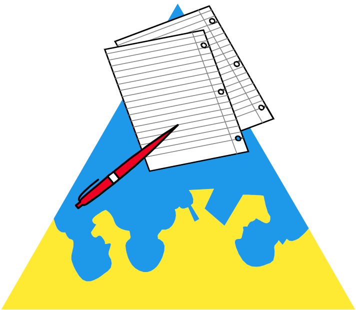
The Computer Workshop

A Professional Development Company

Excel Formulas

The Computer Workshop, Inc.

www.tcworkshop.com
training@tcworkshop.com



Microsoft Excel Formulas

Rel 6.0, 02/17/2020

Course Number: 0200-240-19-W
Course Release Number: 6.0
Software Release Number: 2019

February 24, 2020

Developed by:
Brian Ireson, Suzanne Hixon

Edited by:
Suzanne Hixon,, Thelma Tippie, and Jeffrey DeRamus

Published by:
The Computer Workshop, Inc.
5200 Upper Metro Place, Suite 140
Dublin, Ohio 43017
(614) 798-9505

dba

RoundTown Publishing
5200 Upper Metro Place, Suite 140
Dublin, Ohio 43017

Disclaimer:
Round Town Publishing produced this manual with great care to make it of good quality and accurate, and therefore, provides no warranties for this publication whatsoever, including, but not limited to, the implied warranties of merchantability or fitness for specific uses. Changes may be made to this document without notice.

Table of Contents

Lesson 1: Logical & Lookup Functions

Table of Contents
continued
,

Lesson 2: Complex Summing Functions

Lesson 3: Date & Time Functions

Table of Contents continued

,

Lesson 4: Formula Auditing

Using this Manual

Welcome to the *Excel Formulas* course. This manual and the data files are designed to be used for learning, review and reference after the class. The data files can be downloaded any time from *The Computer Workshop* website.

http://www.tcworkshop.com

There is no login or password required to access these files. You will also find handouts and supplementary materials on the website in the Download section.

To Download Data Files

Once on *The Computer Workshops* website, look at the bottom
of any page to find the link *Download*. Clicking this link opens the *Download* page where you can choose either *Data Files* or
Handouts.

1. ***Data Files*** opens a list of general application types.
2. Click once on the ***Microsoft Office Courses*** link.
3. Click once on the software related to the course.
4. Click once on the version related to the course.
5. If there are multiple folders, click on the ***TCW*** folder.
6. Click on the course name to download the data files.

You can choose to open or save the zipped folders

content to
your computer.

The handouts are in PDF format and also available to

you
without login or password. Simply open the PDF and either
print or save to your computer.

Conventions

Conventions Used in this Manual

The hands-on exercises (Actions) are written in a two-column format. The left column ("Instructions") gives numbered instructions, such as what to type, keys to press, commands to choose from menus, etc. The right column ("Results/ Comments"), contains comments describing results of, reasons for, quick keys, etc. for the instructions listed on the left.

Key names and Functions are bold and enclosed in square brackets:

[Enter], **[Tab]**, **[F5]**, **[F10]**

Keys you press simultaneously are separated by a plus (+) sign, typed in bold and enclosed in square brackets. You do not press the plus.

[Shift + F5]

Keys you press in sequence are separated by a space, bold and enclosed in square brackets.

[Home] [Down Arrow]

Ribbon tab names are in bold and italic: Example: ***Home***

Group names are in bold: Example: Font

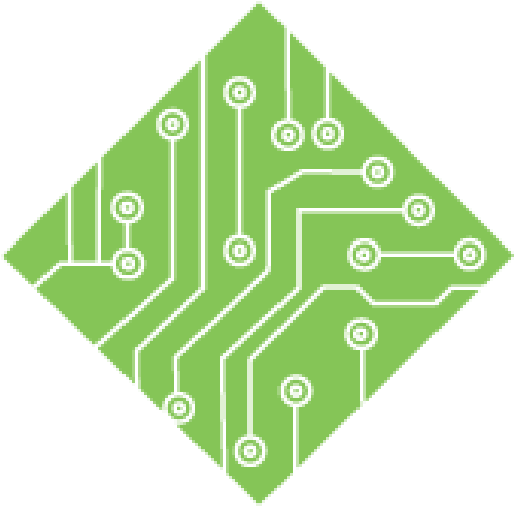
Dialog box names are in italic: Example: *Save As*

Button names are bold and enclosed in square brackets: Example: [Sort]

Information you are to type will be in bold. Example:

This is the first day of the rest of your life.

Information that you need to supply will be indicated with pointed brackets. Example: Type: <your name>.



Lesson 1: Logical & Lookup Functions

Lesson Overview

You will cover the following concepts in this chapter:

Logical Functions

Names

IF Functions

Nested IF Functions

IFS Functions

AND Functions

OR Functions

NOT Functions

Lookup Functions

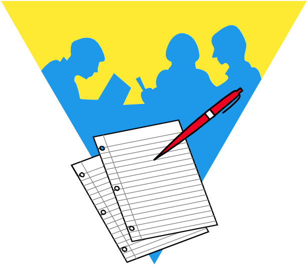
VLOOKUP Functions

HLOOKUP Function

Data Validation List

MATCH Function

INDEX Function



Lesson Notes

Logical Functions

A logical function is one that evaluates an expression or value and returns a value based upon whether the expression is True or False. These are formulas that are commonly used to see if a condition is met or not to determine one action or another.

You use a logical function to check a condition. For example, you wanted to determine if your actual expenses are over budget or determine if a client is eligible for a discount.

Note

If you are searching for text or mathematical expressions, remember to wrap the search values in quotations.

Testing whether conditions are true or false and making logical comparisons between expressions are common to many tasks. You can use the AND, OR, NOT, and IF functions to create conditional formulas.

In the **[Logical]** formulas drop-down button in the **Function Library Group** on the ***Formulas Tab*** you will notice TRUE and FALSE functions. These formulas don't require any arguments and will simply return the same value as the function. In essence you just as easily write True or False into the cell and be done with it.

=TRUE() returns True

=FALSE() returns False

Also within this group of functions you will find the **IF, AND**, **OR**, and **NOT** formulas. These can be used as stand alone functions and also in conjunction with an **IF** to expand the capabilities of the **IF** to much broader parameters.

Names

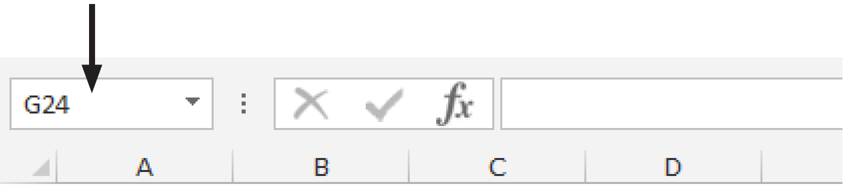
Naming Cells

Instead of referring to cell addresses in formulas it is easier to refer to the cell's name. This concept was introduced in the *Excel 2016 Level 2* book in Lesson 5. Therefore, this will be a short review of using Names in formulas.

Naming cells

Select the cell or cells you want to name.

Click into the **Name Box** and type in the name.



Remember not to use special characters in the name.

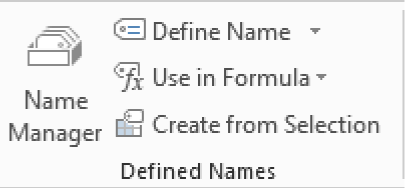
Do not use blank spaces.

Use underscores or camel-case multi-word names.

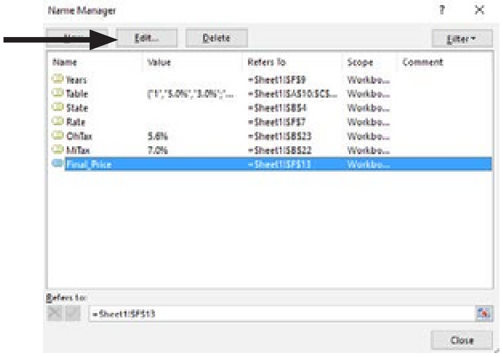
Press the **[Enter]** key or the name will not be applied.

Editing Names

Click the **[Name Manager]** button in the **Defined Names Group** on the ***Formulas Tab***



In the *Name Manager* dialog box, select the name that needs to be edited and click the **[Edit]** button.

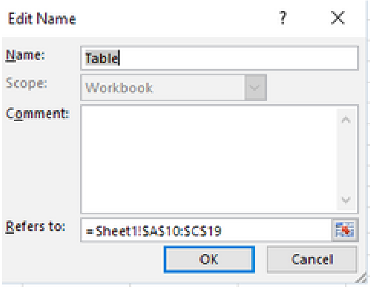


Name	Value	Refers To	Scope	Comment
Years		=Sheet1!$F$9	Workbo...	
Table	{"1","5.0%","3.0%";"...	=Sheet1!$A$10:C...	Workbo...	
State		=Sheet1!$B$4	Workbo...	
Rate		=Sheet1!$F$7	Workbo...	
OhTax	5.6%	=Sheet1!$B$23	Workbo...	
MiTax	7.0%	=Sheet1!$B$22	Workbo...	
Final_Price		=Sheet1!$F$13	Workbo...	

Refers to: =Sheet1!$F$13

Names,
continued

In the *Edit Name* dialog box.



Name: Table

Scope: Workbook

Comment:

Refers to: =Sheet1!$A$10:$C$19

Name: field you can fix typos or rename the cell or range.

Scope: filed is grayed out because that was defined when the name was created. When assigning names with the **Name Box** the scope is by default *global*.

Comment: field you can explain the name.

Refers to: field you can redefine the cell or cells addresses.

If it is only the **Refers to:** field that needs to be edited, then this can be done in the *Name Manager* dialog box.

Select the name to be edited.

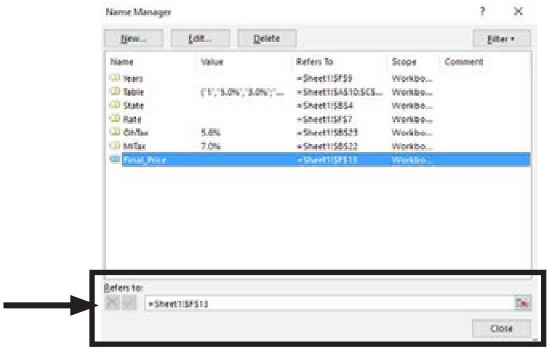
Click into the **Refers To:** field at the bottom of the dialog box.

Click the **[Expand/Collapse]** dialog button and highlight the correct range of cells.

Click the **[Expand/Collapse]** to redisplay the *Name Manager* dialog box.

Click the **[Check]** button to apply the edit. If you forget this last step the edit will not be applied.

Click the **[Close]** button to exit the *Name Manager* dialog box.



Name	Value	Refers To	Scope	Comment
Years		=Sheet1!$F$9	Workbo...	
Table	{"1","5.0%","3.0%","...	=Sheet1!$A$10:C...	Workbo...	
State		=Sheet1!$B$4	Workbo...	
Rate		=Sheet1!$F$7	Workbo...	
OhTax	5.8%	=Sheet1!$B$23	Workbo...	
MiTax	7.0%	=Sheet1!$B$22	Workbo...	
Final_Price		=Sheet1!$F$13	Workbo...	

Refers to: =Sheet1!$F$13

Action 1 – Naming Cells

Instructions:	Results/ Comments:
1. Open **Loan.xlsx**	The file is located in the Data Files folder
2. Save the file as **MyLoan**.	**[F12]**.
	Formulas can become quite complex. To aid you in writing the formulas, create the following range names.
3. Select cell **B6**.	This is the first cell to assign a name to.
4. Click into the Name Box and type: **State** then, press the **[Enter]** key.	The Name Box will now display the name **State** instead of the address **B6**.
5. Repeat steps 3 and 4 for the following names. **Range Range Name** **A13:C22** Table **B25** MiTax **B26** OhTax **I5** Rate **I7** Years	Cells that will be referred to in formulas are now named.
6. Click the drop-down arrow on the Name Box to see that all the names are listed.	If one or more of the names are not listed go back to the cell and apply the name.
7. Save the file.	**[Ctrl + S]**.

IF Functions

IF Functions

One of the most common logical functions is the **IF** function which performs a comparison, or test, and then displays the result in the active cell based on the outcome of that test. The **IF** function will return one value if a condition you specify is TRUE and another value if it is FALSE.

There are three arguments required in the function: **IF**

IF(logical_test,value_if_true,value_if_false)

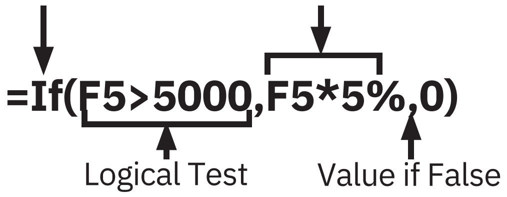
logical_test: isany value or expression that can be evaluated as true or false

value_if_true: is the value returned if the expression is true.

value_if_false: is the value returned if the expression is false.

For example, you need to determine if a salesperson has sold above a set goal in order to earn a bonus. The goal level is a value of $5000. If they sold more than that amount they will receive a bonus of 5% of their total sales; if not they will not receive their bonus. The sales total is in cell **F5** and the bonus calculation is in cell **G5**. The formula in cell **G5** would be as follows:

FunctionValue if True



If the salesperson's sales total is above 5000 then the sales total will be multiplied by 5% to determine the amount of the bonus. Should the sales total be less than 5000 then a zero value is returned.

Action 2 – IF Function

Instructions:	Results/ Comments:
1. **MyLoan** should still be open. If not, open it.	
2. Select cell **I9**.	This cell will show that appropriate sales tax based on which state is entered in cell **B6** (**State**).
3. Enter this formula. **=IF(State=”oh”,OhTax,MiTax)**	The first argument of this formula will determine if cell **B6** (**State**) contains the value Oh. Since Oh is text it must be wrapped in quotation marks. Should cell **B6** (**State**) contain Oh, the formula displays the second argument, **OhTax**; if not, then the third argument is returned, **MiTax**.
4. Click into cell **B6** (**State**).	This is the cell that is being checked by the **IF** function in cell **I9**.
5. Type in **Oh.**	The value in cell **I9** now shows the value from the cell named **OhTax**. The **IF** function Logical_test argument is not case sensitive.
6. Reselect cell **B6** (**State**).	You will change the state to change the sales tax to reflect this change.
7. Type in **Mi**.	The value in cell **I9** now shows the value from the cell named **MiTax**.
8. Save the file.	**[Ctrl + S]**.

Nested IF Functions

Note

A Primary **IF** can have up to seven **IF**s nested in a single formula.

Nested IF Statements

At times you will want to check for multiple conditions. This can be done in a single formula by nesting multiple **IF**'s inside the formula. The subsequent **IF**'s are placed in the *Value if False* part of the formula. Should the *Logical Test* return a *True* value then the formula stops calculating since an answer has been found. If the *Logical Test* returns a *False* value then the next **IF** function is run, this continues until either a *True* response is found or there are no other *Logical Tests* to run and the *False* is returned.

Example of a Nested **IF** statement:

$$\overbrace{\textbf{=IF(F5>5000,F5*5\%,}\underbrace{\textbf{IF(F5>2000,F5*3\%,0)}}_{\text{Nested IF Function}}\textbf{)}}^{\text{Primary IF Function}}$$

In this example the sales total is in cell **F5.** If the salesperson's sales total is greater than 5000 then the sales total will be multiplied by 5% to determine the bonus. If not, next **IF** is run to see if the sales total is greater than 2000. If this is *True,* then the sales total will be multiplied by 3%. If neither of these *Logical Tests* prove *True,* the formula will return a value of 0.

Action 3 – Nesting IF Functions

Instructions:	Results/ Comments:
1. **MyLoan** should still be open. If not, open it.	
2. Select cell **A27.**	We are going to add another state to the region of states we work in.
3. Type in **Indiana** and press the **[Tab]** key.	This is the additional state.
4. Select cell **B27**.	This cell will contain the sales tax rate for the state.
5. Type in **6%** and press the **[Enter]** key.	
6. Select cell **B27** and apply the name **InTax.**	Instead of using the cell address in the formula we want to continue using names.
7. Select cell **I9**.	The **IF** function needs to be edited to include the additional state.
8. Enter this formula. **=IF(State=”oh”,OhTax,** **IF(State=”mi”,MiTax,** **IF(State=”in”,InTax,** **“out of region”)))**	If **B6** (**State**) contains the value oh the formula returns the **OhTax** value. If not, then the *False* argument begins a new **IF** function, to a maximum of seven nested **IF**s. The final *False* in this case returns a text value of *out of region.* In the formula any text values must be wrapped in quotations marks.
9. Select cell **B6** (**State**).	To test the formula.
10. Type in **In** and press the **[Enter]** key.	The value in cell **I9** now shows the value from the cell named **InTax**.
11. Save and close the file.	**[Ctrl + S]** and **[Ctrl + W].**

IFS Functions

If the return value is text, it must be wrapped within quotation marks.

IFS Functions

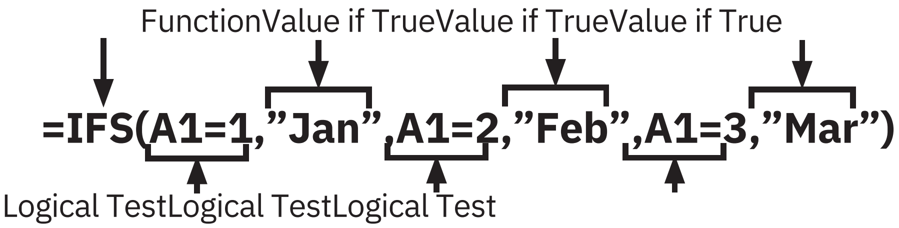
While using nested IF functions is very useful, the limit of seven nested functions may not offer enough options to fulfill your needs. In the latest version of Excel 2019/ 365 a new function called IFS has been added, this function allows up to 127 different conditions in a single formula.

There are two basic arguments required in the **IFS** function:

logical_test: is any value or expression that can be evaluated as true or false

value_if_true: is the value returned if the expression is true.

The syntax of the formula is as follows:



FunctionValue if TrueValue if TrueValue if True

=IFS(A1=1,”Jan”,A1=2,”Feb”,A1=3,”Mar”)

Logical TestLogical TestLogical Test

Each subsequent Logical Test and Value if True adds anther conditional check and return option.

Action 4 – IFS Function

Instructions:	Results/ Comments:
1. Open the **Logical Formulas** file.	
2. Save the file as **My Logical Formulas.**	
3. Click on the ***IFS Function*** sheet tab.	This is the first sheet in the workbook.
4. Select cell **A2**.	This will be the cell that tells us what month is entered in cell A1.
5. Enter this formula, **=IFS(A1=1,”Jan”,A1=2,”Feb”, A1=3,”Mar”,A1=4,”Apr”, A1=5,”May”, A1=6,”Jun”,A1=7,”Jul”, A1=8,”Aug”, A1=9,”Sep”,A1=10,”Oct”, A1=11,”Nov”,A1=12,”Dec”)**	Unlike a nested IF, the IFS function allows up to 127 conditional to be checked within the formula.
6. Select cell **A1.**	This is the cell that the IFS formula is checking.
7. Enter any value between 1 to 12.	As a value is entered the corresponding month is returned in cell **A2**.
8. Save the file and leave it open.	**[Ctrl + S]**.

AND Functions

There are times when you will need to check multiple cells to see if they meet a set of parameters and return either a *True* or *False* answer. If all the conditions are met then the formula will return a *True* value. If any of the conditions are not met then the formula returns a *False* value. One common use for the **AND** function is to expand the usefulness of other functions that perform logical tests. For example, the **IF** function performs a *Logical Test* and then returns one value if the test evaluates to *TRUE* and another value if the test evaluates to *FALSE*. By using the **AND** function as the *Logical Test* argument of the **IF** function, you can test many different conditions instead of just one.

