Excel
Cookbook

Recipes for Mastering Microsoft Excel

Dawn Griffiths

O'REILLY®

Excel Cookbook

Filled with tips, tricks, and techniques, this easy-to-use book is the perfect Excel resource. You'll find more than 350 recipes for over a dozen topics covering formulas, PivotTables, charts, Power Query, and more. Each recipe poses a particular problem and outlines a solution that you can put to use right away—without having to comb through tutorial pages.

Whether you're a data analyst, project manager, financial analyst, or regular Excel user, author Dawn Griffiths directs you straight to the answers you need. Ideal as a quick reference, *Excel Cookbook* is also perfect for learning how to work in a more efficient way, leading to greater productivity on the job. With this book, you'll jump in and get answers to your questions—*fast*.

- Build compelling charts and use Sparklines, 3D Maps, and other visualizations
- Use PivotTables to slice, dice, and summarize datasets
- Perform statistical and financial analyses using formulas, Forecast Sheets, the Analysis ToolPak, and more
- Master dynamic array functions such as SEQUENCE, TEXTSPLIT, and FILTER
- Use Power Query to import, shape, and combine datasets
- Create custom functions using LAMBDA formulas
- Use developer options to write VBA code and create custom UserForms

"The ultimate desk reference, offering a thorough blend of traditional and modern Excel solutions. Perfect for daily users seeking to solve a wide range of problems."

—**George Mount**
Founder, Stringfest Analytics

"Dawn's knowledge of Excel is phenomenal. She explains everything in a way that is clear and easy to follow. I had a good knowledge of Excel already but learned a lot from this book."

—**Ann Brumwell**
Financial controller,
Denfield Advertising & Marketing

Dawn Griffiths is an author and trainer who has taught Excel to thousands of students worldwide. Her previous books include *Head First Statistics, Head First Android Development, Head First Kotlin, Head First C,* and *React Cookbook*.

linkedin.com/company/oreilly-media
youtube.com/oreillymedia

Excel Cookbook
Recipes for Mastering Microsoft Excel

Dawn Griffiths

Table of Contents

Preface

This book contains more than 350 recipes designed to help you get the most out of Microsoft Excel, whether you're writing formulas, importing data, wrangling with PivotTables, or solving complex optimization problems. The recipes result from my over 20 years of experience using Excel and teaching it to thousands of students worldwide.

Each recipe solves a specific problem using one or more techniques, and nearly every recipe uses examples to illustrate the teaching points. You can test the examples for yourself and then put them into practice by applying them to your situation.

You don't have to read the chapters or their recipes in any particular order, so you can read the book from cover to cover or start reading anywhere. Many recipes reference others, so you can quickly jump from one to another.

The book includes the latest features of Excel 365 (at the time of writing), such as `LAMBDA` functions. However, it's also backward compatible with earlier versions. For example, it covers the `XMATCH` function available in Excel 2021 and Excel 365 and the `MATCH` function available in earlier versions. This book also includes differences in using Excel for Microsoft Windows and Excel for Mac, such as how to change the default settings.

Conventions Used in This Book

The following typographical conventions are used in this book:

Italic
> Indicates new terms, URLs, email addresses, filenames, and file extensions.

`Constant width`
> Used for program listings, as well as within paragraphs to refer to program elements such as variable or function names, databases, data types, environment variables, statements, and keywords.

Constant width bold
> Shows commands or other text that should be typed literally by the user.

Constant width italic
> Shows text that should be replaced with user-supplied values or by values determined by context.

This element signifies a tip or suggestion.

This element signifies a general note.

This element indicates a warning or caution.

Using Code Examples

Supplemental material (code examples, exercises, etc.) is available for download at *https://github.com/dogriffiths/excelcookbook*.

If you have a technical question or a problem using the code examples, please send email to *support@oreilly.com*.

This book is here to help you get your job done. In general, if example code is offered with this book, you may use it in your programs and documentation. You do not need to contact us for permission unless you're reproducing a significant portion of the code. For example, writing a program that uses several chunks of code from this book does not require permission. Selling or distributing examples from O'Reilly books does require permission. Answering a question by citing this book and quoting example code does not require permission. Incorporating a significant amount of example code from this book into your product's documentation does require permission.

We appreciate, but generally do not require, attribution. An attribution usually includes the title, author, publisher, and ISBN. For example: "*Excel Cookbook* by Dawn Griffiths (O'Reilly). Copyright 2024 Dawn Griffiths, 978-1-098-14332-9."

If you feel your use of code examples falls outside fair use or the permission given above, feel free to contact us at *permissions@oreilly.com*.

O'Reilly Online Learning

For more than 40 years, *O'Reilly Media* has provided technology and business training, knowledge, and insight to help companies succeed.

Our unique network of experts and innovators share their knowledge and expertise through books, articles, and our online learning platform. O'Reilly's online learning platform gives you on-demand access to live training courses, in-depth learning paths, interactive coding environments, and a vast collection of text and video from O'Reilly and 200+ other publishers. For more information, visit *https://oreilly.com*.

How to Contact Us

Please address comments and questions concerning this book to the publisher:

O'Reilly Media, Inc.
1005 Gravenstein Highway North
Sebastopol, CA 95472
800-889-8969 (in the United States or Canada)
707-827-7019 (international or local)
707-829-0104 (fax)
support@oreilly.com
https://www.oreilly.com/about/contact.html

We have a web page for this book, where we list errata, examples, and any additional information. You can access this page at *https://oreil.ly/excel-cookbook*.

For news and information about our books and courses, visit *https://oreilly.com*.

Find us on LinkedIn: *https://linkedin.com/company/oreilly-media*

Watch us on YouTube: *https://youtube.com/oreillymedia*

Acknowledgments

My heartfelt thanks go to my fantastic editor, Corbin Collins, for telling me when my sentences contained all the right words but not necessarily in the right order. He's been an absolute dream to work with and made me feel valued. I can't thank him enough. I also want to thank David Michelson for commissioning this book, Theresa Jones for helping me with my AsciiDoc queries, Chris Faucher for herding the book

Workbooks, Worksheets, and Cells

Excel has many features and tools to help you save time and produce slick, polished spreadsheets.

