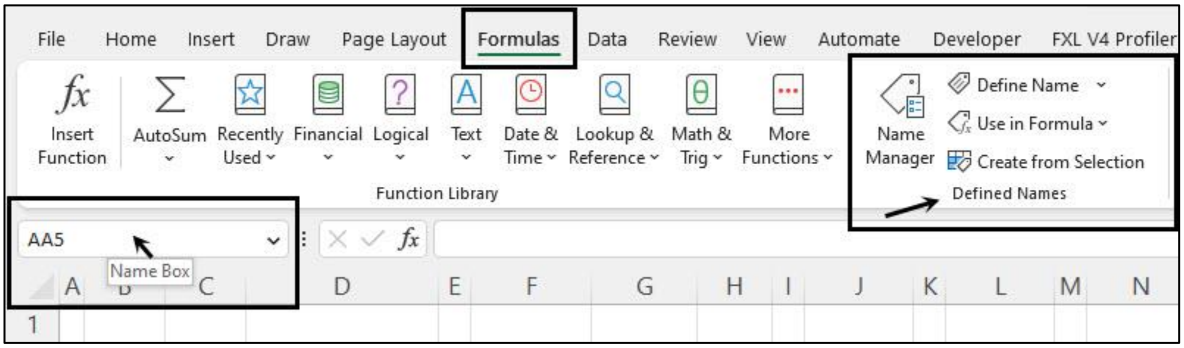
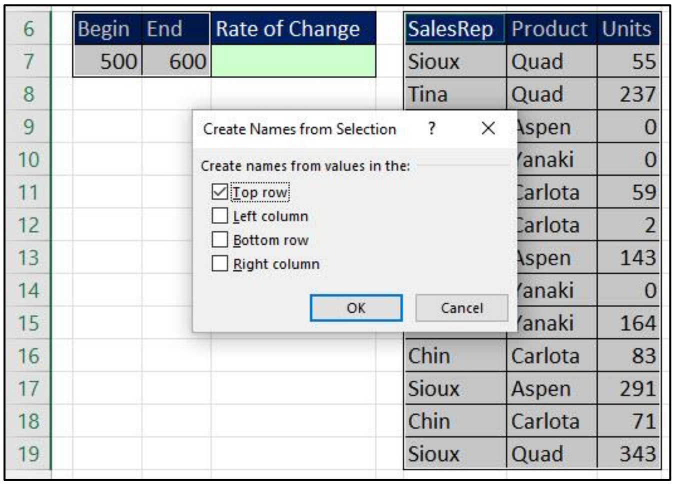
M 365 Excel Class Video 10: LET & LAMBDA Functions and Single Cell Reports

Defined Names

- Picture of Defined Names group in Formula tab in Excel Ribbon:



- Defined Names allow you to assign a name to:
    1. A cell
    2. A range of cells
    3. Non-contiguous ranges of cells
    4. Worksheet Formulas
    5. LAMBDA functions and then use the Defined Names in Formulas.
    6. Excel Table Names, Print Areas and Criteria and Extract Ranges for Advanced Filter for automatically created and listed with all Defined Names.
- Create Defined Names:
    1. You can create a Defined Name for a ***Cell, Range, or Non-Contiguous Ranges of Cells*** by:
        - Selecting cell, range or ranges
        - Type Name in Name Box (shown in picture above)
            - Rules for Defined Names:
                - Must begin with text, not number
                - No spaces
                - No cell references
                - Max of 255 characters in name
                - Cannot use the characters: */+-()^<>+&%~`|][}{@”;:,’$#!
        - Press Enter.
    2. You can create a Defined Name for a ***Cell or Range*** when the name you want to use is above the cell or range, use the keyboard Ctrl + Shift + F3 to open the Create Names From Selection Dialog Box, as shown here:



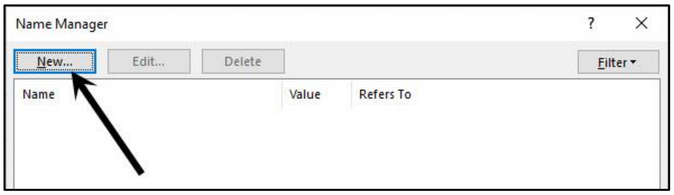
	Begin	End	Rate of Change	SalesRep	Product	Units
7	500	600		Sioux	Quad	55
8				Tina	Quad	237
9					Aspen	0
10					Yanaki	0
11					Carlota	59
12					Carlota	2
13					Aspen	143
14					Yanaki	0
15					Yanaki	164
16				Chin	Carlota	83
17				Sioux	Aspen	291
18				Chin	Carlota	71
19				Sioux	Quad	343

Create Names from Selection

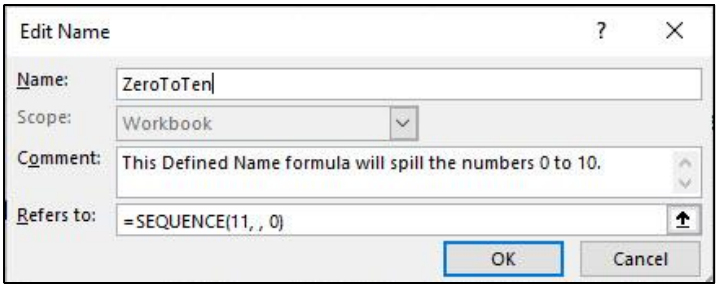
Create names from values in the:

☑ Top row
☐ Left column
☐ Bottom row
☐ Right column

3. You can create a Defined Name for a ***Formula or LAMBDA Function*** using the New Name Dialog Box by:
    - Open Name Manager Dialog Box:
        - Formulas tab in Excel Ribbon, Defined Name group, Name Manager button (shown in above figure).
          or
        - Use the keyboard to open the Name Manager = **Ctrl + F3**
        - Name Manager Dialog Box looks like this:



    - To create a new Defined Name, click the New… button. This opens the New Name Dialog Box as shown below
        - This Defined Name **defines a worksheet formula** that can be called with an equal sign and the name “ZeroToTen”:



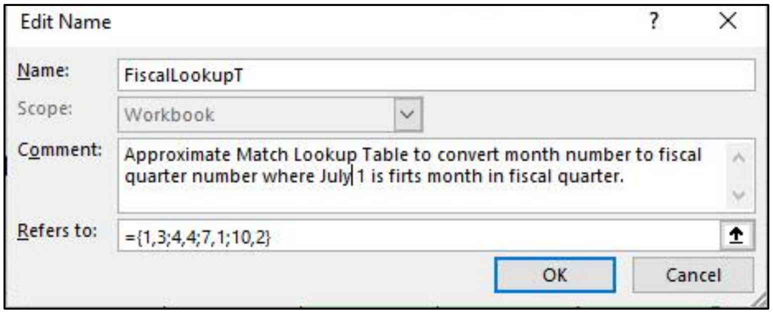
Name: ZeroToTen

Scope: Workbook

Comment: This Defined Name formula will spill the numbers 0 to 10.

Refers to: =SEQUENCE(11, , 0)

        - This Defined Name **defines a lookup table** based on an Array Constant:



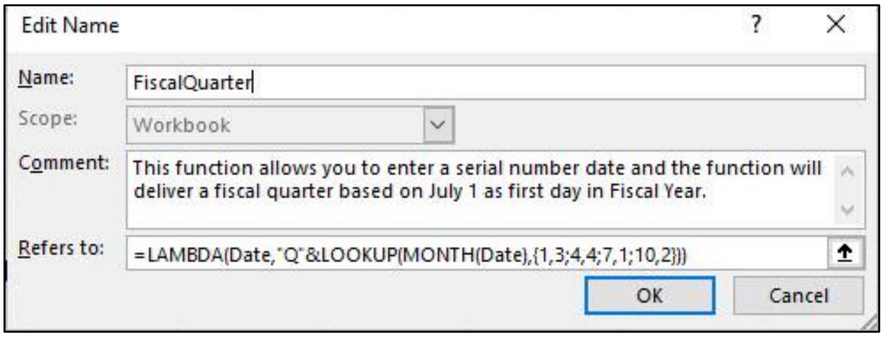
Name: FiscalLookupT

Scope: Workbook

Comment: Approximate Match Lookup Table to convert month number to fiscal quarter number where July 1 is firts month in fiscal quarter.

Refers to: ={1,3;4,4;7,1;10,2}

        - This Defined Name **defines a LAMBDA re-useable custom function** that will create a new function available in the workbook that will calculate fiscal quarter from a serial number date:



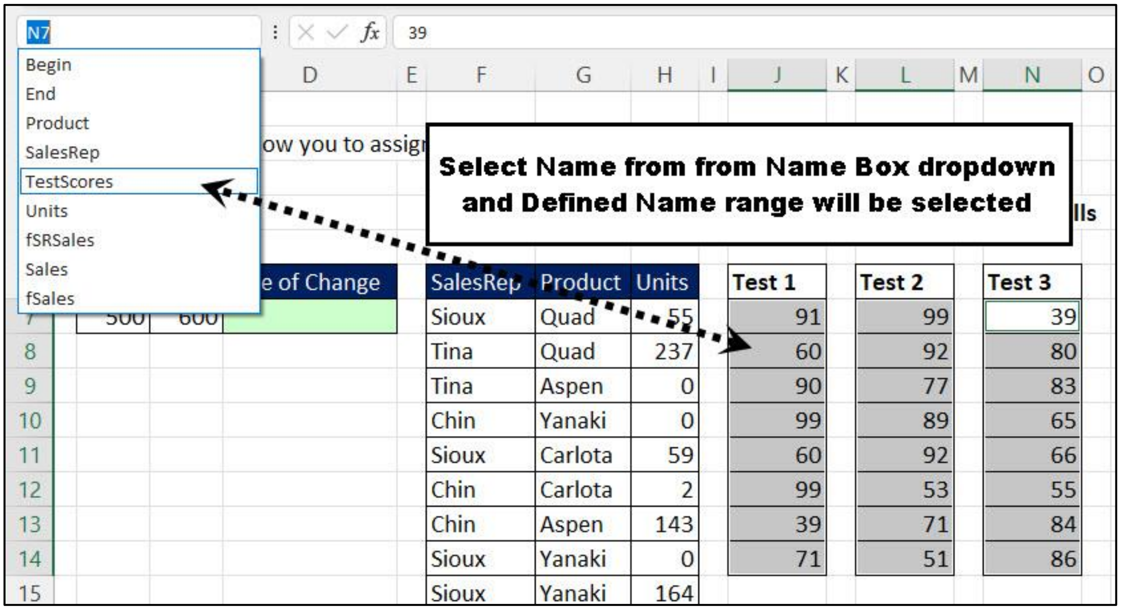
Name: FiscalQuarter

Scope: Workbook

Comment: This function allows you to enter a serial number date and the function will deliver a fiscal quarter based on July 1 as first day in Fiscal Year.

Refers to: =LAMBDA(Date,"Q"&LOOKUP(MONTH(Date),{1,3;4,4;7,1;10,2}))

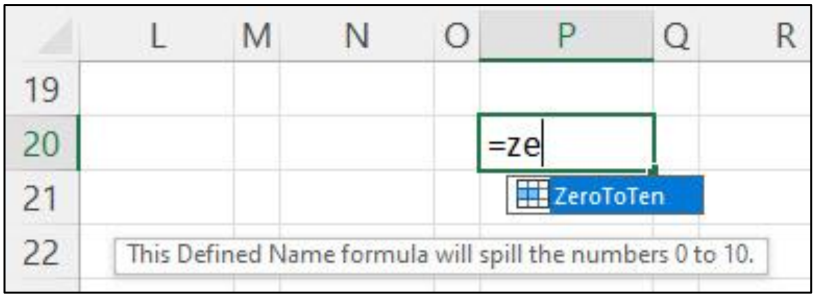
- Use Defined Names:
    1. To Jump to a Defined Name location in workbook, use Name Box dropdown to select name, as shown here:



Select Name from from Name Box dropdown and Defined Name range will be selected

	SalesRep	Product	Units	Test 1	Test 2	Test 3
7	Sioux	Quad	55	91	99	39
8	Tina	Quad	237	60	92	80
9	Tina	Aspen	0	90	77	83
10	Chin	Yanaki	0	99	89	65
11	Sioux	Carlota	59	60	92	66
12	Chin	Carlota	2	99	53	55
13	Chin	Aspen	143	39	71	84
14	Sioux	Yanaki	0	71	51	86
15	Sioux	Yanaki	164			

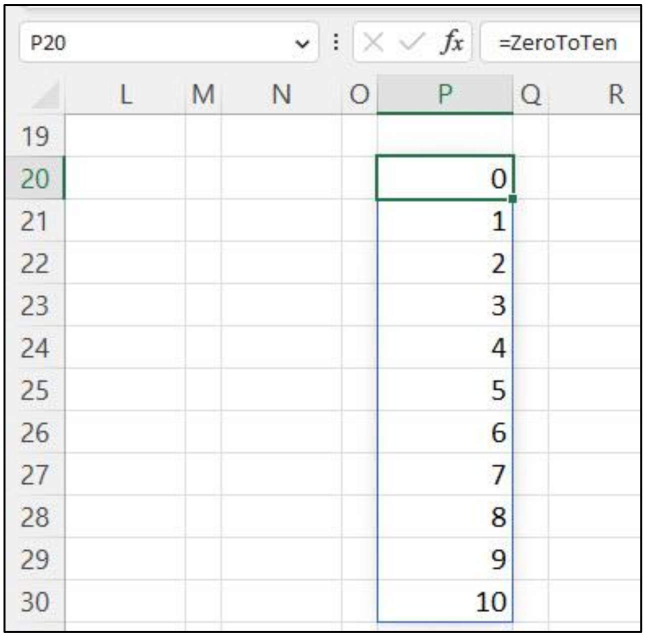
2. To use a Defined Name that is a worksheet formula:
    i. Type an equal sign and then the first few letters in name, as shown here:



=ze

This Defined Name formula will spill the numbers 0 to 10.

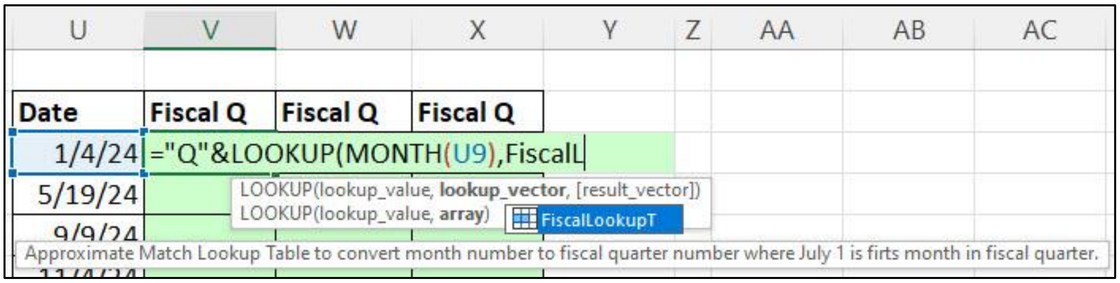
    ii. After entering Defined Name worksheet formula, this is what you see:



=ZeroToTen

	P
19	
20	0
21	1
22	2
23	3
24	4
25	5
26	6
27	7
28	8
29	9
30	10

3. To use a Defined Name that is a lookup table:
   i. In lookup formula, type first few letters in name, as shown here:



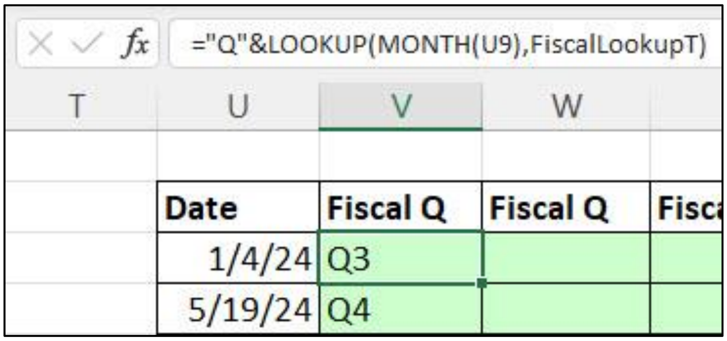
Date	Fiscal Q	Fiscal Q	Fiscal Q
1/4/24	="Q"&LOOKUP(MONTH(U9),FiscalL		
5/19/24			
9/9/24			

LOOKUP(lookup_value, **lookup_vector**, [result_vector])
LOOKUP(lookup_value, **array**)

FiscalLookupT

Approximate Match Lookup Table to convert month number to fiscal quarter number where July 1 is firts month in fiscal quarter.

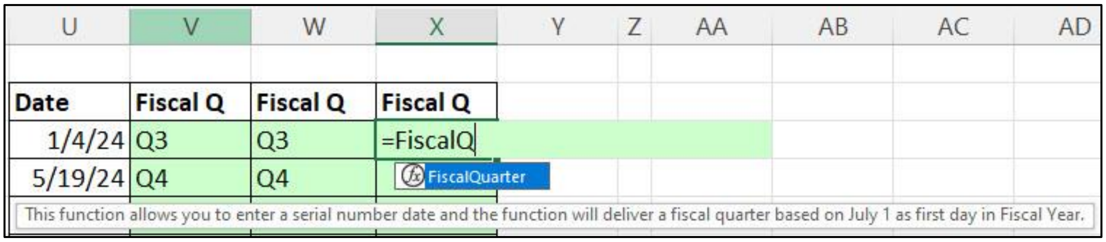
   ii. After entering Defined Name lookup Table, this is what you will see:



```
="Q"&LOOKUP(MONTH(U9),FiscalLookupT)
```

Date	Fiscal Q	Fiscal Q	Fisc
1/4/24	Q3		
5/19/24	Q4		

4. To use a Defined Name that defines a LAMBDA function:
   i. Type an equal sign and then the first few letters in LAMBDA defined function name, as shown here:

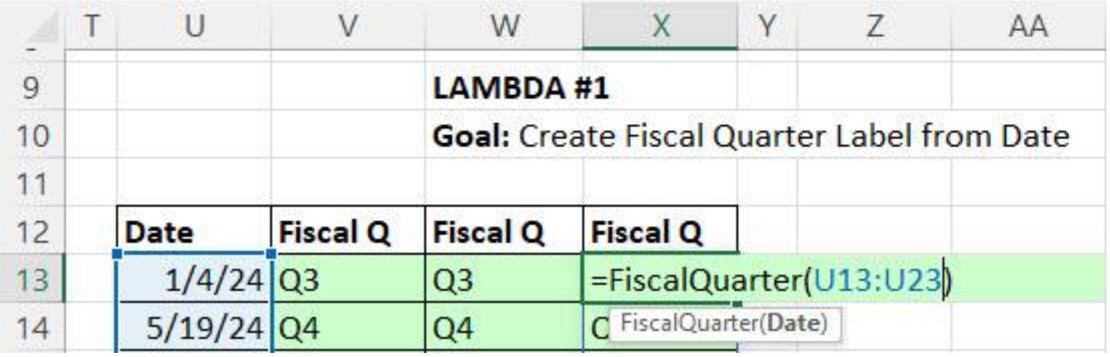


Date	Fiscal Q	Fiscal Q	Fiscal Q
1/4/24	Q3	Q3	=FiscalQ
5/19/24	Q4	Q4	

FiscalQuarter

This function allows you to enter a serial number date and the function will deliver a fiscal quarter based on July 1 as first day in Fiscal Year.

   ii. After invoking the LAMBDA defined function and entering the serial number date (as a single cell or as a range):