The **AND** function syntax has the following arguments:

logical1: The first condition that you want to test that can evaluate to either a TRUE or FALSE value. A required argument.

logical2: Additional conditions that you want to test which can evaluate to either a TRUE or FALSE value. An optional argument, there can be up to a maximum of 255 conditions.

The **AND** function is written like this:

FunctionLogical Test 2

=AND(F5>5000, A5=10)

Logical Test 1

If the value in cell **F5** is greater than 5000 and the value in cell **A5** is equal to 10 then the formula will return a *True* value. If either of the values does not match the parameters of the *Logical Tests* in the **AND** function then the formula will return a *False* value.

Using an AND in an IF Function

To perform multiple *Logical Tests* in an **IF** function, use the **AND** function as the *Logical Test* of the **IF**. In the example below, if both conditions of the **AND** are true, then the **IF** returns the *Value If True.* Should either of the **AND's** *Logical Tests* not be true, then the **IF** will return the *Value If False*.

Primary IF formula Value IF False

=IF(AND(F5>5000,A5=10),F5*5%,0)

Nested AND formula Value IF True

Action 5 – AND Functions

Instructions:	Results/ Comments:
1. **My Logical Formulas** should still be open. If not, open it.	
2. Click on the ***Summary*** sheet tab.	
3. Select cell .**F4**	We are going to see if two conditions are met in order to get a *TRUE* return.
4. Enter this formula, **=AND(D4>1800,E4=”yes”)**	This formula is to check if the commission earned is above $1800 and if the content in cell **E4** is **Yes**. If both conditions are met then the formulas will return a *TRUE* value.
5. Using autofill, fill cells **F5:F14**.	There should a mix of *TRUE* and *FALSE* values in the cells.
6. Select cells **F4:F14**.	We are going to edit all the cells.
7. Click into the formula bar to edit the formula.	Editing in the formula bar will allow us to apply the edit to all the selected cells simultaneously by pressing **[Ctrl + Enter]**.
8. Enter this formula **=IF(AND(D4>1800,E4=”yes”), “Gold”,IF(AND(D4>1600,E4=”Yes”), “Silver”,”Not Eligible”))** then, press **[Ctrl + Enter].**	This nested formula uses two **AND’**s in two **IF’**s to check for multiple variables to determine which membership level should be returned.
9. Save the file.	**[Ctrl + S]**.

OR Functions

OR Functions

While the **AND** function requires that all *Logical Tests* be true to return a *TRUE* value, the **OR** function allows any of the *Logical Tests* to return a true value for the formula to return a *TRUE* value. You may have a situation where multiple conditions must be checked. If any one is true, then the formula will return a *TRUE* value. Use the **OR** function, one of the logical functions, to return a TRUE if any argument is TRUE. It returns *FALSE* if all arguments are false.

The **OR** function syntax uses the following arguments:

logical1: The first condition that you want to test that can evaluate to either TRUE or FALSE. A required argument.

logical2: Subsequent logical values are optional. 1 to 255 conditions you want to test that can be either TRUE or FALSE.

The **OR** function is written like this,

$$\overset{\text{Function}}{=\text{OR}}(\underbrace{\text{F5>5000}}_{\text{Logical Test 1}},\overbrace{\text{A5=10}}^{\text{Logical Test 2}})$$

In this example, if either condition is true, the formula will return a *TRUE* value. Both conditions would have to be false for the formula to return a *FALSE* value.

Using an OR in an IF Function

This again can be nested in an**IF** function to expand the scope of the **IF** function

$$\overbrace{=\text{IF}(\underbrace{\text{OR(F5>5000,A5=10)}}_{\text{Nested OR formula}},\underbrace{\text{F5*5\%}}_{\text{Value If True}},\overbrace{0}^{\text{Value If False}}}^{\text{Primary IF formula}})$$

The **IF** formula will return the *Value If True* if either one of the **OR** formulas *Logical Tests* conditions is met. Should both the **OR** *Logical Tests* conditions not be met, then the **IF** will return the *Value If False*.

Action 6 – OR Functions

Instructions:	Results/ Comments:
1. **My Logical Formulas** should still be open. If not, open it.	
2. Select cell **G4** on the ***Summary Tab.***	This is the cell where we will determine if this salesperson has exceeded a $10,000 total in any quarter, which will entitle them to a higher bonus.
3. Enter this formula **=OR(** Click on sheet ***QT1*** and highlight cell **F4** and type: **>10000** press the **[Comma]** key Click on sheet ***QT2*** and highlight cell **F4** and type: **>10000** press the **[Comma]** key Click on sheet ***QT3*** and highlight cell **F4** and type: **>10000** press the **[Comma]** key Click on sheet ***QT4*** and highlight cell **F4** and type: **>10000** then, press the **[Enter]** key	When finished the formula will read as **=OR('QT1'!F4>10000,'QT2'!F4>100 00, 'QT3'!F4>10000,'QT4'!F4>10000).** If any of these *Logical Tests* prove true then the formula returns a *TRUE* value.
4. Use the autofill to put the formula into cells **G5:G14**.	There should a mix of *TRUE* and *FALSE* values in the cells.
5. Select cell **G4** again.	We will now nest the OR function in an IF to determine the bonus level they earned.
6. Double-click into the cell.	To edit the formula in the cell instead of the formula bar.
7. Add these changes to the formula =**IF(**OR('QT1'!F4>10000,'QT2'!F4>10000, 'QT3'!F4>10000,'QT4'!F4>10000)**, C4*0.07,C4*0.03)** .	Add only the bold text to the formula.
8. Use the autofill to put the formula into cells **G5:G14**.	If any of the sales team sold more than $10,000 in a quarter they receive a 7% bonus otherwise they receive a 3% bonus.
9. Save the file.	**[Ctrl + S]**.

NOT Functions

NOT Functions

Use **NOT** functions when you want to make sure a value is not equal to one particular value since this function reverses the value of its argument. If the *Logical Test* is *FALSE*, **NOT** returns a *TRUE* value; if the ***Logical Test*** is *TRUE*, **NOT** returns a *FALSE* value. Unlike the **AND** or **OR** functions which can contain multiple *Logical Tests*, the **NOT** only contains a single *Logical Test*.

The **NOT** function syntax has the following arguments:

Logical Test: A value or expression that can be evaluated as TRUE or FALSE. A required argument.

The **NOT** function is written like this:

Function

=NOT(F5>5000)

Logical Test

If the value in cell **F5** is less than 5000 then the **NOT** function will return a *TRUE* value. Should the value in cell **F5** prove to be greater than 5000 then the **NOT** function will return a *FALSE* value.

Using a NOT Function in an IF Function

Use this combination when you are making sure a parameter you define is not met in order to get a positive result. Once the NOT logical test is met, the IF function will return the Value if False argument of the IF formula.

Primary IF formulaValue IF True

=IF(NOT(F5>5000),0,F5*5%)

Nested NOT formulaValue IF False

In the example above, until the value in cell **F5** is above 5000 the NOT formula will return a True result and return the Value If True argument of the IF. As the value in **F5** is greater than 5000 the NOT returns a False value and in turn the IF returns the Value if False argument of the IF statement. So you will know when the value in **F5** is greater than 5000 when the cell containing the formula begins returning values instead of 0.

Action 7 – NOT Functions

Instructions:	Results/ Comments:
1. **Logical Formulas** should still be open. If not, open it.	
2. Select cell **H4**.	We now want to see if the salesperson should be under review.
3. Enter this formula **= NOT(F4=”Gold”)**.	If cell **F4** contains the word Gold then the **NOT** formula returns a *FALSE*, for any other content in cell **F4** the formula returns a *TRUE*.
4. Use the autofill to put the formula into cells **H5:H14**.	Only Gold level members should have *FALSE* values. These people do not require reviews.
5. Select cell H4.	We will modify the formula to return a more easily understood response of *YES* or *NO*.
6. Add these changes to the formula =**IF(**NOT(F4=”Gold”)**,”Yes”,”No”**)	Add only the bold text to the formula.
7. Use the autofill to put the formula into cells **H5:H14**.	Now only the Gold level members have a *NO* value in the review column, all others have *YES* values.
8. Save the file.	**[Ctrl + S]**.

Lookup Functions

There are many times where you will need to search for specific data in the worksheet or workbook and the filters will not do. Using lookup functions allows you to reach across worksheets to look for the data you need. You can also add a data validation list to the lookup cell to make the search much more efficient.

There are several lookup functions available to you

Vlookup: A vertical lookup, searches for a value in the first column of a table and returning the value in the same row in the column index number position in the table.

Hlookup: A horizontal lookup, searches for a value in the first row of a table, and returns another value from the same column and row index number position in that table.

Match: Searches for a specified item in a range of cells, and then returns the relative position of that item in that range.

Index: Returns a value or the reference to a value from within a table or range. There are two forms of the INDEX function: the array form and the reference form.

Array: Returns the value of an element in a table or an array, selected by the row and column number indexes. Use the array form if the first argument to INDEX is an array constant.

Reference: Returns the reference of the cell at the intersection of a particular row and column. If the reference is made up of nonadjacent selections, you can pick the selection to look in.

The **MATCH** and **INDEX** functions, when used together can allow you to search a table like a **VLOOKUP** or **HLOOKUP** function when the lookup value is not held in the first column or row of a table or array.

VLOOKUP Functions

VLOOKUP Function

The **VLOOKUP** function is used to look up or search for a value in a table. The **VLOOKUP** function will look at specified data in the first column of a table of data. Once found, it will return a result that is on the same row a specified number of columns from the first column. When using this function, the values in the first column of the table must be in ascending order. For example, you can search for the last name of a student in a list and then return a phone number from another cell on the same row.

The **VLOOKUP** function syntax uses the following arguments:

Lookup_Value—the **Search Value**

Table_Array—the **Range** of the Lookup Table

Col_Index_Num—an **Offset** to tell Excel how many columns to the right of the first column in the table it must look to find the result.

Range_Lookup—this is optional. This should be used only when searching for a text string and not a number. If you place the word "False", this will tell it to find an exact match.

Note

Naming the range of cells that make up the Table_Array will make lookup formulas much easier to both write and read.

The function is written as **VLOOKUP :**

=VLOOKUP(Lookup_Value,Table_Array,Col_Index_Num)

The *Lookup Value* way this works is that the is compared to the first column of the *Table Array* this determines which row of the Table Array will contain the desired information. Once that is established the specifies *Col Index NumTable* what column of the *Array* will contain the desired information. The contents of the cell at the intersection of the row and column will be the value returned by the formula.

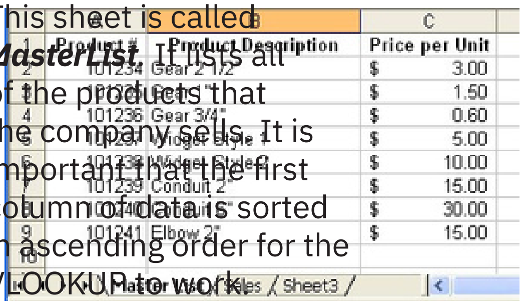
One thing to remember is that the *Col Index Num* argument in the formula refers only to the table and not the worksheet, therefore it doesn't matter wherever the table is located.

VLOOKUP Functions, continued

Example of VLOOKUP

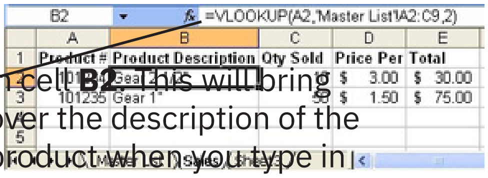
The example below shows how you can use the formula to look up a description of a product on another sheet.

This sheet is called ***MasterList***. It lists all of the products that the company sells. It is important that the first column of data is sorted in ascending order for the VLOOKUP to work.



	A	B	C
1	Product #	Product Description	Price per Unit
2	101234	Gear 2 1/2	$ 3.00
3	101235	Gear 1"	$ 1.50
4	101236	Gear 3/4"	$ 0.60
5	101237	Widget Style 1	$ 5.00
6	101238	Widget Style 2	$ 10.00
7	101239	Conduit 2"	$ 15.00
8	101240	Conduit 3"	$ 30.00
9	101241	Elbow 2"	$ 15.00
10			

This sheet is called ***Sales***. The VLOOKUP formula is placed in cell **B2**. This will bring over the description of the product when you type in the account code in cell **A2**. This is a great function to use because you will not have to type out the product description. It will be brought in automatically from the ***MasterList***. You can also use the formula to automatically bring over the Price Per Unit and place it in Column D.



B2 =VLOOKUP(A2,'Master List'!A2:C9,2)

	A	B	C	D	E
1	Product #	Product Description	Qty Sold	Price Per	Total
2	101234	Gear 2 1/2	10	$ 3.00	$ 30.00
3	101235	Gear 1"	50	$ 1.50	$ 75.00
4					
5					

Note

Vlookup formula will look for Product # 101234 in the Master List Sheet and places the Product description in cell **B2** in the sales sheet.

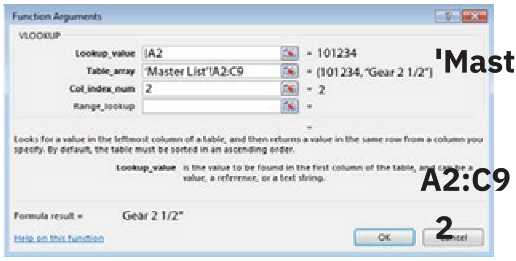
The formula in the above example is:

=VLOOKUP(A2,'Master List'!A2:C9,2)

A2 — this is the Lookup value. This is the value that you are searching for (in this case the product #).

'Master List'!A2:C9 —this is the Table Array. It is the range of the Lookup Table (in this case, it is located on the worksheet ***Master List*** with the data range being **A2:C9**).

2 —this is the Col Index Number. This tells *Excel* what column to the right of the first column in the table array you want to extract information from (in this case, it is in the second column).



Function Arguments

VLOOKUP

Lookup_value A2 = 101234

Table_array 'Master List'!A2:C9 = {101234,"Gear 2 1/2"}

Col_index_num 2 = 2

Range_lookup =

Looks for a value in the leftmost column of a table, and then returns a value in the same row from a column you specify. By default, the table must be sorted in an ascending order.

Lookup_value is the value to be found in the first column of the table, and can be a value, a reference, or a text string.

Formula result = Gear 2 1/2"

This is how the formula would be entered in the *Function Arguments* dialog box.

Action 8– Vlookup Function

Instructions:	Results/ Comments:
1. Open the **MyLoan** file.	
2. Select cell **I5** (**Rate**).	This is where the interest rate will be displayed. Two factors must be considered to find the correct rate, the state and number of years the loan is for. We will use the **VLOOKUP** formula to find which row in the Interest Table contains the correct rate based on the number of years. Since there are three different columns to possibly choose from depending on which state the loan is being issued in, it will be necessary to nest an **IF** within the **VLOOKUP**.
3. Enter this formula **=VLOOKUP(Years,Table,IF(State=”OH”, 3,2))**	At this point the formula can only return a value if the state is equal to either Oh or Mi, since the table name refers to cells **A13:C22**. The values for Indiana need to be added to the Interest table and its name redefined.
4. The formula returns a **#NA**.	This is because there currently is no value in cell **I7 (Years)** for the formula to find.
5. Select cell **B6** (**State**) and enter **<Oh>**.	To set a value that the formula can find.
6. Select cell **I7** (**Years**).	
7. Enter a number between 1 to 10.	This will complete adding values that the VLOOKUP needs and the formula in **I5** (**Rate**) now returns a value.
8. Select cell **D12**, and type: **Indiana**	The header for this column is added. We will begin adding the values for Indiana’s interest rates.
9. In cell ,**D13** type: **4.25%** then, press **[Enter]**.	This is the first value.

Action 8– Vlookup Function, continued

Instructions:	Results/ Comments:
10. In cell type: **D14, 4.75%** then, press . **[Enter]**	This is the second value.
11. Select **D13:D14** cells and use the autofill to populate values to cell . **D22**	Selecting both cells with values establishes the series that autofill will use to populate the remaining cells.
12. Select cell **E13** and enter **Out of Region** Autofill it down to cell **E22**.	This will be the return if no listed state is found by the formula.
13. Click the **[Name Manager]** button in the **Defined Names Group** on the ***Formulas Tab***.	At this point, we will redefine the **Table** name to include the newly added data. The *Name Manager* dialog box opens.
14. Select the **Table** name and click the **[Edit]** button.	The *Edit Name* dialog box opens for the selected name.
15. Click into the **Refers to:** field and change the **$C$22** reference to **$E$22**. Click the **[OK]** button.	By changing the cell address from **C22** to **D22,** we have expanded the reference to include the new data.
16. Click the **[Close]** button.	The *Name Manager* dialog box is closed and the name has been updated.
17. Select cell **I5** (**Rate**).	We will now edit the formula.
18. Add these changes to the formula =VLOOKUP(Years,Table,IF(State=”OH”, 3,**IF(State=”Mi”,2,IF(State=”IN”,4,5)**)).	Add only the bold text to the formula.
19. Change the state to either **Mi** or **In** to test the VLOOKUP formulas functionality.	The results in cell **I5 (Rate)** should change according to the state entered. You may need to increase the number of decimal places for this cell.
20. Change the value in the State cell to**MA**	This state is not part of the table and will return an “Out of Region” in the formula.
21. Save the file.	**[Ctrl + S]**.

HLOOKUP Function

HLOOKUP Function

The **HLOOKUP** function is used to look up or search for a value in a table. The **HLOOKUP** function will look at specified data in the first row of a table of data. Once found, it will return a result that is on the same column a specified number of rows from the first row.

The **HLOOKUP** function syntax uses the following arguments:

Lookup_Value—the **Search Value**

Table_Array—the **Range** of the Lookup Table

Row_Index_Num—an **Offset** to tell *Excel* how many rows down from the first row of the table it must look to find the result.

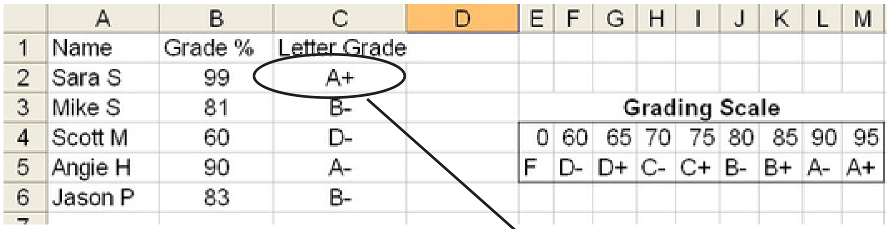
Range_Lookup—This is optional and should be used only when searching for a text string, not a number. If you place the word "False", this will tell it to find an exact match.

The function is written as:

HLOOKUP

=Hlookup(Lookup _Value,Table_Array,Row_Index_Num)

Look at the spreadsheet below, in this example we are trying to lookup the letter grade for each student. The formula in **C2** is **=HLOOKUP(B2,$E$4:$M$5,2)**. *Excel* will look in Column B for the Grade percentage then look in the first row of the table **(E4:M5)** and then place the grade that appears in the second row of the table into Column C.



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Name	Grade %	Letter Grade										
2	Sara S	99	A+										
3	Mike S	81	B-		Grading Scale								
4	Scott M	60	D-		0	60	65	70	75	80	85	90	95
5	Angie H	90	A-		F	D-	D+	C-	C+	B-	B+	A-	A+
6	Jason P	83	B-										

The Formula for HLookup is in cell . **C2 =HLOOKUP(B2,$E$4:$M$5,2)**

Notice how if the value is not found exactly, *Excel* knows that it is within the range. For example, it wasn't able to find the value 99, but it still brought back a letter grade of "A+". *Excel* was able to determine that it was within a range and any number within that range will return a certain letter grade.