This first chapter focuses on features generally useful when working with Excel files, such as themes and cell styles, defining templates, creating custom number formats, and using conditional formatting. It also covers time-saving tools such as Auto Fill, Flash Fill, and the Quick Access Toolbar and includes hidden gems such as custom lists and the Go To Special tool.

1.1 Using Themes

Problem

You have a workbook and want to apply a consistent look and feel to it.

Solution

When you create a new workbook, Excel applies a *theme* to it: a set of colors, fonts, and effects designed to give your workbook a consistent appearance. Excel includes several predefined themes, and you can switch to a different one by choosing Page Layout ⇒ Themes ⇒ Themes and selecting a theme from the Theme gallery.

You can also create a custom theme as follows:

1. Select a set of theme colors by choosing Page Layout ⇒ Themes ⇒ Colors and selecting a predefined color set. Alternatively, create a new custom color set by choosing Page Layout ⇒ Themes ⇒ Colors and selecting the Customize Colors option.

2. Select a set of fonts—a heading font and a body font—by choosing Page Layout ⇒ Themes ⇒ Fonts and selecting a predefined font set. Alternatively, create a new custom font set by choosing Page Layout ⇒ Themes ⇒ Fonts and selecting the Customize Fonts option.

3. Select a set of effects to change the appearance of any drawing objects by choosing Page Layout ⇒ Themes ⇒ Effects and selecting an effect; note that you can use only predefined effects and can't create your own.

4. Save the colors, fonts, and effects as a theme by choosing Page Layout ⇒ Themes ⇒ Themes ⇒ Save Current Theme, and selecting a save location.

Once you've created a custom theme, apply it to a workbook by choosing Page Layout ⇒ Themes ⇒ Themes ⇒ Browse for Themes.

Discussion

A theme lets you apply a consistent look and feel to your workbook. You can choose one of Excel's predefined themes or create a custom one to apply a corporate or personal style.

See Also

To change the default theme Excel applies to new workbooks, see Recipe 1.7.

1.2 Using Cell Styles

Problem

You've applied a theme to a workbook and want to use it to style selected cells.

Solution

You can apply a theme's fonts and colors to cells using cell styles.

A *cell style* is a defined set of formatting characteristics—such as font, font size, and shading—that help you consistently style cells. Each style uses the fonts and colors associated with the workbook's theme (see Recipe 1.1), so if you switch to a different theme, the cell styles update automatically.

To apply a cell style, select the cells you want to format, choose Home ⇒ Styles ⇒ Cell Styles, and select a style. The styles are grouped according to purpose as follows:

Good, Bad, and Neutral
    These styles let you format cells containing particularly good or bad results.

Data and Model
> These are for cells containing calculations, warnings, notes, or for cells that need input values.

Titles and Headings
> This includes styles for different heading levels and a Total cell style.

Themed Cell Styles
> These styles offer color accents based on the theme's color palette.

Number Format
> Use these styles to quickly format a number as a currency or percentage or to use commas as a thousands separator.

If you want to modify a cell style, you can do so by choosing Home ⇒ Styles ⇒ Cell Styles, right-clicking the style you want to modify, then choosing the Modify option from the shortcut menu. This opens the Style dialog box, which you can use to adjust the cell's style with Excel's cell formatting options (see Recipe 1.3).

To create a new style, you can either copy an existing style by right-clicking it and choosing the Duplicate option or create one from scratch by choosing Home ⇒ Styles ⇒ Cell Styles and clicking the New Cell Style option.

> By default, custom styles are available only in the workbook in which you create them. You can, however, import styles from another open workbook by choosing Home ⇒ Styles ⇒ Cell Styles ⇒ Merge Styles.

Discussion

You can use this recipe with Recipe 1.1 to apply a consistent look and feel to a workbook's cells.

1.3 Formatting Cells

Problem

You want to format one or more cells in a worksheet and want to know how.

Solution

Excel includes many options for modifying a cell's format, such as changing the alignment or orientation of its contents, adding a border or background color, and adjusting the font. You can add these options to a cell style (see Recipe 1.2) or apply them to cells ad hoc.

It's generally better to format cells using cell styles since these automatically update to fit your workbook's theme.

You generally update the format using two main methods:

- Selecting the cell and using the available options in the Home menu.
- Using the Format Cells dialog box. You can open this by right-clicking the cells you want to format and selecting Format Cells, choosing Home ⇒ Cells ⇒ Format ⇒ Format Cells, or clicking the Format button in the Style dialog box (see Recipe 1.2).

Discussion

The Home menu and Format Cells dialog box offer convenient ways of modifying a cell or cell style's appearance. Try experimenting with the available options.

1.4 Formatting a Cell's Value

Problem

You want to change how a cell's value is formatted using one of Excel's predefined formats.

Solution

Suppose you have cells that contain numbers, dates, or times, and you want to change how they're formatted. You can do so by applying a number format; for example, you can format numbers as percentages, fractions, or currency and change a date's appearance to include a time component.

To change a cell's number format, you select the cell—or cells—you want to format, open the Format Cells dialog box (see Recipe 1.3), and select a format from the Number tab. The available options include the following:

General
> This is the default format Excel applies to most numbers so they appear how you type them. The format uses scientific (exponential) notation for large numbers with 12 or more digits, and if the cell isn't wide enough, it rounds the numbers.

Number
> This format is for general numbers. You can specify the number of decimal places to display, whether to use commas, and how to display negative numbers.

Currency and Accounting

These options format a number as a monetary value, where the Accounting format lines up the decimal points and currency symbols in a column, and the Currency format doesn't. Excel uses the Currency format by default for any numbers you enter as a monetary value (for example, $14).

Date and Time

These provide date and time formats based on a locale. Excel uses these formats by default for any values it recognizes as a date or time.

Percentage

This option lets you specify a number as a percentage using a specified number of decimal places. Excel uses this format by default for any numbers you enter as a percentage, (for example, 12%).

