X* evaluates to on-zero, or *F* if *X* evaluates to zero.

@ISERROR(*X*) -Returns 1 if *X* ``contains an error, otherwise 0.

@ISNUMBER(*X*) -1 if *X* is a numeric value; otherwise 0.

@ISSTRING(*X*) -1 if *X* is a string value; otherwise 0.

@TRUE -The logical value 1.

Digital Logic Functions

@AND(...) -0 if any arguments are 0; 1 if all arguments are 1; otherwise -1.

@NAND(...) -0 if all arguments are 1; 1 if any arguments are 0; otherwise -1.

@NOR(...) -0 if any arguments are 1; 1 if all arguments are 0; otherwise -1.

@NOT(*X*) -0 if *X*=1; 1 if *X*=0; otherwise -1.

@OR(...) -0 if all arguments are 0; 1 if any arguments are 1; otherwise -1.

@XOR(...) -- 1 if any of the arguments are not 0 or 1; otherwise 0 if the total number of arguments with the value 1 is even; 1 if the total number of arguments with the value 1 is odd.

73.6 Financial Functions

@ACCRINT(I, Ft, S, R, P, F[, B]) -Accrued interest for a security that pays periodic interest.

@ACCRINTM(I, S, R, P[, B]) - Accrued interest for a security that pays interest at maturity.

@COUPDAYBS(S, M, F[, B]) -The number of days between the beginning of the coupon period to the settlement date.

@COUPDAYS(S, M, F[, B]) -The number of days in the coupon period that the settlement date is in.

@COUPDAYSNC(S, M, F[, B]) -The number of days between the settlement date and the next coupon date.

@COUPNCD(S, M, F[, B]) -The next coupon date after the settlement date.

@COUPNUM(S, M, F[, B]) -The number of coupon payments between the settlement date and maturity date.

@COUPPCD(S, M, F[, B]) -The previous (most recent) coupon date before the settlement date.

@CTERM(R, FV, PV) -The number of compounding periods for an investment.

@CUMIPMT(R, NP, PV, S, E, T) -The cumulative interest on a loan between start period S and end period E.

@CUMPRINC(R, NP, PV, S, E, T) -The cumulative principal paid on a loan between start period S and end period E.

@DB(C, S, L, P[, M]) -Fixed- declining depreciation allowance.

@DDB(C, S, L, N) -Double- declining depreciation allowance.

@DISC(S, M, P, R[, B]) -The discount rate for a security.

@DOLLARDE(FD, F) -Converts a dollar amount exclicked as a fraction form into a decimal form.

@DOLLARFR(DD, F) -Converts a dollar amount exclicked as a decimal form into a fraction form.

@DURATION(S, M, R, Y, F[, B]) -

The Macauley duration of a security assuming $100 face value.

@EFFECT(*NR, NP*) -Returns the effective annual interest rate.

@FV(*P, R, N*) -Future value of an annuity.

@FVSCHEDULE(*P, S*) -The future value of an initial investment after compounding a series of interest rates.

@INTRATE(*S, M, I, R[, B]*) -The interest rate for a fully invested security.

@IPMT(*R, P, NP, PV, FV[, T]*) -The interest payment for a specific period for an investment based on periodic, constant payments and a constant interest rate.

@IRR(*G, F*) -The internal rate of return on an investment. (See also @XIRR and @MIRR.)

@MDURATION(*S, M, R, Y, F[, B]*) -The modified Macauley duration of a security assuming $100 face value.

@MIRR(*CF, FR, RR*) -The modified internal rate of return for a series of periodic cash flows.

@NOMINAL(*ER, NP*) -The nominal annual interest rate.

@ODDFPRICE(*S, M, I, FC, R, Y, RD, F[, B]*) -The price per $100 face value of a security with an odd (short or long) first period.

@ODDFYIELD(*S, M, I, FC, R, PR, RD, F[, B]*) -The yield per of a security with an odd (short or long) first period.

@PMT(*PV, R, N*) -The periodic payment for a loan.

@PPMT(*R, P, NP, PV, FV, T*) -The payment on the principal for a specific period for an investment based on periodic, constant payments and a constant interest rate.

@PRICE(*S, M, R, Y, RD, F[, B]*) -The price per $100 face value of a security that pays periodic interest.