Action 9– Hlookup Function

Instructions:	**Results/ Comments:**
1. Open the **Discount** file.	We will be using the to**HLOOKUP** compare the sales total to a table that outlines the discounts to determine how much of a discount the sale is eligible for.
2. Save the file as **My Discount**	
3. Click the ***Discount Table*** sheet.	This is the sheet that has the discount table data.
4. Select cells **A2:F3**	
Name the range **Table**.	Use the **Name Box** to apply the name to the cell range.
6. Click the ***Sales*** sheet.	This is the sheet that we need the **HLOOKUP** formula on.
7. Select cell **I2**.	This is the cell that requires the formula to see what discount is to be applied to the sale.
8. Enter this formula **=HLOOKUP(H2,Table,2)**.	The value in **H2** will be compared to the first row of the Table and when a match is found the value from the second row will be returned.
9. Double-click the Autofill handle.	Since the formula is in a cell directly adjacent to the data, the Autofill runs down the column up to the first blank row is encountered.
10. Select cell **J2** and enter the following formula **=H2-(H2*I2)** .	Now we need to calculate what the discounted price will be.
11. Select cell **J2,** Autofill and format the column for currency.	Using the Autofill completes this column of data in the worksheet.
12. Save the file.	**[F12]**.

Data Validation List

Data Validation List

If you want to be able to select what you are looking for from a list, then this feature will prove invaluable. This tool does not filter the data in any way, but will populate the cell with choices made from the list and use only unique values in the list. The cell can be referred to as the *Lookup Value* argument of a **VLOOKUP** or any other formula that uses a *Lookup Value* argument, which makes the retrieval of data much simpler.

Creating a Data Validation List

Click the **[Data Validation]** button in the **Data Tools Group** on the ***Data Tab.***

In the *Data Validation* dialog, select the ***Settings*** tab.

Click the drop-down for the **Allow** field and choose *List*.

Click into the **Source** field. There are two ways that a list can be entered: by selection or manually typing it in.

Selection: click the **[Expand/Collapse]** button at the right edge of the field. Then select the cells that contain the list entries. Click the **[Expand/Collapse]** button again to redisplay the *Data Validation* dialog box.

Manual Entry: Type each list entry. Separate each entry with a comma.

While it is not necessary, you can also click the ***Input Message*** tab and add a message that will be displayed when the cell is selected.

If you do not want to allow an entry that is not on the list, click the ***Error Alert*** tab. Set the **Style** field to ***Stop***. While it is not necessary, you can add a message that will be displayed if invalid data is entered.

Click **[OK]**.

Action 10 – Setting Up Drop-down Lists

Instructions:	Results/ Comments:
1. **My Discount** should still be open. If not, open it.	We will now search through a much larger list of possible discount variables.
2. Click on the ***Advanced Search*** sheet tab and select cell **D1**.	To make this much easier to find, we will now add drop-down lists to ensure that users will search for only valid data.
3. Click the **[Data Validation]** button in the **Data Tools Group** on the ***Data Tab***	The *Data Validation* dialog box opens. We will not worry about the ***Input Message*** and ***Error Alert*** tabs.
4. In the ***Settings*** tab of the *Data Validation* dialog box, choose *List* from the **Allow:** drop-down.	This is the option that allows us to create drop-down lists.
5. Click into the **Source**: field Select cell **A8**, and hold both the **[Ctrl + Shift]** keys Tap the down arrow key once, then click the **[OK]** button.	Since the worksheet has all the entries already listed, we can use those cell addresses.
6. Select cell **D2**.	We will now get the drop-down list of the Percentages using the same technique as above.
7. Click the **[Data Validation]** button in the **Data Tools Group** on the ***Data Tab***.	
8. In the ***Settings*** tab of the *Data Validation* dialog box, choose *List* from the **Allow:** drop-down.	
9. Click into the **Source**: field, Select cell **B7**, and hold both the **[Ctrl + Shift**] keys Tap the right arrow key once, then click the **[OK]** button.	
10. Test the drop-down lists, then save the file.	**[Ctrl + S]**.

MATCH Function

MATCH Function

When the *Lookup Value* you are searching for is not held in the first column or row of a *Table Array*, the **MATCH** function will search for a specific value within a range of cells and return the position of the *Lookup Value* within the range.

Use **MATCH** instead of one of the **LOOKUP** functions when you need the position of an item in a range instead of the item itself. For example, you might use the **MATCH** function to provide a value for the row_num argument of the **INDEX** function.

The **MATCH** function syntax has the following arguments:

lookup_value: A required argument. The value that you want to match in the *Lookup_array*. For example, when you look up someone's number in a telephone book, you are using the person's name as the lookup value, but the telephone number is the value you want.

The *Lookup_Value* argument can be a value (number, text, or logical value) or a cell reference to a number, text, or logical value.

Lookup_Array: A required argument. The range of cells being searched.

Match_Type An optional argument. The number ***-1***, ***0***, or ***1***. The *Match_Type* argument specifies how *Excel* matches the *Lookup_Value* with values in the *Lookup_Array*. The default value for this argument is ***1***.

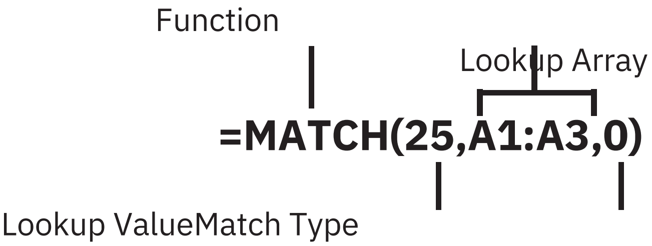
1= finds the largest value that is less than or equal to the *Lookup_Value*. The values in the *Lookup_Array* argument must be placed in ascending order.

0= finds the first value that is exactly equal to the *Lookup_Value*. The values in the *Lookup_Array* argument can be in any order.

-1= finds the smallest value that is greater than or equal to the *Lookup_Value*. The values in the *Lookup_Array* argument must be placed in descending order.

MATCH Function, continued

The **Match**function is written like this;



Function

Lookup Array

=MATCH(25,A1:A3,0)

Lookup ValueMatch Type

In this example, the range **A1:A3** contains the values 5, 25, and 38 respectively. The formula returns the number 2, because 25 is the second item in the range.

Action 11 – Creating a Match Function

Instructions:	Results/ Comments:
1. **My Discount** should still be open. If not, open it.	
2. Select cell **H1**.	We want to find out what row contains the value we are looking for.
3. Enter the following formula, **=MATCH(D1,A8:A34,0)** .	The formula will first check to see what value is in cell **D1**. Then it will take that value and compare it to what is in the cell range, **A8:A34**. Since we are searching for exact matches, the third element of the formula is set to **0**.
4. Select cell **D1** and choose any value from the drop-down list.	As the value changes, our formula tells us what row number we can find that value in.
5. Select cell **H2**.	This time we will be looking for the column number that holds the percentage.
6. Enter the following formula, **=MATCH(D2,B7:V7,0)** .	The formula will first check to see what value is in cell **D2**. Then it will take that value and compare it to what is in the cell range, **B7:V7**. Since we are searching for exact matches, the third element of the formula is set to **0**.
7. Select cell **D2** and choose any value from the drop-down list.	As the value changes, our formula tells us what column number we can find that value in.
8. Save the file.	**[Ctrl + S]**.

INDEX Function

INDEX Function

The **INDEX** function can return a value or the reference to a value from within a table or range. You can refer to both the column and/or row within the cell range. You can use the results of a **MATCH** function as the Lookup Value to pull data from another column along the row defined by the **MATCH**.

There are two forms of the **INDEX** function: the array form and reference form.

INDEX Array Form

Returns the value of an element in a table or an array, selected by the row and column number indexes.

Use the array form if the first argument to **INDEX** is an array constant.

The **INDEX** function syntax has the following arguments:

Array: A required argument. The range of cells or a constant array of cells that are being searched.

If the array contains only one row or column, the corresponding *Row_num* or *Column_num* argument is optional.

If array has more than one row and more than one column, and only *Row_num* or *Column_num* is used, **INDEX** returns an array of the entire row or column in array.

Row_num: A required argument. Selects the row in the array from which to return a value. This can be considered the *Lookup Value* that is being searched for within the *Array.* You can refer to the cell containing the **MATCH** function. If *Row_num* is omitted, *Column_num* is required.

Column_num: An optional argument. Selects the column in the array from which to return a value. This can be considered the *Lookup Value* that is being searched for within the *Array.* You can refer to the cell containing the **MATCH** function. If *Column_num* is omitted, *Row_num* is required.

INDEX Function,
continued

The **INDEX** function is written like this:

=INDEX(array, row_num, [column_num])

If both the *Row_num* and *Column_num* arguments are used, **INDEX** returns the value in the cell at the intersection of *Row_ num* and *Column_num*.

Note

Row_num and *Column_num* must point to a cell within array; otherwise, **INDEX** returns the *#REF*! error value

INDEX Reference form

Returns the reference of the cell at the intersection of a particular row and column. If the reference is made up of nonadjacent selections, you can pick the selection to look in.

The **INDEX** function syntax has the following arguments:

Reference: A required argument. A reference to one or more cell ranges, defines the area that contains the raw data to search through.

If you are entering a nonadjacent range for the reference, enclose reference in parentheses. =**INDEX((reference,reference),row_num)**

If each area in reference contains only one row or column, the *Row_num* or *Column_num* argument, respectively, is optional. For example, for a single column reference, use =**INDEX(reference,column_num)**.

Row_num: A required argument. The number of the row that contains the data wanted from within the referenced range of cells.

Column_num: An optional argument. The number of the column in reference that contains the data wanted from within the referenced range of cell.

Area_num: An optional argument. If there is more than one reference in the formula, this defines which of the references from which to return the intersection of *Row_num* and *Column_num*. If *Area_num* is omitted, **INDEX** uses area 1.

For example, if Reference describes the cells (**A1:B4,D1:E4,G1:H4**)

Area_num 1 is the range **A1:B4**
Area_num 2 is the range **D1:E4**
Area_num 3 is the range **G1:H4**.

INDEX Function, continued

This type of **INDEX** function is written like this:

=INDEX(reference, row_num, [column_num], [area_num])

After *Reference* and *Area_num* have selected a particular range, *Row_num* and *Column_num* select a particular cell. *Row_num* 1 is the first row in the range, *Column_num* 1 is the first column, and so on. The reference returned by **INDEX** is the intersection of *Row_num* and *Column_num*.

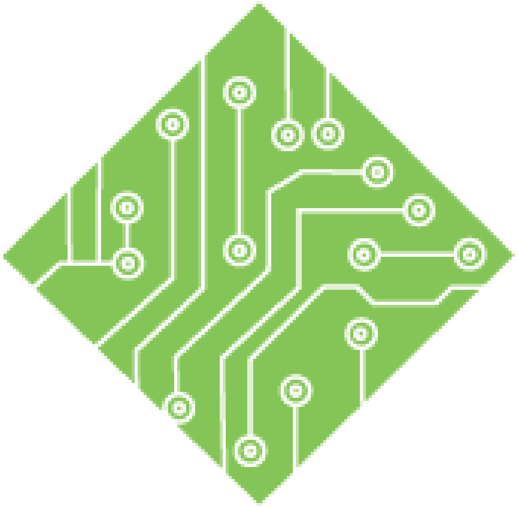
If you set *Row_num* or *Column_num* to 0 (zero), **INDEX** returns the reference for the entire column or row, respectively.

Row_num, *Column_num*, and *Area_num* must point to a cell within *reference*; otherwise, **INDEX** returns the **#REF**! error value. If *Row_num* and *Column_num* are omitted, **INDEX** returns the area in reference specified by *Area_num*.

The result of the **INDEX** function is a reference and is interpreted as such by other formulas. Depending on the formula, the return value of **INDEX** may be used as a reference or as a value. For example, the formula **CELL("width",INDEX(A1:B2,1,2))** is equivalent to **CELL("width",B1)**. The **CELL** function uses the return value of **INDEX** as a cell reference. On the other hand, a formula such as **2*INDEX(A1:B2,1,2)** translates the return value of **INDEX** into the number in cell **B1**.

Action 12 – Creating an INDEX Function

Instructions:	Results/ Comments:
1. **My Discount** should still be open. If not, open it.	
2. Select cell **H3**.	Now we want to look for the value in the cell that lies at the intersection of the row and column numbers we found using the **MATCH** functions.
3. Enter the following formula, **=INDEX(B8:V34,H1,H2)** .	This formula first defines the area where we are looking for data. Then the row number that is being returned by the **MATCH** in cell **H1**. The last element of this formula is the column number which is again being returned by the **MATCH** in cell **H2**.
4. Try changing the values in cells **D1** and **D2**.	Both the **INDEX** and **MATCH** functions are updated based on the choices made from the drop-down lists in cells **D1** and **D2**.
5. Select cell **C4**.	Now, we will nest **MATCH** functions inside and **INDEX** to accomplish the same result in a single cell with a single formula.
6. Enter the following formula, **=INDEX(B8:V34, MATCH(D1,A8:A34,0), MATCH(D2,B7:V7,0))** .	In this **INDEX** formula, a **MATCH** formula is replacing the row and column elements of the formula. We are running multiple checks in one place.
7. Save and close the file.	**[Ctrl + S]** and **[Ctrl + W]**.



Lesson 2: Complex Summing Functions

Lesson Overview

You will cover the following concepts in this chapter:

Filtering Data for Unique Values

SUMIF Functions

AVERAGEIF Function

COUNTIF Functions

SUMIFS Functions

COUNTIFS Functions

MAXIFS and MINIFS Functions

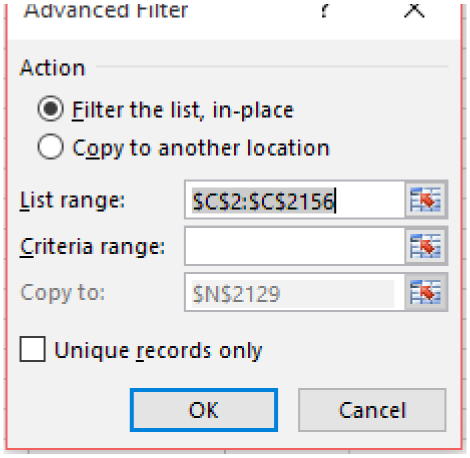
Lesson Notes

Filtering Data for Unique Values

Using the Advanced Filter

When starting to work on a new file, the data may come to you as a CSV file or simply as raw data in an *Excel* file. In this case, you may want to extract information out of the raw data to format information that you need. Instead of searching for individual records then copying and pasting them one at a time, it is more effective to spend a little time preparing the file and using the Advanced Filter.

Once the file is open, click on the ***Data Tab*** and look in the **Sorting & Filtering Group** of commands for the **[Advanced]** button. Once the button is clicked the *Advanced Filter* dialog box will open.


Advanced Filter

Action

● Filter the list, in-place

○ Copy to another location

List range: $C$2:$C$2156

Criteria range:

Copy to: $N$2129

□ Unique records only

This dialog box allows you to choose the range of cells to filter for specific items and either filter the data in place or place the filtered list in a location of your choosing. It can also find unique values in the define list range.

The following buttons allow you to choose how the results of the filter will be handled.

If *Filter list in place* is selected, then the **Copy To:** field is grayed out.

If *Copy to another location* is selected, then the **Copy to:** field is available.

If you are looking to create a list of unique values, check the *Unique records only* checkbox. Since the search is for unique values only, there is no need to enter anything into the **Criteria range:** field.

Action 1 - Advanced Filtering and Naming cell ranges

Instructions:	**Results/ Comments:**
1. Open the file **Sumif**.	
2. Save the file as **My Sumif**	This file contains the raw data that we need to extract information from, but as you can see it would take a very long time to figure out who sold how much to whom. Before we start, naming and filtering the cell ranges will help speed the process up.
3. Select cell **A2,** and hold **[Shift + Ctrl]** keys. Tap the down arrow key once.	The entire column of customers is selected.
4. Click into the **NameBox** and type: **Customer**.	The range is now named to be used in our formulas.
5. Select cell **C2,** and hold **[Shift + Ctrl]** keys. Tap the down arrow key once.	
6. Click into the **NameBox** and type: **Salesperson**. Press the **[Enter]** key.	
7. Select cell **J2**, and hold **[Shift + Ctrl]** keys. Tap the down arrow key once.	
8. Click into the **NameBox** and type: **QTY**. Press the **[Enter]** key.	
9. Select cell **K2,** and hold **[Shift + Ctrl]** keys. Tap the down arrow key once.	
10. Click into the **NameBox** and type: **Sales.** Press the **[Enter]** key.	Now main ranges needed for our formulas are named. The first thing we need is to extract a list of the Salespeople.
11. Click the **[Advanced]** button in the **Sort & Filter Group** on the ***Data Tab***.	The *Advanced Filter* dialog box opens.

Action 1 - Advanced Filtering and Naming cell ranges, continued

Instructions:	Results/ Comments:
12. Click the button to ***Copy to another location***	*Excel* allows to place the results of the filter in a location of your choosing.
13. In the **List range:** field enter **$C$1:$C$2156**	This is the range of cells to be filtered. Include the header row since the filter will automatically take the first entry as the header of the list.
14. In the **Copy to:** field enter **N1**	This is where we will place the list.
15. Check the ***Unique Records Only*** checkbox.	We will get only unique values returned and it also negates the need to define what is being searched for by the filter.
16. Click the **[OK]** button.	Now a complete list of the salespeople is displayed in cells **N1** through **N10**.
17. Click the button in the **[Advanced]** **Sort & Filter Group** on the ***Data Tab***	We will now extract a full list of the customers.
18. Click the button to ***Copy to another location***	
19. In the **List range:** field enter **$A$1:$A$2156**	
20. In the **Copy to:** field enter **N15**	
21. Check the ***Unique Records Only*** checkbox.	
22. Click the **[OK]** button.	
23. Save the file.	**[Ctrl +S].**

SUMIF Functions

Using the SUMIF Formula

This formula saves a lot of time when there is a specific subset of data that needs to be totaled. You may have found that you sorted the data by a subset value to run a calculation but when the data is resorted the calculated values were off. This is because the formula used a cell range but those values change in relation to sorting and resorting. To avoid this issue, use the **Sumif** formula since it will only calculate a result based in specified criteria from within your overall data set.

There are three arguments required in the **SUMIF** function:

Note
In this case, using names to specify ranges will prove much easier than trying to select the cell ranges.

Note
When entering the criteria, if the criteria is text or a mathematic expression it must be wrapped in quotation marks. You can also use a cell reference as a criteria.

=SUMIF(Range,Criteria,[Sum_Range])

Range: required - the range to search for the specified criteria

Criteria: required - what is being searched for within the specified search range

Sum_Range: optional - once the criteria is found for each row, sum the data from this range

As an example, should you require a total of all sales made by a salesperson, but the data is sorted by sale date or time. The first thing to consider doing would be to select the column containing all the salespersons names, excluding the column header, and apply a name to the range of cells. Click into the Name Box and type in a name that is clearly descriptive of the content (in this case, Salesperson would be a good choice). Then select and name the range relating to sales totals. Once the ranges are named you can easily refer to the names instead of having to highlight those ranges.

For more explanation of Naming, refer back to the Naming section of Lesson 1.

=SUMIF(salesperson,”smith”,sales)

This translates as: Search the range of cells named *Salesperson* for any instances of text *Smith* and add only those rows in the column named *Sales*.

The **SUMIF** function will return incorrect values if you try matching strings of more than 255 characters.