Fraction

Use this option to format a number as a fraction. Excel uses this format by default for any numbers you enter as a fraction (for example, 1 1/2).

Scientific

This option formats numbers using scientific (exponential) notation, where you can specify the number of decimal places. Excel uses this format by default for any numbers you enter using scientific notation (for example, 1.23E+10).

Text

This treats the cell value as text, displaying it how it's entered. You should use this option only for values you don't want to use in calculations since it can lead to misleading results; if you type a number into a cell with a Text format, Excel may store the number as a text value, which number functions such as SUM interpret as 0.

Special

This lets you format a number as a zip code, phone number, or Social Security number, depending on the locale.

Custom

This option lets you apply a custom format (see Recipe 1.5).

Changing a cell's format modifies only its appearance, leaving its underlying value unchanged. To determine a cell's actual value, select the cell and look in the formula bar, change the format to General, or choose Formulas ⇒ Formula Auditing ⇒ Show Formulas.

Discussion

This recipe offers an overview of Excel's predefined number formats and what they're for. Formatting numbers is generally a good idea since it makes them more consistent and easier to read.

1.5 Defining a Custom Number Format

Problem

You want to define your own format for numbers, dates, and times instead of using one of Excel's predefined formats.

Solution

Suppose you have cells containing numbers, dates, or times and want to customize their appearance. You can do so by defining a custom number format.

You define a custom number format by opening the Format Cells dialog box (see Recipe 1.3), selecting the Number tab, choosing the Custom option, and typing the format in the Type box. You can either select and modify an existing format or create one from scratch.

Each custom number format has one to four code sections, each one separated by a semicolon. If you supply a single code section, Excel applies that format to all numbers, so the custom number format `0`, for example, displays all numbers as integers. If you supply two code sections, Excel uses the first to format positive numbers and zeros and the second to format negative numbers. The custom number format `0.00;` `[Red]-0.00`, for example, contains two code sections: the first formats positive numbers and zeros to two decimal places, and the second formats negative numbers to two decimal places, includes a minus sign, and makes the font color red. If you supply four code sections, Excel uses the sections to format positive numbers, negative numbers, zeros, and text, in that order.

You control how numbers are displayed using the following characters:

`#` *(hash) and* `0` *(zero)*

Use `#` to specify a number's significant digits, so the custom number format `#.##` displays 1.678 as `1.68` and 0.6 as `.6`. Use `0` to include extra zeros, so the custom number format `0.##` displays 0.678 as `0.68`, and the format `000.00` displays 1.2 as `001.20`.

? (question mark)

Use ? to specify significant digits and align numbers by the decimal point. So the custom number format 0.?? displays 123 as 123. and 1.678 as 1.68, lining up the decimal points.

, (comma)

Use a comma as a thousands separator to scale a number by a multiple of 1,000. So the custom number format #,##0.00 displays 1234 as 1,234.00, and the format 0, displays 1234 as 1.

% (percent)

Use % to display a number as a percentage, so the custom number format 0% displays 123 as 12300%.

_ (underscore)

Use an underscore followed by a character to display a space the same width as the character. The format _(0.00_);(0.00), for example, puts negative numbers in parentheses and adds a parenthesis-width space before and after any positive numbers so the numbers line up.

* (asterisk)

To repeat a character so that it fills the remaining width of the cell, prefix it with an asterisk, so the format *0# adds leading zeros.

Figure 1-1 shows some examples of using custom number formats to control how numbers are displayed.

General	#.##	0.##	#,##0.00	000.00	0.??	0,	0%
123	123.	123.	123.00	123.00	123.	0	12300%
12345	12345.	12345.	12,345.00	12345.00	12345.	12	1234500%
0.6	.6	0.6	0.60	000.60	0.6	0	60%
1.678	1.68	1.68	1.68	001.68	1.68	0	168%
12345.678	12345.68	12345.68	12,345.68	12345.68	12345.68	12	1234568%

Figure 1-1. Examples of using custom number formats

Placeholders after a decimal point specify how many decimal places to round a number to. All digits to the left of the decimal point are displayed, irrespective of the number of placeholders.

You can also add extra characters to a number format. To add a text string, enclose it in double quotation marks and add it to the relevant code section. The format `"Balance: "$0.00` prefixes a dollar amount with the text *Balance:*. The double quotes aren't needed for many commonly used single characters, such as $, (,), +, and -, so you can type these directly in the relevant code section. So the format `0.00;(0.00)` puts any negative numbers in parentheses, and `$0.00` includes a dollar symbol, positioned according to your system's regional settings (see Figure 1-2).

General	0.00;(0.00)	$0.00	"Balance: "$0.00	"Balance: "$0.00;"Balance: "-$0.00	_(0.00_);(0.00)	*0#
123	123.00	$123.00	Balance: $123.00	Balance: $123.00	123.00	000123
-123	(123.00)	-$123.00	-Balance: $123.00	Balance: -$123.00	(123.00)	-00 123
0	0.00	$0.00	Balance: $0.00	Balance: $0.00	0.00	000000

Figure 1-2. Adding extra characters to custom number formats

If you want to display the default format for any code section, use `General` instead of a format code. For example, the custom format `General;-General;0;General` displays zeros as `0` and other values using the default format, prefixing negative numbers with a minus sign. You can also use the `@` character as a placeholder for any entered text. See Figure 1-3 for some examples that use these format codes.

General	General;-General;0;General	General;(General);0;@	General;-General;0;"Text: "@
123		123	123
-123		-123	(123)	-123
0		0	0	0
Hello	Hello		Hello	Text: Hello

Figure 1-3. Examples of the `General` format and `@`

To change the font color for any code section, put the name of the color in square brackets and add it to the start of the section. The possible options are [Black], [Blue], [Cyan], [Green], [Magenta], [Red], [White], and [Yellow], so the format code [Green]0.00;[Red]-0.00;0;@ uses green for positive numbers, red for negative numbers, and the default color for zeros and text. You can also refer to a color by number using the syntax [ColorN], where N is a number from 1 to 56 that refers to one of the 56 colors in the standard color palette. For example, [Color10] refers to a different shade of green than using [Green].