LAMBDA #1

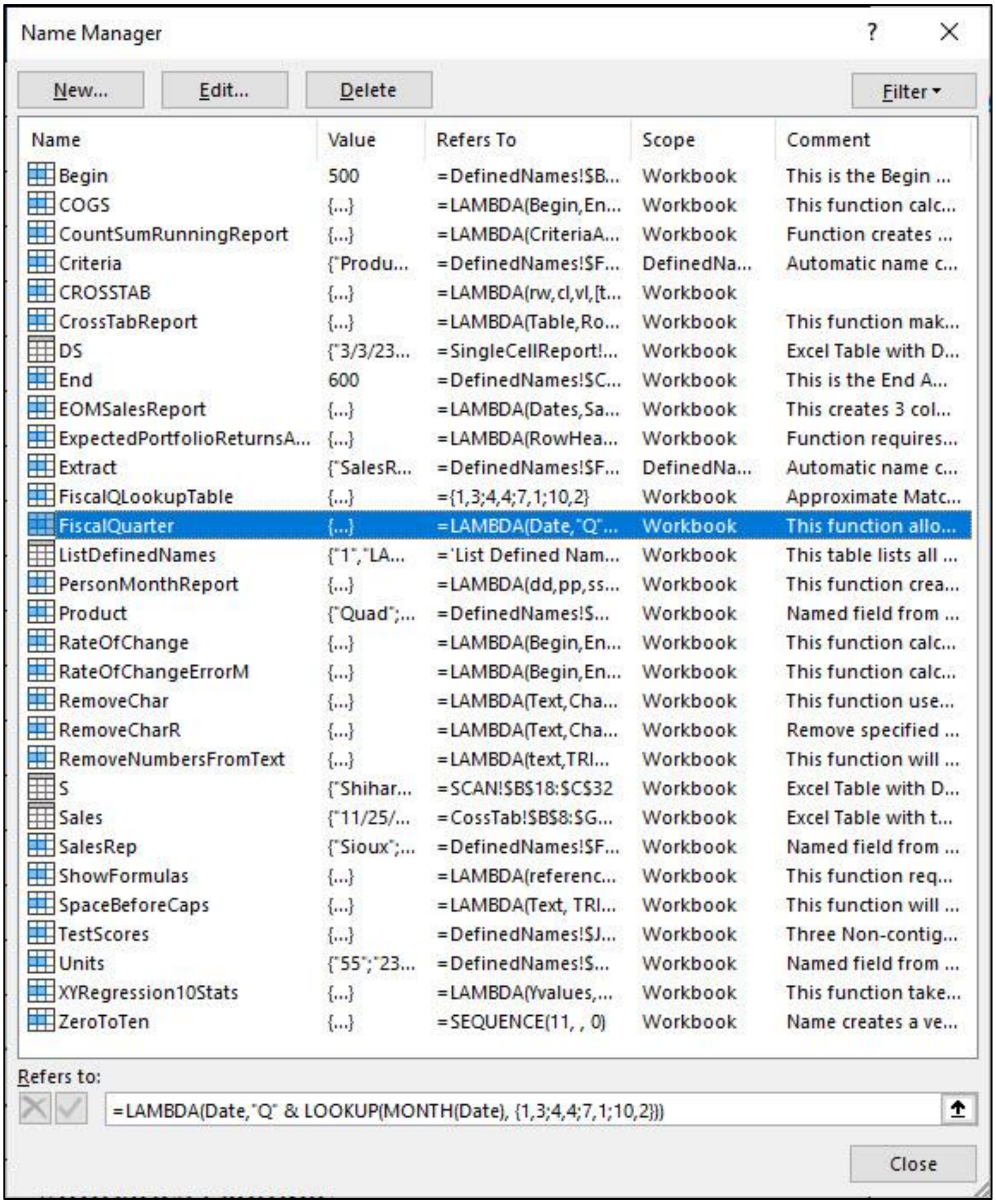
Goal: Create Fiscal Quarter Label from Date

Date	Fiscal Q	Fiscal Q	Fiscal Q
1/4/24	Q3	Q3	=FiscalQuarter(U13:U23)
5/19/24	Q4	Q4	C

FiscalQuarter(**Date**)

Here is what the Name Manager (Ctrl + F3) looks like when you have many Defined Names (Selected Cells Icon) and Excel Tables (Table Icon):

1. **Name column** = name of Defined Name or Excel Table.
2. **Value column** = preview of value if available (formulas and functions show no values).
3. **Refers to column** = formula that defined a cell, range, noncontiguous ranges, worksheet formulas, LAMBDA defined functions, Excel Tables, memorized Print Areas and memorized Criteria and Extract ranges from Advanced Filter feature.
4. **Scope column** = tells you where you can use the Defined Name:
    - **Workbook** => means you can use it anywhere in workbook.
    - Worksheet Name, like "**Topic**" => means you can only use it on that specified worksheet.
    - **Defined Names** => is a label given to the automatic Defined Names that are created when you use the Advanced Filter Feature. The two names created are "Criteria" and "Extract".
5. **Comment** = Comment added to help users understand what the Defined Name does. When you add a comment to a LAMBDA defined custom function, it appears in the function screen tip when you are typing the function name in a formula.



Name	Value	Refers To	Scope	Comment
Begin	500	=DefinedNames!$B...	Workbook	This is the Begin ...
COGS	{...}	=LAMBDA(Begin,En...	Workbook	This function calc...
CountSumRunningReport	{...}	=LAMBDA(CriteriaA...	Workbook	Function creates ...
Criteria	{"Produ...	=DefinedNames!$F...	DefinedNa...	Automatic name c...
CROSSTAB	{...}	=LAMBDA(rw,cl,vl,[t...	Workbook	
CrossTabReport	{...}	=LAMBDA(Table,Ro...	Workbook	This function mak...
DS	{"3/3/23...	=SingleCellReport!...	Workbook	Excel Table with D...
End	600	=DefinedNames!$C...	Workbook	This is the End A...
EOMSalesReport	{...}	=LAMBDA(Dates,Sa...	Workbook	This creates 3 col...
ExpectedPortfolioReturnsA...	{...}	=LAMBDA(RowHea...	Workbook	Function requires...
Extract	{"SalesR...	=DefinedNames!$F...	DefinedNa...	Automatic name c...
FiscalQLookupTable	{...}	={1,3;4,4;7,1;10,2}	Workbook	Approximate Matc...
FiscalQuarter	{...}	=LAMBDA(Date,"Q"...	Workbook	This function allo...
ListDefinedNames	{"1","LA...	='List Defined Nam...	Workbook	This table lists all ...
PersonMonthReport	{...}	=LAMBDA(dd,pp,ss...	Workbook	This function crea...
Product	{"Quad";...	=DefinedNames!$...	Workbook	Named field from ...
RateOfChange	{...}	=LAMBDA(Begin,En...	Workbook	This function calc...
RateOfChangeErrorM	{...}	=LAMBDA(Begin,En...	Workbook	This function calc...
RemoveChar	{...}	=LAMBDA(Text,Cha...	Workbook	This function use...
RemoveCharR	{...}	=LAMBDA(Text,Cha...	Workbook	Remove specified ...
RemoveNumbersFromText	{...}	=LAMBDA(text,TRI...	Workbook	This function will ...
S	{"Shihar...	=SCAN!$B$18:$C$32	Workbook	Excel Table with D...
Sales	{"11/25/...	=CossTab!$B$8:$G...	Workbook	Excel Table with t...
SalesRep	{"Sioux";...	=DefinedNames!$F...	Workbook	Named field from ...
ShowFormulas	{...}	=LAMBDA(referenc...	Workbook	This function req...
SpaceBeforeCaps	{...}	=LAMBDA(Text, TRI...	Workbook	This function will ...
TestScores	{...}	=DefinedNames!$J...	Workbook	Three Non-contig...
Units	{"55";"23...	=DefinedNames!$...	Workbook	Named field from ...
XYRegression10Stats	{...}	=LAMBDA(Yvalues,...	Workbook	This function take...
ZeroToTen	{...}	=SEQUENCE(11, , 0)	Workbook	Name creates a ve...

Refers to:

=LAMBDA(Date,"Q" & LOOKUP(MONTH(Date), {1,3;4,4;7,1;10,2}))

- Defined Name Keyboards:
  1. Open the Name Manager = **Ctrl + F3**
  2. Create Names From Selection = **Ctrl + Shift + F3**
  3. Paste Name = **F3**
  4. List all Defined Names in Worksheet: **F3, then click "Paste Names"**.
- Defined Name Warning: Worksheets copied from one workbook to another bring all Defined Names from source workbook into destination workbook.

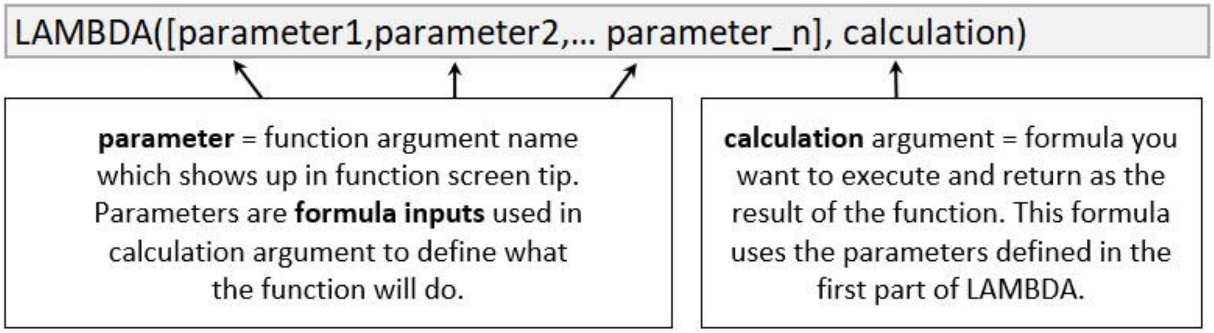
LAMBDA Function

- Define LAMBDA function
  - The LAMBDA function allows to create a custom function value, which:

    1] Can be stored in a Defined Name to create a reusable function

    or

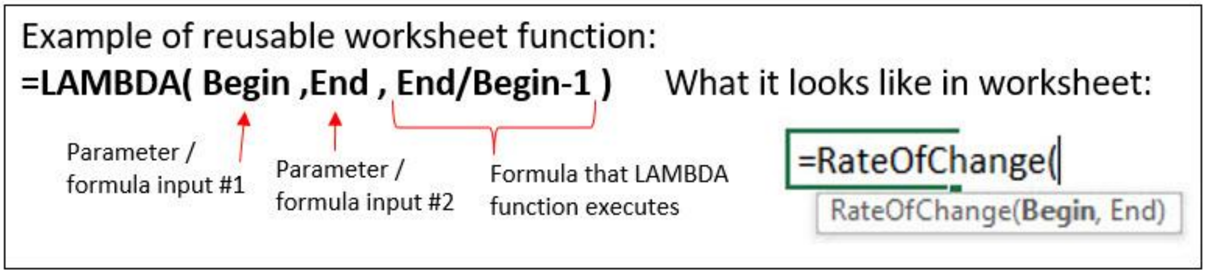
    2] Can be used in one of six LAMBDA Helper Functions for specific tasks such as spilling an aggregate calculation across rows. When you use LAMBDA in a helper function you can use the formula directly in the worksheet or you can store it in a Defined Name to create a reusable function.
  - The arguments for the LAMBDA function are shown here:



LAMBDA([parameter1,parameter2,... parameter_n], calculation)

parameter = function argument name which shows up in function screen tip. Parameters are **formula inputs** used in calculation argument to define what the function will do.

calculation argument = formula you want to execute and return as the result of the function. This formula uses the parameters defined in the first part of LAMBDA.



Example of reusable worksheet function:

=LAMBDA(Begin ,End , End/Begin-1)

What it looks like in worksheet:

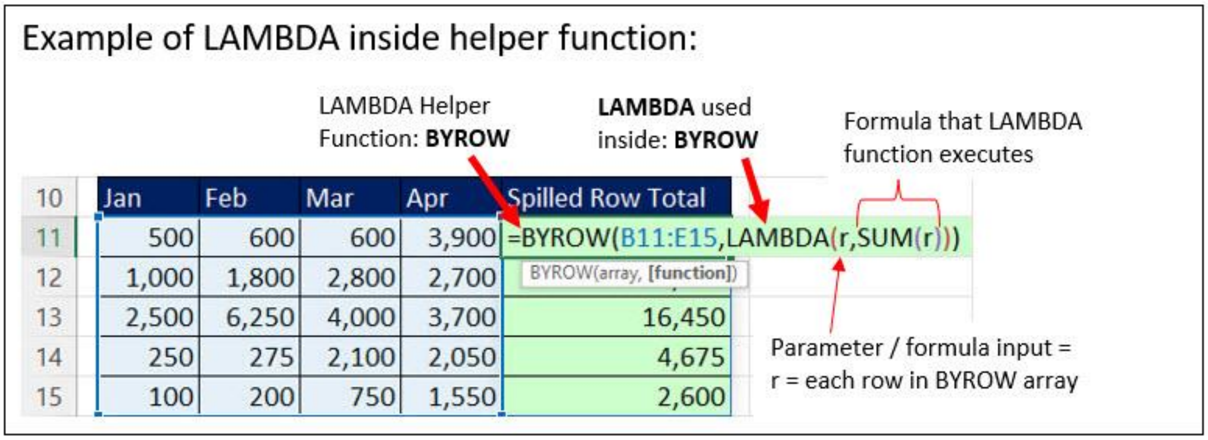
Parameter / formula input #1

Parameter / formula input #2

Formula that LAMBDA function executes

=RateOfChange(

RateOfChange(**Begin**, End)



Example of LAMBDA inside helper function:

LAMBDA Helper Function: **BYROW**

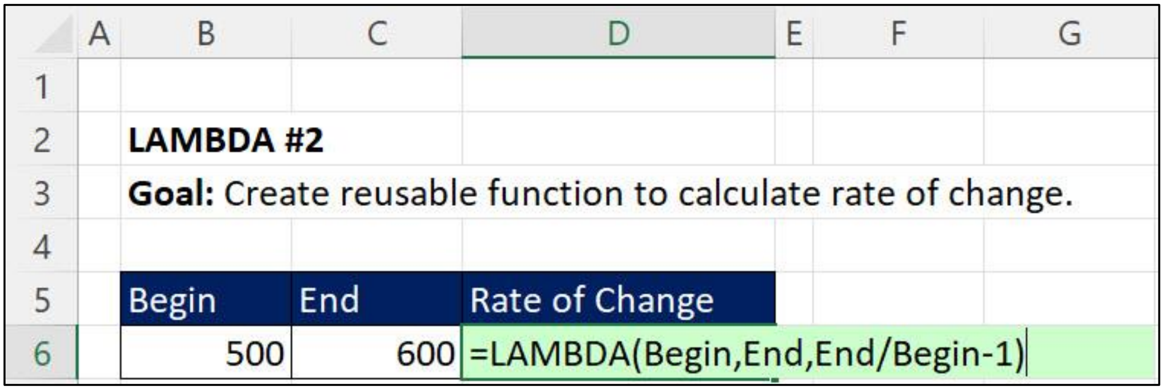
LAMBDA used inside: **BYROW**

Formula that LAMBDA function executes

10	Jan	Feb	Mar	Apr	Spilled Row Total
11	500	600	600	3,900	=BYROW(B11:E15,LAMBDA(r,SUM(r)))
12	1,000	1,800	2,800	2,700	BYROW(array, **[function]**)
13	2,500	6,250	4,000	3,700	16,450
14	250	275	2,100	2,050	4,675
15	100	200	750	1,550	2,600

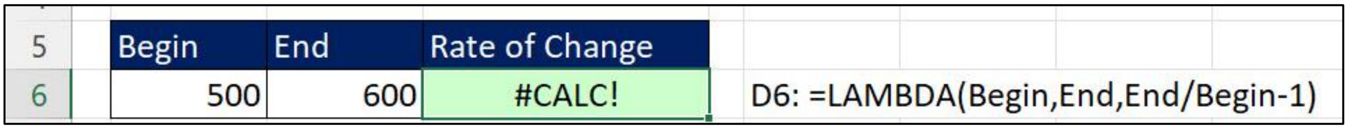
Parameter / formula input = r = each row in BYROW array

- Steps to create a Defined Name LAMBDA custom function for a Rate of Change calculation
  1. In the worksheet type your LAMBDA function, as shown below. Note, the parameters that you enter in the first few arguments of LAMBDA, become the argument labels in the function screen tip. This means that you should name the parameters in a way that can help the user to understand what to enter in the function arguments.



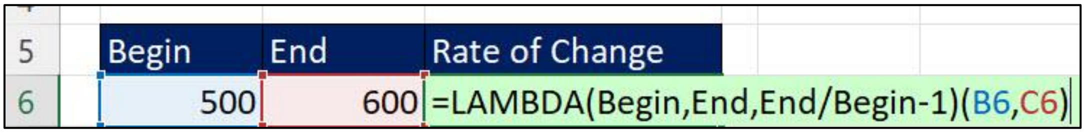
	A	B	C	D	E	F	G
1							
2		**LAMBDA #2**					
3		**Goal:** Create reusable function to calculate rate of change.					
4							
5		Begin	End	Rate of Change			
6		500	600	=LAMBDA(Begin,End,End/Begin-1)			

If you enter the LAMBDA into the worksheet without testing the formula with formula inputs, you will see this:

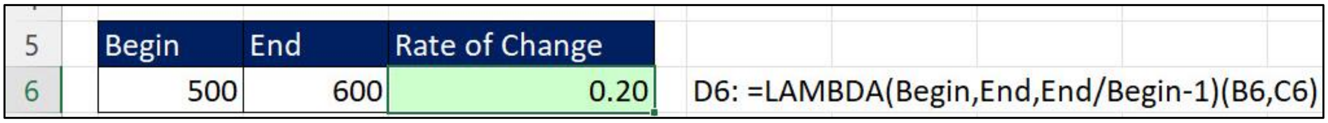


5		Begin	End	Rate of Change	
6		500	600	#CALC!	D6: =LAMBDA(Begin,End,End/Begin-1)

  2. To test the LAMBDA function in the worksheet, you type open parentheses, the formulas inputs in the same order as you entered the parameters in first part of the LAMBDA function, then close parentheses, as shown here:

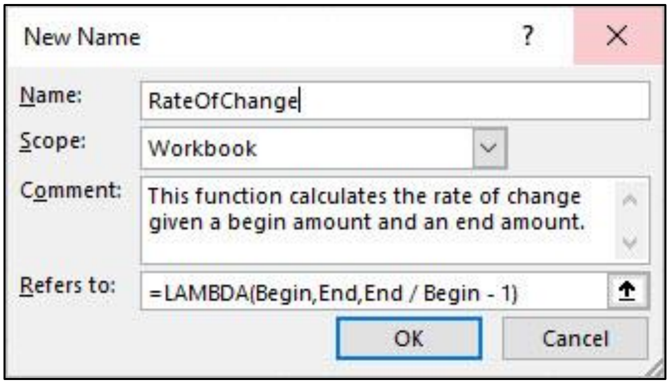


5		Begin	End	Rate of Change
6		500	600	=LAMBDA(Begin,End,End/Begin-1)(B6,C6)



5		Begin	End	Rate of Change	
6		500	600	0.20	D6: =LAMBDA(Begin,End,End/Begin-1)(B6,C6)

3. To create a Defined Name for the LAMBDA function (this allows you to re-use function anywhere in workbook), copy the LAMBDA function you created in the worksheet (without the testing parentheses at end) and then:
   i. Use Ctrl + F3 to open Name Manager
   ii. Click New Name button
   iii. In the New Name dialog box, enter:
      1) **Name:** The name of the function (will appear in function dropdown list when you create formulas)
      2) **Scope:** Where you want the function to appear: in Workbook (available everywhere in workbook) or a specified worksheet name.
      3) **Comment**: Create a description of the function (this appears in function screen tip when typing function name in a formula)
      4) **Refers to**: Paste the LAMBDA function you created in the worksheet (without the testing parentheses at end).