@PRICEDISC(*S, M, D, RD[, B]*) -The price per $100 face value of a discounted security.

@PRICEMAT(*S, M, I, R, Y[, B]*) -The price per $100 face value of a security that pays interest at maturity.

@PV(*P, R, N*) -The present value of an annuity

@RATE(*FV, PV, N*) -The interest rate required to reach future value *FV*.

@RECEIVED(*S, M, I, D, [, B]*) -The amount received at maturity for a fully vested security.

@SLN(*C, S, L*) -The straight-line depreciation allowance.

@SYD(*C, S, L, N*) -The ``sum-of-years-digits depreciation allowance.

@TBILLEQ(*S, M, D*) -The bond-equivalent yield (BEY) for a Treasury Bill.

@TBILLYIELD(*S, M, D*) -The yield on a Treasury bill.

@TERM(*P, R, FV*) -The number of payment periods for an investment.

@VDB(*C, S, L, S, E*) -Fixed- declining depreciation allowance between two periods.

@XIRR(*G, V, D*) -Internal rate of return for a series of cash flows with variable intervals.

@XNPV(*R, V, D*) -Returns the net present value for a series of cash flows with variable intervals.

@YIELD(*S, M, R, PR, RD, F[, B]*) -Yield of a security that pays periodic interest.

@YIELDMAT(*S, M, I, R, PR[, B]*) -Annual yield of a security which pays interest at maturity.

3.7 Date and Time Functions

@DATE(*Y, M, D*) -The date value for year *Y*, month *M*, and day *D*.

@DATEVALUE(*S*) -The corresponding date value for a given string *S*.

@DAYS360(*S, E*) -The number of days between two dates, based on a 30/360 day count system.

@DAY(*DT*) -The day number in the date/time value *DT*.

@EDATE(*S, M*) -The date/time value representing number of months (*M*) before or after start date (*S*).

@EOMONTH(*S, M*) -The date/time value representing the last day of the month *M* months after *S*, if M is positive, or *M* months before if *M* is negative.

@HOUR(*DT*) -The hour value (0-23) of date/time value *DT*.

@MINUTE(*DT*) -The minute value (0-59) of date/time value *DT*.

@MONTH(*DT*) -The number of the month in date/time value *DT*.

@NETWORKDAYS(*S, E[, H]*) -The number of whole working days, starting at *S* and going to *E*, excluding weekends and holidays.

@NOW -The date/time value of the current system date and time.

@SECOND(*DT*) -The seconds value (0-59) of the date/time value *DT*.

@TIME(*H, M, S*) -The time value for hour *H*, minute *M*, and second *S*.

@TIMEVALUE(*S*) -The corresponding time value for a given string value *S*.

@TODAY -The date value of the current system date.

@WEEKDAY(*D*) -The integer representing the day of the week on which the day *D* falls. 1 is Sunday, 7 is Saturday.

@WORKDAY(*S, D[, H]*) -The day that is *D* working days after *S*, if *D* is positive, or before *S*, if *D* is negative, excluding weekends and all holidays specified as dates in range *H*.

@YEAR(*DT*) -The year value of date/time value *DT*.

@YEARFRAC(*S, E[, B]*) -The portion of the year represented by the number of days between start date (*S*) and end date (*E*).

3.8 Miscellaneous Functions

@CELLREF(*N1, N2*) -A reference to the cell in column *N1* and row *N2*.

@CHOOSE(*N, ...*) -The *N*th argument from the list.

@COL(*C*) -The column address of the cell referenced by *C*.

@COLS(*R*) -The number of columns in the specified range *R*.

@HLOOKUP(*X, S, R*) -The value of the cell in range *S* that is *R* number of rows beneath *X*.

@INIT(*X1, X2*) -The first argument on the first recalculation pass and the second argument on all subsequent recalculation passes when Spreadsheet is performing iterative calculations.

@INTERP2D(*R1, R2, N*) -The interpolation value for a 2-dimensional vector.

@INTERP3D(*R, X, Y*) -The interpolation value for a 3-dimensional vector.

@MATCH(*V, R[, T]*) -The relative position in range *R* of value *V* based on positioning criteria *T*.

@N(*R*) -The numeric value of the top left cell in range *R*.

@RANGEREF(*N1, N2, N3, N4*) -A reference to the range defined by coordinates *N1* through *N4*.