If you want to change the color of values subject to some condition, put the condition in a separate set of square brackets. The format [Red][<100]#0;[Blue][>=100]#0 changes the font color to red for numbers less than 100 and blue for numbers greater than or equal to 100. Similarly, the format [Red][<100]#0;[Blue][>1000]#0;#0 changes the font color to red for numbers less than 100, blue for numbers greater than 1,000, and uses the default font color for all other numbers.

You can specify only up to two conditions in a custom number format. See Recipe 1.9 for a more flexible alternative.

You can also use custom number formats to format dates and times since, behind the scenes, dates and times are stored as numbers (see "Discussion" on page 136). You format the different components of a date/time value as follows:

Years
> Use yy to format the year using two digits and yyyy to format it using four digits.

Months
> Use m to format the month using one or two digits (1 to 12), mm to format it as two digits (01 to 12), mmm to display its short name (Jan to Dec), mmmm to display its full name (January to December), and mmmmm to display its initials (J to D).

Days
> Use d to format the day using one or two digits (1 to 31), dd to format it using two digits (01 to 31), ddd to display the short name of its weekday (Sun to Sat), and dddd to display the full name (Sunday to Saturday).

Hours
> Use h to format the hours using one or two digits (0 to 23) and hh to display two digits (00 to 23). By default, hours are based on a 24-hour clock; use AM/PM to use a 12-hour clock instead.

Minutes
> Use m to format the minutes using one or two digits (0 to 59) and mm to display two digits (00 to 59).

Seconds
> Use s to format the seconds using one or two digits (0 to 59) and ss to display two digits (00 to 59). You can also include hundredths of a second using ss.00.

Elapsed time
> You can display elapsed time by putting the relevant time component in square brackets. The format [h]:mm shows the elapsed time in hours and minutes, while [mm]:ss shows it in minutes and seconds.

See Figure 1-4 for examples of using date and time format codes.

dd mmmm yyyy hh:mm:ss	d-mmm-yyyy h:mm:ss AM/PM	dddd, mm/dd/yyyy
01 November 2023 16:45:30	1-Nov-2023 4:45:30 PM	Wednesday, 11/01/2023

Figure 1-4. Using custom number formats to control how dates and times are displayed

Discussion

This recipe describes how to define a custom number format, which is handy if none of Excel's predefined formats (see Recipe 1.4) display cell values how you want.

> You can skip sections using semicolons (;) to hide any values in that section. The format `0;-0;;@`, for example, hides zeros, while using `;;;` hides all values.

1.6 Merging Cells

Problem

You want to create a larger cell by merging several cells.

Solution

Suppose you want to make a cell larger by making it span more than one row and/or column. You can do so by merging cells as follows:

1. Select the cells you want to merge.
2. Choose Home ⇒ Alignment ⇒ Merge & Center to merge the cells into one and center its contents. Alternatively, open the Merge & Center drop-down menu and select Merge Cells to merge the cells without centering the contents or Merge Across to merge the selected cells in rows.

To unmerge cells, choose Home ⇒ Alignment, open the Merge & Center command's drop-down menu, and select Unmerge.

Discussion

This recipe shows how to create a larger cell spanning multiple rows and/or columns; this can be handy if, for example, you want to insert text or a formula that's relevant to multiple rows. It's also handy if you want to increase the size of a sparkline; see Recipe 13.10.

1.7 Creating Templates

Problem

You want to be able to quickly create a workbook that uses a specified theme or contains prefilled cells.

Solution

Suppose you want Excel to automatically apply a specific theme to each new workbook you create, or you want to prefill new workbooks with the same layout, formatting, and formulas. In these situations, you can use a *template*: a partially completed workbook as a starting point for other workbooks.

Before creating the template, you must first set or find the location of the default personal templates folder. How you do this depends on whether you're using Excel for Windows or Mac.

If you're using Windows, follow these steps (see Figure 1-5):

1. In Excel, choose File ⇒ Options ⇒ Save.

2. Make a note of the folder in the Default personal templates location box, which is in the Save Workbooks section. If this box is empty, you'll need to type the location yourself; this is typically *c:\Users\<UserName>\Documents\Custom Office Templates*, where *<UserName>* is your Windows username.

3. Click OK if you've made any changes, or Cancel if you haven't.
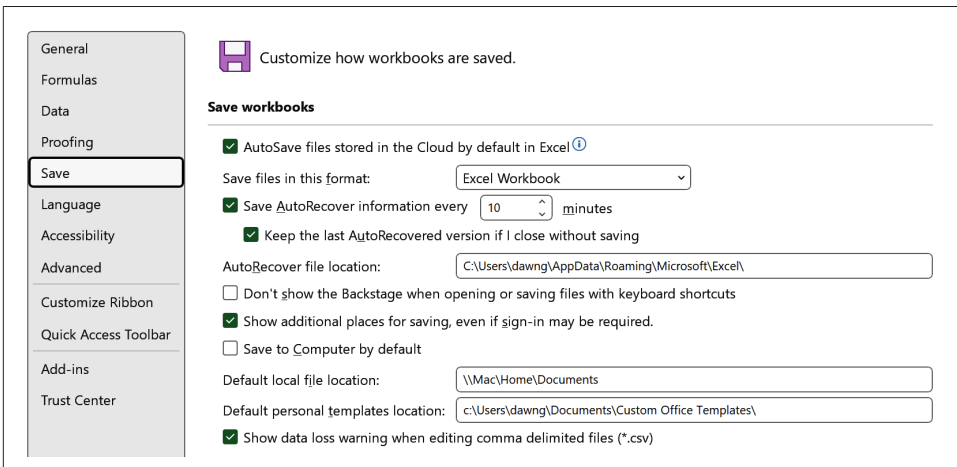
Figure 1-5. The Windows Default personal templates location box is toward the end of the Save Workbooks section

If you're using Excel for Mac:

1. Open Word[1] and choose Word ⇒ Preferences ⇒ File Locations.

2. Make a note of the folder in the User templates box; this is typically */Users/<User-Name>/Library/Group Containers/UBF8T346G9.Office/User Content/Templates*, where *<UserName>* is your username.