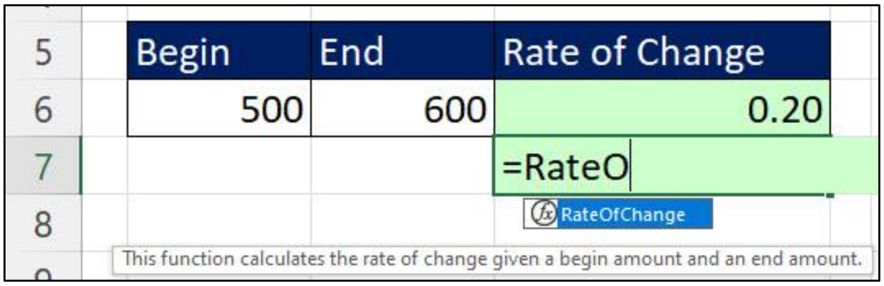
Name: RateOfChange

Scope: Workbook

Comment: This function calculates the rate of change given a begin amount and an end amount.

Refers to: =LAMBDA(Begin,End,End / Begin - 1)

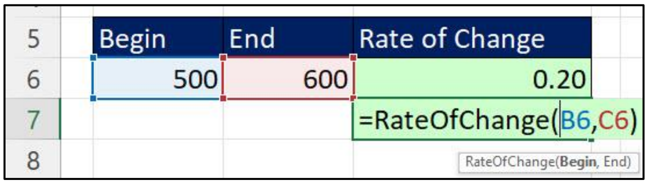
4. Test new function in worksheet by typing an equal sign and then the first few letters in function name (screen tip with description pops up), as shown here:



5	Begin	End	Rate of Change
6	500	600	0.20
7			=RateO
8			RateOfChange

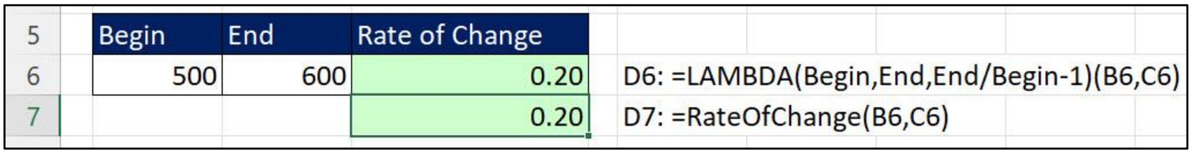
This function calculates the rate of change given a begin amount and an end amount.

5. Test new function with arguments, as shown here:



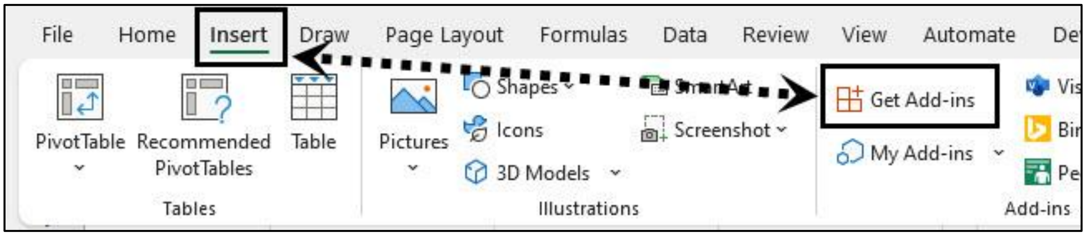
5	Begin	End	Rate of Change
6	500	600	0.20
7			=RateOfChange(B6,C6)
8			RateOfChange(**Begin**, End)

6. Test the output of the new Defined Name LAMBDA function, as shown here:

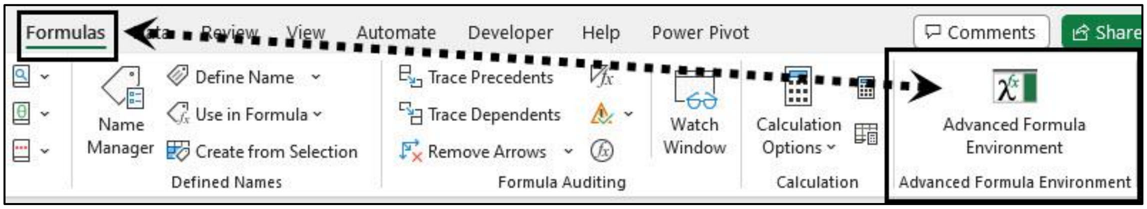


5	Begin	End	Rate of Change	
6	500	600	0.20	D6: =LAMBDA(Begin,End,End/Begin-1)(B6,C6)
7			0.20	D7: =RateOfChange(B6,C6)

- You can use the Advanced Formula Environment to create a Defined Name LAMBDA custom functions
    - The Advanced Formula Environment is an Excel Worksheet Add-in that is not installed by default in your Excel version. As shown below, you can install it with the Get-Add-ins button in the Add-ins group in the Insert tab in the Excel Ribbon. However, many work entities have the Add-ins feature disabled. You can contact your network administrator to have it installed. Otherwise, you can create your LAMBDA custom functions with using this task pane and use the Defined Name Dialog Box to create your LAMBDA re-usable custom function.

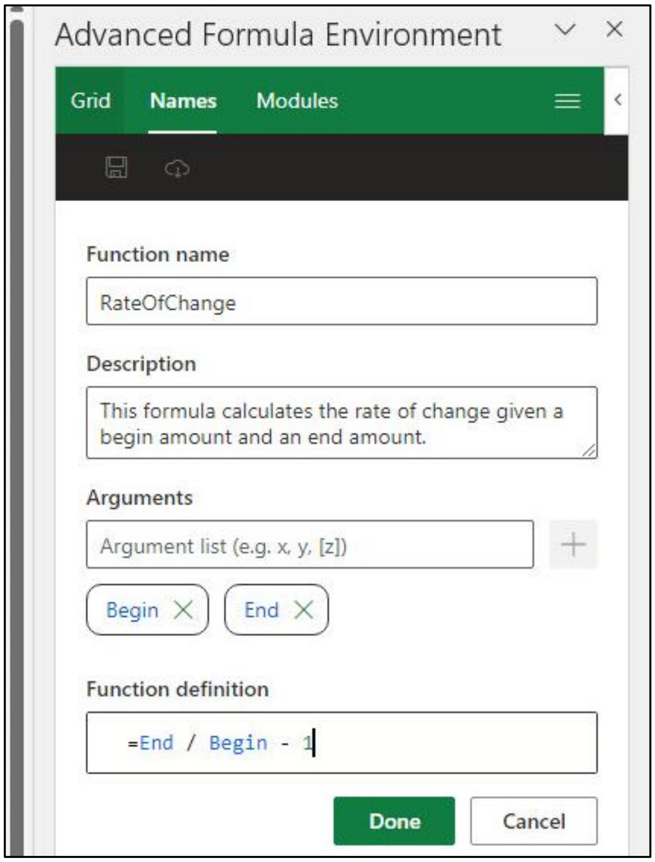


    - To open the Advanced Formula Environment, go to the Formula tab in the Excel Ribbon as shown below:



Using Advanced Formula Environment:

1. When you create a LAMBDA defined re-usable custom function, you can use the Advanced Formula Environment Task Pane as a user interface for the Defined Name Dialog Box. When you create a LAMBDA function in the Advanced Formula Environment Task Pane it is saved in the Defined Name Dialog Box. The reverse is also true: When you create a LAMBDA function in the Defined Name Dialog Box, it will appear in the Advanced Formula Environment Task Pane.
2. The advantage of using the Advanced Formula Environment Task Pane is that it is easier to type and create a LAMBDA function in the Advanced Formula Environment Task Pane than it is in the Defined Name Dialog Box.
3. The disadvantage of using the Advanced Formula Environment Task Pane is that you cannot directly access references from the worksheet, and you cannot directly test it in the worksheet.
4. An example of the Advanced Formula Environment Task Pane is shown to the right:



Function name

RateOfChange

Description

This formula calculates the rate of change given a begin amount and an end amount.

Arguments

Argument list (e.g. x, y, [z])

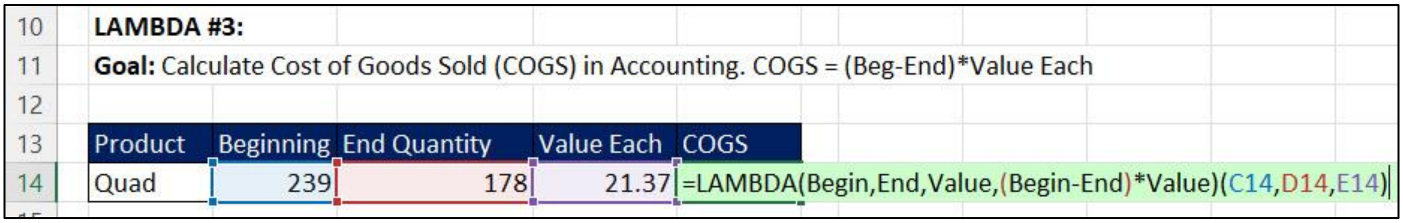
Begin ✕ End ✕

Function definition

```
=End / Begin - 1
```

- Steps to create a Defined Name LAMBDA custom function for a COGS calculation

  1. **Create and test in worksheet:**

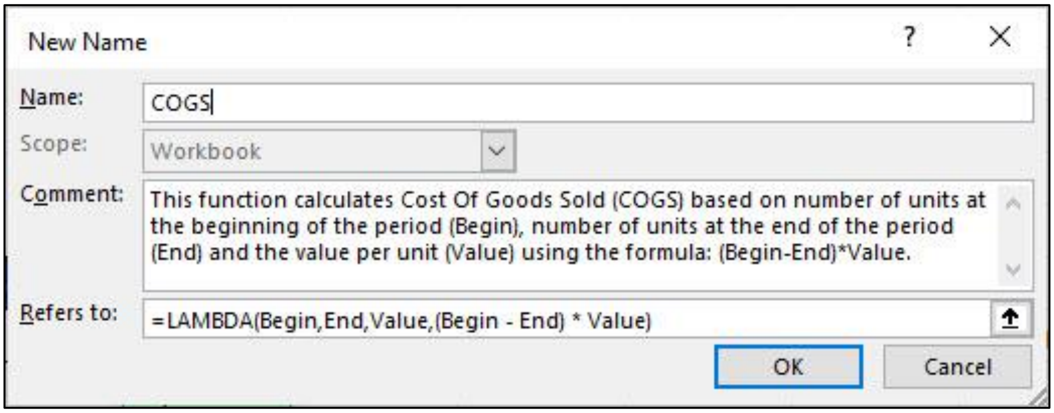


10 **LAMBDA #3:**

11 **Goal:** Calculate Cost of Goods Sold (COGS) in Accounting. COGS = (Beg-End)*Value Each

	Product	Beginning	End Quantity	Value Each	COGS
14	Quad	239	178	21.37	=LAMBDA(Begin,End,Value,(Begin-End)*Value)(C14,D14,E14)

  2. **Create Defined Name LAMBDA custom function:**



New Name

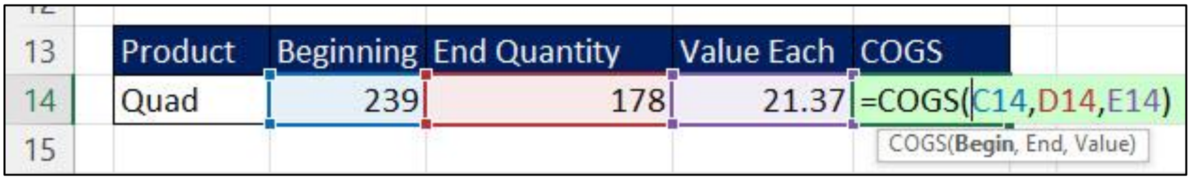
Name: COGS

Scope: Workbook

Comment: This function calculates Cost Of Goods Sold (COGS) based on number of units at the beginning of the period (Begin), number of units at the end of the period (End) and the value per unit (Value) using the formula: (Begin-End)*Value.

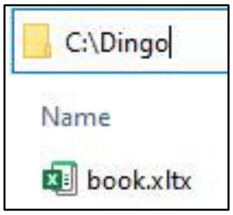
Refers to: =LAMBDA(Begin,End,Value,(Begin - End) * Value)

  3. **Use new COGS function in worksheet:**



	Product	Beginning	End Quantity	Value Each	COGS
14	Quad	239	178	21.37	=COGS(C14,D14,E14)
15					COGS(**Begin**, End, Value)

- Save LAMBDA functions to Excel file that opens with Ctrl + N:
  1. Create new Excel file with a single worksheet
  2. Add Defined Name LAMBDA functions to workbook
  3. Save file as Excel Template file with name book.xltx to C:\Dingo Folder:



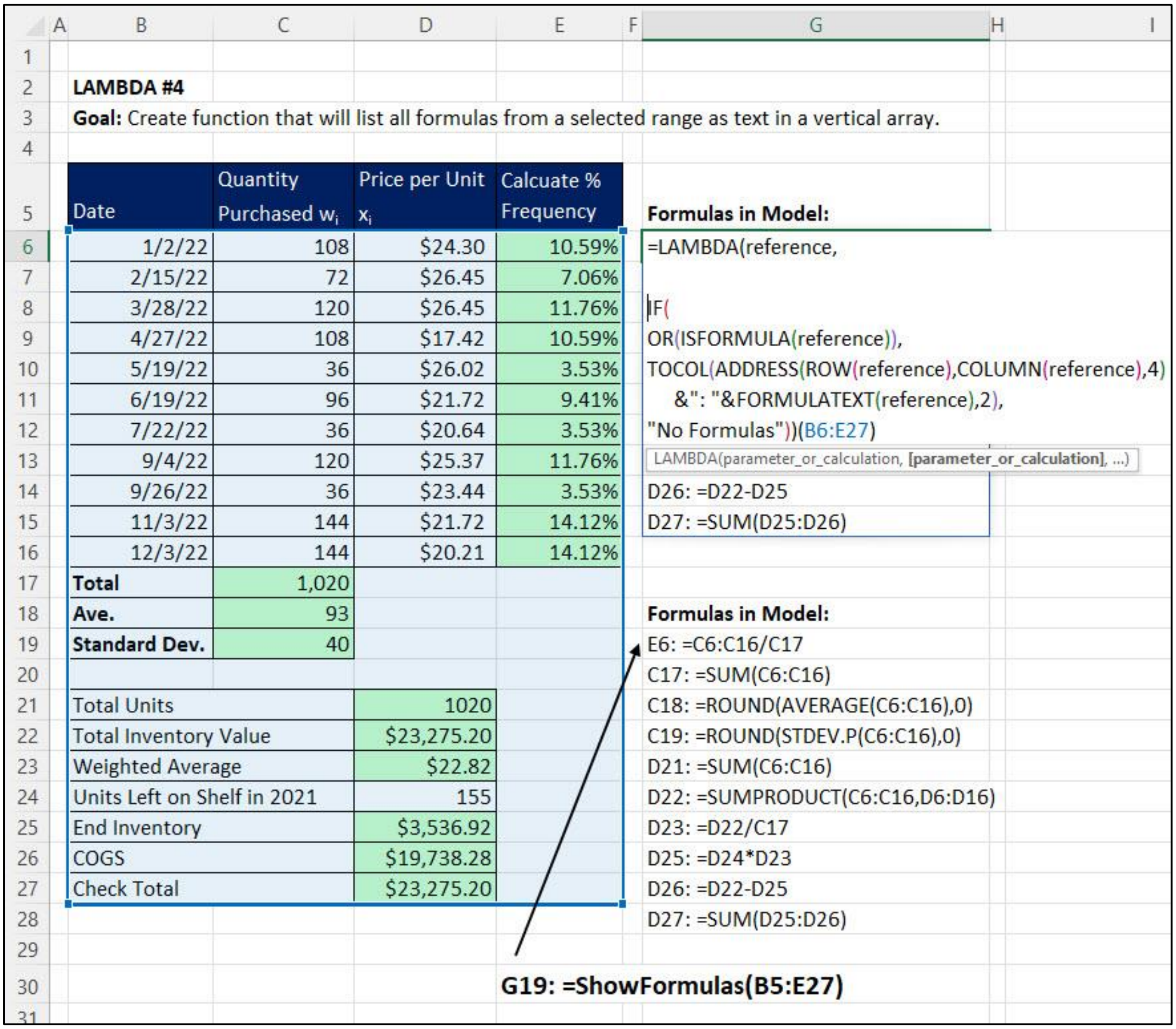
C:\Dingo

Name

book.xltx

  4. When you save this custom file:
     - Blank Workbook button DOES NOT open custom file.
     - Open Excel App and click Esc (not Blank Workbook button) DOES open custom file.
     - Ctrl + N opens custom file.
  5. Templates, to always open with Excel and also using the CTRL+N command, can be copied in the C:\Program Files\Microsoft Office\Root\Office16\XLSTART\ folder, with administrator permissions.

- Defined Name LAMBDA custom function for a show formula text in vertical column formula



LAMBDA #4

Goal: Create function that will list all formulas from a selected range as text in a vertical array.