Action 2 - Using SUMIF to create totals

Instructions:	Results/ Comments:
1. file **MySumif** should still be open. If not, reopen it.	
2. Select cell **O1** and type: **Sales Total** then, press the **[Enter]** key.	The header is added to this cell.
3. Select cell **O2**, if necessary.	This will be where the formula will be added.
4. Enter the following formula: **=SUMIF(Salesperson,$N2,Sales)** press the **[Ctrl +Enter]** key combination.	The formula will examine the range named Salesperson for every instance of the name in cell **N2** (use the **[F4]** key to set the column as absolute). Once those are found it will sum only those instances in the Sales range. Using the **[Ctrl +Enter]** key combination applies the formula and keeps the cell active.
5. Use the autofill handle to fill in cells **O3:O10**.	Click and drag down to cell **O10**.
6. Select cell **P1** and type: **Units Sold** then, press the **[Enter]** key.	Another header is set in place.
7. Select cell **O2** and use the autofill handle to pull the formula into cell **P2**.	You will copy this formula into cell **P2**. Making the criteria use an absolute column and relative row address allows the formula to be dragged into other columns but still refer to the correct column in the formula.
8. Select cell **P2** and Double-click into the cell to edit the formula.	The cell now displays its formula.
9. Change the Sales name to QTY and press the **[Ctrl + Enter]** key combination.	Now the formula will total the number of units sold by the salesperson on this row.

Action 2 - Using SUMIF to create totals, continued

Instructions:	Results/ Comments:
10. Use the autofill to complete the totals for the remaining salespeople.	Now you know how many units were sold by each salesperson.
11. Select cell **O15** and type: **Sales Total** then, press the **[Enter]** key.	Here we want to see how much each customer has purchased.
12. In cell **O16,** enter the following formula: **=SUMIF(Customer,$N16,Sales)** then, press the **[Ctrl +Enter]** key combination.	The formula will examine the range named Customer for every instance of the name in cell **N16** (use the **[F4]** key to set the column as absolute). Once those are found it will add only those instances in the Sales range. Using the **[Ctrl +Enter]** key combination applies the formula and keeps the cell active.
13. Use the autofill handle to fill in cells **O17:O104**.	Double-click the autofill handle to run the formula down as many rows as needed.
14. Try creating another column that calculates how many units each customer has purchased.	Repeat steps 7 through 10 to fill cells **P15:P104**.
15. Apply any desired cell formatting.	
16. Save the file.	**[Ctrl +S]**.

AVERAGEIF Function

Using the AVERAGEIF Formula

This formula uses the same syntax as the SUMIF formula but returns an average derived from the specified criteria.

=AVERAGEIF(Range,Criteria,[Average_Range])

Range: required - the range to search for the specified criteria

Criteria: required - what is being searched for within the specified search range

Average_Range: optional - once the criteria is found for each row, average the data from this range

Note

In criteria, a question mark (?) matches any single character and an asterisk (*) matches any sequence of characters. If you want to find an actual question mark or asterisk, type a tilde (~) before the character.

Things to keep in mind when using this formula:

If the cells in range contain TRUE or FALSE values, they are ignored.

Empty cells within the average_range will be ignored by the **AVERAGEIF** function.

AVERAGEIF will return the **#DIV0!** error if the range is blank or has text values.

Empty cells within the criteria are treated as 0 by the **AVERAGEIF** function.

When the **AVERAGEIF** does not find any matches to the criteria it will return a **#DIV/0!** error.

You can use the wildcard characters within the criteria in this formula.

Average_range and Range do not have to be of the same size. The formula uses the top-left cell in the Average_range as the starting point, it then adapts to meet the size of the Range.

Action 3 - Using the AVERAGEIF formula

Instructions:	Results/ Comments:
1. file **MySumif** should still be open. If not, reopen it.	
2. Select cell **Q1** and type: **Average Sales** then, press the **[Enter]** key.	The next header is added to the spreadsheet. Adjust the column width by placing the cursor between columns Q and R, in the column header area. When the horizontal double arrow cursor is displayed, Double-click the line that separates the columns.
3. Enter the following formula: **=AVERAGEIF(Salesperson,N2,Sales)** then, press the **[Ctrl +Enter]** key combination.	This formula will return the average sale for the salesperson on this row.
4. Use the autofill to fill in the remaining salespeople's values.	Now you know what each salesperson's average sale is.
5. Select cell **Q15** and type: **Average Purchase** then, press the **[Enter]** key.	You are beginning to create another subset of calculated data here.
6. In cell **Q16,** enter the following formula: **=AVERAGEIF(Customer,N16,Sales)** then, press the **[Ctrl +Enter]** key combination.	This returns the average purchase of the customer listed in cell **N16**.
7. Use the autofill to fill in the remaining Customers values.	Now you know the average purchase each customer makes.
8. Save the file.	**[Ctrl + S]**.

COUNTIF Functions

Using the COUNTIF Formula

In the file we have been working on, we have successfully been able to calculate several important data points in relation to both the sales staff and customers. Now we need to know how many sales each salesperson has made, as well as how many times each customer has made an order. To get this information we will use the **COUNTIF** formula.

This formula will work the same way as the previous formulas. The syntax of this formula differs in one way only. It does not use the optional third argument. Both arguments in this formula are required.

=COUNTIF(Range,Criteria)

Range: where to count what will be searched for by the criteria.

Criteria: what is to be searched for and counted within the range.

Action 4 - Using the COUNTIF formula

Instructions:

1. **Mysumif** file should still be open. If not, reopen it.

2. Select cell **R1** and type:

Number of Sales
then, press the **[Enter]** key.

3. In cell **R2,** enter the following formula:

=COUNTIF(Salesperson,N2)
then, press the **[CTRL + Enter]** key combination.

4. Use the autofill to complete the counts for the remaining salespeople.

5. Select cell **R15** and type:

Number of Purchases
then, press the **[Enter]** key.

6. In cell **R16,** enter the following formula:

=COUNTIF(Customer,N16)
then, press the **[CTRL + Enter]** key combination.

7. Use the autofill to complete the counts for the remaining customers.

8. Adjust the width of column **R** to show all the contents.

9. Save the file.

Results/ Comments:

The new header is placed.

This formula will count the number instances the name in cell **N2** occur within the Salesperson range.

The other salespeople's values are filled in.

The next header is placed.

Now you are counting the number of times the customer appears in the Customer range.

We now know how often the customer has made a purchase.

The column is now fully displayed.

[Ctrl + S].

SUMIFS Functions

Using the SUMIFS Formula

Summing by one subset of data is extremely useful. There may come a time when you will need to add more subsets to even further refine the data coming out of your raw data. This is done by using the **SUMIFS**, **AVERAGEIFS**, or **COUNTIFS.** These functions allow you to search for multiple criteria in multiple ranges.

We need to make calculations based on multiple criteria. In the file we have been working on, you may need to know who made sales that met or exceeded target goals. This is where the **SUMIFS** function comes into play.

The syntax for this formula is:

**=SUMIFS(Sum_Range,Range,Criteria,
Range,Criteria)**

In this formula:

The first criteria argument defines the range containing the data to be summed. Unlike a **SUMIF** the Sum_ Range is no longer an optional argument.

The first range defines what will be searched.

The first Criteria defines what will be searched for within the first range.

The subsequent Range and Criteria pairs will narrow the results to a fine point.

There can be up to 127 Range and Criteria pairs used in a single **SUMIFS** formula. The **AVERAGEIFS** formula uses the same syntax as the **SUMIFS**.

COUNTIFS Functions

Using the COUNTIFS Formula

The easiest method to determine the number of instances something occurs within a given set of parameters is to use the **COUNTIFS** function. Since this formula can have multiple Range and Criteria pairs it allows you to do very precise counts.

The syntax of this formula is:

=COUNTIFS(Range,Criteria,Range,Criteria,...)

The first range will be where the count is run.

The first criteria is what is being searched for within the range.

Subsequent Range and Criteria pairs further refine the search results based on their definitions.

If you needed to know how many sales were made by a sales representative that met or exceeded a target goal, you would use a **COUNTIFS** formula. The first Range and Criteria pair defines the primary information needed. In this case you may want to find the desired salesperson from that range.

$$=\text{COUNTIFS(}\underset{\substack{\uparrow\\ \text{Range}}}{\text{Salesperson}}\text{,}\underset{\substack{\uparrow\\ \text{Criteria}}}{\text{Name}}\text{,}$$

The second pairing would search within the number of items sold range for a specified amount. Here you can use mathematic expressions to boost the functionality of the formula. When including a mathematic expression in these formulas, you must wrap the expression in double quotes. This is also true if the criteria is a text value.

$$=\text{COUNTIFS(}\underbrace{\text{Salesperson,Name,}}_{\text{First Range and Criteria Pair}}\overbrace{\text{QTY}}^{\text{Second Range}}\text{,}\underbrace{\text{”>10”}}_{\text{Second Criteria, using mathematic expression}}\text{)}$$

Action 5 - Using the SUMIFS and COUNTIFS formulas

Instructions:	Results/ Comments:
1. **MySumif** file should still be open. If not, reopen it.	
2. Select cell **S1** and type: **Sales above 10** then, press the **[Enter]** key.	The next header is in place.
3. In cell **S2**, enter the following formula: **=SUMIFS(Sales,Salesperson,N2,QTY, ">10")** then, press the **[CRTL + Enter]** key combination.	This formula will add the values in the sales column only if both conditions are found to be true. The name in cell **N2** is found in the Salesperson range and the number for the record is higher than ten in the QTY range.
4. Use autofill to complete the calculations for the other salespeople.	
5. Select cell **T1** and type: **# of Sales Above 10** then, press the **[Enter]** key	Now we will start to breakdown the data into subsets that will offer a more in depth understanding of the data.
6. In cell **T2**, enter the following formula: **=COUNTIFS(Salesperson,N2,QTY,">10")** then, press the **[CRTL + Enter]** key combination.	This will count the number of cells that have a value greater than 10 within the QTY range only if the name Salesperson range match the criteria. You now can see how many sales of more than ten units were made by each salesperson.
7. Use the autofill to complete the calculations for the remaining salespeople.	
8. Save the file.	**[Ctrl +S]**.

Action 6 - Adding more criteria to a COUNTIFS formulas

Instructions:	Results/ Comments:
1. **MySumif** file should still be open. If not, reopen it.	
2. Select cell **U1** and type: **Sales between 5 and 9 Units** then, press the **[Enter]** key.	The new header is added.
3. In cell **U2** enter the following formula: **=COUNTIFS(Salesperson,N2, QTY,"<10",QTY,">=5")** then, press the **[Ctrl + Enter]** key combination.	This formula will narrow the search by adding a third Range and Criteria pair. It will search the same range twice, first to find values below 10 and then for any values '*equal to and above*' 5 within the QTY range but count only those records matching the salespersons' name in cell **N2**.
4. Use the autofill to complete the calculations for the remaining salespeople.	You now see how many sales of between 5 and 9 each salesperson made.
5. Select cells **E2:E2156**.	Select cell **E2**, hold down both the [Shift + CTRL] keys and press the down arrow key once to quickly select the range.
6. Click into the **Namebox,** enter: **Order_Date** then, press the **[Enter]** key.	Since you will need to use this range for the next formula, it is a good idea to name the range.
7. Select cell **S14,** and type: **Number of purchases by month** then, press the **[Enter]** key.	You are adding in another header to define an area where you are going to track purchases by month.
8. In cell **S15**, type: **Jan** then, press the **[Enter]** key.	The first month header.
9. In cell **S16,** enter the following formula: =**COUNTIFS(Customer,N16, Order_Date ,">=1-1-2015", Order_Date,"<=1-31-2015")** then, press the **[CTRL + Enter]** key combination.	This formula is going to count the records for the customer listed in cell **N16** if they made an order between *Jan 1, 2015* and *Jan 31, 2015*.

Action 6 - Adding more criteria to a COUNTIFS formula, continued

Instructions:	Results/ Comments:
10. Try creating another **COUNTIFS** in cell **T16** to find the number of orders each customer made in February.	**=COUNTIFS(Customer,N16, Order_Date ,”>=2-1-2015”, Order_Date,”<=2-28-2015”)**
11. Select cells **S16:T16** and use the autofill to fill in each customer’s monthly order.	
12. Save the file.	**[Ctrl + S]**.

MAXIFS and MINIFS Functions

The **MAXIFS** and **MINIFS** functions will return either the highest or lowest value from a range of cells based on criteria you define. These functions are similar to **SUMIFS**, and **AVERAGEIFS** functions except that these functions can be run with a single criteria range and criteria pair or up to 127 criteria pairs.

The syntax of these formulas is;

=MAXIFS(Max_Range,Criteria_Range1,Criteria,[Criteria_Range2],[Criteria],....)

The first argument defines where the highest or lowest value will be pulled from, depending on which function is being used.

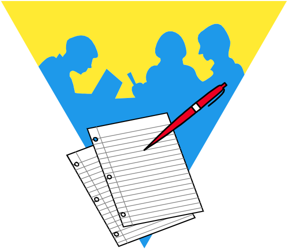
The first Criteria_Range defines where the criteria will be searched for.

The first Criteria is what is being searched for within the criteria range.

Subsequent Criteria_Range and Criteria pairs further refine the search results based on their definitions.

Action 7 - Using the MAXIFS and MINIFS Functions

Instructions:	Results/ Comments:
1. **MySumif** file should still be open. If not, reopen it.	
2. Select cell **W1** and enter: **Highest Sale Value.**	The new header is added.
3. Select cell **W2**.	
4. Enter the following formula; **=MAXIFS(Sales,Salesperson,N2)** then, press the **[CTRL + Enter]** key combination.	This formula will find the highest value in the Sales column of data based on the Salesperson.
5. Use the autofill to complete the calculations for the remaining salespeople.	Now you know the value of each salespersons best sale.
6. Select cell **X1** and enter: **Lowest Sale Value.**	The new header is added.
7. Select cell **X2**.	
8. Enter the following formula; **=MINIFS(Sales,Salesperson,N2)** then, press the **[CTRL + Enter]** key combination.	This formula will find the lowest value in the Sales column of data based on the Salesperson.
9. Use the autofill to complete the calculations for the remaining salespeople.	Each salespersons' lowest sale value is shown.
10. Save and close the file.	**[Ctrl + S]** and **[Ctrl + W]**.



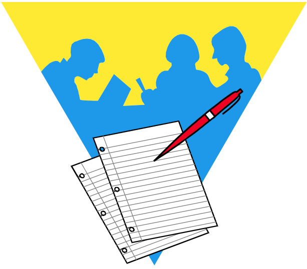
Tips and Notes

Lesson 3: Date & Time Functions

Lesson Overview

You will cover the following concepts in this chapter:

- Time
- NOW Function
- Dates
- Date Functions
- WORKDAY Functions
- NETWORKDAYS Functions
- Finding the Day of the Week
- SWITCH Function
- Finding Week Number



Lesson Notes

Time

Using the Time Functions

Until a cell's value is formatted as time, *Excel* displays the time of day as a decimal number. *Excel* divides a day down into decimal increments ranging from 0 (zero) to 0.99988426, representing the times from 0:00:00 (12:00:00 AM) to 23:59:59 (11:59:59 P.M.). Time can be entered into a cell in several ways:

Manually typing in the time and applying the correct formatting.

Using the **[CTRL + ;]** keyboard shortcut. This method enters the time as a static value, so when it needs to be updated it will have to be done manually again.

Using the **NOW()** function, inserts the time as a dynamic value. When the file is opened or refreshed the time value reflects the system time.

This function does not require any arguments in order to return a value.

By default the cell is formatted as Date and Time, it may need to be reformatted as time only, if needed.

Using the **TIME** function which is similar to manually typing in the time.

The **TIME** function syntax has the following arguments:

Hour: Required. A number from 0 (zero) to 32767 representing the hour. Any value greater than a multiple of 23 will be divided by 24 and the remainder will be treated as the hour value. For example, TIME(27,0,0) = TIME(3,0,0) = .125 or 3:00 AM.

Minute: Required. A number from 0 to 32767 representing the minute. Any value greater than a multiple of 59 will be converted to hours and minutes. For example, TIME(0,750,0) = TIME(12,30,0) = .520833 or 12:30 PM.

Second: Required. A number from 0 to 32767 representing the second. Any value greater than a multiple of 59 will be converted to hours, minutes, and seconds. For example, TIME(0,0,2000) = TIME(0,33,22) = .023148 or 12:33:20 AM

Time,
continued

Note

To create a number format that includes text that is typed in a cell, insert an “at” sign (@) in the text section of the number format code section at the point where you want the typed text to be displayed in the cell. If the @ character is not included in the text section of the number format, any text that you type in the cell is not displayed; only numbers are displayed.

Adding and Subtracting Times

Adding or subtracting times to get an accumulated time is not an uncommon necessity, *Excel* has simple formulas to accomplish this type of task.

Add time

As an example, consider two (or more) separate tasks where each have their own duration. The tasks must be completed in order
to move forward on the project and you need to know how much time the tasks will take to be completed.

Each cell records the time for each task:
A1=6:45
A2=5:30
A3=4:30
In cell **A4** you can enter the following formula;

=A1+A2+A3

The result will be 19 hours and 45 minutes. Whether or not the cells are formatted for time makes no difference to the calculation. The Autosum function will work also in this case. Easy enough when the total time will be less than 24 hours. When the total will be greater than 24 hours, the cell needs a custom number format to return a useful value.

A custom number formatting of **[h]:mm** is required since there is no built-in format in *Excel*. Creating the custom number format is done by:

Opening the *Format Cells* dialog

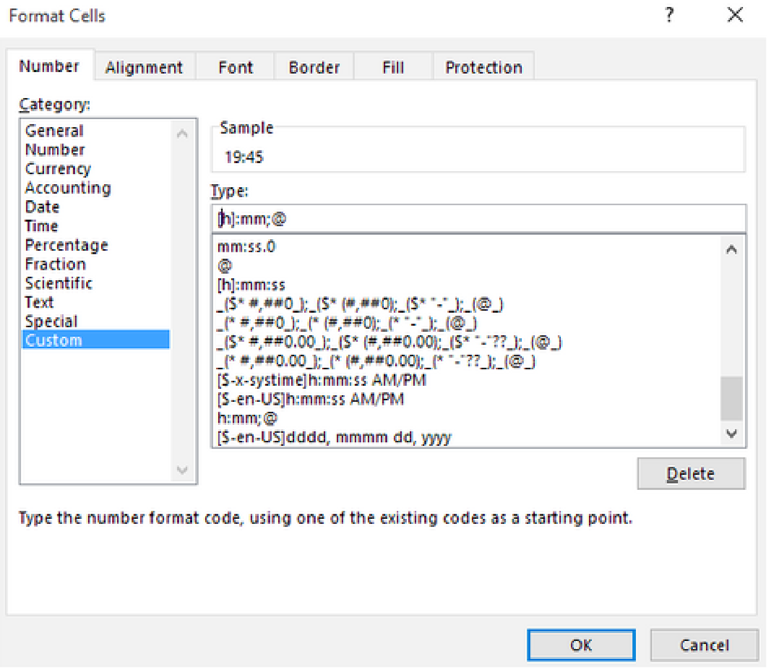
Right-click the cell and choose *Format Cells*

Click the **[Dialog Launcher]** button on the **Number** or **Font Groups** on the ***Home Tab***

Choose the ***Number Tab*** in the dialog.

Choose *Custom* from the bottom of the list of categories on the left

Time,
continued



Category:

- General
- Number
- Currency
- Accounting
- Date
- Time
- Percentage
- Fraction
- Scientific
- Text
- Special
- Custom

Sample

19:45

Type:

```
[h]:mm;@
```

```
mm:ss.0
@
[h]:mm:ss
_($* #,##0_);_($* (#,##0);_($* "-"_);_(@_)
_(* #,##0_);_(* (#,##0);_(* "-"_);_(@_)
_($* #,##0.00_);_($* (#,##0.00);_($* "-"??_);_(@_)
_(* #,##0.00_);_(* (#,##0.00);_(* "-"??_);_(@_)
[$-x-systime]h:mm:ss AM/PM
[$-en-US]h:mm:ss AM/PM
h:mm;@
[$-en-US]dddd, mmmm dd, yyyy
```

Type the number format code, using one of the existing codes as a starting point.

In the **Type** field, typing in **[h]:mm**.