3. Close the File Locations dialog box.

Once you've made a note of the default personal templates folder, you can create the template as follows:

1. Create an Excel workbook that includes everything you want to appear in the template. For example, you may want to apply a custom theme or prefill some of the cells with formulas.

2. Choose File ⇒ Save As if you're using Excel for Windows, or File ⇒ Save as Template if you're using Excel for Mac. Browse to the default personal templates folder and change the Save As Type to Excel Template—or Excel Macro-Enabled Template if the file contains macros (see Chapter 18).

3. Click Save to save the template; then close the file.

To create a new workbook based on the template, choose File ⇒ New and select your template from the Personal section.

Discussion

Custom templates offer a handy way of prefilling cells in a new workbook or automatically applying a default theme. They're helpful in situations where, for example, you make a copy of an older workbook and edit the parts you need to change.

This recipe describes how to create a workbook template. To create a chart template, see Recipe 12.22.

1.8 Protecting Excel Files, Workbooks, Worksheets, and Cells

Problem

You want to protect a workbook and need to know Excel's available options.

---

1 Yes, Word, not Excel.

Solution

Excel includes various features to help protect your work from accidental changes, depending on what you want to achieve and your version of Excel.

To add passwords that control whether users have read-only access to an Excel file or can modify it, use file-level protection. To find these options, open the workbook and choose File ⇒ Info ⇒ Protect Workbook in Excel for Windows, or use the options in the File menu in Excel for Mac.

> Using a password to protect your work doesn't guarantee 100% security, especially when using older versions of Excel. Be careful when sharing files and passwords with others, and consider storing your files in a secure location.

Use workbook-level protection to control what users can do inside a workbook's structure, such as adding or deleting worksheets. You can enable this option by choosing Review ⇒ Protect ⇒ Protect Workbook.

To control what users can do in a worksheet, use worksheet-level protection. You can enable this option by choosing Review ⇒ Protect ⇒ Protect Sheet or by right-clicking the worksheet's tab and choosing Protect Sheet.

To protect a cell, apply a cell style (see Recipe 1.2) or format that includes protection. Set protection at this level by opening the Format Cells dialog box (see Recipe 1.3) and using the options on the Protection tab to lock or hide the cell.

> You can hide rows, columns, and worksheets by right-clicking them and selecting Hide. Alternatively, select the rows, columns, or worksheets you want to hide and choose Home ⇒ Cells ⇒ Format ⇒ Hide & Unhide.

Discussion

This recipe offers a helpful overview of Excel's built-in protection features, which you can use to prevent users from accidentally changing your work.

1.9 Using Conditional Formatting

Problem

You want to format one or more cells differently based on their contents.

Solution

Conditional formatting lets you format a cell range based on each cell's contents. You can use it to identify large, small, or intermediate values, for example, or highlight cells that meet a certain set of rules that you specify.

To apply conditional formatting, select the range you want to format and choose Home ⇒ Styles ⇒ Conditional Formatting, and select one of the following options (see Figure 1-6):

Highlight Cells Rules
> This option lets you apply a format (see Recipe 1.3) to cells whose values are within a specific numeric range, contain a given text string, are within a specific date range relative to today's date, or that contain duplicate or unique values.

Top/Bottom Rules
> This lets you apply a format to the top or bottom 10 or 10% values or ones above or below the range's average value.

Data Bars
> This adds data bars proportional to each cell's value to help identify relatively large, small, or intermediate values.

Color Scales
> This adds shading to each cell that's proportional to each cell's value. Similarly to data bars, this option helps you identify relatively large, small, or intermediate values.

Icon Sets
> This adds an icon to each cell based on its value. As with data bars and color scales, this option helps you identify relatively large, small, or intermediate values.

New Rule
> This option lets you define other conditional formatting rules, such as when a formula evaluates to TRUE (see Recipes 2.2 and 7.5); you can also define new rules by choosing one of the previous options and selecting More Rules.

Data bars	Color scales	Icon sets	Top 10%	Bottom 10%	Duplicates
1	1 ◆	1	1	1	1
3	3 ▲	3	3	3	3
5	5 ●	5	5	5	5
5	5 ●	5	5	5	5

Figure 1-6. Examples of using conditional formatting

The Conditional Formatting menu also includes a Clear Rules option, which lets you remove conditional formatting from the selected cells, a table, a PivotTable, or the worksheet, and a Manage Rules option, which opens the Conditional Formatting Rules Manager. The Rules Manager provides an overview of the rules in particular worksheets, tables, or selected cells, and lets you create new conditional formatting rules; view, edit, delete, and duplicate any existing ones; and update which cells they apply to (see Figure 1-7).
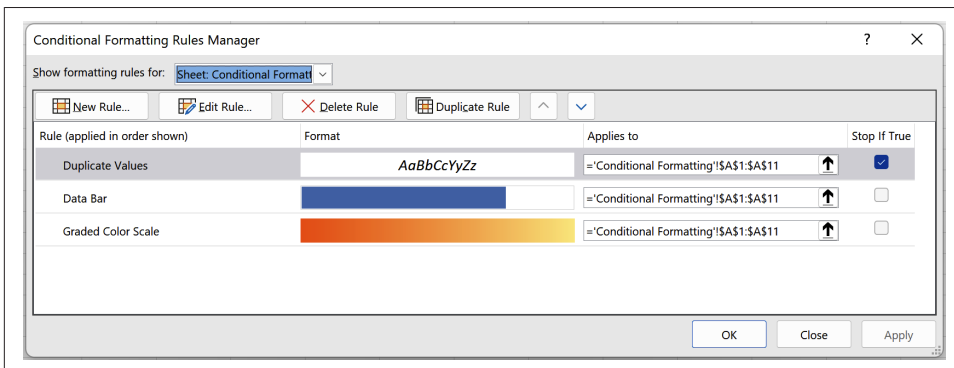
Figure 1-7. The Conditional Formatting Rules Manager

The Rules Manager lists rules in priority order, where the highest-priority rule is at the top and the lowest-priority rule is at the bottom. Excel applies rules in priority order, from the lowest rule upward, so if two rules conflict—for example, they apply different color scales to the same cells—Excel applies the rule with the highest priority that's highest in the list. You can change a rule's priority by selecting it and clicking the up or down arrows in the Rules Manager.