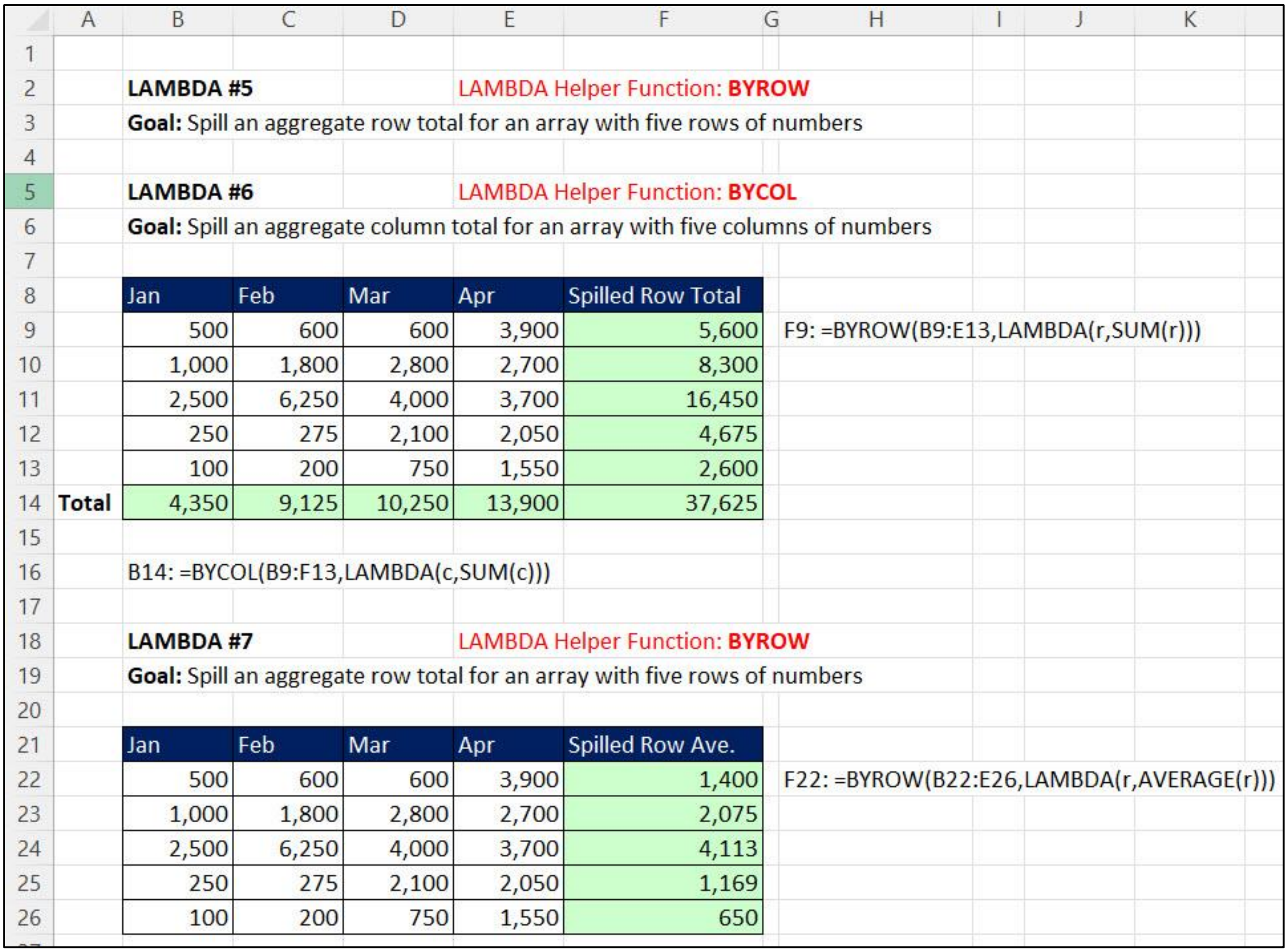
Date	Quantity Purchased w_i	Price per Unit x_i	Calcuate % Frequency	**Formulas in Model:**
1/2/22	108	$24.30	10.59%	=LAMBDA(reference,
2/15/22	72	$26.45	7.06%	
3/28/22	120	$26.45	11.76%	IF(
4/27/22	108	$17.42	10.59%	OR(ISFORMULA(reference)),
5/19/22	36	$26.02	3.53%	TOCOL(ADDRESS(ROW(reference),COLUMN(reference),4)
6/19/22	96	$21.72	9.41%	&": "&FORMULATEXT(reference),2),
7/22/22	36	$20.64	3.53%	"No Formulas"))(B6:E27)
9/4/22	120	$25.37	11.76%	LAMBDA(parameter_or_calculation, **[parameter_or_calculation]**, ...)
9/26/22	36	$23.44	3.53%	D26: =D22-D25
11/3/22	144	$21.72	14.12%	D27: =SUM(D25:D26)
12/3/22	144	$20.21	14.12%	
Total	1,020			
Ave.	93			**Formulas in Model:**
Standard Dev.	40			E6: =C6:C16/C17
				C17: =SUM(C6:C16)
Total Units		1020		C18: =ROUND(AVERAGE(C6:C16),0)
Total Inventory Value		$23,275.20		C19: =ROUND(STDEV.P(C6:C16),0)
Weighted Average		$22.82		D21: =SUM(C6:C16)
Units Left on Shelf in 2021		155		D22: =SUMPRODUCT(C6:C16,D6:D16)
End Inventory		$3,536.92		D23: =D22/C17
COGS		$19,738.28		D25: =D24*D23
Check Total		$23,275.20		D26: =D22-D25
				D27: =SUM(D25:D26)

G19: =ShowFormulas(B5:E27)

- Why Do We Need LAMBDA Helper Functions?
  - ✓ If the new LAMBDA is so amazing at creating custom functions, why do we need helper functions for LAMBDA? The reason is because there are some tasks that Built-in Excel Functions and LAMBDA Designed Custom Functions have a hard time doing. Here are two examples of tasks that functions have a hard time doing:
    1. Without matrix algebra, functions cannot spill an aggregate row total down a column. But by using LAMBDA inside the LAMBDA Helper Function BYROW, you can do exactly that with this formula: =BYROW(B11:E15,LAMBDA(r,SUM(r))), where the range B11:E15 contains rows of numbers.
    2. Functions cannot spill an account balance formula that must always refer to the previous cell's balance. But by using LAMBDA inside the LAMBDA Helper Function SCAN, you can do exactly that with this formula: =SCAN(M17,L18:L22,LAMBDA(i,a,i+a)), where the cell M17 contains the starting balance and the range L18:L22 contains the array of change values.
  - ✓ You can use LAMBDA inside the LAMBDA Helper Functions directly in the worksheet or in Defined Name Re-usable LAMBDA Custom Functions.

- These are the six LAMBDA Helper Functions
  1. BYCOL (array, LAMBDA()
     - The BYCOL function iterates a function defined by the LAMBDA function over a specified array, making a function calculation for each column and returning an array of the results. This function is good for spilling an aggregate calculation for each column in a specified array.
  2. BYROW(array, LAMBDA())
     - The BYROW function iterates a function defined by the LAMBDA function over a specified array, making a function calculation for each row and returning an array of the results. This function is good for spilling an aggregate calculation for each row in a specified array.
  3. MAKEARRAY(rows, cols, LAMBDA())
     - The MAKEARRAY function returns a calculated array of a specified row and column size, by applying a function defined by the LAMBDA function.
  4. MAP (array1, [array2], [array_n]… LAMBDA())
     - The MAP function Iterates a function defined by the LAMBDA function over a specified set of arrays (1 or more), making a function calculation for each cell in the corresponding arrays and then returning an array of the results. If you are iterating over a single row or single column, then it is safer to use the MAP function rather than the BYROW or BYCOL function, respectively. When the LAMBDA function expands as it iterates, BYCOL and BYROW may give you the wrong answer, whereas the MAP function will not. For example, a formula like: =BYROW(SEQUENCE(3),LAMBDA(r,CONCAT(SEQUENCE(,r)))) will yield the text array: {1;1;1}, whereas a formula like: =MAP(SEQUENCE(3),LAMBDA(x,CONCAT(SEQUENCE(,x)))) will yield the text array: {1;12;123}.
  5. SCAN ([starting_value], array, LAMBDA())
     - The SCAN function scans an array by applying a LAMBDA to each value and returns an array that has each intermediate value. For example, it can create a spilled cumulative total formula. This function is good for spilling a cumulative total or an account balance formula, where the spilled account balance formula must always refer to the previous cell's balance (a relative cell reference that always looks "one cell above").
  6. ISOMITTED(LAMBDA_parameter_name)
     - The ISOMITTED function checks whether the value in a LAMBDA formula input argument is missing and returns TRUE or FALSE.
  7. REDUCE([starting _value], array, LAMBDA())
     - The REDUCE function reduces an array to an accumulated value by applying a LAMBDA to each value and returning the last value in the accumulator array. This is similar to the SCAN function, but only returns the last value of a parallel SCAN spilled array.

- Examples of BYROW and BYCOL LAMBDA Helper Functions to spill aggregate calculations



LAMBDA #5 LAMBDA Helper Function: **BYROW**

Goal: Spill an aggregate row total for an array with five rows of numbers

LAMBDA #6 LAMBDA Helper Function: **BYCOL**

Goal: Spill an aggregate column total for an array with five columns of numbers

	Jan	Feb	Mar	Apr	Spilled Row Total	
	500	600	600	3,900	5,600	F9: =BYROW(B9:E13,LAMBDA(r,SUM(r)))
	1,000	1,800	2,800	2,700	8,300	
	2,500	6,250	4,000	3,700	16,450	
	250	275	2,100	2,050	4,675	
	100	200	750	1,550	2,600	
Total	4,350	9,125	10,250	13,900	37,625	

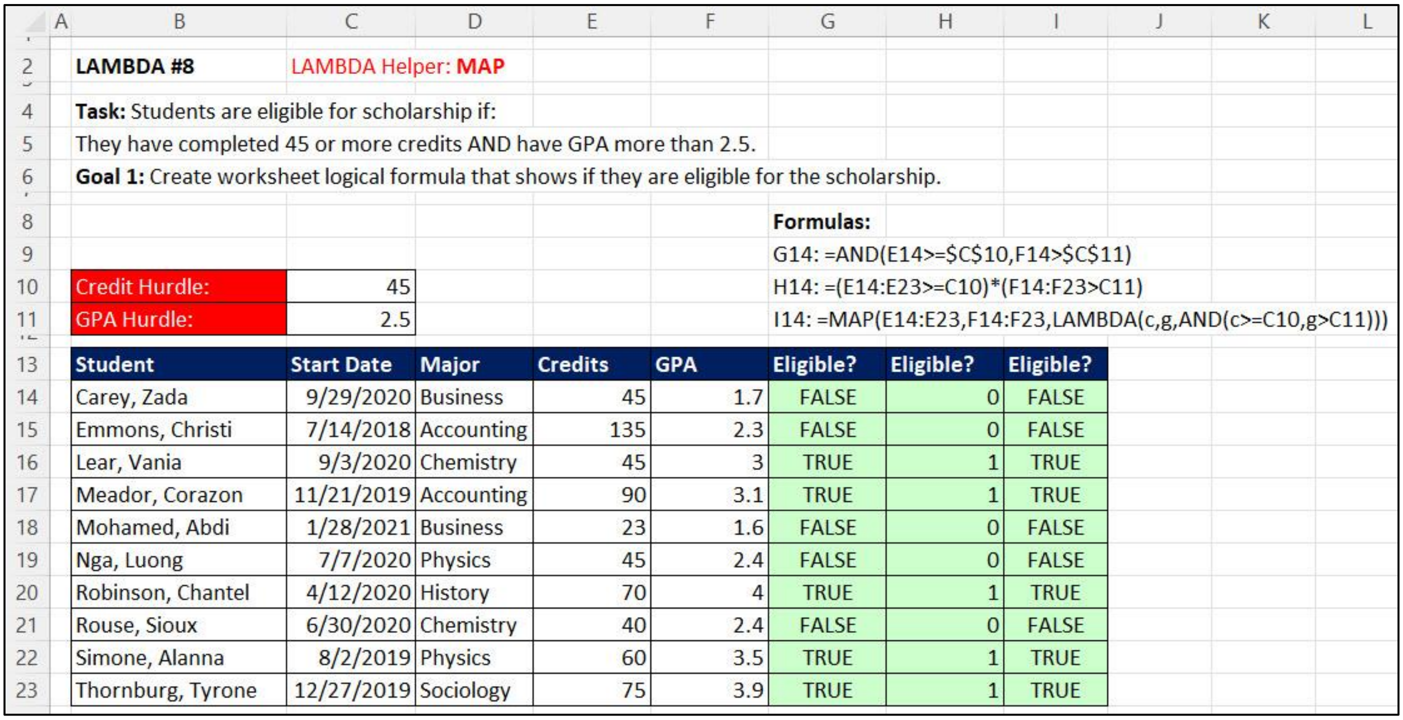
B14: =BYCOL(B9:F13,LAMBDA(c,SUM(c)))

LAMBDA #7 LAMBDA Helper Function: **BYROW**

Goal: Spill an aggregate row total for an array with five rows of numbers

Jan	Feb	Mar	Apr	Spilled Row Ave.	
500	600	600	3,900	1,400	F22: =BYROW(B22:E26,LAMBDA(r,AVERAGE(r)))
1,000	1,800	2,800	2,700	2,075	
2,500	6,250	4,000	3,700	4,113	
250	275	2,100	2,050	1,169	
100	200	750	1,550	650	

- Examples of MAP LAMBDA Helper Function



LAMBDA #8 LAMBDA Helper: **MAP**

Task: Students are eligible for scholarship if:

They have completed 45 or more credits AND have GPA more than 2.5.

Goal 1: Create worksheet logical formula that shows if they are eligible for the scholarship.

Formulas:

G14: =AND(E14>=$C$10,F14>$C$11)

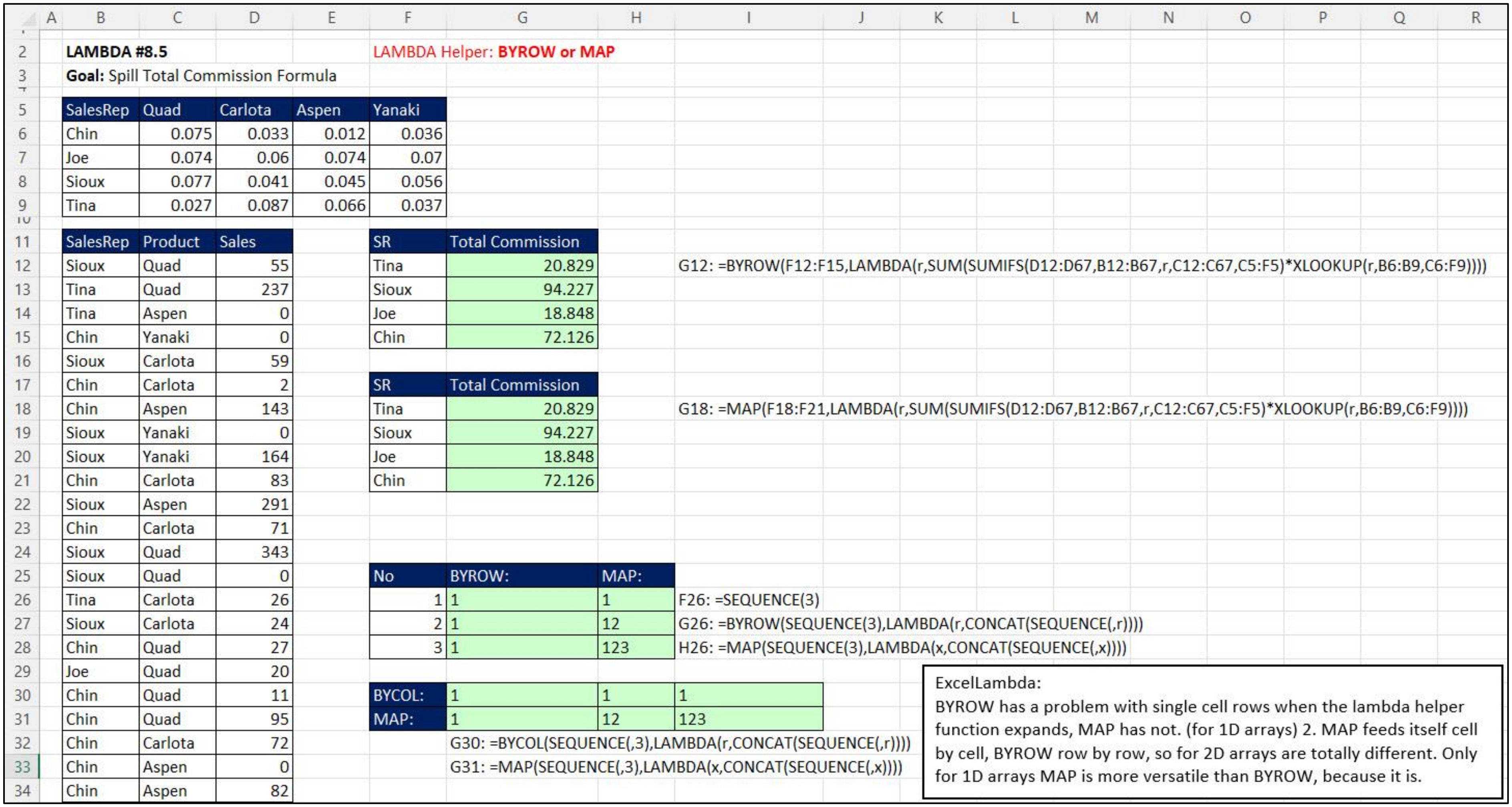
H14: =(E14:E23>=C10)*(F14:F23>C11)

I14: =MAP(E14:E23,F14:F23,LAMBDA(c,g,AND(c>=C10,g>C11)))

Credit Hurdle:	45
GPA Hurdle:	2.5

Student	Start Date	Major	Credits	GPA	Eligible?	Eligible?	Eligible?
Carey, Zada	9/29/2020	Business	45	1.7	FALSE	0	FALSE
Emmons, Christi	7/14/2018	Accounting	135	2.3	FALSE	0	FALSE
Lear, Vania	9/3/2020	Chemistry	45	3	TRUE	1	TRUE
Meador, Corazon	11/21/2019	Accounting	90	3.1	TRUE	1	TRUE
Mohamed, Abdi	1/28/2021	Business	23	1.6	FALSE	0	FALSE
Nga, Luong	7/7/2020	Physics	45	2.4	FALSE	0	FALSE
Robinson, Chantel	4/12/2020	History	70	4	TRUE	1	TRUE
Rouse, Sioux	6/30/2020	Chemistry	40	2.4	FALSE	0	FALSE
Simone, Alanna	8/2/2019	Physics	60	3.5	TRUE	1	TRUE
Thornburg, Tyrone	12/27/2019	Sociology	75	3.9	TRUE	1	TRUE

- Use MAP on Single Column or Single Row rather than BYROW or BYCOL



LAMBDA #8.5

LAMBDA Helper: **BYROW or MAP**

Goal: Spill Total Commission Formula

SalesRep	Quad	Carlota	Aspen	Yanaki
Chin	0.075	0.033	0.012	0.036
Joe	0.074	0.06	0.074	0.07
Sioux	0.077	0.041	0.045	0.056
Tina	0.027	0.087	0.066	0.037

SalesRep	Product	Sales
Sioux	Quad	55
Tina	Quad	237
Tina	Aspen	0
Chin	Yanaki	0
Sioux	Carlota	59
Chin	Carlota	2
Chin	Aspen	143
Sioux	Yanaki	0
Sioux	Yanaki	164
Chin	Carlota	83
Sioux	Aspen	291
Chin	Carlota	71
Sioux	Quad	343
Sioux	Quad	0
Tina	Carlota	26
Sioux	Carlota	24
Chin	Quad	27
Joe	Quad	20
Chin	Quad	11
Chin	Quad	95
Chin	Carlota	72
Chin	Aspen	0
Chin	Aspen	82

SR	Total Commission
Tina	20.829
Sioux	94.227
Joe	18.848
Chin	72.126

G12: =BYROW(F12:F15,LAMBDA(r,SUM(SUMIFS(D12:D67,B12:B67,r,C12:C67,C5:F5)*XLOOKUP(r,B6:B9,C6:F9))))

SR	Total Commission
Tina	20.829
Sioux	94.227
Joe	18.848
Chin	72.126

G18: =MAP(F18:F21,LAMBDA(r,SUM(SUMIFS(D12:D67,B12:B67,r,C12:C67,C5:F5)*XLOOKUP(r,B6:B9,C6:F9))))

No	BYROW:	MAP:
1	1	1
2	1	12
3	1	123

F26: =SEQUENCE(3)

G26: =BYROW(SEQUENCE(3),LAMBDA(r,CONCAT(SEQUENCE(,r))))

H26: =MAP(SEQUENCE(3),LAMBDA(x,CONCAT(SEQUENCE(,x))))

BYCOL:	1	1	1
MAP:	1	12	123

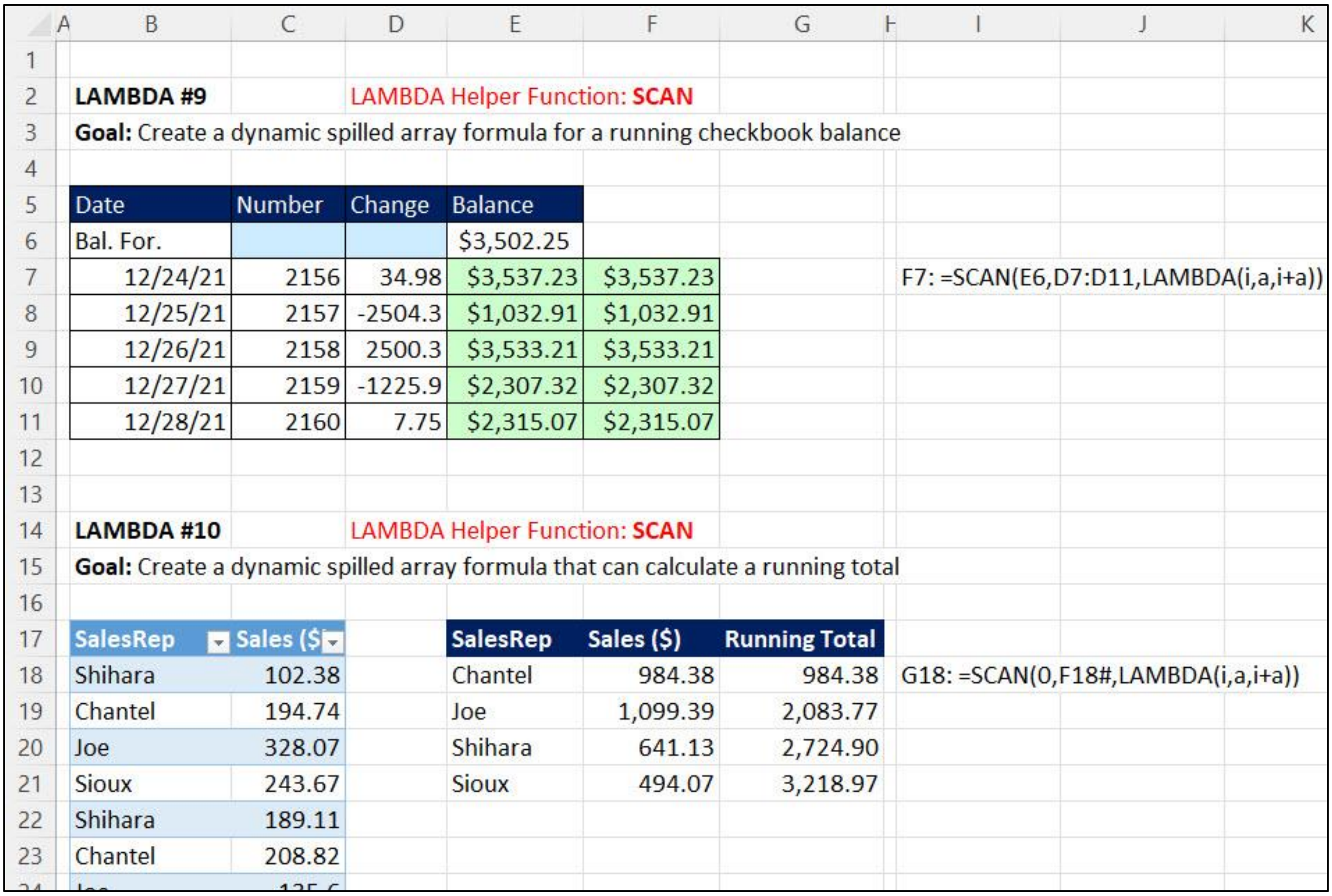
G30: =BYCOL(SEQUENCE(,3),LAMBDA(r,CONCAT(SEQUENCE(,r))))

G31: =MAP(SEQUENCE(,3),LAMBDA(x,CONCAT(SEQUENCE(,x))))

ExcelLambda:
BYROW has a problem with single cell rows when the lambda helper function expands, MAP has not. (for 1D arrays) 2. MAP feeds itself cell by cell, BYROW row by row, so for 2D arrays are totally different. Only for 1D arrays MAP is more versatile than BYROW, because it is.