Click the **[OK]** button

Now the desired result of the formula is displayed correctly.

Note

In cells that are formatted as time with AM and PM, type a space then "a" for AM or "p" for PM, and press **[Enter]**.

Subtract time

Another common function is tracking the time spent on a project when there is a start and end time recorded in the data. To quickly calculate how much time was spent on the project, simply subtract the start time from the end time.

=End time - start time

When the times being subtracted will be less than 24 hours use the Time format that includes the AM or PM. This ensures *Excel* returns the correct values.

When the time could be greater than 24 hours, the cells containing the start and end times need to be formatted as Date and Time using another Custom Format:

m/d/yyyy h:mm AM/PM.

(Notice the empty space at the end of yyyy, and at the end of mm.)

Time, continued

The cell containing the formula will use the same custom format as discussed earlier; **[h]:mm;@** .

Custom Time Formatting Codes

To display As	*Use this code*
Years as 00-99yy	
Years as 1900-9999yyyy	
Months as 1-12m	
Months as 01-12mm	
Months as Jan-Decmmm	
Months as January-Decembermmmm	
Months as J-Dmmmmm	
Days as 1-31d	
Days as 01-31dd	
Days as Sun-Satddd	
Days as Sunday-Saturdaydddd	
Hours as 0-23h	
Hours as 00-23hh	
Minutes as 0-59m	
Minutes as 00-59mm	
Seconds as 0-59s	
Seconds as 00-59ss	
Time as 4 AMh AM/PM	
Time as 4:36 PMh:mm AM/PM	
Time as 4:36:03 PMh:mm:ss A/P	
Time as 4:36:03.75 PMh:mm:ss.00	
Elapsed time (hours and minutes) as 1:02	[h]:mm
Elapsed time (minutes and seconds) as 62:16	[mm]:ss
Elapsed time (seconds and hundredths) as 3735.80	[ss].00

Action 1 - Adding and Subtracting Times

Instructions:	**Results/ Comments:**
1. Open the **Times** file.	
2. Save the file as **My Times**.	You will look at the formatting applied to the cells containing the times.
3. Select the ***Weekly Totals*** sheet.	
4. Right-click cell **B4** and choose *Format Cells* from the menu.	The *Format Cells*dialog opens.
5. Click the ***Number*** tab.	
6. Notice the cell is formatted as time without AM/PM.	The *Format Cells* dialog shows what formatting is applied to the active cell.
7. Close the *Format Cells* dialog box.	
8. Select cell **G4**.	You will calculate the total hours this person worked throughout the week.
9. Enter the following formula; **=SUM(B4:F4)** press **[CTRL + Enter].**	This is a simple addition formula, but the value of 4:00 being returned is not usable because any value above 24 is not shown.
10. Right-click the cell and choose *Format Cells* from the menu.	The *Format Cells* dialog opens.
11. Click the ***Number*** tab and choose *Custom* from the list of categories at the left.	The list of existing custom formats are displayed along with the **Type** field, where you can create your own custom number formats.
12. In the **Type** field enter the following: **[h]:mm** click the **[OK]** button.	Adding the brackets around the hour section of the format allows *Excel* to display more than 24 hours.
13. Use autofill to fill in cell **G5:G8**.	Each person's total is calculated.
14. Select cell **B9**.	Now to calculate the daily totals.

Action 1 - Adding and Subtracting Times, continued

Instructions:	Results/ Comments:
15. Enter the following formula; **=SUM(B4:B8)** press . **[CTRL + Enter]**	The formula is a simple addition again.
16. Repeat steps 9 through 11.	To apply the correct formatting to the cells. This completes getting the cumulative hours spent on this project.
17. Use autofill to populate cells	
C9:G9. 18. Save the file.	**[CTRL + S]**.
19. Make the ***Daily Totals*** sheet	Click the sheet tab titled ***Daily Totals***.
active. 20. Select cell **E4**.	You will calculate the number of hours worked by this team member, based on start and end times.
21. Enter the following formula **=D4-C4** press **[CTRL + Enter]**.	This subtracts the start time from the end time. The result should be 3:45 AM. Next you will clear the AM or PM from cell.
22. Right-click the cell and choose *Format Cells* from the menu.	The *Format Cells* dialog box opens.
23. Click the ***Number*** tab and choose *Custom* from the list of categories at the left.	The list of available number formats is displayed.
24. In the Type field enter the following **[h]:mm** click the **[OK]** button.	The value in the cell will be displayed in hours and minutes without the AM or PM.
25. Use autofill to fill in cells **E5:E9**.	Double-click the Autofill handle to run the formulas and formatting down to cell **E9**.
26.**E9**. Select cell 27. Click **[Autosum]** button in the **Editing Group** on the ***Home Tab*** the	The total hours worked are added up.
28..Save the file	**[Ctrl +S]**.

NOW Function

Using the NOW Function

There are times when you need the time in a cell to update automatically each time the file is opened or refreshed. The **NOW** function does not require any arguments in order to work.

The syntax of this function is:

=NOW()

Once the formula is added to a cell, apply the appropriate formatting.

Note

By default the **NOW** function is formatted as date and time until the format is changed.

Refreshing the NOW Function

The **=NOW()** is refreshed each time a formula anywhere is the spreadsheet is added or recalculated. When no new data is added or changed, the **NOW** function can be refreshed in two ways. By calculating the entire workbook or the single worksheet.

The entire sheet

Select any cell in the spreadsheet.

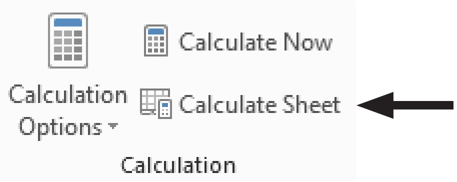
Make the ***Formula Tab*** active.

Click the **[Calculate Sheet]** button in the **Calculation Group**.

All formulas in the spreadsheet are updated.

- OR-

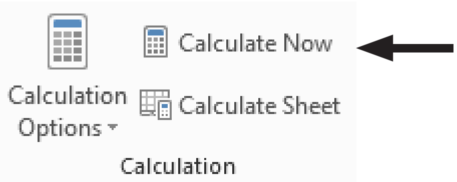
Select any cell in the spreadsheet.



Click on the ***Formulas Tab.***

Click the **[Calculate Now]** button in the **Calculation Group.**

All formulas in the workbook are updated.



Action 2 - Adding a NOW Function

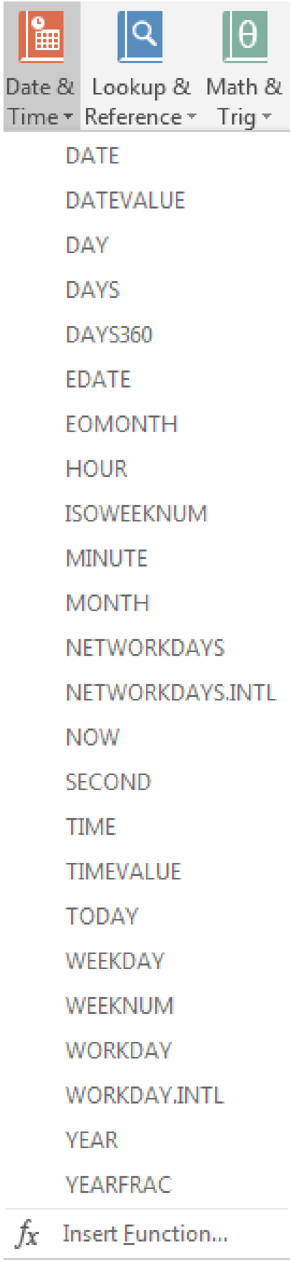
Instructions:	Results/ Comments:
1. The file **My Times** should still be open. If not, reopen it.	
2. Make the ***Hours Today*** sheet active.	
3. Select cell **G3**.	The current time is needed in this cell.
4. Enter the following formula: **=NOW()** press the **[Enter]** key.	The current date and time are inserted in the cell.
5. Right-click cell **G3** and choose *Format Cells* from the menu.	The *Format Cells* dialog box opens.
6. Click the ***Number*** tab.	
7. Choose *Time* from the list of categories on the left and choose *1:30 PM* for the options. Click **[OK].**	The list of time formatting options is displayed. Choose the appropriate format and apply it.
8. Select cell **C5**.	The total hours worked today will be calculated here.
9. Enter the following formula: **=$G$3-B5** press the **[CTRL + Enter]** key combination.	Notice the time should have updated to match the system clock time as the total hours are returned.
10. Right-click cell **C5** and choose *Format Cells* from the menu.	The AM/PM needs to be removed from the formatting of this cell.
11. Choose *Time* from the list of categories on the left and click the time without AM/ PM. Click **[OK].**	The correct formatting is applied but the time is not updated.
12. Use AutoFill to complete the other calculations.	
13. Click the **[Calculate Sheet]** button in the **Calculation Group** on the ***Formula Tab***.	All formulas in the spreadsheet are updated, including the time.
14. Save and close the file.	**[Ctrl + S]** and **[Ctrl + W]**

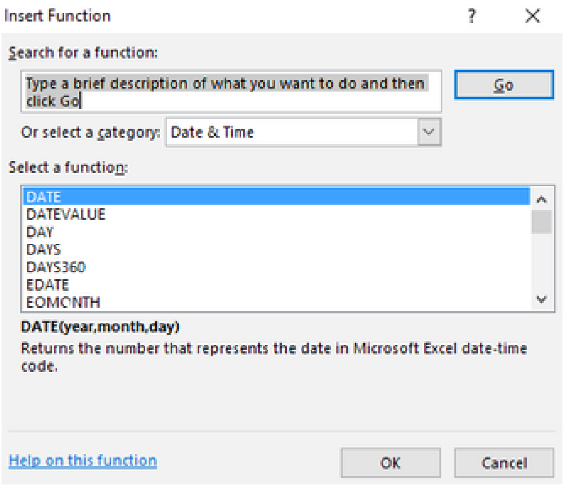
Dates

Understanding Dates

Excel calculates a date as a number, with number 1 being January 1, 1900, and successive dates as the next numbers. Unless the cell is formatted as a date, *Excel* will simply display a numeric value. As a number, *Excel* is able to run simple math and formulas to return relevant information about dates.

There is a wide variety of formulas built into *Excel* which are found on the ***Formula Tab*** in the ribbon or within the *Insert Function* dialog box.





Search for a function: Type a brief description of what you want to do and then click Go

Or select a category: Date & Time

DATE(year,month,day)

Returns the number that represents the date in Microsoft Excel date-time code.

Date Functions

Using the TODAY Function

While you can always manually type in a date into a cell or use the keyboard shortcut of **[CTRL + ;]**, the date is static and does not change to reflect the current date. Should you need to always have a cell display the current date, *Excel* has the **TODAY** function. This formula does not require any arguments in order to function correctly.

The syntax of the function is:

=TODAY()

This will always return the current date. It is based on the system's date (the date on the computer). Formulas which refer to the cell will be updated every time the file is opened.

As an example: if you need to track the number of days that an order takes to be processed. The date the order was made is subtracted from the current date.

Note
Dates may be entered as text strings within quotation marks (for example, "2001/1/30"), as serial numbers (for example, 36921, which represents January 30, 2001, if you're using the 1900 date system), or as the results of other formulas or functions

Finding the Difference Between Two Dates

This can be a simple math formula that subtracts a start date from an end date. It is a math formula because even though the cell is displaying a date, it only does so due to the cell's formatting. In the background, *Excel* starts counting days from January 1, 1900 as 1, each successive day adding another number. You will notice this numeric value displayed for dates if you turn on **[Show Formulas]**.

To find the number of days separating two dates, write the formula as

=End_Date-Start_Date

If you have the dates in cells, use the cell references. When you plan on using the same type of formula to calculate a range of cells with autofill, set the End_Date cell reference as an absolute reference.

Date Functions,
continued

Using the DATEDIF Function

When more specific information is needed, the **DATEDIF** formula is ideal since it calculates the number of days, months, or years between two dates. It must be noted that the **DATEDIF** function is not a standard *Excel* formula.

The syntax of the **DATEDIF** formula is:

Note

The DATEDIF function is not part of Excels' list of functions but it is commonly used and understood by the program.

=DATEDIF(start_date,end_date,unit)

Start_date: the starting date of the period of time.

End_date: the ending date of the period of time.

Unit: The type of information that you want returned:

The table below lists the types of available units.

Unit	Returns
"Y"	The number of complete years in the period.
"M"	The number of complete months in the period.
"D"	The number of days in the period.
"MD"	The difference between the days in start_date and end_date. The months and years of the dates are ignored.
"YM"	The difference between the months in start_date and end_date. The days and years of the dates are ignored
"YD"	The difference between the days of start_date and end_date. The years of the dates are ignored.

The quotation marks are required when adding the Unit to the formula.

Date Functions,
continued

If you need to see the years, months, and days between two dates then the formula requires a little modification in order to return the information in the format needed.

To calculate the number of years, months and days between two dates (more than a year apart) you can use this formula:

=DATEDIF(Start_date,End_date,"y") & " years, " & DATEDIF(Start_date,End_date,"ym") & " months, " & DATEDIF(Start_date,End_date,"md") & " days"

The first **DATEDIF** will return the number of years between to two dates.

Adding the **&" years, "** will add the text and spaces after the number of years.

The next **&** adds the next section of this formula.

The second **DATEDIF**, using the "YM" unit returns the number of months while ignoring the number of years.

Adding the **&" months, "** will add the text and spaces after the number of months.

The next **&** adds the next section of this formula.

The third **DATEDIF**, using the "MD" unit returns the number of days while ignoring the number of months.

Adding the **&" days, "** will add the text and spaces after the number of months.

The result between *Jan15, 2012* and *Feb 27, 2015* would be:

3years,1month,12Days

If you need each unit in it's own cell write each DATEDIF in it's own cell instead of combining them in a single cell.

Date Functions, continued

Using the Days Formula

This is a new formula available in *Excel* to quickly find the number of days between a pair of start and end dates. The **DAYS** formula returns a numeric value and should not be formatted as a date.

The syntax of the **DAYS** formula is:

=DAYS(End_date,Start_date)

End_date: required - the last day to be taken into account by the formula.

Start_date: required - the first day to be taken into account by the formula.

Action 3 - Date Calculations

Instructions:	**Results/ Comments:**
1. Open the **Employee_Dates** file.	You will be calculating the employees’ Ages and Years of Service.
2. Save the file as **My Employee Dates.**	
3. Select cell **F1**.	
4. Enter the following formula; **=TODAY()** press the **[Enter]** key	This puts today’s date into the cell. It will always reflect the current date so the data in the file will be accurate.
5. Select cell **J7**, and Enter the following formula **=$F$1-I7** Then, press the **[CTRL + Enter]** key combination.	This formula works but returns a value in days. It is possible to use **=($F$1-I7)/365** but that does not help when considering leap years.
6. In cell **J7**, change the formula to: **=DATEDIF(I7,$F$1,”Y”)** press the **[CTRL +Enter]** key combination.	Using the **DATEDIF** formula will return an accurate age. The first argument is the birth date, the second argument is the current date, and the last argument defines the unit of time you want.
7. Use Autofill to complete calculating the other employee ages.	Double-click the Autofill handle.
8. Select cell **L7**.	Here you need to know how many years, month, and days each employee has worked.
9. Enter the following formula; **=DATEDIF(K7,$F$1,”y”)**	To return a value in years.
& “ Years, “	To add the text held within the quotes
& DATEDIF(K7,$F$1,”ym”)	To add and return a value in months.
& “ months, “	To add the text held within the quotes
& DATEDIF(K7,$F$1,”md”)	To add and return a value in days.
& “ days”	To add the text held within the quotes

Action 3 - Date Calculations, continued

Instructions:	Results/ Comments:
10. Press the **[CTRL + Enter].**	Using this key combination applies the formula to the cell and keeps the cell actively selected.
11. Use Autofill to complete calculating the other employees years of service data.	Double-click the Autofill handle.
12. Save the file and leave it open.	**[Ctrl + S]**.

WORKDAY Functions

Using the WORKDAY Formula

Calculating a day in the future can be as simple as adding the number of days in the future to the current date since the date is really a number. If you need to add only working days to a date to calculate a future date, use the **WORKDAY** formula. This formula assumes the work week runs Monday through Friday. It can also exclude holidays like the **NETWORKDAYS** formulas.

The syntax of the WORKDAY formula is:

=WORKDAY(Start_date,Number_of_days, [Holidays])

Date: required - the starting date to which you want to add days.

Number of Days: required - the number of days to add to the start date.

Holidays: optional - can be a range of cells that lists holiday dates.

Using the EDATE Formula

If you need to add months to a date, the **EDATE** formula provides a simple solution. It can return both future and past dates with equal ease. The cell containing the formula must be formatted as a date.

The syntax of the **EDATE** formula is:

=EDATE(Start_Date,Number_of_months)

Start_date: required - the starting date to which you want to add months to.

Number_of_Months: required - the number of months to add to or subtract from the start date.

To subtract months, place a minus sign before the Number_of_months

Action 4 - Adding Working Days to a date

Instructions:	Results/ Comments:
1. The file **My Employee Dates** should still be open. If not, reopen it.	
2. Select cell **M5**.	
3. Type in **Probation period** press the **[Enter]** key.	To add the Header for the column.
4. Adjust the column width.	Double-click between columns M and N.
5. Select cell **M7**.	
6. Enter the following formula; **=WORKDAY(K7,90)** press the **[CTRL + Enter]** key combination.	This formula will add 90 working days (excluding weekends) to the hire date in cell **K7**.
7. Right-click the cell and choose *Format Cells* from the menu.	The result is correct but needs to be formatted as a date.
8. On the ***Number Tab,*** choose *Date* from the left and select *3/14/12* formatting. Click **[OK].**	The value in cell **M7** is now clear.
9. Use AutoFill to complete the other employee's probation period calculations.	Double-click the AutoFill handle.
10. Select cell **N5**.	
11. Type in **Anniversary** press the **[Enter]**	The column header is added.
12. Select cell **N7**.	
13. Enter the following formula **=EDATE(K7,6)** press the **[CTRL + Enter]** key combination.	This formula adds six months to the hire date.

Action 4 - Adding Working Days to a date, continued

Instructions:	Results/ Comments:
14. Use AutoFill to complete the rest of the column.	Double-click the AutoFill handle.
15. Save and close the file.	**[Ctrl + S]** then **[Ctrl + W]**.

NETWORKDAYS Functions

Calculating Working Days

When working long projects it may be necessary to calculate the number of working days the project will take, excluding weekends and holidays. *Excel* has two built-in formulas
to run this type of calculation: **NETWORKDAYS** and **NETWORKDAYS.INTL**.

The syntax of the **NETWORKDAYS** formula is:

=NETWORKDAYS(start_date, end_date, [holidays])

Start_date: required - represents the start date.

End_date: required - represents the end date.

Holidays: is an optional argument, a range of one or more dates to exclude from the working calendar, such as state and federal holidays and floating holidays. The list can be either a range of cells that contains the dates or an array constant of the serial numbers that represent the dates.

The syntax of the **NETWORKDAYS.INTL** formula is:

=NETWORKDAYS.INTL(start_date, end_date, [weekend],[holidays])

Start_date and end_date: Required. The dates for which the difference is to be calculated. The start_date can be earlier than, the same as, or later than the end_date.

Weekend: Optional. Indicates the days of the week that are weekend days and are not included in the number of whole working days between start_date and end_date. Weekend is a weekend number or string that specifies when weekends occur.

Holidays: is an optional argument, a range of one or more dates to exclude from the working calendar, such as state and federal holidays and floating holidays. The list can be either a range of cells that contains the dates or an array constant of the serial numbers that represent the dates.

NETWORKDAYS Functions, continued

The weekend argument can be a string of seven characters with each character representing a day of the week, beginning with Monday. 1 represents a non-workday while 0 represents a workday. Only the characters 1 and 0 are permitted in the string. As an example, 1000001 would result in a weekend that is Monday and Sunday.