@ROW(*C*) -The row address of the cell referenced by *C*.

@ROWS(*R*) -The number of rows in the specified range *R*.

@S(*R*) -The string value of the top left cell in range *R*.

@VLOOKUP(*X, S, C*) -The value of the cell in range *S* that is *C* number of columns to the right of *X*.

IMPORTANT: Some Spreadsheet functions return a result that is a range or cell reference.

Spreadsheet does not include these indirect references in determining the pattern of recalculation.

Plan carefully before using these functions. See the section, Computed Cell References at the end of this chapter for more information.

3.9 Embedded Tools

@DFT(*R*) -The Discrete Fourier Transform of the range *R*.

@EIGEN(*M*) -The eigenvalues of the matrix *M*.

@FFT(*R*) -The Discrete Fourier Transform of the range *R* using a fast Fourier Transform algorithm.

@FREQUENCY(*R, B*) -Returns a frequency distribution for values *R* with a set of intervals *B*.

@INVDFT(*R*) -The inverse of the Discrete Fourier Transform of the range *R*.

@INVERT(*M*) -The inverse of matrix *M*.

@INVFFT(*R*) -The inverse of the Discrete Fourier Transform of the range *R* using a fast Fourier Transform algorithm.

@LINFIT(*X, Y*) -The straight line least squares fit. This function is equivalent to @POLYFIT(*X, Y, 1*).

@LLS(*A, Y*) -The linear least squares solution *X* to the over-determined system of equations *AX=Y*.

@MMUL(*M1, M2*) -The product of multiplying matrix *M2* by matrix *M1*.

@PLS(*X, Y, d*) -Analyzes the least squares polynomial model *Y=P(X)*, where *P* is a polynomial of degree *d*.

@POLYCOEF(*X, Y, d*) -The least squares coefficients for the polynomial fit *Y=P(X)*, where *P* is a polynomial of degree *d*.

@TRANSPOSE(*M*) -The transpose of matrix *M*.

@TREND(*NX, KX, KY*) -The y values for new x values given existing x and y values.

Note

Embedded tools should not be contained within other functions or arithmetic operations in a single formula. You may, however, copy, move and format embedded tools just as any other function.

4. Using Spreadsheet Built-in Functions

You enter a function in a cell in the same way you enter a formula or any other entry, with a few additional guidelines.

- Type in the function name. Spreadsheet recognizes the string as a function. Function names are abbreviations that indicate what the function does. For instance, ABS computes absolute value, ROUND rounds to the specified number of places, and AVG computes the average of a list of arguments. Function names may be preceded with an @ sign, but this is not required.

- After typing the function name, enter arguments in parentheses. Most functions use one or more arguments to define the task to be performed. For example, the @AVG function averages the value of two or more arguments. The @LENGTH function returns the length of an argument that is a character string.

- Use only the arguments required by the function, in the exact order specified in the function syntax. If you enter other arguments or enter them in the wrong order, Spreadsheet will misinterpret their meaning or return an error message.

- All the function names in this chapter are typed in uppercase letters, but you can enter them in upper or lower-case for your entries.

4.1 Arguments

Arguments specify the values the function should use in its calculations. The number of arguments, their types, and their formats varies from one function to another. Arguments are usually numeric values, cell or range references, or string values. Most functions have at least one argument; a few have none.

The following chart shows different types of arguments used in Spreadsheet functions.

Argument Example

Numeric Value 123

Address of a cell A10

Address of a range F9..F99

String Value ``Quarterly Report

4.2 Using Operators with Functions

The result of a function depends on the order in which Spreadsheet handles the calculations. Please see Chapter Calculations, for more information on operators and their precedence.

4.3 Computed Cell References

Several Spreadsheet functions such as @CELLREF and @RANGERE return a result that is itself a cell reference or range reference. This is a powerful facility, but it must be used with caution because Spreadsheet can not take these indirect references into account when determining the order of recalculation. The same caution applies to constraint exclickions used in conditional statistical functions. As a rule, cells that are indirectly referenced by a function are not automatically recalculated. Spreadsheet provides a special construct to force a recalculation, referred to as an explicit dependency.

Spreadsheet does not recalculate the spreadsheet unless explicit dependencies have been changed, so you may need to force recalculation if you change the value of a cell that is referenced only indirectly through a function.