- Examples of SCAN LAMBDA Helper Function

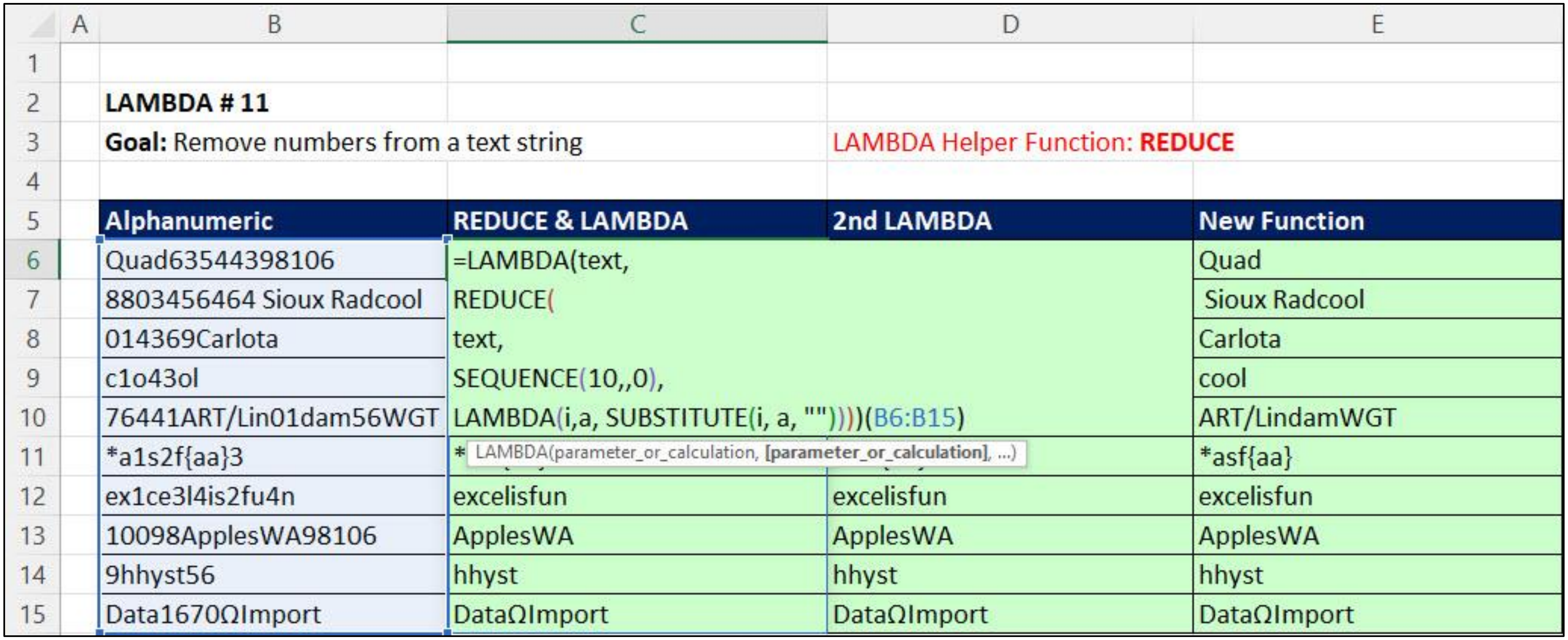
This example shows how to use SACN to iterate across an array and deliver each intermediate amount, like in a running total. SCAN also is the function method to simulate a Relative Cell Reference in a spilled array.



	A	B	C	D	E	F	G	H	I	J	K
1											
2		**LAMBDA #9**		LAMBDA Helper Function: **SCAN**							
3		**Goal:** Create a dynamic spilled array formula for a running checkbook balance									
4											
5		Date	Number	Change	Balance						
6		Bal. For.			$3,502.25						
7		12/24/21	2156	34.98	$3,537.23	$3,537.23			F7: =SCAN(E6,D7:D11,LAMBDA(i,a,i+a))		
8		12/25/21	2157	-2504.3	$1,032.91	$1,032.91					
9		12/26/21	2158	2500.3	$3,533.21	$3,533.21					
10		12/27/21	2159	-1225.9	$2,307.32	$2,307.32					
11		12/28/21	2160	7.75	$2,315.07	$2,315.07					
12											
13											
14		**LAMBDA #10**		LAMBDA Helper Function: **SCAN**							
15		**Goal:** Create a dynamic spilled array formula that can calculate a running total									
16											
17		SalesRep	Sales ($		SalesRep	Sales ($)	Running Total				
18		Shihara	102.38		Chantel	984.38	984.38		G18: =SCAN(0,F18#,LAMBDA(i,a,i+a))		
19		Chantel	194.74		Joe	1,099.39	2,083.77				
20		Joe	328.07		Shihara	641.13	2,724.90				
21		Sioux	243.67		Sioux	494.07	3,218.97				
22		Shihara	189.11								
23		Chantel	208.82								

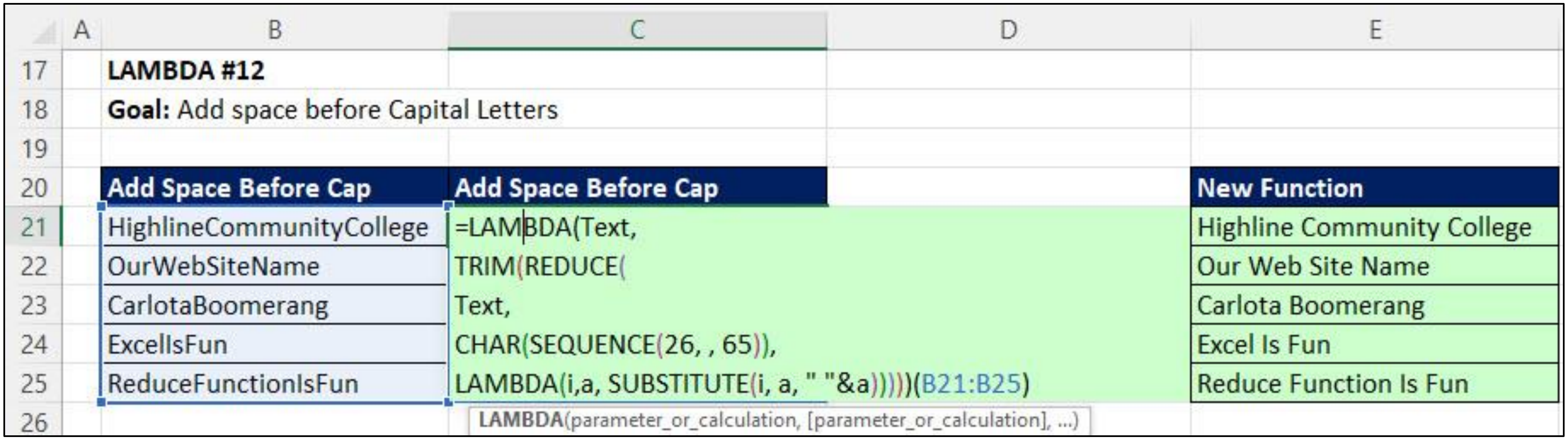
- Examples of REDUCE LAMBDA Helper Function

This example shows how to use REDUCE to iterate across an array and deliver the final result after all iterations.

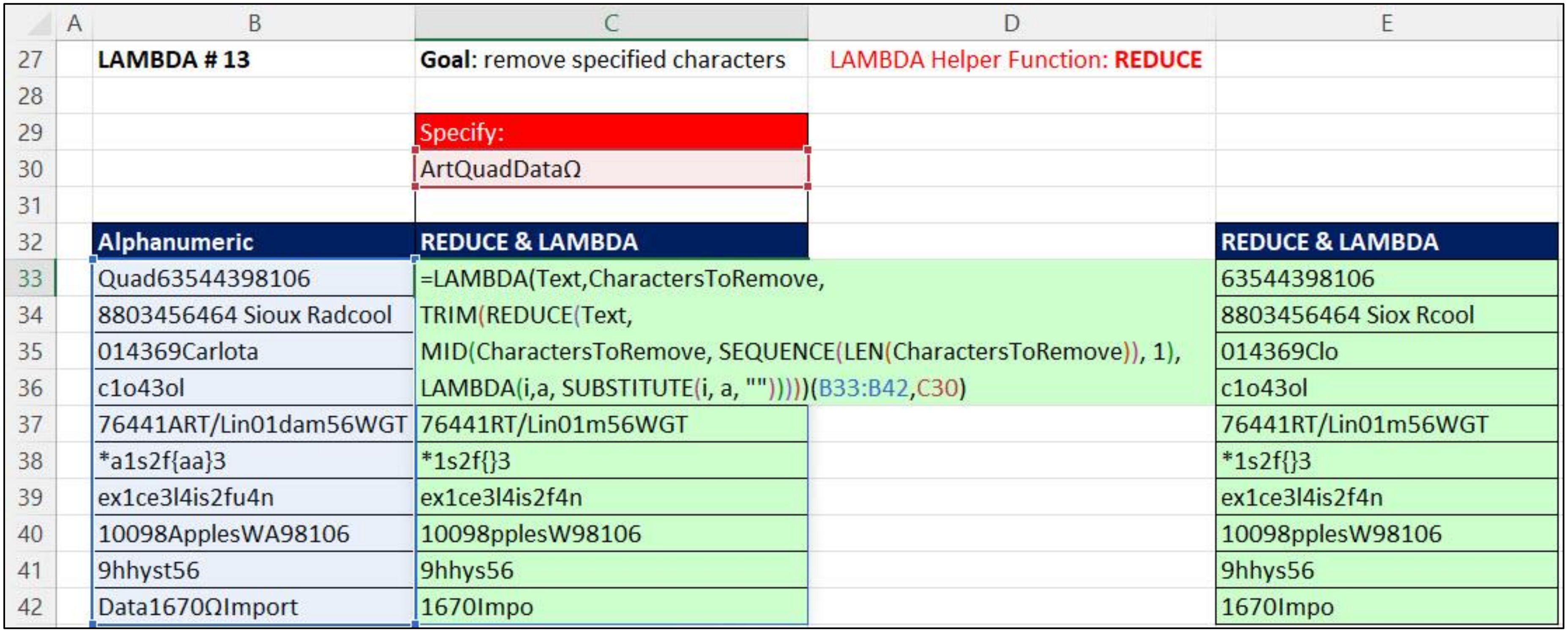


	A	B	C	D	E
1					
2		**LAMBDA # 11**			
3		**Goal:** Remove numbers from a text string		LAMBDA Helper Function: **REDUCE**	
4					
5		**Alphanumeric**	**REDUCE & LAMBDA**	**2nd LAMBDA**	**New Function**
6		Quad63544398106	=LAMBDA(text,		Quad
7		8803456464 Sioux Radcool	REDUCE(		Sioux Radcool
8		014369Carlota	text,		Carlota
9		c1o43ol	SEQUENCE(10,,0),		cool
10		76441ART/Lin01dam56WGT	LAMBDA(i,a, SUBSTITUTE(i, a, "")))(B6:B15)		ART/LindamWGT
11		*a1s2f{aa}3	* LAMBDA(parameter_or_calculation, [parameter_or_calculation], ...)		*asf{aa}
12		ex1ce3l4is2fu4n	excelisfun	excelisfun	excelisfun
13		10098ApplesWA98106	ApplesWA	ApplesWA	ApplesWA
14		9hhyst56	hhyst	hhyst	hhyst
15		Data1670ΩImport	DataΩImport	DataΩImport	DataΩImport

Two more examples of REDUCE on next page:

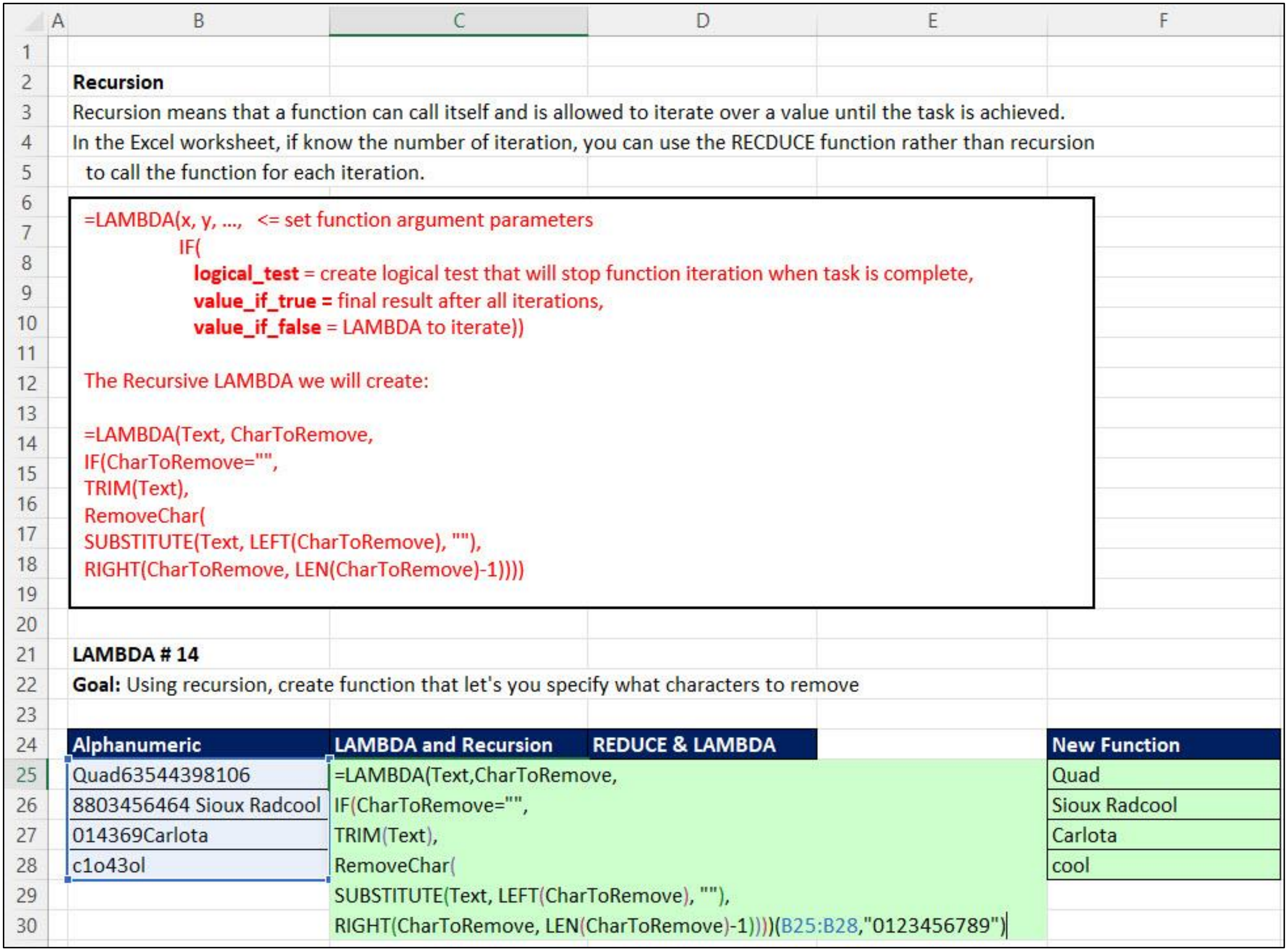


	A	B	C	D	E
17		**LAMBDA #12**			
18		**Goal:** Add space before Capital Letters			
19					
20		**Add Space Before Cap**	**Add Space Before Cap**		**New Function**
21		HighlineCommunityCollege	=LAMBDA(Text,		Highline Community College
22		OurWebSiteName	TRIM(REDUCE(		Our Web Site Name
23		CarlotaBoomerang	Text,		Carlota Boomerang
24		ExcelIsFun	CHAR(SEQUENCE(26, , 65)),		Excel Is Fun
25		ReduceFunctionIsFun	LAMBDA(i,a, SUBSTITUTE(i, a, " "&a))))(B21:B25)		Reduce Function Is Fun
26			**LAMBDA**(parameter_or_calculation, [parameter_or_calculation], ...)		



	A	B	C	D	E
27		**LAMBDA # 13**	**Goal**: remove specified characters	LAMBDA Helper Function: **REDUCE**	
28					
29			Specify:		
30			ArtQuadDataΩ		
31					
32		**Alphanumeric**	**REDUCE & LAMBDA**		**REDUCE & LAMBDA**
33		Quad63544398106	=LAMBDA(Text,CharactersToRemove,		63544398106
34		8803456464 Sioux Radcool	TRIM(REDUCE(Text,		8803456464 Siox Rcool
35		014369Carlota	MID(CharactersToRemove, SEQUENCE(LEN(CharactersToRemove)), 1),		014369Clo
36		c1o43ol	LAMBDA(i,a, SUBSTITUTE(i, a, "")))))(B33:B42,C30)		c1o43ol
37		76441ART/Lin01dam56WGT	76441RT/Lin01m56WGT		76441RT/Lin01m56WGT
38		*a1s2f{aa}3	*1s2f{}3		*1s2f{}3
39		ex1ce3l4is2fu4n	ex1ce3l4is2f4n		ex1ce3l4is2f4n
40		10098ApplesWA98106	10098pplesW98106		10098pplesW98106
41		9hhyst56	9hhys56		9hhys56
42		Data1670ΩImport	1670Impo		1670Impo

- Example of Recursion in the LAMBDA function



Recursion

Recursion means that a function can call itself and is allowed to iterate over a value until the task is achieved.