You can also use specific numbers to represent the weekend as listed below.

Weekend number	Weekend days
1 or omitted	Saturday, Sunday
2	Sunday, Monday
3	Monday, Tuesday
4	Tuesday, Wednesday
5	Wednesday, Thursday
6	Thursday, Friday
7	Friday, Saturday
11	Sunday only
12	Monday only
13	Tuesday only
14	Wednesday only
15	Thursday only
16	Friday only
17	Saturday only

Action 5 - Calculating Working Days and Hours

Instructions:	Results/ Comments:
1. Open the **Project** file.	
2. Save the file as **My Project**	
3. On the ***Billing*** worksheet, select cell **F2**.	
4. Enter the following formula; **=NETWORKDAYS(D2,E2,Holidays)** press the **[CTRL + Enter]** key combination.	This formula will return the number of working days between the Start and End dates, excluding weekends and holidays.
5. Format the cell as a number and Autofill the rest of the column.	Use the **[Number Format]** drop-down in the **Number Group** on the ***Home Tab***.
6. Press the **[Tab]** key to select cell **G2**.	
7. Enter the following formula; **=NETWORKDAYS(D2,E2,Holidays)*8** press the **[CTRL + Enter]** key combination.	This will take the value returned by the **NETWORKDAYS** formula and multiply it by 8. You now have the number of working hours based on an eight hour workday.
8. Format the cell as a number and Autofill the rest of the column.	Use the **[Number Format]** drop-down in the **Number Group** on the ***Home Tab***.
9. Press the **[Tab]** key to select cell **H2**.	This column has the number of allocated hours per project. You will create a formatting rule to highlight projects where the hours worked exceed the allotted time.
10. Hold the **[CTRL +Shift]** keys and press the down arrow key once.	The rest of the column is selected.
11. Click the **[Conditional Formatting]** button drop-down in the **Styles Group** on the ***Home Tab*** and choose *New Rule*.	The *New Formatting Rule* dialog box opens.
12. In the **Select a Rule Type:** field, click *Use a formula to determine which cells to format*	You will be using a formula to determine whether or not to apply formatting.

Instructions:	Results/ Comments:
13. In the **Format values where this formula is true** field type; **=G2>H2**	This checks to see if the hours worked are greater than the allotted time.
14. Click the **Format** button.	The *Format Cells* dialog box opens.
15. Click the **Fill** tab.	
16. Choose a color *Red* and click the **[OK]** button.	
17. Click the **[OK]** button to apply the formatting.	Now when the hours worked exceeds the allotted time, the cell is filled with red.
18. Save the file.	**[Ctrl + S]**.

Finding the Day of the Week

Using the WEEKDAY Formula

This formula allows you to determine what day of the week any date is. Use this function when it is important to know on what day of the week a date falls.

The syntax of the **WEEKDAY** formula is:

=WEEKDAY(serial_number,[return_type]

Serial_number: [required] what day of the week is the date you have.

Return_type: [optional] the argument tells the function how to number the days of the week. If you leave this argument empty, Sunday will be 1 and Monday will be 2 etc.

Below is the list of available Return_types

Value	Description
1 or	Numbers 1 (Sunday) through 7 (Saturday).
omitted 2	Numbers 1 (Monday) through 7 (Sunday).
3	Numbers 0 (Monday) through 6 (Sunday).
11	Numbers 1 (Monday) through 7 (Sunday).
12	Numbers 1 (Tuesday) through 7 (Monday).
13	Numbers 1 (Wednesday) through 7 (Tuesday).
14	Numbers 1 (Thursday) through 7 (Wednesday).
15	Numbers 1 (Friday) through 7 (Thursday).
16	Numbers 1 (Saturday) through 7 (Friday).
17	Numbers 1 (Sunday) through 7 (Saturday).

Action 6 – Using the WEEKDAY formula

Instructions:	Results/ Comments:
1. **My Project** file should still be open. If not, reopen it.	
2. Activate the ***Holidays*** worksheet.	Click the **[New Sheet]** button.
3. Select cell **H2 and type** **1** then, press **[Enter]** and type: **2** in cell **H3.**	This will begin the lookup list to use in a **VLOOKUP** formula later .
4. Select both cells and use autofill series down to cell **H8**.	You now have numbers 1 through 7.
5. Select cell **I2** and type: **Sunday** then, press the **[CTRL + Enter]**key combination.	These will be the days that correspond to the numbers.
6. Autofill cell I2 down to cell **I8**.	The rest of the weekdays should now be listed beside a number.
7. In cell **B2,** enter the following formula **=WEEKDAY(A2,1)** then, press the **[Ctrl + Enter]** keys.	The formula will now return a numeric value representing the day of week the holiday falls on.
8. Autofill cell **B2** down to cell **B17**.	Now each holiday date is shown as a numeric value of 1 to 7.
9. In cell **C2,** enter the following formula **=VLOOKUP(B2,$H$2:$I$8,2)** then, press the **[Ctrl + Enter]** keys.	This will look for the value in cell **A2**, compare it to the table and return the correct day as a text value.
10. Autofill cell **C2** down to cell **C17**.	You now know what day of the week each holiday falls.
11. Save the file.	**[Ctrl +S]**.

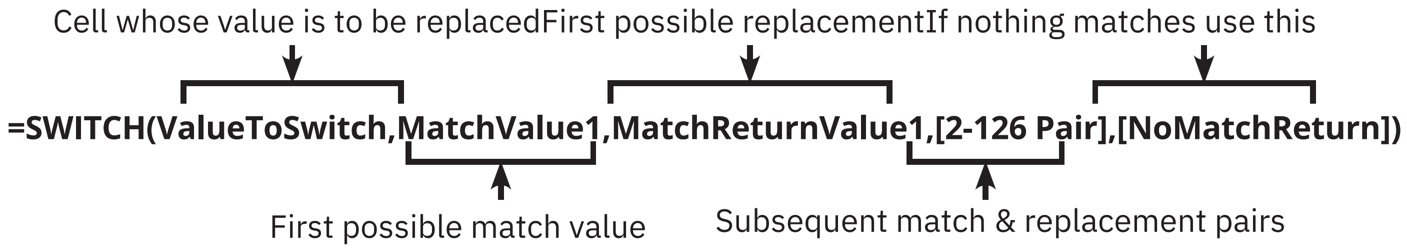
SWITCH Function

In the last exercise the **WEEKDAY** function returned a numeric value to represent the day of the week. Once that value was returned a **VLOOKUP** was used to compare the returned value and pull the day from a table. While this worked to show what day of the week the date occurred on it does require the addition of the table array needed in the **VLOOKUP**. Using the **SWITCH** function with the **WEEKDAY** function nested inside would simplify the process greatly.

Using the SWITCH Function

The **SWITCH** function can be seen like a merged **VLOOKUP** and **IFS** formula. It changes a cells' existing value by comparing it to up to 126 values. Once a match is found, the match return value replaces the original value. Should no matches be found, an optional value may be returned.

The syntax of the **SWITCH** function is ,

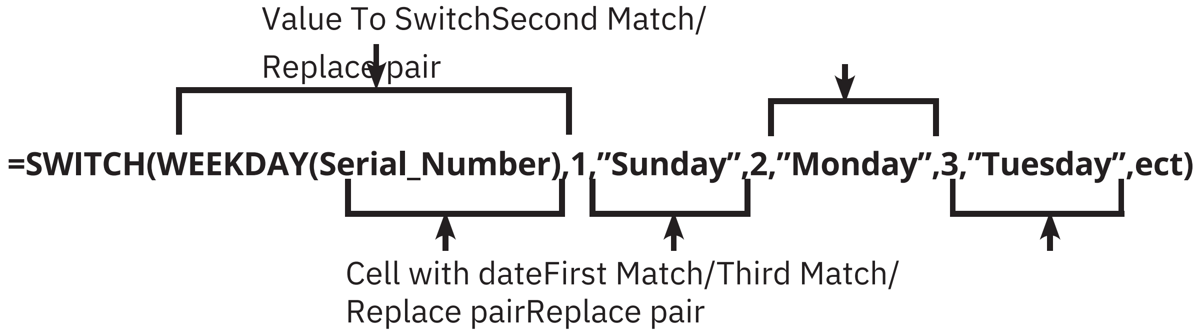


Cell whose value is to be replacedFirst possible replacementIf nothing matches use this

=SWITCH(ValueToSwitch,MatchValue1,MatchReturnValue1,[2-126 Pair],[NoMatchReturn])

First possible match value Subsequent match & replacement pairs

The first argument of the formula (ValueTo Switch) is where the **WEEKDAY** function can nested so the day of the week the date falls on is returned.



Value To SwitchSecond Match/
Replace pair

=SWITCH(WEEKDAY(Serial_Number),1,"Sunday",2,"Monday",3,"Tuesday",ect)

Cell with dateFirst Match/Third Match/
Replace pairReplace pair

This negates the need for adding lookup tables and writing multiple formulas in cells to get a desired answer.

Action 7 – Using the SWITCH formula

Instructions:	Results/ Comments:
1. **My Project** file should still be open. If not, reopen it.	
2. Select cell **D2** and enter the following formula; **=SWITCH(WEEKDAY(A2),1,”Sunday”, 2,”Monday”,3,”Tuesday”,4,”Wednesday”, 5,”Thursday”,6,”Friday”,7,”Saturday”)** use the **[Ctrl + Enter key]** combo	This formula will return the day of the week without any other lookups.
3. Autofill cell **D2** down to cell **D17**.	
4. Save the file.	**[Ctrl + S]**.

Finding Week Number

Using the WeekNum Formula

This formula will return the week number of a specific date. As an example, the week containing January 1 is the first week of the year, and is numbered week 1 up to a possible week 52.

There are two systems used for this function:

System 1: The week containing January 1 is the first week of the year, and is numbered week 1.

System 2: The week containing the first Thursday of the year is the first week of the year, and is numbered as week 1. This system is the methodology specified in ISO 8601, which is commonly known as the European week numbering system.

The syntax of the **WEEKNUM** formula is:

=WEEKNUM(serial_number,[return_type])

Serial_number : Required. A date within the week. Dates should be entered by using the DATE function, or as results of other formulas or functions. For example, use DATE(2008,5,23) for the 23rd day of May, 2008. Problems can occur if dates are entered as text.

Return_type: Optional. A number that determines on which day the week begins. The default is 1.

Return_type	Week begins on	System
1 or omitted	Sunday	1
2	Monday	1
11	Monday	1
12	Tuesday	1
13	Wednesday	1
14	Thursday	1
15	Friday	1
16	Saturday	1
17	Sunday	1
21	Monday	2

Finding Week Number, continued

Using the ISOWEEKNUM Formula

ISO8601: 2000 is becoming mandatory in the European Union and is becoming more commonly used throughout the rest of the World.

There may come a time when you need to know what week number of the year a specific date falls in. *Excel* has added a new formula to make this easy to determine, the **ISOWEEKNUM** formula.

The syntax of the **ISOWEEKNUM** formula is:

=ISOWEEKNUM(Date)

Date: Required- any date after January 1 1900 is valid. Dates before the January 1 1900 will return a #NUM error.

The date cell must be formatted as a date in order for the formula to work.

The Standard issued in 1986, prescribes amongst other things,

Dates must be formatted as either:

A non-separated form of yyyymmdd (20160130 for 30-Jan-2016)

A separated form of yyyy-mm-dd (2016-01-30 for 30-Jan-2016)

Week numbering, must use one of the two following systems:

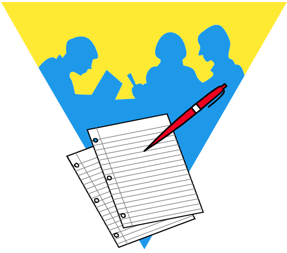
Week 1 begins on the Monday where 4th January falls.

Week 1 begins on the Monday that contains the first Thursday of the calendar year.

So it will be up to your requirements as to which of these formulas to use.

Action 8 - Find the Week number

Instructions:	Results/ Comments:
1. The file **MyProject** should still be open. If not, reopen it.	
2. Select cell **E2**.	
3. Enter the following formula; **=WEEKNUM(A2)** press the **[Ctrl + Enter]** keys.	This formula will return the week number of the year you were born in.
4. Autofill cell **E2** down to cell **E17**.	You now know the week number each holiday falls in.
5. Save and close the file	**[Ctrl + S]** and **[Ctrl + W]**.



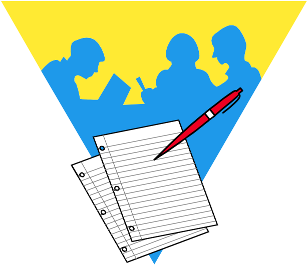
Tips and Notes

Lesson 4: Formula Auditing

Lesson Overview

You will cover the following concepts in this chapter:

- Formula Auditing
- Tracing Formulas
- Errors
- Error Checking
- Evaluating Formulas
- Watch Window
- Calculations



Lesson Notes

Formula Auditing

You may frequently be given spreadsheets that were created by other users or created in previous versions of *Excel* in which the formulas no longer function or show error messages. In this lesson, you will learn to use the tools *Excel* has to help you audit and correct these formula errors and issues.

The Formula Auditing Group

The **Formula Auditing Group** on the ***Formulas Tab*** has tools to
facilitate management of your formulas. A variety of tools allow you to:

Trace Precedents and Dependents: displays blue lines showing where formulas draw their data.

Precedents show where the selected cell is drawing data from.

Dependants show connections to cells that are referencing the selected cell.

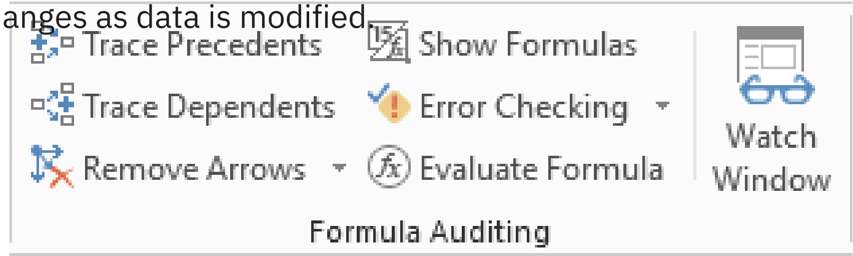
Remove Arrows: clears the lines from the spreadsheet.

Show Formulas: displays the formulas in their cells as opposed to the formula results.

Error Checking: examines the spreadsheet for formula errors in a similar manner as using spell check.

Evaluate Formulas: opens a window where you can move through the formula in a step by step process.

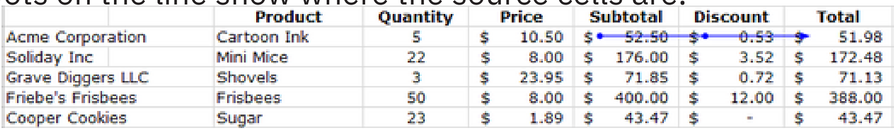
Watch Window: opens a window that can dynamically display results of multiple formula changes as data is modified.



Tracing Formulas

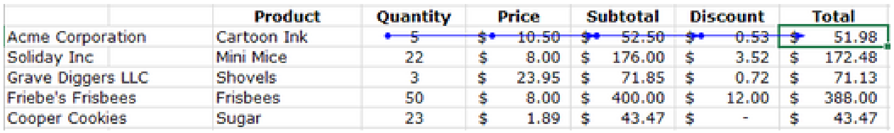
Tracing Precedents

The **[Trace Precedents]** button will indicate which cells the active cell is drawing data from. A **Precedent** is a cell that contributes
to the result.
When the button is clicked a blue line appears. It traces backward to the cells that influence the current cell's formula. Dots on the line show where the source cells are.



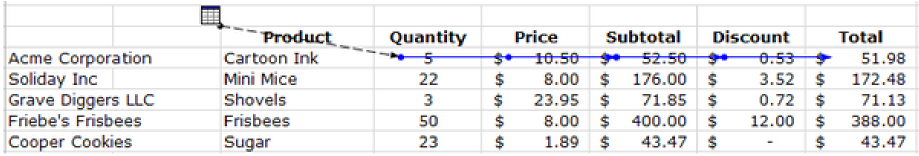
	Product	Quantity	Price	Subtotal	Discount	Total
Acme Corporation	Cartoon Ink	5	$ 10.50	$ 52.50	$ 0.53	$ 51.98
Soliday Inc	Mini Mice	22	$ 8.00	$ 176.00	$ 3.52	$ 172.48
Grave Diggers LLC	Shovels	3	$ 23.95	$ 71.85	$ 0.72	$ 71.13
Friebe's Frisbees	Frisbees	50	$ 8.00	$ 400.00	$ 12.00	$ 388.00
Cooper Cookies	Sugar	23	$ 1.89	$ 43.47	$ -	$ 43.47

Clicking the button again will show the precedents of any preceding cells that contain formula references. Continue clicking the **[Trace Precedents]** button to continue examining formula connections.



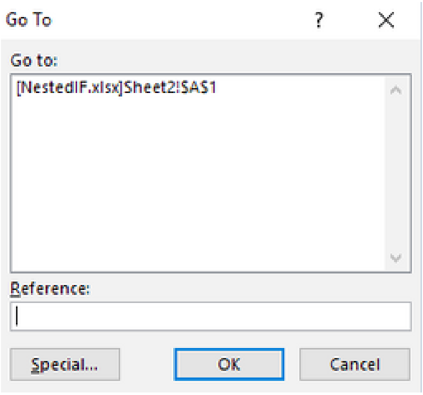
	Product	Quantity	Price	Subtotal	Discount	Total
Acme Corporation	Cartoon Ink	5	$ 10.50	$ 52.50	$ 0.53	$ 51.98
Soliday Inc	Mini Mice	22	$ 8.00	$ 176.00	$ 3.52	$ 172.48
Grave Diggers LLC	Shovels	3	$ 23.95	$ 71.85	$ 0.72	$ 71.13
Friebe's Frisbees	Frisbees	50	$ 8.00	$ 400.00	$ 12.00	$ 388.00
Cooper Cookies	Sugar	23	$ 1.89	$ 43.47	$ -	$ 43.47

If there are any cells referencing content from other worksheets or workbooks, a black dotted line and worksheet icon will appear. To follow the precedents, double-click the dotted line (not the icon) to open the *Go To* dialog box.



	Product	Quantity	Price	Subtotal	Discount	Total
Acme Corporation	Cartoon Ink	5	$ 10.50	$ 52.50	$ 0.53	$ 51.98
Soliday Inc	Mini Mice	22	$ 8.00	$ 176.00	$ 3.52	$ 172.48
Grave Diggers LLC	Shovels	3	$ 23.95	$ 71.85	$ 0.72	$ 71.13
Friebe's Frisbees	Frisbees	50	$ 8.00	$ 400.00	$ 12.00	$ 388.00
Cooper Cookies	Sugar	23	$ 1.89	$ 43.47	$ -	$ 43.47

Any 3D references will be listed in the *Go To* dialog box. Select the one you want to follow and click the **[OK]** button to move to the sheet or file. If the file is not currently open, it will be opened.



Go To

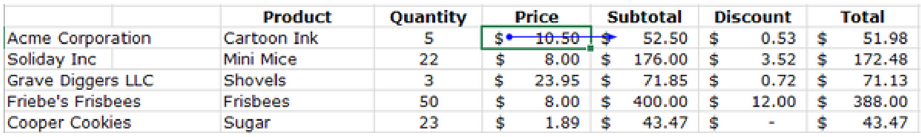
Go to: [NestedIF.xlsx]Sheet2!$A$1

Reference:

Tracing Formulas, continued

Tracing Dependents

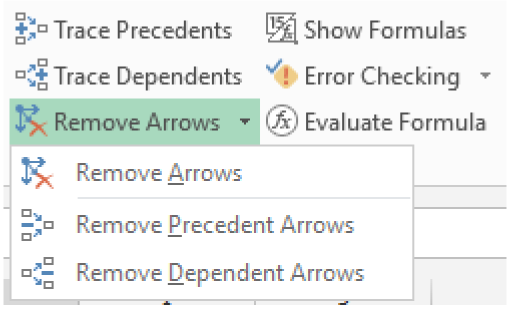
This tool is used to see what other formulas are drawing data from the active cell. The Dependent cell does have to contain a formula. It can be a static value. The arrows will now point to cells that the active cell contributes to. Just like the **[Trace Precedent]** button, if you continue to click the button it will show the next level of cells that the active cell contributes to.