For example, suppose you want to count the numeric values in the range C3..J100 that fall within the limits specified in cells A1 and A2. The Spreadsheet formula to compute this is

@CCOUNT(C3..J100,#A1 && #<A2)

This formula will correctly count the numeric values in the range C3..J100. However, if you change the value in A1, Spreadsheet will not automatically recalculate the result, because A1 is referenced only indirectly through the constraint exclickion.

- To force Spreadsheet to recalculate the entire spreadsheet you should call the Recalc() command. You should also add Recalculate menu in your application that calls Recalc().

- You can also force Spreadsheet to do a partial recalculation with respect to that cell, edit the cell and append a blank and click the [Return] key on the cell containing the @CCOUNT formula.

- You can also use explicit dependencies to circumvent the limitation described above, if you entered the formula below in the form

@CCOUNT(C3..J100,#A1 && #<A2)\A1\A2

Spreadsheet would take into account the dependencies on A1 and A2 and update the spreadsheet just as you expect.

- Another approach is to construct the condition string with an exclickion that references the cells directly. For example,

@CCOUNT(C3..J100, @STRCAT("#",A1,"&_<",A2))

In this example, A1 and A2 are directly referenced and thus will properly trigger recalculation.

Explicit Dependency is described in more detail in Section Explicit Dependency.

5. Spreadsheet Error Messages

Spreadsheet checks for a variety of errors. Depending on the error type, the most recent error message is displayed either inside the affected cell(s), on the Message Line or is displayed inside the Spreadsheet Message dialog box.

5.1 Types of Errors

5.1.1 Errors in Functions

Errors that occur inside functions are reported along with the name of the function in which the error occurred.

5.1.2 Formula Syntax Errors

These errors occur only when you are typing in a formula. When you finish entering the formula, Spreadsheet will attempt to read the formula and convert it to an internal representation. If it is unable to do so, it continues to display the erroneous formula, switches into ``edit mode, places the text cursor at the beginning of the text which it had difficulty parsing, and displays the error message.

The problem must be corrected before Spreadsheet can continue.

5.1.3 Formula Evaluation Errors

Formula evaluation error occurs when Spreadsheet reads in a formula and converts it into its internal formula representation, but is not able to evaluate the formula and produce a correct numeric or string formula. In some cases, the formula has been entered incorrectly, for example, an operand or parenthesis is missing. In other cases, an error has occurred as a result of computation that cannot be handled properly by the computers floating point hardware, or there is an error condition in a cell or range that is referenced in the context of this formula. Errors can also occur in the evaluation of Spreadsheet built-in functions.

5.2 Summary of Error Messages

argument must be an integer

@FACT has been passed a non-integer argument.

argument not a cell or range

@@ has been passed an argument that is neither a cell nor a range.

argument out of range

An argument to a function is not within the correct range for the function and its other arguments.

arguments must be numeric

The function requires numeric arguments, which may be literal numbers, formulas which return numeric values, or references to cells containing numeric values.

arguments must be positive

The arguments in this function must be all positive values.

can not parse condition string

Spreadsheet has encountered a malformed conditional exclickion.

cannot find interpolation

@INTERP2D or @INTERP3D is unsuccessful in finding interpolated values.

cash flow series must be a range

@NPV and @MIRR require that their cash flow series must be a range, which must represent a single column or row.

cash flow series must be single column or row

@NPV and @MIRR require that their cash flow series must be a range, which must represent a single column or row.

cell operand contains error condition

A cell which is referenced from the cell in which the error occurs contains an error condition.

cell reference out of range

A cell reference has been made which is outside the range A1..FAN32767

coefficient matrix has linearly dependent columns

The existence of a unique solution to a linear least squares (@LLS) problem, Ax=b, requires that the columns of A are linearly independent.

column offset out of range

The third argument to the @VLOOKUP function specifies an offset that is less than 0 or is greater than the width of the range specified in the second argument.

constraint check not supported with ``As Needed

Constraint checking is not supported when the recalculation is set to ``As Needed.

contains an error indicator

A cell in one or more of the data ranges for a graph contains an error condition. The error condition must be resolved before Spreadsheet can plot the graph.

could not find real root

@IRR could not find a real root. This suggests that the data given to @IRR is probably wrong.