In the Excel worksheet, if know the number of iteration, you can use the RECDUCE function rather than recursion to call the function for each iteration.

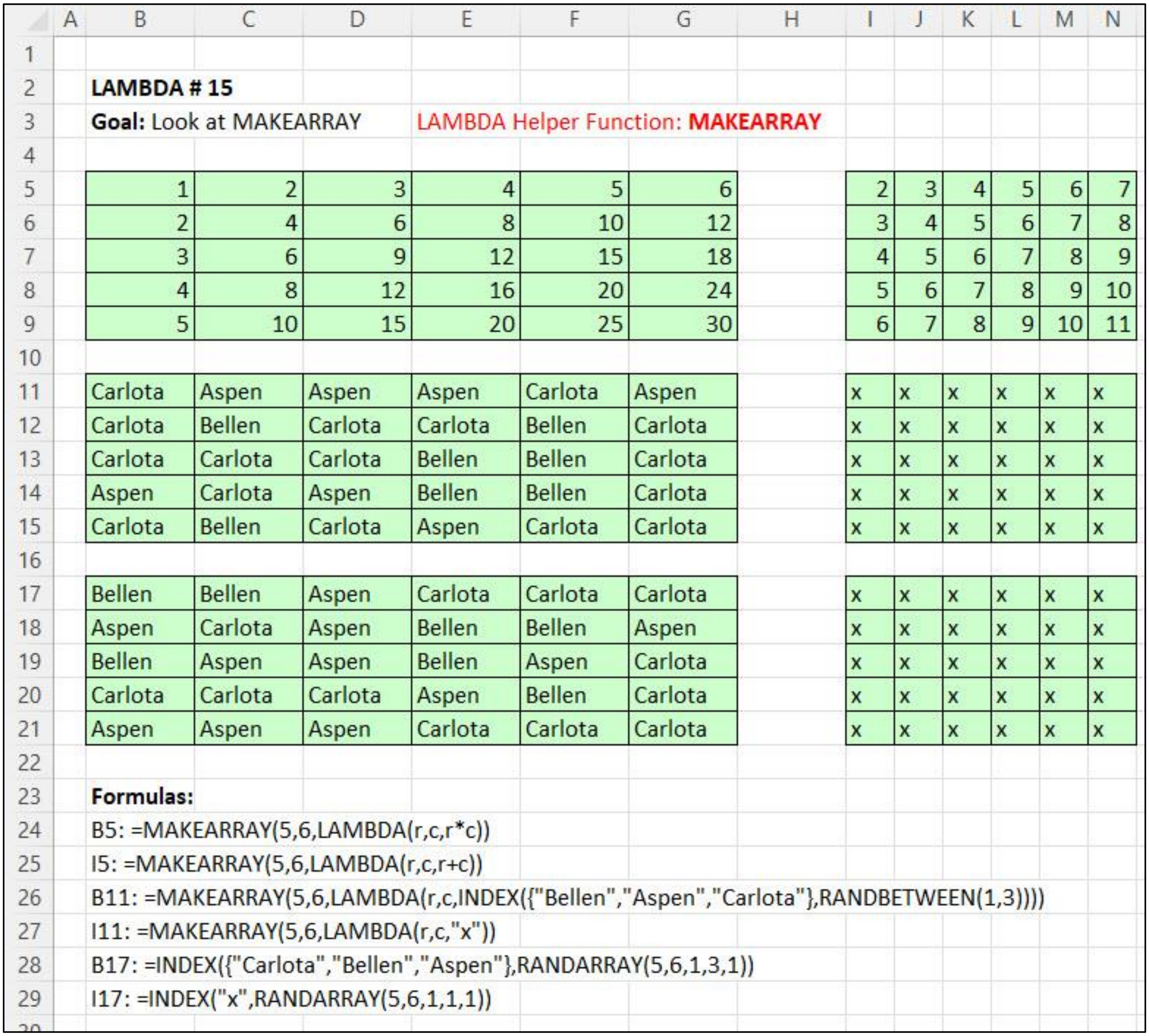
```
=LAMBDA(x, y, ...,   <= set function argument parameters
            IF(
              logical_test = create logical test that will stop function iteration when task is complete,
              value_if_true = final result after all iterations,
              value_if_false = LAMBDA to iterate))

The Recursive LAMBDA we will create:

=LAMBDA(Text, CharToRemove,
IF(CharToRemove="",
TRIM(Text),
RemoveChar(
SUBSTITUTE(Text, LEFT(CharToRemove), ""),
RIGHT(CharToRemove, LEN(CharToRemove)-1))))
```

LAMBDA # 14

Goal: Using recursion, create function that let's you specify what characters to remove

Alphanumeric	LAMBDA and Recursion	REDUCE & LAMBDA		New Function
Quad63544398106	=LAMBDA(Text,CharToRemove,			Quad
8803456464 Sioux Radcool	IF(CharToRemove="",			Sioux Radcool
014369Carlota	TRIM(Text),			Carlota
c1o43ol	RemoveChar(			cool
	SUBSTITUTE(Text, LEFT(CharToRemove), ""),			
	RIGHT(CharToRemove, LEN(CharToRemove)-1))))(B25:B28,"0123456789")			

Example of MAKRARRAY LAMBDA Helper Function



LAMBDA # 15

Goal: Look at MAKEARRAY LAMBDA Helper Function: **MAKEARRAY**

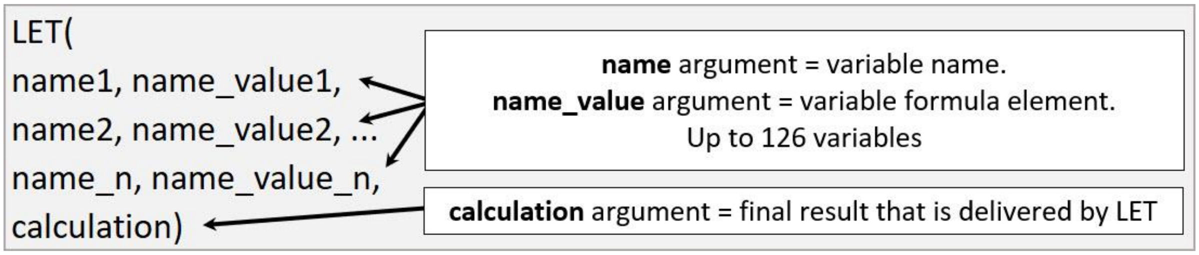
	B	C	D	E	F	G	H	I	J	K	L	M	N
5	1	2	3	4	5	6		2	3	4	5	6	7
6	2	4	6	8	10	12		3	4	5	6	7	8
7	3	6	9	12	15	18		4	5	6	7	8	9
8	4	8	12	16	20	24		5	6	7	8	9	10
9	5	10	15	20	25	30		6	7	8	9	10	11
10													
11	Carlota	Aspen	Aspen	Aspen	Carlota	Aspen		x	x	x	x	x	x
12	Carlota	Bellen	Carlota	Carlota	Bellen	Carlota		x	x	x	x	x	x
13	Carlota	Carlota	Carlota	Bellen	Bellen	Carlota		x	x	x	x	x	x
14	Aspen	Carlota	Aspen	Bellen	Bellen	Carlota		x	x	x	x	x	x
15	Carlota	Bellen	Carlota	Aspen	Carlota	Carlota		x	x	x	x	x	x
16													
17	Bellen	Bellen	Aspen	Carlota	Carlota	Carlota		x	x	x	x	x	x
18	Aspen	Carlota	Aspen	Bellen	Bellen	Aspen		x	x	x	x	x	x
19	Bellen	Aspen	Aspen	Bellen	Aspen	Carlota		x	x	x	x	x	x
20	Carlota	Carlota	Carlota	Aspen	Bellen	Carlota		x	x	x	x	x	x
21	Aspen	Aspen	Aspen	Carlota	Carlota	Carlota		x	x	x	x	x	x

Formulas:

```
B5: =MAKEARRAY(5,6,LAMBDA(r,c,r*c))
I5: =MAKEARRAY(5,6,LAMBDA(r,c,r+c))
B11: =MAKEARRAY(5,6,LAMBDA(r,c,INDEX({"Bellen","Aspen","Carlota"},RANDBETWEEN(1,3))))
I11: =MAKEARRAY(5,6,LAMBDA(r,c,"x"))
B17: =INDEX({"Carlota","Bellen","Aspen"},RANDARRAY(5,6,1,3,1))
I17: =INDEX("x",RANDARRAY(5,6,1,1,1))
```

LET Function

- Define LET function: The LET function allows you to define variables within the function and use those variables to make a calculation. Use LET when you have repeating formula elements or when you want to make complex formulas easier to read. Let arguments are here:



```
LET(
name1, name_value1,
name2, name_value2, ...
name_n, name_value_n,
calculation)
```

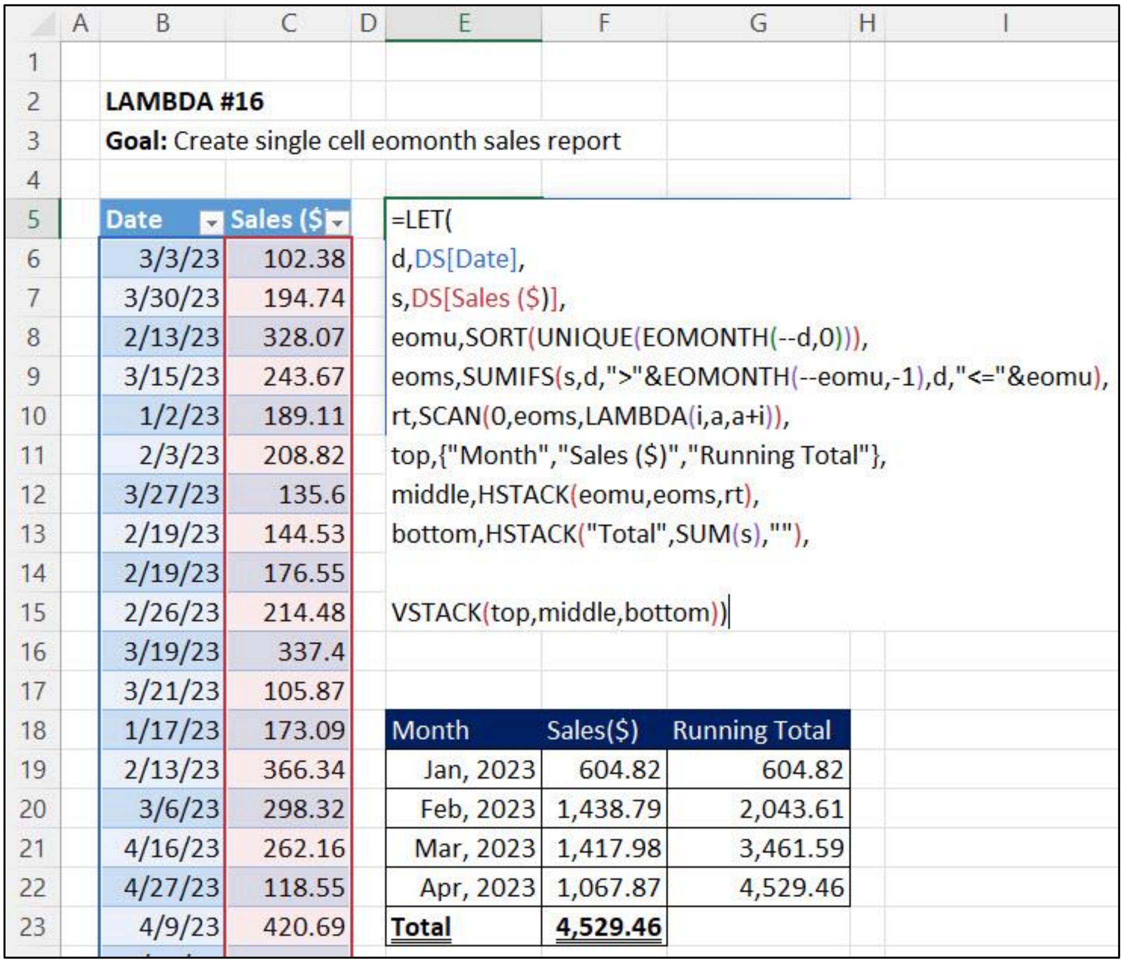
name argument = variable name.
name_value argument = variable formula element.
Up to 126 variables

calculation argument = final result that is delivered by LET

- Advantages of LET:
  - A variable is evaluated a single time, and the result is stored in memory so that it can be used throughout a formula. For formulas with repeating formula elements, this can reduce overall calculation time by preventing duplication of evaluation procedures.
  - Formulas with repeating elements are easy to edit because you have to edit in only one location.
  - Complex formulas can be visually easy to read because each element has a name and can be placed on a different line with the line feed keyboard shortcut, Alt+Enter.
  - You can build single cell report formulas more efficiently with the LET function.

Steps to Build LET & LAMBDA Functions for Single Cell Report

- Step 1: Build LET Part of formula:



LAMBDA #16

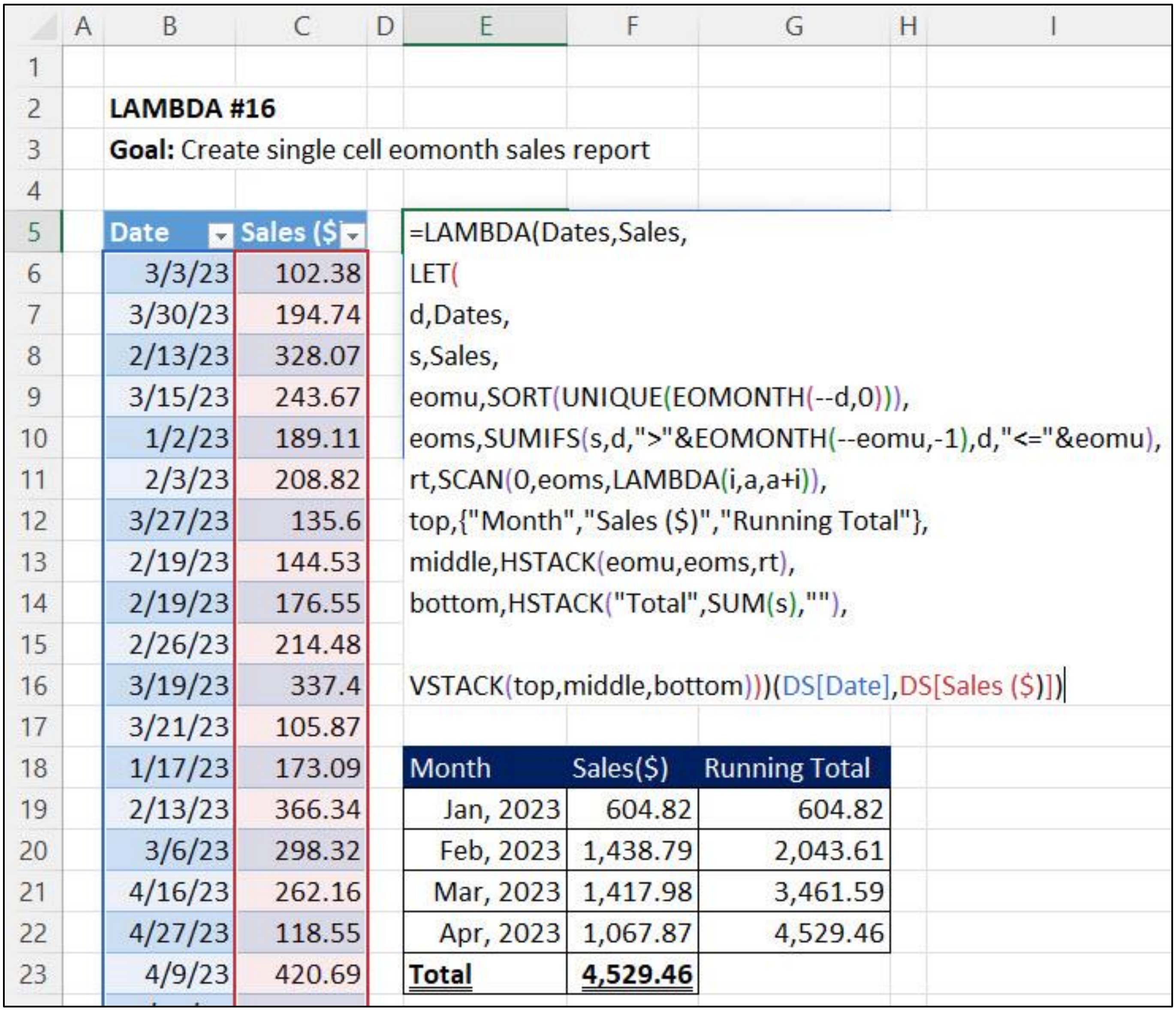
Goal: Create single cell eomonth sales report

Date	Sales ($)
3/3/23	102.38
3/30/23	194.74
2/13/23	328.07
3/15/23	243.67
1/2/23	189.11
2/3/23	208.82
3/27/23	135.6
2/19/23	144.53
2/19/23	176.55
2/26/23	214.48
3/19/23	337.4
3/21/23	105.87
1/17/23	173.09
2/13/23	366.34
3/6/23	298.32
4/16/23	262.16
4/27/23	118.55
4/9/23	420.69

```
=LET(
d,DS[Date],
s,DS[Sales ($)],
eomu,SORT(UNIQUE(EOMONTH(--d,0))),
eoms,SUMIFS(s,d,">"&EOMONTH(--eomu,-1),d,"<="&eomu),
rt,SCAN(0,eoms,LAMBDA(i,a,a+i)),
top,{"Month","Sales ($)","Running Total"},
middle,HSTACK(eomu,eoms,rt),
bottom,HSTACK("Total",SUM(s),""),

VSTACK(top,middle,bottom))
```

Month	Sales($)	Running Total
Jan, 2023	604.82	604.82
Feb, 2023	1,438.79	2,043.61
Mar, 2023	1,417.98	3,461.59
Apr, 2023	1,067.87	4,529.46
Total	**4,529.46**	

- Step 2: Add LAMBDA part to formula and then save as a Defined Name:



LAMBDA #16

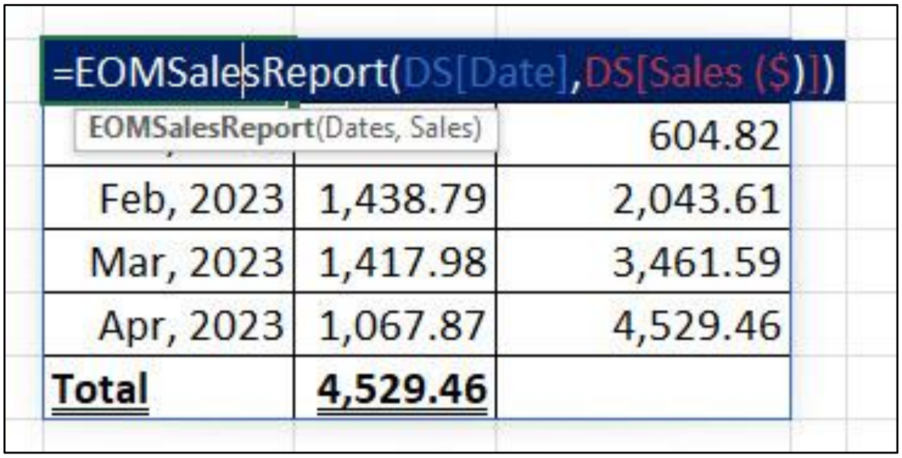
Goal: Create single cell eomonth sales report