	Product	Quantity	Price	Subtotal	Discount	Total
Acme Corporation	Cartoon Ink	5	$ 10.50	$ 52.50	$ 0.53	$ 51.98
Soliday Inc	Mini Mice	22	$ 8.00	$ 176.00	$ 3.52	$ 172.48
Grave Diggers LLC	Shovels	3	$ 23.95	$ 71.85	$ 0.72	$ 71.13
Friebe's Frisbees	Frisbees	50	$ 8.00	$ 400.00	$ 12.00	$ 388.00
Cooper Cookies	Sugar	23	$ 1.89	$ 43.47	$ -	$ 43.47

Removing Arrows

Since you can show both types of arrows at the same time, the **[Remove Arrows]** button will remove any and all **Precedent** or **Dependent** arrows with a single click. The drop-down for the **[Remove Arrows]** button will offer three choices:



Remove Arrows: this will remove all the arrows at once.

Remove Precedent Arrows: removes the last level of precedent arrows applied.

Remove Dependent Arrows: removes the last level of dependent arrows applied.

Action 1- Tracing Formulas

Instructions:	Results/ Comments:
1. Open the **Tracing.xlsx** file.	The file is stored in the class data files folder, the instructor will point out where to find the folder.
2. Save the file as **MY Tracing**	
3. Make the **QTR 1** sheet active.	Click the **QTR 1** sheet tab below the spreadsheet.
4. Select cell **F13**.	This cell is the result of a summing formula.
5. Click the **[Trace Precedents]** button in the **Formula Auditing Group** on the ***Formulas Tab***.	A blue line indicates what cells are referred to by the formula in cell F13.
6. Click the **[Trace Precedents]** button again.	A new set of lines indicate the cells being referred to by the first set of cells that are being calculated in cell **F13**.
7. Click the **[Remove Arrows]** button in the **Formula Auditing Group** on the ***Formulas Tab***.	The arrows are removed from the spreadsheet.
8. Select cell **B6**.	This cell contains a value that another cells' formula is referring to.
9. Click the **[Trace Dependents]** button in the **Formula Auditing Group** on the ***Formulas Tab***.	Blue lines point to the cells that include the active cell in their formulas.
10. Click the **[Trace Dependents]** button again.	More lines appear, pointing to the next cells that are dependent on the active cell in their formulas. A black dotted line pointing to a spreadsheet icon also appears, indicating that formulas in other locations depend on the cell in their calculations.
11. Click the **[Remove Arrows]** button in the **Formula Auditing Group** on the ***Formulas Tab***.	The arrows are removed from the spreadsheet.

Action 1- Tracing Formulas, continued

Instructions:	**Results/ Comments:**
12. Make the ***Summary*** sheet active.	Click the tab below the spreadsheet.
13. Select cell **F13**.	This cell contains a formula referring to other cells in the spreadsheet.
14. Click the **[Trace Precedents]** button three times.	Each time the button is clicked the formula is traced back another level.
15. Double-click any of the dotted lines..	The *Go to* dialog box opens, showing the reference or related references.
16. Select the reference and click the **[OK]** button.	The reference location is now active.
17. Make the ***Summary*** sheet active.	Click the tab below the spreadsheet.
18. Click the **[Remove Arrows] drop-down** button and choose *Remove Precedents Arrows*.	One level of Precedent lines is removed from the spreadsheet.
19. Click the **[Remove Arrows]** button.	The arrows are removed from the spreadsheet.
20. Save and close the file.	**[CTRL + S]** and **[CTRL + W]**.

Errors

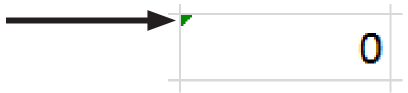
Errors

Errors can be marked and corrected in two ways:

Using the **[Error Checking]** button (like a spelling checker) checks for errors on the entire spreadsheet

Using the **Error Checking Options** immediately when they occur on the worksheet as you work.

When an error occurs, a green triangle appears in the upper-left corner of the cell.



Error Messages

Some errors cannot return a result and will display an error value in the cell. The error messages are simple ways *Excel* explains problems in formulas.

#NAME?	occurs when *Excel* does not recognize text in a formula, if using a name in a formula and the name is misspelled or does not exist
#VALUE!	occurs when a formula has the wrong type of argument, if the formula is expecting numeric values but a cell contains text or dates
#DIV/0!	occurs when a formula tries to divide a number by 0 or an empty cell
#REF!	occurs when a formula refers to a cell that is not valid, if the formula refers to a cell in a deleted row or column
#NULL!	occurs when you specify a intersecting range which in fact does NOT intersect, the value in a cell may be a negative value, percent, or currency value
#N/A	occurs when the wrong type of argument or operand is used in the formula, most common in relation to lookup formulas

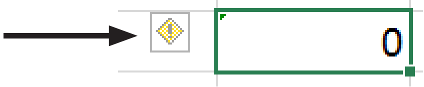
Error Checking

Note

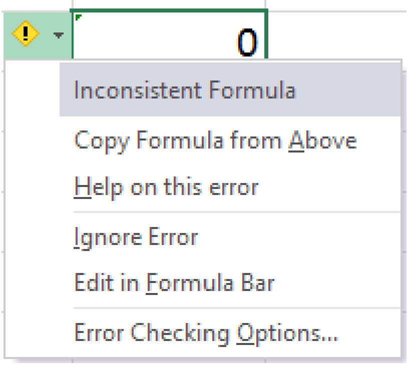
When multiple cells containing errors are selected, the **Error Checking Options** smart tag is not displayed.

Error Checking Options

Other errors still return values but show error warnings, such as being inconsistent with adjacent formulas. The error may advise you to examine the formula. When the cell with the error is active, a small warning icon is displayed to the left of the cell.



When the icon is clicked a drop-down menu displays offering suggestions on how to resolve the potential error. This could be a potential error because you may not need to refer to the same relative range in this instance. In which case *Ignore Error* would be the best option to avoid having the icon appear when the cell is selected.



Inconsistent Formula : the basis error (it can change depending on the formula) that has determined *Excel*

Copy Formula from Above : will copy the formula from the cell above to remove the perceived error

Help on This Error : opens the dialog *Help* box to the specific error type help page

Ignore Error : leaves the *Excel* formula as is without continuing to display the **Error Checking Options** smart tag

***Edit in Formula Bar*:** displays the formula in the formula bar

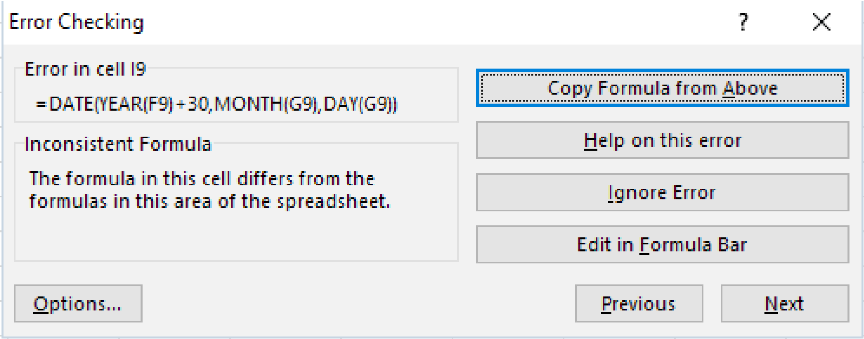
Error Checking Options *Options* : opens the dialog box, where you can select the rules governing error checking

Error Checking, continued

Error Checking

The tool works in a similar fashion as running a spell check in other programs. It will inspect the entire spreadsheet looking for errors which may not have been noticed while creating the spreadsheet.

If the **[Error Checking]** button is clicked, the *Error Checking* dialog opens. You could go through the spreadsheet checking each error at one time. These are similar options to those found in the **Error Checking Options** Smart Tag menu.



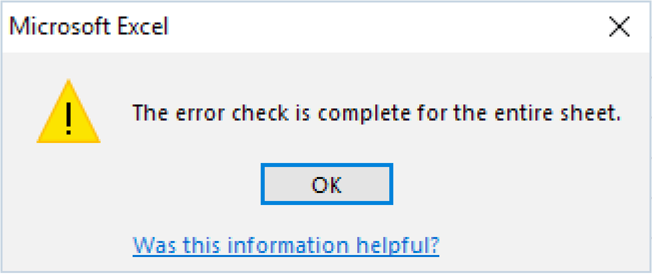
Error in cell I9

=DATE(YEAR(F9)+30,MONTH(G9),DAY(G9))

Inconsistent Formula

The formula in this cell differs from the formulas in this area of the spreadsheet.

As one formula is corrected, click the **[Next]** button to advance to the next found error. When all errors have been checked*Excel* displays a message to that effect.



The error check is complete for the entire sheet.

While still in the process of checking for error in the *Error Checking* dialog, clicking the **[Options]** button will open the*Excel Options* dialog where there are more controls relating to how *Excel* will treat errors.

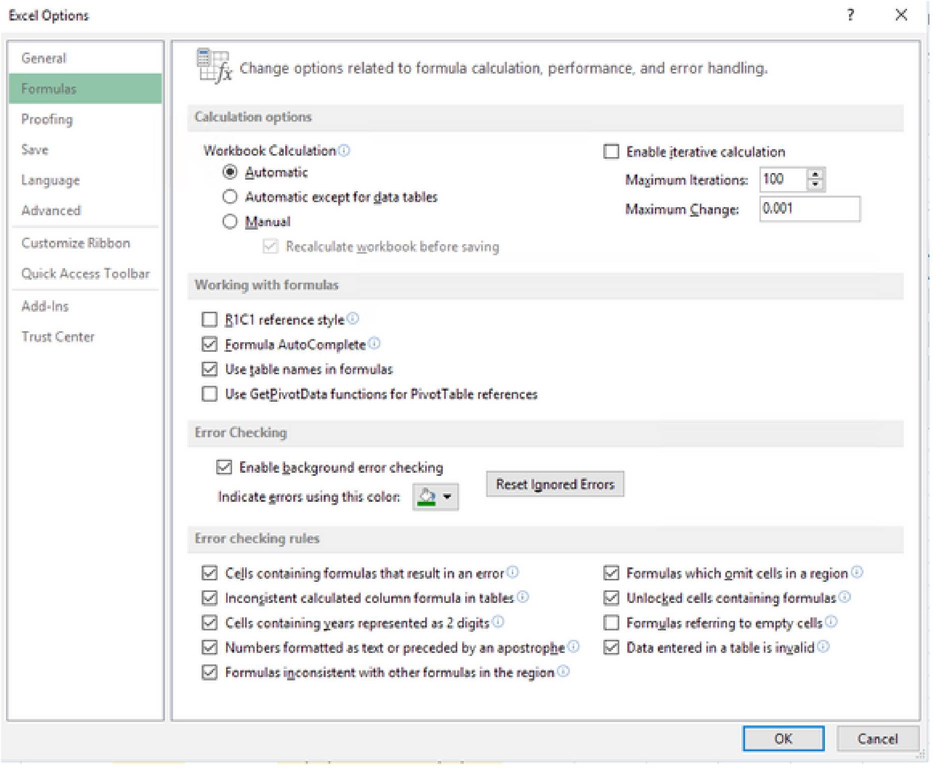
Error Checking, continued

Note

Try using the keyboard to navigate the program to access the **Options** by sequentially tapping the **ALT, F, T** keys.

Formula Options

Turning off error checking is done through the program options. When you click the **[Options]** button in the *Error Checking* dialog you can edit the program options directly. A more direct method to access the program options is to click ***File Tab***, then choose **Options**, and then the **Formulas** category.



Change options related to formula calculation, performance, and error handling.

Calculation options

Workbook Calculation
● Automatic
○ Automatic except for data tables
○ Manual
☑ Recalculate workbook before saving

□ Enable iterative calculation
Maximum Iterations: 100
Maximum Change: 0.001

Working with formulas

□ R1C1 reference style
☑ Formula AutoComplete
☑ Use table names in formulas
□ Use GetPivotData functions for PivotTable references

Error Checking

☑ Enable background error checking
Indicate errors using this color:
Reset Ignored Errors

Error checking rules

☑ Cells containing formulas that result in an error
☑ Inconsistent calculated column formula in tables
☑ Cells containing years represented as 2 digits
☑ Numbers formatted as text or preceded by an apostrophe
☑ Formulas inconsistent with other formulas in the region
☑ Formulas which omit cells in a region
☑ Unlocked cells containing formulas
□ Formulas referring to empty cells
☑ Data entered in a table is invalid

When in the options you are able to turn error checking on or off, and even change the color of the error warning triangle from withing the **Error Checking** group of controls. In the **Error Checking Rules** group of commands you are able to determine what errors will be ignored or tracked.

Cells containing formulas that result in an error:

Turning this off will stop the display of error messages

Inconsistent calculated column formula in tables:

Turning this off will stop displaying Error Checking Options smart tags for any formula that could be perceived as being inconsistent inside an *Excel* table only.

Error Checking, continued

Note
If the copied data contains a formula, this formula overwrites the data in the calculated column.

Exceptions within a calculated column in a table are:

If you type data other than a formula in a calculated column cell.

If you type a formula in a calculated column cell and then click **[Undo]** button on the Quick Access Toolbar.

If you type in a new formula within a calculated column cell containing one or more exceptions.

If you copy data into the calculated column using a different formula.

If you move or delete a cell on another worksheet area that is being referenced by a row within a calculated column.

Cells containing years represented as 2 digits: Turning off this rule will stop *Excel* from checking for unclear dates, when a cell contains dates which could be misinterpreted as the wrong century.

Numbers formatted as text or preceded by an apostrophe: Turning this rule off will stop *Excel* warning you that a numeric value could be formatted as text. When this occurs *Excel* will display the **Error Checking Options** smart tag triangle in the upper left corner of the cell. When there are numbers stored as text, it will cause unexpected sorting behaviors and prevent the values from being calculated.

Formulas inconsistent with other formulas in the region: Turning this rule off will stop *Excel* from warning you when a formula does not match the pattern of other formulas near by. In many cases, formulas that are adjacent to other formulas differ only in the references used.

Action 2- Error Checking

Instructions:	**Results/ Comments:**
1. Open the **Errors** file.	
2. Save the file as **My Errors**.	
3. Select cells **E6:E17**.	We will add a formula to average the grades here.
4. Click the **[Autosum]** button drop-down in the **Function Library Group** on the ***Formulas Tab*** and choose *Average*.	The formula is added to all the cells. You can also find the **[Autosum]** button in the **Editing Group** on the ***Home Tab***.
5. Select column **E**.	Click the column header **E**.
6. Right-click the column and choose *Insert* from the menu.	A new column is added after column **D**.
7. Select cell **E5** and type in: **Photo Corrections** press the **[Enter]** key.	The project header is added. If you want to add a soft return, use the **[ALT + Enter]** key combination.
8. In cell **E6**, type: **85** then, press the **[Enter]** key.	
9. In cell **E7**, type: **78** then, press the **[Enter]** key.	
10. Select cells **E6:E7**.	
11. Use autofill to fill in the remaining grades. Click the ***Autofill Options*** smart tag and choose *Copy Cells*.	Choosing to copy the cells will fill the cells with the number pair.
12. Select cell **F8** and click the *Error Checking Options* smart tag. Choose *Update formula to include cells*.	The original formula reads as: **=AVERAGE(B8:D8)**. After updating the formula, now reads as: **=AVERAGE(B8:E8)**.
13. Click the **[Error Checking]** button in the **Formula Auditing Group** on the ***Formula Tab***.	The *Error Checking* dialog box opens, this acts like a spell check dialog box. You can check every error that potentially exists in the spreadsheet.

Action 2- Error Checking, continued

Instructions:

14. Click the **[Update formula to include cells]** button, repeat this to correct all the errors. When all have been updated, click **[OK]** the button to close the dialog box.

15. Click the ***File Tab*** and choose ***Options***. Choose the ***Formulas*** category and examine the **Error Checking** options.

16. Click **[OK]**.

17. Save and close the file.

Results/ Comments:

Each error is displayed and you are given options on how to correct them. When all errors have been checked a dialog box lets you know the checking is completed.

The **Formula Error** options can also be accessed from within the *Error Checking* dialog box by clicking the **[Options]** button.

To close the options.

[CTRL + S] and **[CTRL + W]**.

Evaluating Formulas

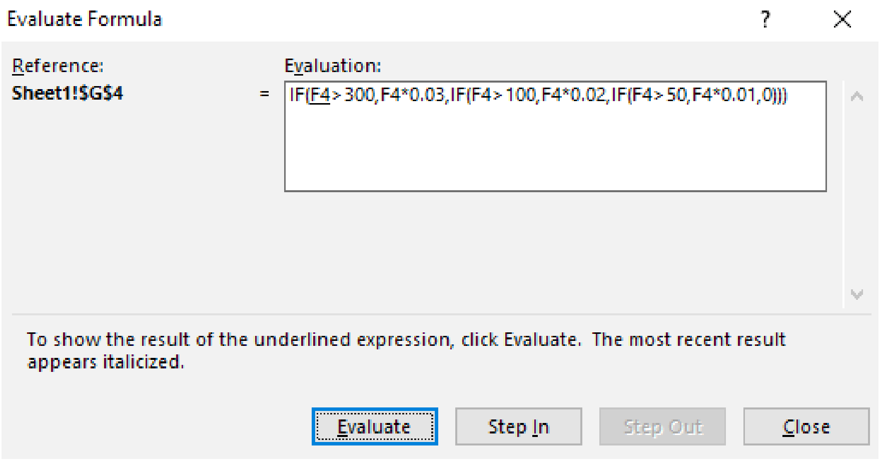
Using the Evaluate Formula Tool

The **[Evaluate Formula]** button will open an *Evaluate Formula* dialog box, allowing you to see the formula and go through it step by step. Showing each step of the formula's calculation, so you can see the logic behind the formula. This tool is also extremely useful in helping to understand where problems exist within a formula.

Select the cell containing the formula you would like to evaluate.

Click the **[Evaluate Formula]** button in the **Formula Auditing Group** of the ***Formulas Tab***.

The *Evaluate Formula* dialog box will open. In the **Evaluation:** field, the formula from the selected cell is displayed.



Evaluate Formula

Reference: Sheet1!$G$4

Evaluation: = IF(F4>300,F4*0.03,IF(F4>100,F4*0.02,IF(F4>50,F4*0.01,0)))

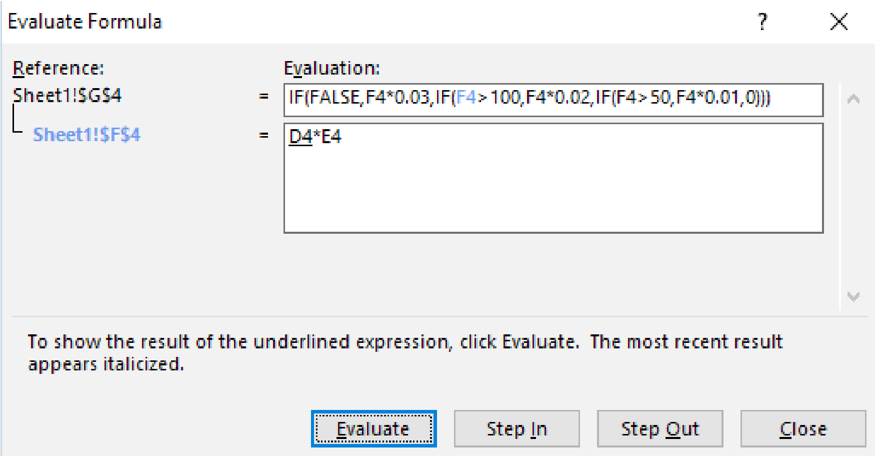
To show the result of the underlined expression, click Evaluate. The most recent result appears italicized.