count less than zero

User has passed a negative argument to a function which requires a count, for example, with @LEFT, it is impossible to take the -2 leftmost characters of a string.

data set size must be = 3

@LINFIT and @LINCOEF require a data set of size 3 or larger.

data set size must be = polynomial degree + 2

@PLS, @POLYFIT, and @POLYCOEF require that the data set size be greater than or equal to the polynomial degree + 2.

date series must be single column or row

@XIRR and @XNPV require the argument D (date series) to be a single column or single row.

decimal places out of range

@STRING only takes a decimal place argument between 0 and 15.

degrees of freedom must be 0

@F and @T require degrees of freedom greater than zero, as ``degrees of freedom is mathematically undefined for zero or less.

dimension must be power of 2

@FFT and @INVFFT require matrices whose dimensions are powers of two. The somewhat slower functions @DFT and @INVDFT, respectively, are equivalent functions which do not share this requirement.

divide by zero

An attempt has been made to divide by zero. Note that Spreadsheet considers cells which are empty or contain text strings to have the value zero in the context of a numerical calculation.

does not accept arguments

Several Spreadsheet functions, including @PI, @TRUE, @FALSE, @RAND, and @GRAND, do not accept any arguments.

domain is -1 < x < 1

@ATANH only takes arguments between -1 and 1, exclusive.

domain is -1 <= x <= 1

@ACOS and @ASIN only take arguments between -1 and 1, inclusive.

domain is 0 <= x <= 170

@FACT only takes arguments between 0 and 170, inclusive. (Most platforms)

domain is 0 <= x <= 33

@FACT only takes arguments between 0 and 33, inclusive. (VAX platforms)

domain is x 0

@LN, @LOG2, @LOG, @GAMMA, and @LNGAMMA only take arguments greater than zero.

domain is x = 1

@ACOSH only takes arguments greater than or equal to 1.

``End Period must be = 1

@CUMIPMT and @CUMPRINC require the argument E (end period) to be greater than or equal to 1.

``End Period must be = ``Start Period

@CUMIPMT, @CUMPRINC and @VDB require the argument E (end period) to be greater than or equal to S (start period).

ending line with a

\

The \ is an escape sequence introducer, which should be followed by another character for interpretation, but the string ended prematurely.

ending line with a superscript command

When displaying text in the context of graphics, a **^** is a superscript introducer. Like y**^**2 means ``y squared. This message occurs when a **^** occurs at the end of the string.

ending line with subscript command

When displaying text in the context of graphics, an `_ is a subscript introducer. Like y_2 means ``y subscript 2. This message occurs when an `_ occurs at the end of the string.

error in regular exclickion

An error occurred while parsing the regular exclickion used in a search or extract operation, or while executing @REGEX or @MATCH.

expected the right hand side of a range here

The outer range reference is missing.

expected to find [*something*] here

There was a parsing error. The cursor will be placed in the edit window in edit mode. Read the documentation for the function and correct the error.

expecting a function

There is something wrong with the formula you have entered on the edit line. The parser was expecting to find a function name at the point indicated by the cursor position.

expecting an operand

There is something wrong with the formula you have entered on the edit line. The parser was expecting to find a function name at the point indicated by the cursor position.

expecting an operator

There is something wrong with the formula you have entered on the edit line. The parser was expecting to find a function name at the point indicated by the cursor position.

extraneous operands

There is something wrong with the formula you have entered on the edit line. The parser was expecting to find a function name at the point indicated by the cursor position.

F must be = 0

The third argument to @F must be greater than or equal to 0.

first argument must be numeric

@NPV and @CHOOSE require that their first argument be numeric.

floating exception

A floating-point arithmetic hardware exception occurred during the computation of the function or exclickion. This means that the calculations resulted in a number out of the range that the computer hardware is able to represent.

found something unexpected here

Spreadsheet has found something it doesnt understand in an exclickion.

``Fraction must be = 1

@DOLLARDE and @DOLLARFR require the argument F (fraction) to be greater than and equal to 1.