Date	Sales ($
3/3/23	102.38
3/30/23	194.74
2/13/23	328.07
3/15/23	243.67
1/2/23	189.11
2/3/23	208.82
3/27/23	135.6
2/19/23	144.53
2/19/23	176.55
2/26/23	214.48
3/19/23	337.4
3/21/23	105.87
1/17/23	173.09
2/13/23	366.34
3/6/23	298.32
4/16/23	262.16
4/27/23	118.55
4/9/23	420.69

```
=LAMBDA(Dates,Sales,
LET(
d,Dates,
s,Sales,
eomu,SORT(UNIQUE(EOMONTH(--d,0))),
eoms,SUMIFS(s,d,">"&EOMONTH(--eomu,-1),d,"<="&eomu),
rt,SCAN(0,eoms,LAMBDA(i,a,a+i)),
top,{"Month","Sales ($)","Running Total"},
middle,HSTACK(eomu,eoms,rt),
bottom,HSTACK("Total",SUM(s),""),

VSTACK(top,middle,bottom)))(DS[Date],DS[Sales ($)])
```

Month	Sales($)	Running Total
Jan, 2023	604.82	604.82
Feb, 2023	1,438.79	2,043.61
Mar, 2023	1,417.98	3,461.59
Apr, 2023	1,067.87	4,529.46
Total	**4,529.46**	

- Step 3: Test new Monthly Sales Report LAMBDA Custom Function:

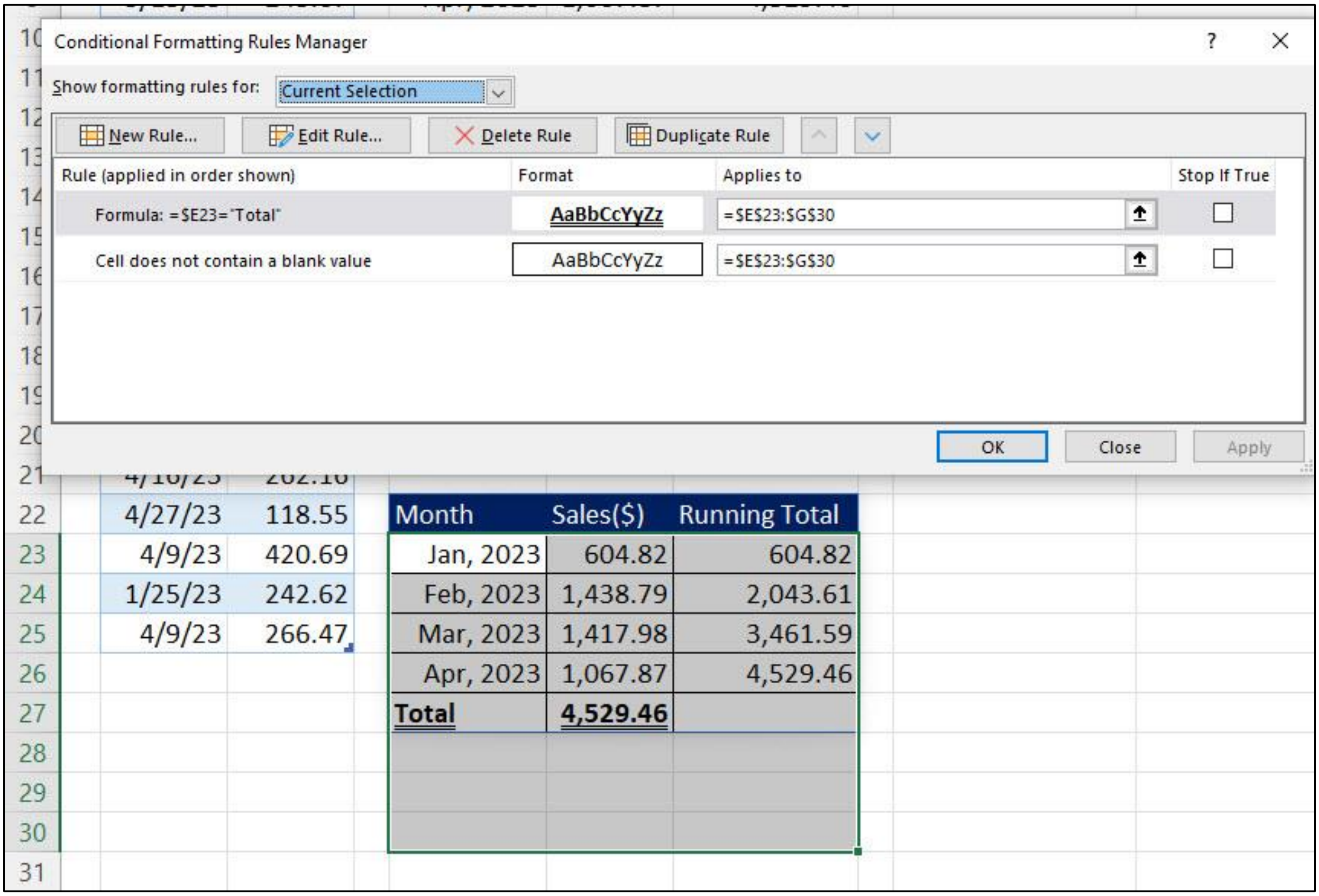


```
=EOMSalesReport(DS[Date],DS[Sales ($)])
```

EOMSalesReport(Dates, Sales)

		604.82
Feb, 2023	1,438.79	2,043.61
Mar, 2023	1,417.98	3,461.59
Apr, 2023	1,067.87	4,529.46
Total	**4,529.46**	

- Step 4: Add Conditional Formatting



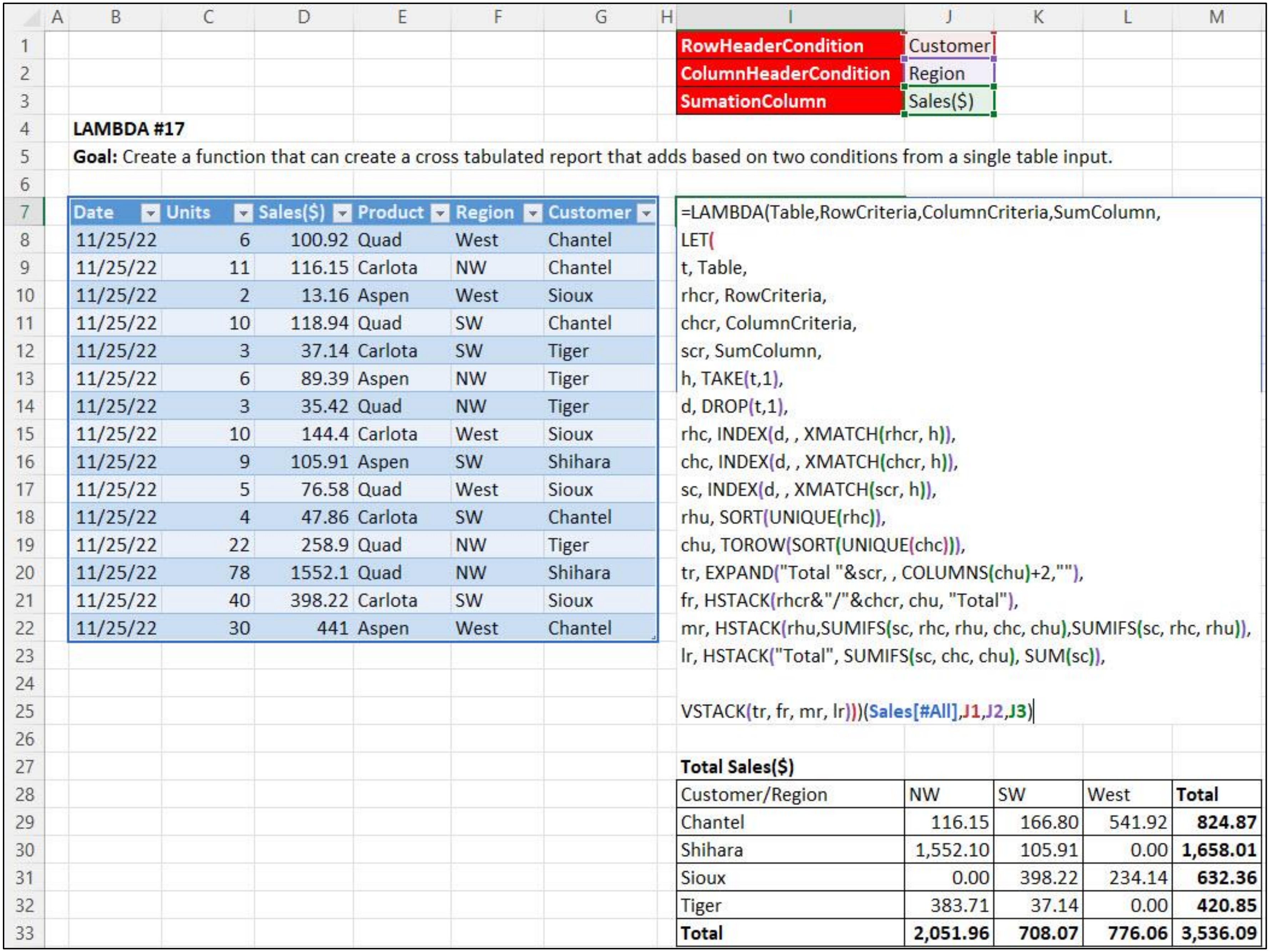
Conditional Formatting Rules Manager

Show formatting rules for: Current Selection

Rule (applied in order shown)	Format	Applies to	Stop If True
Formula: =$E23="Total"	**AaBbCcYyZz**	=$E$23:$G$30	☐
Cell does not contain a blank value	AaBbCcYyZz	=$E$23:$G$30	☐

22	4/27/23	118.55	Month	Sales($)	Running Total
23	4/9/23	420.69	Jan, 2023	604.82	604.82
24	1/25/23	242.62	Feb, 2023	1,438.79	2,043.61
25	4/9/23	266.47	Mar, 2023	1,417.98	3,461.59
26			Apr, 2023	1,067.87	4,529.46
27			**Total**	**4,529.46**	
28					
29					
30					
31					

LET & Lambda Functions to make Dynamic Cross Tab Report



RowHeaderCondition	Customer
ColumnHeaderCondition	Region
SumationColumn	Sales($)

LAMBDA #17

Goal: Create a function that can create a cross tabulated report that adds based on two conditions from a single table input.

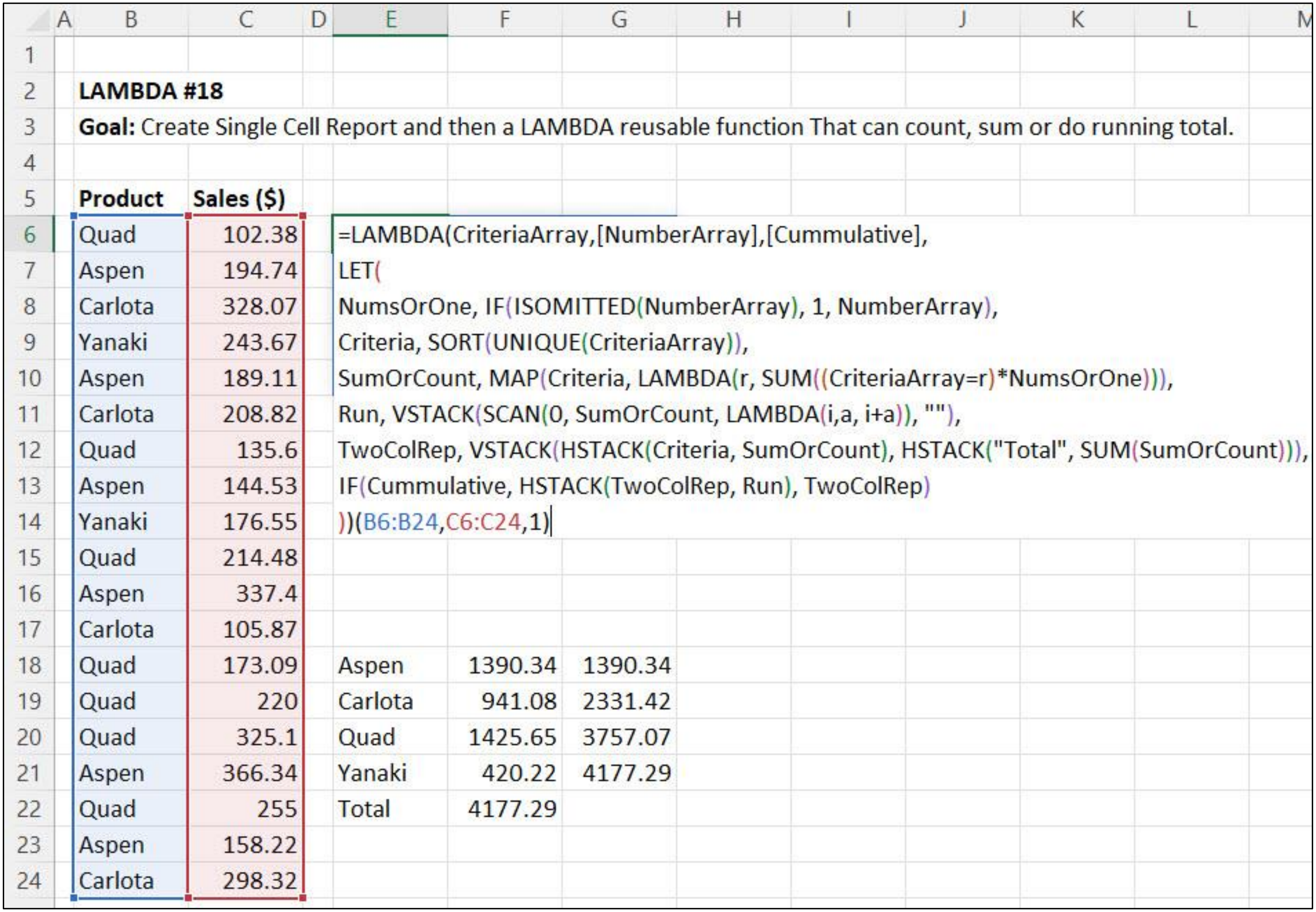
Date	Units	Sales($)	Product	Region	Customer
11/25/22	6	100.92	Quad	West	Chantel
11/25/22	11	116.15	Carlota	NW	Chantel
11/25/22	2	13.16	Aspen	West	Sioux
11/25/22	10	118.94	Quad	SW	Chantel
11/25/22	3	37.14	Carlota	SW	Tiger
11/25/22	6	89.39	Aspen	NW	Tiger
11/25/22	3	35.42	Quad	NW	Tiger
11/25/22	10	144.4	Carlota	West	Sioux
11/25/22	9	105.91	Aspen	SW	Shihara
11/25/22	5	76.58	Quad	West	Sioux
11/25/22	4	47.86	Carlota	SW	Chantel
11/25/22	22	258.9	Quad	NW	Tiger
11/25/22	78	1552.1	Quad	NW	Shihara
11/25/22	40	398.22	Carlota	SW	Sioux
11/25/22	30	441	Aspen	West	Chantel

```
=LAMBDA(Table,RowCriteria,ColumnCriteria,SumColumn,
LET(
t, Table,
rhcr, RowCriteria,
chcr, ColumnCriteria,
scr, SumColumn,
h, TAKE(t,1),
d, DROP(t,1),
rhc, INDEX(d, , XMATCH(rhcr, h)),
chc, INDEX(d, , XMATCH(chcr, h)),
sc, INDEX(d, , XMATCH(scr, h)),
rhu, SORT(UNIQUE(rhc)),
chu, TOROW(SORT(UNIQUE(chc))),
tr, EXPAND("Total "&scr, , COLUMNS(chu)+2,""),
fr, HSTACK(rhcr&"/"&chcr, chu, "Total"),
mr, HSTACK(rhu,SUMIFS(sc, rhc, rhu, chc, chu),SUMIFS(sc, rhc, rhu)),
lr, HSTACK("Total", SUMIFS(sc, chc, chu), SUM(sc)),

VSTACK(tr, fr, mr, lr)))(Sales[#All],J1,J2,J3)
```

Total Sales($)

Customer/Region	NW	SW	West	**Total**
Chantel	116.15	166.80	541.92	**824.87**
Shihara	1,552.10	105.91	0.00	**1,658.01**
Sioux	0.00	398.22	234.14	**632.36**
Tiger	383.71	37.14	0.00	**420.85**
Total	**2,051.96**	**708.07**	**776.06**	**3,536.09**

Example of LAMBDA Function that can Create Four Different Types of Reports



LAMBDA #18

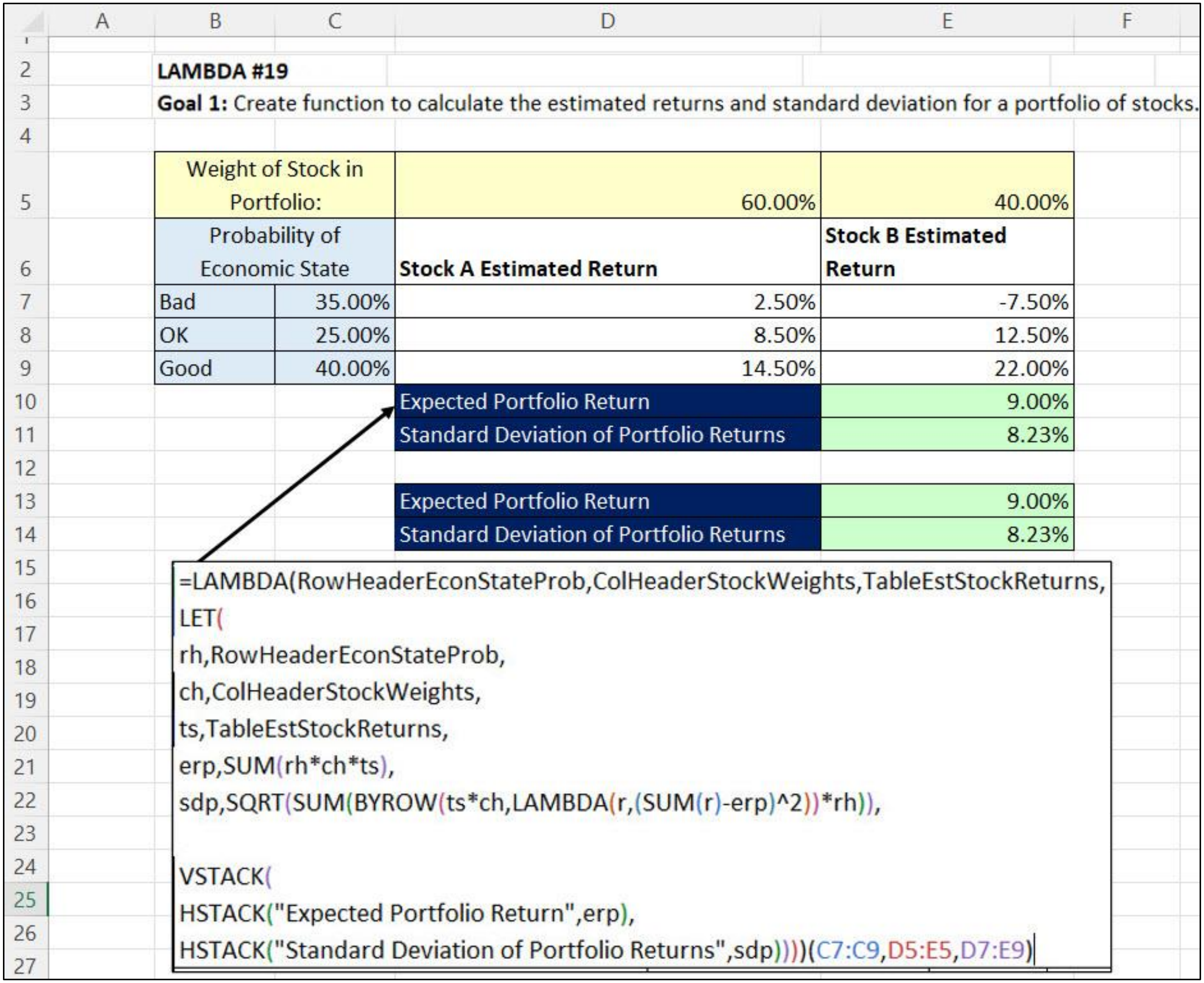
Goal: Create Single Cell Report and then a LAMBDA reusable function That can count, sum or do running total.