The Rules Manager also displays a Stop If True check box next to each rule. This check box controls whether lower-priority rules should run; once checked, it stops any lower-priority rules from being applied to cells where the checked rule is true.

Discussion

Applying conditional formatting offers a handy way of visualizing numeric data or highlighting cells whose values meet some condition. You can either use Excel's predefined rules or create your own.

See Recipe 2.2 for more details about using conditional formatting with formulas.

1.10 Using the Format Painter

Problem

You want to quickly apply one cell's format to another cell or range.

Solution

Excel's Format Painter lets you copy a cell's format and apply it to another cell or range. You use it as follows:

1. Select the cell whose format you want to copy.
2. Choose Home ⇒ Clipboard ⇒ Format Painter—the button with a paintbrush image.
3. Select the cells you want to apply the format to.
4. When you release the mouse button, Excel applies the original cell's format.

To apply the format to multiple locations, double-click the Format Painter button. The Format Painter remains active until you select the Format Painter again or press the Esc key.

Discussion

This recipe offers a quick, convenient way of copying a cell's format to one or more locations. Note that if you want to copy a cell's contents in addition to its format, Recipe 1.11 may be more efficient.

1.11 Using Paste Special

Problem

You want to copy and paste a cell's or range's contents and/or attributes or use it to perform arithmetic operations on another range.

Solution

Excel's Paste Special tool lets you copy a cell or range and select which contents or attributes—such as the values calculated by a formula or selected formatting—to paste elsewhere. You can also use this tool to transpose rows and columns, add the copied values to another set of values, or subtract, multiply, or divide by them.

To use Paste Special, follow these steps (see Figure 1-8):

1. Select the cells you want to copy.

2. Copy the cells by pressing Ctrl+C or choosing Home ⇒ Clipboard ⇒ Copy.

3. Select the first cell of the area you want to paste values to.

4. Select Home ⇒ Clipboard, open the Paste button's menu, and choose Paste Special to open the Paste Special dialog box.

5. Select the options you want to use and click OK.
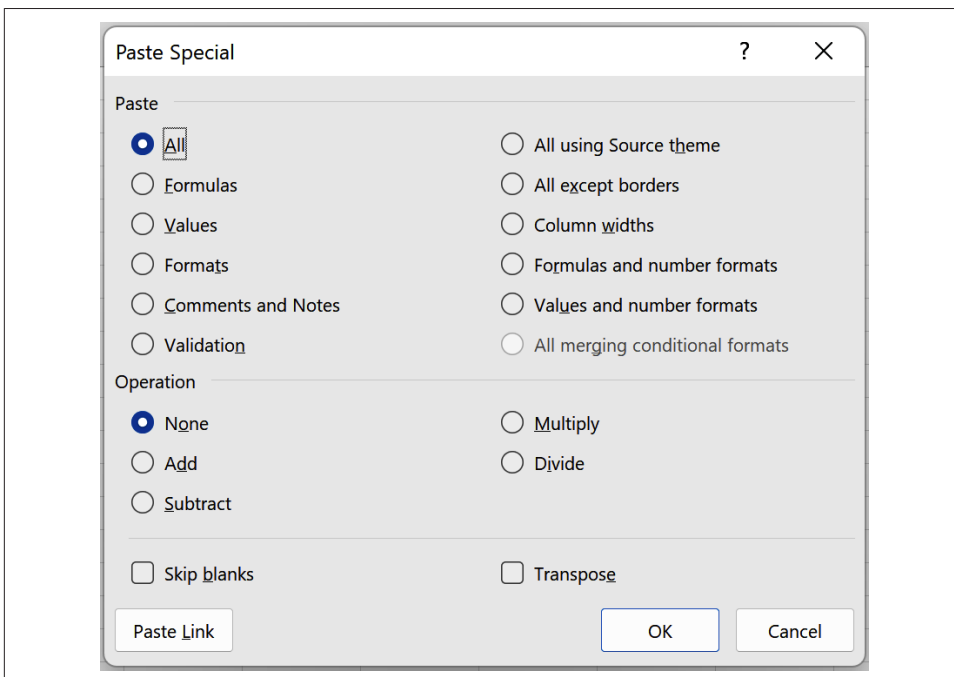
Figure 1-8. The Paste Special dialog box

Discussion

Paste Special offers a convenient way of copying one or more cells, pasting their contents and/or attributes to another location, and transposing cells. Note that if you want to only copy a cell's format, you may find Recipe 1.10 more efficient.

1.12 Using Auto Fill

Problem

You want to fill cells with data that follows a series or pattern based on the contents of other cells.

Solution

Suppose you want to fill a cell range with a sequence of values, such as the numbers 1 to 10, or the weekdays from April 8, 2024 to May 6, 2024. You can do so using Excel's Auto Fill tool.

Auto Fill helps you quickly fill cells based on the contents of other cells. You can use it, for example, to complete a series that increases in regular steps, copy a repeating pattern, complete a growth trend, or copy a formula to a range of cells. You generally use Auto Fill as follows:

1. Enter the first one or two values in the first cells of the range you want to fill. To fill the range A1:A10 with the numbers 1 to 10, for example, you'd type **1** in cell A1 and **2** in cell A2.

2. Select the cells you've just entered values into, click the selection's square fill handle in its bottom-right corner, and drag it across the cells you want to fill. When you release the mouse button, Excel fills the cells.

If Auto Fill doesn't work, you can enable it in Excel for Windows by choosing File ⇒ Options ⇒ Advanced and checking the Enable fill handle check box in the Editing options section. You can find this option in Excel for Mac by choosing Excel ⇒ Preferences ⇒ Edit.