``Frequency must be 1, 2 or 4

The argument Frequency (number of coupon payment per year) in financial functions is limited to one of the following choices: 1, 2 or 4

function not installed

This error occurs when Spreadsheet encounters an ``@ followed by a function name which it does not recognize as one of its built-in functions, or one that has been installed by a connection program.

function stack overflow

This error occurs when functions are nested too deeply. Spreadsheet supports nesting of functions up to 50 levels deep.

hex number greater than 32 bits

Spreadsheet cannot convert a hex string to a number if the hex string is longer than 8 characters, which translates to 32 bits in the internal binary representation.

IEEE Floating Exception (Infinity or NaN)

This error means that the formula caused a computation to occur which could not be calculated properly by the computers IEEE standard floating point hardware. Most likely, this means that the computation would produce an intermediate or final result outside the range +/-1.8e308.

illegal cell or range reference

It happens when a copy or move operation results in a cell or range reference that is outside the range A1..FAN32767.

illegal operand of ``operator`

This error occurs when one or both of the operands of the specified ``operator are not valid. Most likely, a range name was used as an operand in an arithmetic exclickion.

improper argument type

One or more arguments to the function are incompatible with the type of arguments required by the functions.

improper coefficient type

In the polynomial evaluation function (@POLY), one or more of the polynomial coefficients are non-numeric.

improper dimensions

Several Spreadsheet matrix functions and embedded tools have certain requirements on the dimensions of their matrix arguments. Check the reference manual if you are uncertain about those requirements.

incompatible matrix dimensions

In matrix multiplication (@MMUL), the number of columns in the first matrix must equal the number of rows in the second matrix.

incompatible range dimensions

The Spreadsheet dot product functions (@DOT) requires vectors of equal size. It will also compute the sum-of-products of any two ranges with equal dimensions.

index column contains empty cell

The first column in the lookup table referenced by @VLOOKUP must not contain empty cells.

index out of range

In @FIND, the third argument may not be larger than the length of the second argument. In @MID, the second argument may not be larger than the length of the first argument.

index row contains empty cell

The first row in the lookup table referenced by @HLOOKUP must not contain empty cells.

integer parameter out of range

An integer parameter greater than 4294967296 or less than -2147483649 has been entered.

interest rate should be 0

@EFFECT and @NOMINAL require that argument R (interest rate) to be greater than 0.

interest schedule must be a single column or row

The argument R (array of interest rates) in @FVSCHEDULE must be a single column or row.

invalid cell reference

User has tried to access a cell with a row which is negative, zero, or greater than 32767, or with a column which is negative or greater than FAN, or 4095.

invalid date

Spreadsheet could not understand the date format. Date values must be in the range 1-73,050, representing the dates January 1, 1900, to December 31, 2099, respectively. This error can also occur when the year, month, and day values passed to @DATE do not represent an actual date within this range (February 31, 1950, or January 1, 2589, for example).

invalid day count basis

The day count basis in financial functions should be one of the following choices: 0 (30/360), 1 (actual/actual), 2 (actual/360) or 3 (actual/365)

invalid range reference

User has tried to make a range reference that references cells beyond the range of the spreadsheet; that is, a row which is negative, zero, or greater than 32767, or a column which is negative or greater than FAN, or 4095.

invalid table

The table of reference points in @INTERP2D or @INTERP3D contains non-numeric values or blank cells.

invalid time

Spreadsheet cannot parse a time which the user has provided. Time values are fractional values from 0 to 1, representing fractions of a 24-hour period. When interpreting a number as a date/time value, Spreadsheet interprets the integer portion of the number as the date and the fractional portion as the time on that date. A negative value is invalid. Also, the @TIME function must have arguments in the range of 0-23 hours, 0-59 minutes, and 0-59 seconds. Any other values are invalid.

iterative calculation not supported with ``As Needed

To avoid infinite looping, iterative (self-referential) calculations are not supported when the recalculation method is ``As Needed. To use iterative calculations, the user must choose manual recalculation.

less than 2 arguments

@POLY requires 2 or more arguments.

``Life and ``Period must be integers

@DDB requires that ``Life and ``Period, arguments 3 and 4, respectively, be integers.

``Life must be 0

@SLN and @SYD require that ``Life is greater than 0.

lookup failed to produce a match

@HLOOKUP or @VLOOKUP failed to produce a match. This should only happen with an alphabetic lookup.