Clicking the **[Evaluate]** button begins replacing cell addresses with their values and moving through the formula, step by step. Keep clicking the button until the entire formula has been shown and you see the end result.

When the result is displayed in the **Evaluation:** field, you can click the **[Restart]** button to go back through the formula again if necessary.

Evaluating Formulas, continued

Clicking the **[Step In]** button will evaluate each section of the formula that is underlined. If a cell address is underlined and it is in fact the end result of a formula, the formula in that cell address is displayed in another new window below the main **Evaluation:** field.



Evaluate Formula

Reference:		Evaluation:
Sheet1!$G$4	=	IF(FALSE,F4*0.03,IF(F4>100,F4*0.02,IF(F4>50,F4*0.01,0)))
Sheet1!$F$4	=	D4*E4

To show the result of the underlined expression, click Evaluate. The most recent result appears italicized.

Continue to click the **[Step In]** button to see the string of cells needed for your formula to function. Each connection to cells that use formulas is displayed in a new level field of the **Evaluation:** field.

Click the **[Step Out]** button to move back one level at a time until you are back to the original formula being evaluated.

Action 3- Evaluating Formulas

Instructions:	Results/ Comments:
1. Open the **Evaluation** file.	
2. Save as **My Evaluation**	
3. Select cell **H2**.	This cell contains a formula to be evaluated.
4. Click the **[Evaluate Formula]** button in the **Formula Auditing Group** on the ***Formulas Tab***.	The *Evaluate Formula* dialog box opens.
5. Examine the formula in the *Evaluate Formula* dialog box.	The formula in cell **H2** is displayed in the **Evaluation** field. The underlined portion of the formula is the active part of the formula being evaluated. As the formula is evaluated step by step, the underline changes to show what part of the formula is active .
6. Click the **[Step in]** button.	Since the value is cell **G2** results from a formula, the formula used to determine the value in cell **G2** is displayed below. Notice, it is indented to show it is a secondary formula to the formula being evaluated.
7. Click the **[Step In]** button again.	Each time the **[Step in]** button is clicked, the formula is traced back to the original data the formula is drawing on. Once all levels of the formulas have been displayed, the value returned by the formula is displayed.
8. Click the **[Step out]** buttons twice.	Click this button to step back through the interconnected formula until all indented fields are removed.
9. Click the **[Evaluate]** button.	The first cell reference is replaced by the value in the cell.

Action 3- Evaluating Formulas, continued

Instructions:	Results/ Comments:
10. Continue clicking the **[Evaluate]** button until the formula results are displayed.	Each click of the **[Evaluate]** button reveals the next step in the formula until all steps are shown and the result of the completed formula is displayed. The **[Step in]** button is only active when the cell being examined has a formula.
11. Click the **[Restart]** button to begin going through the formula again, or click the **[Close]** button to exit the dialog box.	Choose whether to re-evaluate the formula or close the dialog box.
12. Save and close the file.	**[CTRL + S]** and **[CTRL + W]**.

Watch Window

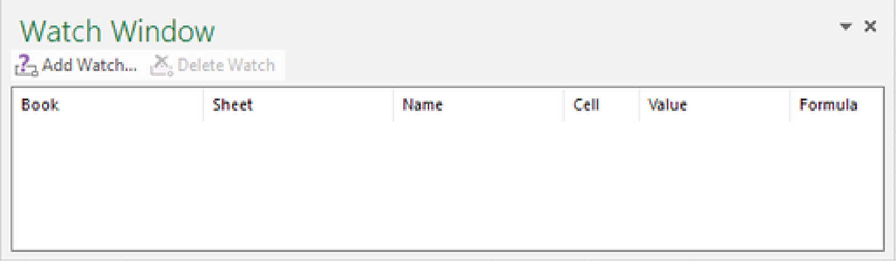
Using the Watch Window

The *Watch Window* allows you to monitor the results of a cell when another cell is changed. The Watched cell can be on the same sheet, a different sheet, or even in a different file.

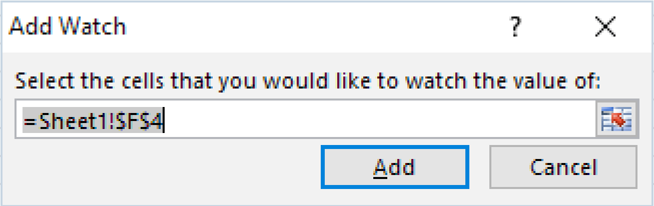
In the *Watch Window,* you can add watches to the cells that you want to track changes being made somewhere else in the file without having to scroll from one place to another. Once a *Watch* has been added to the window, it shows the workbook, worksheet, name of the cell (if the cell has a name), cell address, the cell values, and formulas. Add as many *Watches* as needed, so you are able to easily keep track of changes from multiple locations in one window. The *Watch Window* can be docked to the top, bottom, or either side of the interface by dragging it to the desired position. It can also be left as a floating window, and will remain open until it is closed using the **[Close]** button in the top right corner.

To Add a Cell to the Watch Window

Click the **[Watch Window]** button in the **Formula Auditing Group** of the ***Formulas Tab*** to open the *Watch Window* dialog box.



In the *Watch Window* dialog box, click the **[Add Watch...]** button.



Select the cells that you would like to watch the value of:

=Sheet1!$F$4

Click on the cell in the worksheet or workbook that you would like to watch.

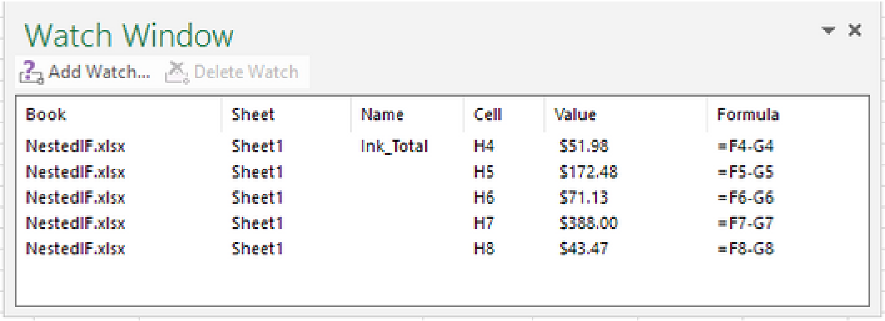
Click the **[Add]** button.

Note

If you select a range of cells, each of the selected cells will be added as it's own watch.

Watch Window, continued

As long as the file and the *Watch Window* dialog box remain open, you will see the values update when the data is altered.



Book	Sheet	Name	Cell	Value	Formula
NestedIF.xlsx	Sheet1	Ink_Total	H4	$51.98	=F4-G4
NestedIF.xlsx	Sheet1		H5	$172.48	=F5-G5
NestedIF.xlsx	Sheet1		H6	$71.13	=F6-G6
NestedIF.xlsx	Sheet1		H7	$388.00	=F7-G7
NestedIF.xlsx	Sheet1		H8	$43.47	=F8-G8

To Delete a Cell from the Watch Window

Select the Watch you would like to delete in the *Watch Window* dialog box.

Click the **[Delete Watch]** button.

When you close a file, the associated cell in the *Watch Window* dialog box will disappear. However, when the file is re-opened, the cell will reappear in the *Watch Window*.

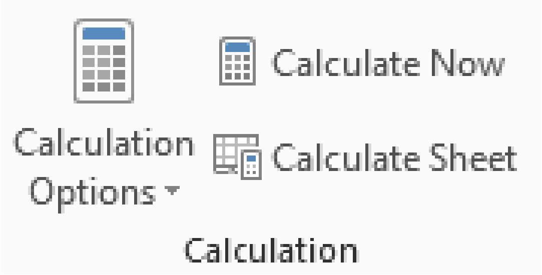
To close the *Watch Window* dialog click the **[X]** button.

Action 4 - Using the Watch Window

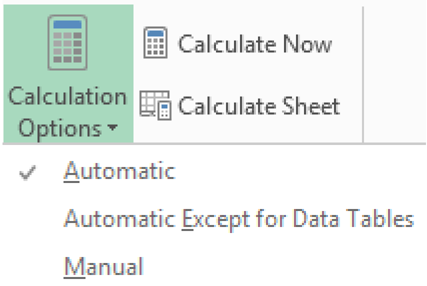
Instructions:	Results/ Comments:
1. Open the file **WatchWindow**	
2. Save as **My WatchWindow**	
3. Select cell and type: **A67 Totals**	A new header is added.
4. Select cell **G67** and enter the following formula: **=SUM(G2:G65)** then, press the **[CTRL + Enter]**.	A total of units sold is added.
5. Use autofill to populate cell **H67** with a relative copy of the formula.	A total of all sales is added.
6. Select cell **A1**.	**[CTRL + Home]** key combination takes you back to cell **A1** from anywhere in the spreadsheet.
7. Click the **[Watch Window]** button in the **Formula Auditing Group** on the ***Formulas Tab***.	The *Watch Window* opens.
8. Click the **[Add Watch...]** button.	The *Add Watch* dialog box opens.
9. Highlight the field and select cell **G67** and click the **[Add]** button.	The cell is added to the *Watch Window.* Any relevant information regarding the cell is displayed for tracking proposes.
10. Add another watch for cell **H67**.	Repeat the Add Watch process for cell **H67**.
11. Dock the Watch Window above the Formula bar.	Drag the window by the title-bar up toward the formula bar. It will snap into position.
12. Select cell **G5** and type: **25** then, press the **[Enter]** key.	Editing a cell's contents will impact the results of the formulas in cells **G67** and **H67**. These changes are shown in the *Watch Window*.
13. Save the file and leave it open.	**[CTRL + S]**.

Calculations

By default, *Excel* is set to update formula results as changes are made. There are times when having *Excel* automatically update the formula as you enter new or change data can be prove confusing. You may want the calculation to update once all your changes have been completed, instead of being done in real time. You can change the default calculation setting by using the **[Calculation Options]** button in the **Calculation Group** on the ***Formulas Tab***.



Clicking the **[Calculation Options]**button offers three choices:



Automatic: This is the default choice. The formula is recalculated any time data is updated.

Automatic Except for Data Tables: If there are Data Tables being used in the file, they are not recalculated as data is updated.

Manual: The cell will not recalculate until you decide to run the calculation.

Changing the Calculation Option

Click the **[Calculation Options]** button in the **Calculations Group** on the ***Formulas Tab*** and choose *Manual*.

Change data in the cells that the formula needs to calculate.

Calculations, continued

Click the **[Calculate Now]** button in the **Calculation Group** on the ***Formulas Tab*** to recalculate the formulas on all open sheets and all open workbooks.

-OR-

Click the **[Calculate Sheet]** button in the **Calculation Group** on the ***Formulas Tab*** to recalculate all the formulas on the *active sheet only*, not on other open sheets or open workbooks.

Both the ***Manual*** option and the **[Calculate Now]** button operate independently of the cells that are selected. Both commands apply to *all* cells, *all* sheets, and *all* files.

Both commands affect the entire *Excel* program and neither command is limited to any specific cell, cells, sheet, sheets, workbook, or workbooks.

Changing the ***Manual*** option on the ribbon is the same as changing the ***Manual*** option by going to the ***File Tab***, selecting **Options**, then clicking the ***Formulas Tab*** on the left, and choosing *Workbook Calculation* under the ***Calculation options*** section.

Instructions:	Results/ Comments:
1. The file should still be **Watch Window** open. If not, reopen it.	
2. Close the *Watch Window*.	Click the **[X]** button at the upper right corner of the window.
3. Select cell **G65** and type: **25** then, press the **[Enter]** key.	The values in cells **G67:H67** change to reflect the changes to the data.
4. Select cells **G67:H67**.	These cells currently are set to calculate automatically.
5. Click the **[Calculation Options]** button and choose .*Manual*	This turns off the automatic calculation for these cells.
6. Select cell **G65** and type: **50** then, press the **[Enter]** key.	Notice the values in cells **G67:H67** do not update to reflect the changes to the data.
7. Click the **[Calculate Sheet]** button.	The values in cells **G67:H6** **7** are updated.
8. Try changing the value in cell **G65** again.	The values in cells **G67:H6** **7** are not updated.
9. Recalculate the sheet.	The values in cells **G67:H67** are updated.
10. Save and close the file.	**[CTRL + S]** and **[CTRL + W]**.

TCW BOOK CODES

Excel Level 1 L-1 Excel Level 2 L-2 Excel Level 3 L-3 Excel Formulas FM Excel Data Analysis DA Excel Charts CH Excel PivotTables PT Excel Data Analysis with PowerPivot PPT

MICROSOFT OFFICE EXCEL ASSOCIATE EXAM MO-200

Import data into workbooks	
Import data from .txt file	DA
Import data from .csv files	DA

Navigate within workbooks	
Search for data within a workbook	L-
Navigate to named cells, ranges, or workbook elements	1
Insert and remove hyperlinks	L-

2

Format worksheets and workbooks	
Modify page setup	L- L-
Adjust row height and column width	3 1
Customize headers and footers	L-

1

Customize options and views	L-
Customize the Quick Access toolbar	1 L-
Display and modify workbook content in different views	1
Freeze worksheet rows and columns	L-
Change window views	2
Modify basic workbook properties	
Display formulas	L-

2

Configure content for collaboration	L- 2
Set a print area	L- L-
Save workbooks in alternative file	1
formats Configure print settings	2 L-
Inspect workbooks for issues	1 L- 1

L-
1
L-
1

TCW BOOK CODES

Excel Level 1 L-1 Excel Level 2 L-2 Excel Level 3 L-3 Excel Formulas FM Excel Data Analysis DA Excel Charts CH Excel PivotTables PT Excel Data Analysis with PowerPivot PPT

Manipulate data in worksheets	
Paste data by using special paste options	L-
Fill cells by using Auto Fill	1
Insert and delete multiple columns or	L-
rows Insert and delete cells	1

L-

Format cells and ranges	1
Merge and unmerge cells	L- L-
Modify cell alignment, orientation, and indentation	1 1
Format cells by using Format Painter	L-
Wrap text within cells	1
Apply number formats	
Apply cell formats from the Format Cells dialog box	L- 1
Apply cell styles	L-
Clear cell formatting	1

L-

Define and reference named ranges	1
Define a named	L-2 / FM L-
range Name a table	DA

1

Summarize data visually	L-
Insert Sparklines	L- 1
Apply built-in conditional formatting	2
Remove conditional formatting	L- L- 1

2

Create and format tables	L-
Create Excel tables from cell	2 L-
ranges Apply table styles	2
Convert tables to cell ranges	L-

2

Modify tables	L-
Add or remove table rows and	2 L-
columns Configure table style options	2
Insert and configure total rows	L-

2

L-

2

TCW BOOK CODES

Excel Level 1 L-1 Excel Level 2 L-2 Excel Level 3 L-3 Excel Formulas FM Excel Data Analysis DA Excel Charts CH Excel PivotTables PT Excel Data Analysis with PowerPivot PPT

Filter and sort table data	
Filter records	L-
Sort data by multiple columns	2

L-

2

Insert references	
Insert relative, absolute, and mixed	L-
references Reference named ranges and named tables in	1

formulas L-

Calculate and transform datas	2
Perform calculations by using the AVERAGE(), MAX(), MIN(), and SUM() functions	L-1
Count cells by using the COUNT(), COUNTA(), and COUNTBLANK() functions	DA
Perform conditional operations by using the IF() function	FM

Format and modify text	
Format text by using RIGHT(), LEFT(), and MID() functions	DA
Format text by using UPPER(), LOWER(), and LEN() functions	DA
Format text by using the CONCAT() and TEXTJOIN() functions	DA

Create charts	
Create charts	L-2 / CH
Create chart sheets	L-2 / CH

Modify charts	
Add data series to charts	L-2 / CH
Switch between rows and columns in source data	L-2 / CH
Add and modify chart elements	L-2 / CH

TCW BOOK CODES

Excel Level 1 L-1 Excel Level 2 L-2 Excel Level 3 L-3 Excel Formulas FM Excel Data Analysis DA Excel Charts CH Excel PivotTables PT Excel Data Analysis with PowerPivot PPT

MICROSOFT OFFICE EXCEL EXPERT EXAM MO-201

Manage workbooks	
Copy macros between workbooks	L-
Reference data in other workbooks	3
Enable macros in a workbook	L-
Manage workbook versions	3

L-

Prepare workbooks for collaboration	3
Restrict editing	L- L-
Protect worksheets and cell ranges	2 2
Protect workbook structure	L-
Configure formula calculation options	2
Manage comments	L-

2

Use and configure language options	FM
Configure editing and display languages	L- L-
Use language-specific features	2 1

L-

Fill cells based on existing data	1
Fill cells by using Flash Fill	L-
Fill cells by using advanced Fill Series options	1

L-

Format and validate data	2
Create custom number formats	L-1
Configure data validation	L-3 / FM
Group and ungroup data	L-3
Calculate data by inserting subtotals and totals	L-3
Remove duplicate records	DA

TCW BOOK CODES

Excel Level 1 L-1 Excel Level 2 L-2 Excel Level 3 L-3 Excel Formulas FM Excel Data Analysis DA Excel Charts CH Excel PivotTables PT Excel Data Analysis with PowerPivot PPT

Apply advanced conditional formatting and filtering	
Create custom conditional formatting rules	L-
Create conditional formatting rules that use formulas	2
Manage conditional formatting rules	L-

2

Perform logical operations in formulas	
Perform logical operations by using nested functions including the IF(), IFS(), SWITCH(),	L- 2 FM
SUMIF(), AVERAGEIF(), COUNTIF(), SUMIFS(), AVERAGEIFS(), COUNTIFS(), MAXIFS(),	FM
MINIFS(), AND(), OR(), and NOT() functions	FM

Look up data by using functions	
Look up data by using the VLOOKUP(), HLOOKUP(), MATCH(), and INDEX() functions	FM

Use advanced date and time functions	
Reference date and time by using the NOW() and TODAY() functions	FM
Calculate dates by using the WEEKDAY() and WORKDAY() functions	FM

Perform data analysis	
Summarize data from multiple ranges by using the Consolidate feature	L-
Perform what-if analysis by using Goal Seek and Scenario Manager	3
Forecast data by using the AND(), IF(), and NPER() functions	L-
Calculate financial data by using the PMT() function	3

FM

FM

TCW BOOK CODES

Excel Level 1 L-1 Excel Level 2 L-2 Excel Level 3 L-3 Excel Formulas FM Excel Data Analysis DA Excel Charts CH Excel PivotTables PT Excel Data Analysis with PowerPivot PPT

Troubleshoot formulas	
Trace precedence and dependence	FM
Monitor cells and formulas by using the Watch Window	FM
Validate formulas by using error checking rules	FM
Evaluate formulas	FM

Create and modify simple macros	
Record simple	L-
macros Name simple	3
macros Edit simple	L-
macros	3

Create and modify advanced charts	L-
Create and modify dual axis charts	3 CH
Create and modify charts including Box & Whisker, Combo, Funnel, Histogram, Map, Sunburst, and Waterfall charts	CH

Create and modify PivotTables	
Create PivotTables	PT
Modify field selections and	PT
options Create slicers	PT
Group PivotTable data	PT
Add calculated fields	PT
Format data	PT

Create and modify PivotCharts	
Create PivotCharts	PT
Manipulate options in existing PivotCharts	PT
Apply styles to PivotCharts	PT
Drill down into PivotChart details	PPT