You can adjust the Auto Fill tool's output by clicking the Auto Fill Options button Excel displays next to the filled cells (see Figure 1-9) and selecting one of the options from its menu. So instead of filling the cells with a sequence, you can copy the original cells to create a repeating pattern, copy only the cell formatting, complete the series without including any formatting, or use Flash Fill (see Recipe 1.14). If the cells contain dates, you can also fill the series using days, weekdays, months, or years (see Figure 1-9).
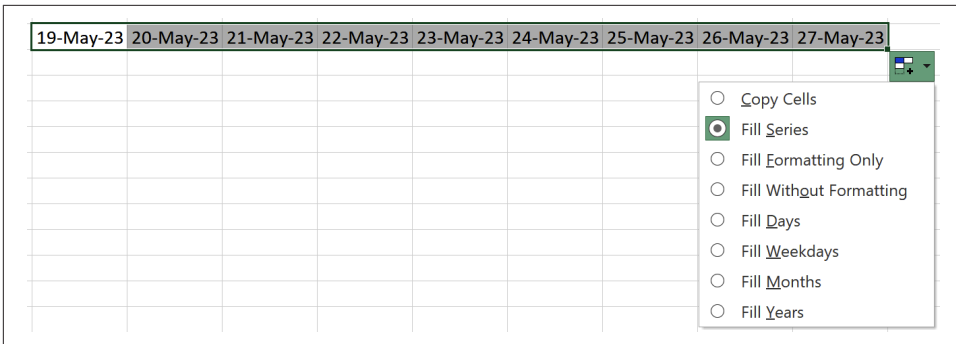
Figure 1-9. The Auto Fill Options menu

If the Auto Fill Options button doesn't appear, you can enable it in Excel for Windows by choosing File ⇒ Options ⇒ Advanced and checking the Show Paste Options check box in the "Cut, copy, and paste" section. You can find this option in Excel for Mac by choosing Excel ⇒ Preferences ⇒ Edit.

If there are values next to the column or row you need to fill, you can double-click the fill handle in step 2 instead of dragging it, and Excel will deduce how many cells it needs to fill. This facility is handy if, for example, you need to fill an extensive range with sequential values.

If you want to fill the cells with a linear or growth trend or make further adjustments to the fill series, you can do so by right-clicking and dragging the fill handle in step 2 instead of clicking it. Doing so opens a menu containing extra Linear, Growth, and Series options; selecting the Series option opens the Series dialog box shown in Figure 1-10, which you can use to fine-tune how you want to fill cells.
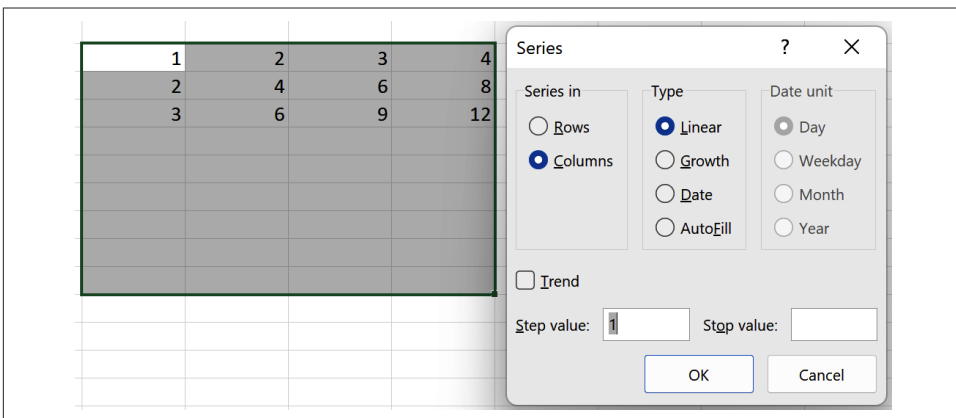
Figure 1-10. The Series dialog box

Discussion

Auto Fill is one of Excel's most flexible tools because you can use it to copy values, formulas, attributes, and continue series. You can even use it to populate cells from a custom list (see Recipe 1.13). This recipe gives an overview of its many features, including those less well-known.

1.13 Using Custom Lists

Problem

You want to create a custom list of text values to populate cells with Auto Fill or use the list to sort values.

Solution

Suppose you frequently need to populate a cell range with a list of text values—for example, the values North, South, East, and West—and you want to be able to do this using Auto Fill. You additionally want to be able to sort these values in a specific order.

You can achieve this by defining a *custom list*: an ordered list of text values stored on your computer. To define a custom list (see Figure 1-11):

1. If you're using Excel for Windows, choose File ⇒ Options ⇒ Advanced and click the Edit Custom Lists option in the General section to open the Custom Lists dialog box. If you're using Excel for Mac, choose Excel ⇒ Preferences ⇒ Custom Lists instead.

2. Select NEW LIST from the Custom Lists section, click in the List Entries box, and type the orders you want to appear in the list using the Enter/Return key to separate entries. To define a custom list containing the values North, South, East, and West, for example, you'd type **North** in the List Entries box, press Enter/Return, then type **South**, and so on.

3. When you've finished typing all the entries, click the Add button to add the list to your system. Then click the OK button to close the dialog box.

You can also define a custom list by importing values from a cell range. Instead of typing the entries in the List Entries box in step 2, add the cell range to the Import list from cells box in the Custom Lists dialog box and click the Import button.