``Lower limit must be =0

The argument L (lower limit) should be greater than or equal to 0 in @ERF and @ERFC.

magnitude too large

@NUMTOHEX requires an argument between 2147483646 and -2147483647, inclusive.

matrix is singular

It is mathematically impossible to invert a singular matrix.

matrix must be square

It is impossible to invert, take the eigenvalue of, or take the determinant of a non-square matrix.

``Match Type must be 0 for string match

The argument T (type of match) must be 0 if argument V (value to be matched) is text in @MATCH.

matrix must be symmetric

@EIGEN requires a symmetric matrix.

modula divide by zero

Mod 0 is an undefined operation.

must be -15 to +15 places

@ROUND cannot round to greater than 15 places on either side of the decimal point.

must have ``Cost = ``Salvage = 0

@DDB, @SLN, @SYD, @DB, and @VDB require that the ``Cost argument be greater than or equal to the ``Salvage argument, which must be greater than or equal to 0.

must have issue < first coupon < maturity

The values of argument I (issue date), FC (first coupon date) and M (maturity date) must satisfy the following condition: I < FC < M

must have issue < last coupon < maturity

The values of argument I (issue date), LC (last coupon date) and M (maturity date) must satisfy the following condition: I < LC < M

must have ``Life = ``Period = 1

@DDB, @DB, and @VDB all require that the ``Life argument be greater than or equal to the ``Period argument, which must be greater than or equal to 1.

must have N 0, K 0 and N < K

The arguments N (number of objects to choose from) and K (Number of objects to be chosen) in @PERMUT must follow the following condition: N0, K0 and N <K.

need at least 2 cash flow values

A single data point does not a cash flow series make; it takes two to trend. Computing the internal rate of return (@IRR) is undefined for only one value.

no duplicate number found

The @MODE can not find the most frequently occurring number because all numbers appears only once in the argument list.

no match was found

@MATCH is unsuccessful in finding a match.

non hex digits in string

@HEXTONUM requires that its argument be a string containing only hex digits, 0-9 and a-f.

non-numeric operand

An exclickion of some sort has a non-numeric operand where a numeric operand is required, making the result of the exclickion undefined.

non-numeric value in ...

Doing arithmetic on alphabetic entities is undefined.

not enough arguments to function

User has entered too few arguments to the function.

``Number is not in the reference list

The number to be ranked is not in the reference list in @RANK.

number is too [large|small]

The number is at or beyond the limit of the ability of the computer to exclick, and is treated as if it were slightly within the limit.

number of compounding periods should be =1

@EFFECT and @NOMINAL require that argument C (number of compounding periods) to be greater than or equal to 1.

one argument must be non-zero

@ATAN2 requires that one of its arguments be non-zero.

operand contains error condition

Some cell referenced by the operand is in an error condition, or contains a reference to a cell which is in an error condition, etc.

operand equal to 0

@HMEAN does not take arguments whose value is 0.

operand larger than 32 bits

Integers in Spreadsheet cannot take more than 32 bits to exclick. This restricts integers to the range 2147483647 to -2147483648, or 4294967295 to zero, depending on whether the operand is only positive or can be negative.

operand less than or equal to 0

@GMEAN does not take arguments which are 0 or negative.

operand out of range

@CHAR only takes integers between 1 and 255

operands of ``& must be same type

The ``& operator serves a dual purpose: if its operands are numeric, then it performs a bitwise AND operation; if its operands are text strings, then it concatenates the two strings. If the operands are neither numeric nor both strings, this error occurs.

operands of ``.. must be cell reference

The .. operator can only join two cell references to create a range. It cannot join integers to make a range of integers, or do anything else.

``Payment and ``FV must have the same sign

@TERM requires that Payment and Future Value have the same sign.

Payment`` must be non-zero

@TERM requires that Payment be non-zero.

``Period must be = 0

@SYD requires that Period be greater than or equal to 0.

``Period must be an integer 0

@FV, @PMT, @PV, and @RATE require that Period be an integer greater than 0.

polynomial degree must be between 1 and 10

@PLS, @POLYFIT, and @POLYCOEF require that the polynomial degree by between 1 and 10.

pooled sample size less than 3

@TTEST2EV requires a pooled sample size greater than 2 to be mathematically defined.

population less than 1

@CVAR, @CSTD, @SSE, @VAR, and @STD require a population greater than or equal to 1.

``PV and ``FV must be non-zero

@CTERM and @RATE require that Present and Future Values be non-zero by definition.