Product	Sales ($)
Quad	102.38
Aspen	194.74
Carlota	328.07
Yanaki	243.67
Aspen	189.11
Carlota	208.82
Quad	135.6
Aspen	144.53
Yanaki	176.55
Quad	214.48
Aspen	337.4
Carlota	105.87
Quad	173.09
Quad	220
Quad	325.1
Aspen	366.34
Quad	255
Aspen	158.22
Carlota	298.32

```
=LAMBDA(CriteriaArray,[NumberArray],[Cummulative],
LET(
NumsOrOne, IF(ISOMITTED(NumberArray), 1, NumberArray),
Criteria, SORT(UNIQUE(CriteriaArray)),
SumOrCount, MAP(Criteria, LAMBDA(r, SUM((CriteriaArray=r)*NumsOrOne))),
Run, VSTACK(SCAN(0, SumOrCount, LAMBDA(i,a, i+a)), ""),
TwoColRep, VSTACK(HSTACK(Criteria, SumOrCount), HSTACK("Total", SUM(SumOrCount))),
IF(Cummulative, HSTACK(TwoColRep, Run), TwoColRep)
))(B6:B24,C6:C24,1)
```

Aspen	1390.34	1390.34
Carlota	941.08	2331.42
Quad	1425.65	3757.07
Yanaki	420.22	4177.29
Total	4177.29	

Custom LAMBDA & LET Function to Calculate Return and Standard Deviation for Portfolio of Stocks



LAMBDA #19

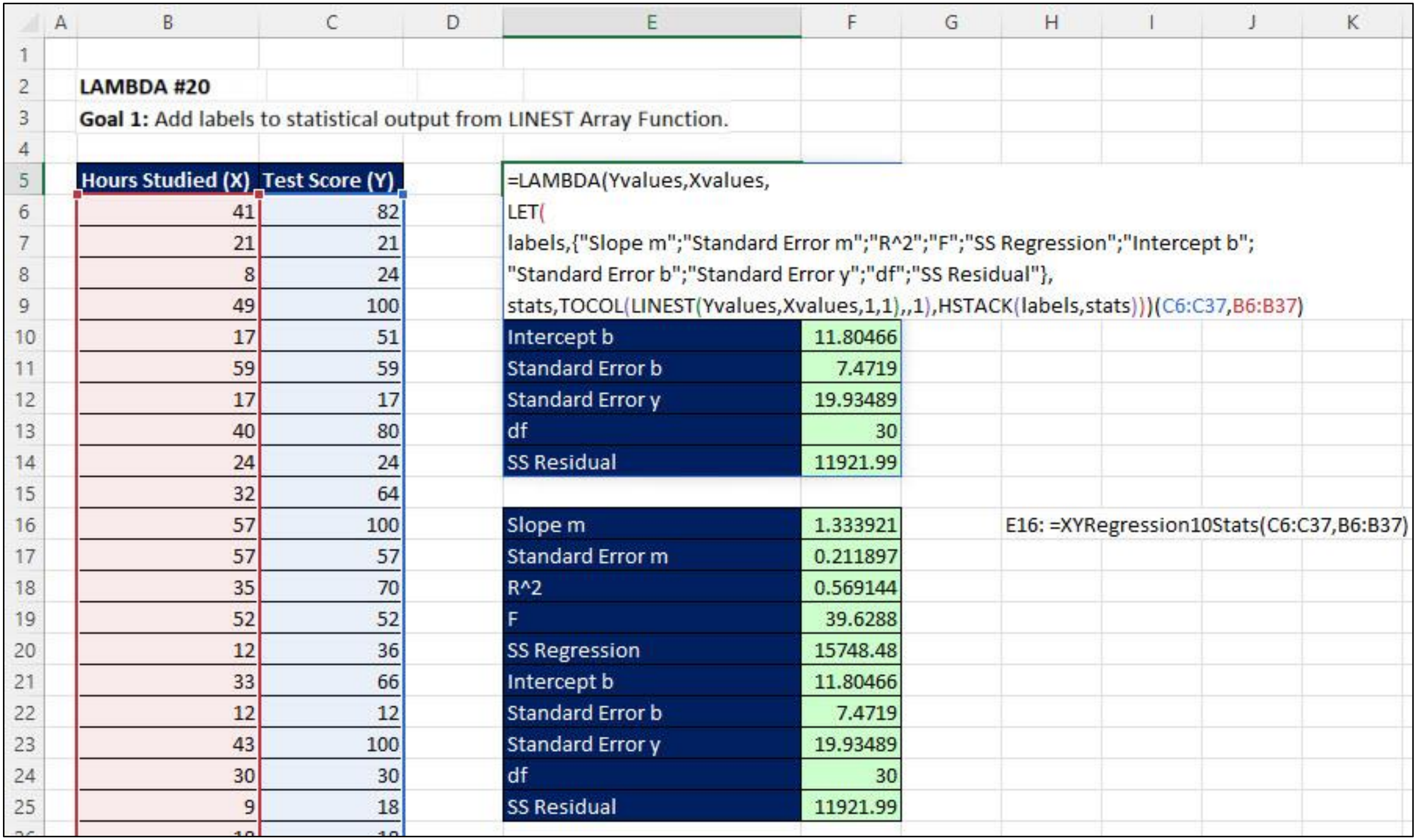
Goal 1: Create function to calculate the estimated returns and standard deviation for a portfolio of stocks.

Weight of Stock in Portfolio:		60.00%	40.00%
Probability of Economic State		**Stock A Estimated Return**	**Stock B Estimated Return**
Bad	35.00%	2.50%	-7.50%
OK	25.00%	8.50%	12.50%
Good	40.00%	14.50%	22.00%
		Expected Portfolio Return	9.00%
		Standard Deviation of Portfolio Returns	8.23%

Expected Portfolio Return	9.00%
Standard Deviation of Portfolio Returns	8.23%

```
=LAMBDA(RowHeaderEconStateProb,ColHeaderStockWeights,TableEstStockReturns,
LET(
rh,RowHeaderEconStateProb,
ch,ColHeaderStockWeights,
ts,TableEstStockReturns,
erp,SUM(rh*ch*ts),
sdp,SQRT(SUM(BYROW(ts*ch,LAMBDA(r,(SUM(r)-erp)^2))*rh)),

VSTACK(
HSTACK("Expected Portfolio Return",erp),
HSTACK("Standard Deviation of Portfolio Returns",sdp))))(C7:C9,D5:E5,D7:E9)
```

Custom LAMBDA & LET Function to Enhance the built-in Statistics Function LINEST



LAMBDA #20

Goal 1: Add labels to statistical output from LINEST Array Function.

Hours Studied (X)	Test Score (Y)
41	82
21	21
8	24
49	100
17	51
59	59
17	17
40	80
24	24
32	64
57	100
57	57
35	70
52	52
12	36
33	66
12	12
43	100
30	30
9	18

```
=LAMBDA(Yvalues,Xvalues,
LET(
labels,{"Slope m";"Standard Error m";"R^2";"F";"SS Regression";"Intercept b";
"Standard Error b";"Standard Error y";"df";"SS Residual"},
stats,TOCOL(LINEST(Yvalues,Xvalues,1,1),,1),HSTACK(labels,stats)))(C6:C37,B6:B37)
```

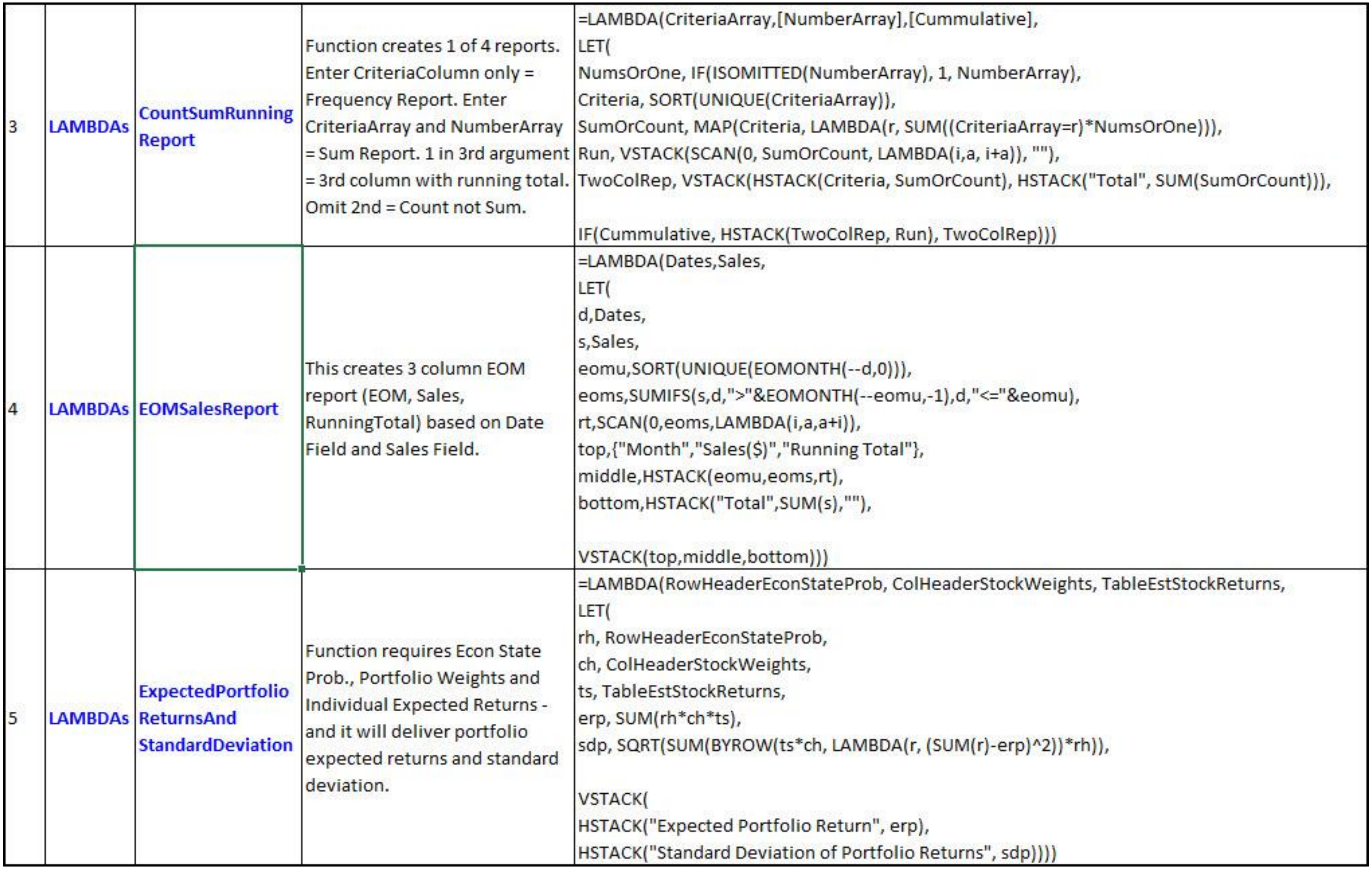
Intercept b	11.80466
Standard Error b	7.4719
Standard Error y	19.93489
df	30
SS Residual	11921.99

Slope m	1.333921
Standard Error m	0.211897
R^2	0.569144
F	39.6288
SS Regression	15748.48
Intercept b	11.80466
Standard Error b	7.4719
Standard Error y	19.93489
df	30
SS Residual	11921.99

E16: =XYRegression10Stats(C6:C37,B6:B37)

All Defined Names From Video

No.	Type	Name	Description of Defined Name	Defined Name Definition
1	LAMBDAs	COGS	This function calculates Cost Of Goods Sold (COGS) based on number of units at the beginning of the period (Begin), number of units at the end of the period (End) and the value per unit (Value) using the formula: (Begin-End)*Value.	=LAMBDA(Begin, End, Value, (Begin - End) * Value)
2	LAMBDAs	CrossTabReport	This function makes a summation cross tabulated report with the function argument formula inputs: Table (headers and data), field name for row header condition, field name for column header condition and field name for column to sum.	=LAMBDA(Table,RowCriteria,ColumnCriteria,SumColumn, LET(t, Table, rhcr, RowCriteria, chcr, ColumnCriteria, scr, SumColumn, h, TAKE(t,1), d, DROP(t,1), rhc, INDEX(d, , XMATCH(rhcr, h)), chc, INDEX(d, , XMATCH(chcr, h)), sc, INDEX(d, , XMATCH(scr, h)), rhu, SORT(UNIQUE(rhc)), chu, TOROW(SORT(UNIQUE(chc))), tr, EXPAND("Total "&scr, , COLUMNS(chu)+2,""), fr, HSTACK(rhcr&"/"&chcr, chu, "Total"), mr, HSTACK(rhu,SUMIFS(sc, rhc, rhu, chc, chu),SUMIFS(sc, rhc, rhu)), lr, HSTACK("Total", SUMIFS(sc, chc, chu), SUM(sc)), VSTACK(tr, fr, mr, lr)))



3	LAMBDAs	CountSumRunning Report	Function creates 1 of 4 reports. Enter CriteriaColumn only = Frequency Report. Enter CriteriaArray and NumberArray = Sum Report. 1 in 3rd argument = 3rd column with running total. Omit 2nd = Count not Sum.	=LAMBDA(CriteriaArray,[NumberArray],[Cummulative], LET(NumsOrOne, IF(ISOMITTED(NumberArray), 1, NumberArray), Criteria, SORT(UNIQUE(CriteriaArray)), SumOrCount, MAP(Criteria, LAMBDA(r, SUM((CriteriaArray=r)*NumsOrOne))), Run, VSTACK(SCAN(0, SumOrCount, LAMBDA(i,a, i+a)), ""), TwoColRep, VSTACK(HSTACK(Criteria, SumOrCount), HSTACK("Total", SUM(SumOrCount))), IF(Cummulative, HSTACK(TwoColRep, Run), TwoColRep)))
4	LAMBDAs	EOMSalesReport	This creates 3 column EOM report (EOM, Sales, RunningTotal) based on Date Field and Sales Field.	=LAMBDA(Dates,Sales, LET(d,Dates, s,Sales, eomu,SORT(UNIQUE(EOMONTH(--d,0))), eoms,SUMIFS(s,d,">"&EOMONTH(--eomu,-1),d,"<="&eomu), rt,SCAN(0,eoms,LAMBDA(i,a,a+i)), top,{"Month","Sales($)","Running Total"}, middle,HSTACK(eomu,eoms,rt), bottom,HSTACK("Total",SUM(s),""), VSTACK(top,middle,bottom)))
5	LAMBDAs	ExpectedPortfolio ReturnsAnd StandardDeviation	Function requires Econ State Prob., Portfolio Weights and Individual Expected Returns - and it will deliver portfolio expected returns and standard deviation.	=LAMBDA(RowHeaderEconStateProb, ColHeaderStockWeights, TableEstStockReturns, LET(rh, RowHeaderEconStateProb, ch, ColHeaderStockWeights, ts, TableEstStockReturns, erp, SUM(rh*ch*ts), sdp, SQRT(SUM(BYROW(ts*ch, LAMBDA(r, (SUM(r)-erp)^2))*rh)), VSTACK(HSTACK("Expected Portfolio Return", erp), HSTACK("Standard Deviation of Portfolio Returns", sdp))))

No.	Type	Name	Description of Defined Name	Defined Name Definition
6	LAMBDAs	FiscalQuarter	This function allows you to enter a serial number date and the function will deliver a fiscal quarter based on July 1 as first day in Fiscal Year.	=LAMBDA(Date,"Q" & LOOKUP(MONTH(Date), {1,3;4,4;7,1;10,2}))
7	LAMBDAs	RateOfChange	This function calculates the rate of change given a begin amount and an end amount.	=LAMBDA(Begin, End, End/Begin-1)
8	LAMBDAs	RateOfChangeErrorM	This function calculates the rate of change given a begin amount and an end amount. This has a message if an argument is left empty.	=LAMBDA(Begin, End, IFS(AND(ISOMITTED(Begin), ISOMITTED(End)), "Enter Both Begin and End Amounts", ISOMITTED(Begin), "Please Enter Begin", ISOMITTED(End), "Please Enter End", TRUE, End/Begin-1))
9	LAMBDAs	RemoveChar	This function uses recursion to remove the specified digits from a text string.	=LAMBDA(Text, CharToRemove, IF(CharToRemove="", TRIM(Text), RemoveChar(SUBSTITUTE(Text, LEFT(CharToRemove), ""), RIGHT(CharToRemove, LEN(CharToRemove)-1))))
10	LAMBDAs	RemoveCharR	Remove specified characters listed in a single cell using REDUCE.	=LAMBDA(Text, CharactersToRemove, TRIM(REDUCE(Text, MID(CharactersToRemove, SEQUENCE(LEN(CharactersToRemove)), 1), LAMBDA(i, a, SUBSTITUTE(i, a, "")))))
11	LAMBDAs	RemoveNumbersFromText	This function will remove the digits 0,1,2,3,4,5,6,7,8,9 from a test string.	=LAMBDA(text, REDUCE(text, SEQUENCE(10,,0), LAMBDA(i,a, SUBSTITUTE(i, a, ""))))

No.	Type	Name	Description of Defined Name	Defined Name Definition

12	LAMBDAs	ShowFormulas	This function requires a worksheet range and the function will list all formulas from the range in a vertical array.	=LAMBDA(reference,
				IF(
				OR(ISFORMULA(reference)),
				TOCOL(ADDRESS(ROW(reference), COLUMN(reference), 4)&": "&FORMULATEXT(reference),2), ""))

13	LAMBDAs	SpaceBeforeCaps	This function will insert spaces before capital letters.	=LAMBDA(Text, TRIM(REDUCE(Text, CHAR(SEQUENCE(26, , 65)), LAMBDA(i,a, SUBSTITUTE(i, a, " "&a)))))
14	LAMBDAs	XYRegression10Stats	This function takes the ourput from the LINEST function and labels each statistical output and lists the labels and statistics in a vertical array.	=LAMBDA(Yvalues,Xvalues, LET(labels,{"Slope m";"Standard Error m";"R^2";"F"; "SS Regression";"Intercept b";"Standard Error b";"Standard Error y";"df";"SS Residual"}, stats,TOCOL(LINEST(Yvalues,Xvalues,1,1),,1),HSTACK(labels,stats)))

No.	Type	Name	Description of Defined Name	Defined Name Definition
15	**Formulas**	**FiscalQLookup Table**	Approximate Match Lookup Table to convert month number to fiscal quarter number where July 1 is first month in fiscal quarter.	={1,3;4,4;7,1;10,2}
16	**Formulas**	**TestScores**	Three Non-contiguous ranges with related test scores.	=DefinedNames!$J$7:$J$14, DefinedNames!$L$7:$L$14, DefinedNames!$N$7:$N$14
17	**Formulas**	**ZeroToTen**	Name creates a vertical array of numbers from 0 to 10.	=SEQUENCE(11, , 0)
18	**Ranges**	**Begin**	This is the Begin Amount that is related to an End Amount.	=DefinedNames!$B$7
19	**Ranges**	**End**	This is the End Amount that is related to an Begin Amount.	=DefinedNames!$C$7
20	**Ranges**	**Product**	Named field from sales table on the Defined Name worksheet.	=DefinedNames!$G$7:$G$19
21	**Ranges**	**SalesRep**	Named field from sales table on the Defined Name worksheet.	=DefinedNames!$F$7:$F$19
22	**Ranges**	**Units**	Named field from sales table on the Defined Name worksheet.	=DefinedNames!$H$7:$H$19