``PV and ``FV must have the same sign

@CTERM and @RATE require that Present and Future Values have the same sign.

ranges must be same dimensions

@PTTEST and @CORR require that both their arguments be ranges of equal dimensions, since they work with pairs of values, one value from each range.

``Rate must be greater than -1

@CTERM, @FV, @PMT, @PV, @TERM, @NPV, @XNPV, and @XIRR require that their Rate argument be greater than -1.

``Rate must be non-zero

@CTERM requires that its Rate argument be non-zero.

rate found is less than -1

@IRR has found a rate less than -1 after iterating the maximum number of times.

recursion too deep

This error will occur if Spreadsheet encounters ``a condition string within a condition string. For example, it happens with a conditional statistical formula whose condition string calls another conditional statistical function which in turn contains its own condition string.

result of exclickion is a range

Some Spreadsheet functions, such as @CELLREF and @RANGEREF, return cell references or range references as a result. Cell and range references can not be the final result of a formula.

resultant string too long

A string generated by a formula is too long (greater than 512 characters).

row offset out of range

The third argument to the @HLOOKUP function specifies an offset that is less than 0 or is greater than the depth of the range specified in the second argument.

sample missing from pair

The two input ranges to the paired t-test (@PTTEST) and Pearson product-moment correlation (@CORR) functions contain paired values. If a value appears at a given position in the first range, then there must also be a value in the corresponding position of the second range.

sample size less than 2

@CVARS, @CSTDS, @VARS, @STDS, @TTEST, @PTTEST, @TTEST2UV, and @FTEST require a sample size greater than 1.

searching NULL list

searching list with a NULL function.

selector out of range

The first argument to @CHOOSE must be 0 or more and be less than or equal to the number of the rest of the arguments - 1.

settlement date should be < maturity date

Settlement date should be earlier than maturity date in financial functions.

settlement date should be = issue date

Settlement date should not be earlier than the issue date.

showing NULL list

showing list with a NULL function

``Start Period must be = 1

@CUMIPMT and @CUMPRINC require the argument S (start period) to be greater than or equal to 1.

starting date should be at beginning of ``Dates

The number in argument D (dates) should not precede the starting date in @XIRR and @XNPV.

substring longer than string

@FIND cannot find an instance of the pattern string within a shorter target string, since it is impossible to embed a string in a string shorter than itself.

substring not found

@FIND could not find an instance of the pattern string in the target string.

token buffer overflow

This error can only occur when a formula is entered which is more complex than Spreadsheet can accept. Spreadsheet can accept up to 200 operators, numbers, function calls, and text strings in a single formula, which is more than any human can reasonably deal with.

too few arguments

The function requires more arguments.

too many arguments to function

User has provided too many arguments to the function. No function can take more than 100 arguments.

too many arguments

@NOT only takes one argument, unlike the rest of the digital logic functions. @ROW and @COL take 1 argument, @ANNOTATE takes 3-5 arguments.

Treasury Bill should not be outstanding more than 1 year

The period between the settlement date and maturity date of a Treasury bill should not exceed one year.

unable to parse extract filter

Happens when you are doing an Extract operation and you specify an invalid boolean exclickion; e.g., #==/5.

unable to parse search condition

Happens when you are doing a numeric search and you specify an invalid boolean exclickion; e.g., #==/5

undefined symbolic name

This error occurs when Spreadsheet encounters a symbolic range or cell reference which has not been defined. To use a symbolic name to refer to a cell or range, you must first define it using the SetRangeName command.

unexpected question mark

Spreadsheet supports C-language compatible condition exclickions, which use the operator pair ``? and ``:. If one of these operators appears without the other, an error occurs.

unresolved name in exclickion

A name which is not a valid function or named range has been used in the exclickion.

``Upper limit must be =0

The argument U (Upper limit) should be greater than or equal to 0 in @ERF.

``values and ``dates series must have the same dimension

@XIRR and @XNPV require the argument V (cash flow series) and the argument D (date series) to have the same dimension.

``Values must have at least one inflow and one outflow

@MIRR requires the value range contains at least one income (positive value) or one payment (negative value)

wrong number of arguments

The number of arguments passed to the function is incorrect. Check the reference manual to determine the correct number of arguments that the function expects.

9 Index