Once you've created the custom list, you can use it to populate cells using Auto Fill. So if you've defined a custom list containing North, South, East, and West, for example, you can populate the range A1:A4 with these values by typing **North** in A1, **South** in A2, and then using Auto Fill to populate A3 and A4 with the values East and West (see Recipe 1.12).
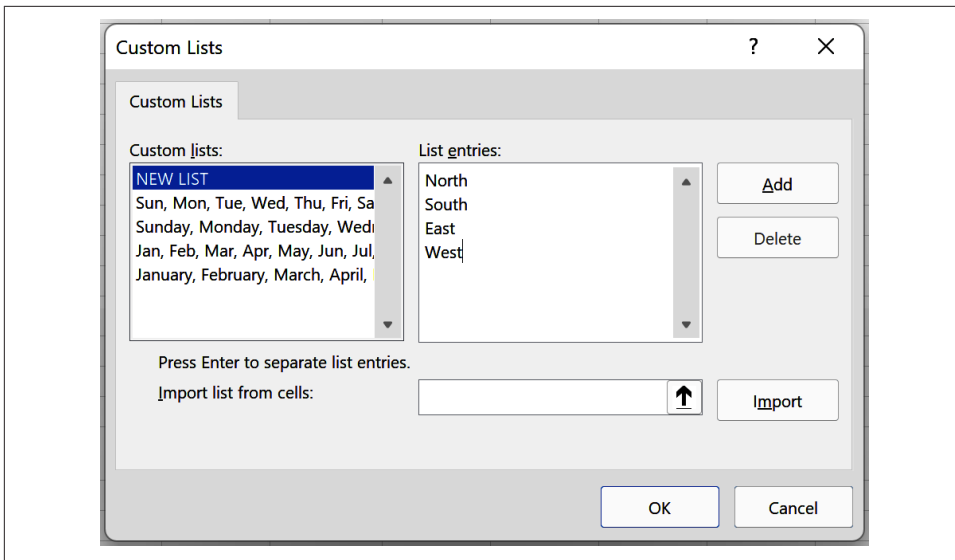
Figure 1-11. The Custom Lists dialog box

If you want to edit or delete a custom list, select it in the Custom Lists dialog box, edit its entries, or click the Delete button. Note that you can't edit or delete any of Excel's built-in lists, such as days of the week and months of the year.

Discussion

Custom lists can be helpful if you regularly want to Auto Fill cell ranges with the same list of text values. You can also use them to sort data in a specific order.

1.14 Using Flash Fill

Problem

You need to extract or concatenate data and want Excel to fill the remaining cells when it spots a pattern.

Solution

Suppose cells A2:A10 list customer first names and last names, and you want to extract the customer first names to cells B2:B10 (see Figure 1-12). You can do so using Flash Fill: a tool that analyzes the date you're entering and uses pattern recognition to fill the remaining cells.

To use Flash Fill, you first enter a few examples of the results you want in a column next to the data. If you're using Excel for Windows, it automatically launches Flash Fill when it spots a pattern and shows you a preview of the data it will fill, which you can accept by pressing Enter/Return. If you start typing the customer first names in cells B2 and B3, for example, Excel quickly recognizes what you're doing and starts Flash Fill (see Figure 1-12).

	A	B	C	D	E	F
1	Name	First name				
2	Amy Pond	Amy				
3	Clara Oswald	Clara				
4	Rose Tyler	Rose				
5	Jackie Tyler	Jackie				

Figure 1-12. Customer names and the Flash Fill preview

> If you're using Excel for Windows and Flash Fill doesn't start automatically, you may need to enable this facility by choosing File ⇒ Options ⇒ Advanced and checking the Automatically Flash Fill check box in the Editing options section.

If you're using Excel for Mac or you want to start Flash Fill manually, enter a few examples of the results you want, and then choose Data ⇒ Data Tools ⇒ Flash Fill or select Flash Fill from the Auto Fill Options menu (see Recipe 1.12).

Once you've accepted a Flash Fill preview, Excel displays a Flash Fill Options button next to the filled cells. When you click this button, Excel displays a menu that lets you undo the Flash Fill, accept the suggestions, select any blank cells, or select the cells that the Flash Fill completed (see Figure 1-13).

> If you want to extract numeric codes—for example, telephone numbers or zip codes—you can get Flash Fill to include any leading zeros and numeric symbols by prefixing the text you type with an apostrophe. The apostrophe tells Excel to treat the codes as text, so typing **'+44** is interpreted as the text +44, while typing **+44** is interpreted as the positive number 44 without a leading plus sign.
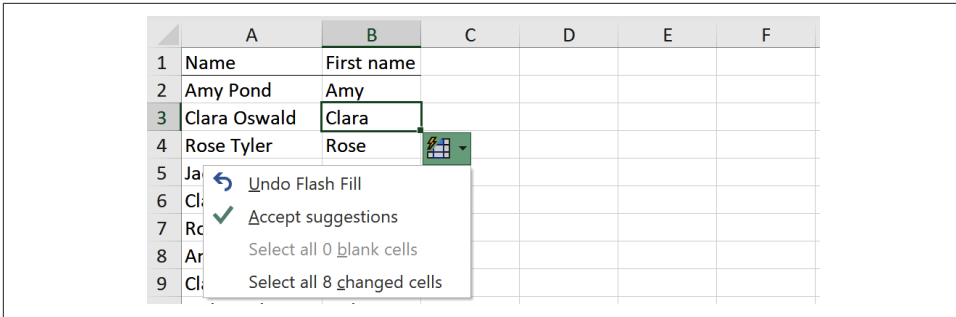
Figure 1-13. The Flash Fill Options menu

Discussion

Flash Fill offers a fast and simple solution when you want to extract or concatenate data. Note that data populated using Flash Fill isn't dynamically updated, so if the underlying data changes, you'll need to update the filled text or use Flash Fill again. Alternatively, use the recipes in Chapter 5 to create a more dynamic, formula-based solution.

> If you have one or more cells containing text separated by a delimiter—for example, a comma—you can split it into columns using the Text to Columns Wizard. To access this tool, select the cells you want to split and choose Data ⇒ Data Tools ⇒ Text to Columns.

1.15 Customizing AutoCorrect

Problem

You want Excel to change abbreviations to full text, longer phrases, or symbols automatically.

Solution

Suppose that each time you type a short code in a cell, you want Excel to replace it with longer text. You can do so by adding the code and text to Excel's AutoCorrect options (see Figure 1-14):

1. If you're using Excel for Windows, choose File ⇒ Options ⇒ Proofing and click AutoCorrect Options to open the AutoCorrect dialog box. If you're using Excel for Mac, choose Excel ⇒ Preferences ⇒ AutoCorrect.

2. Enter the code and the text you want to replace it with in the Replace and With boxes; then click Add.

3. Update any other options you want; then click OK to close the dialog box.
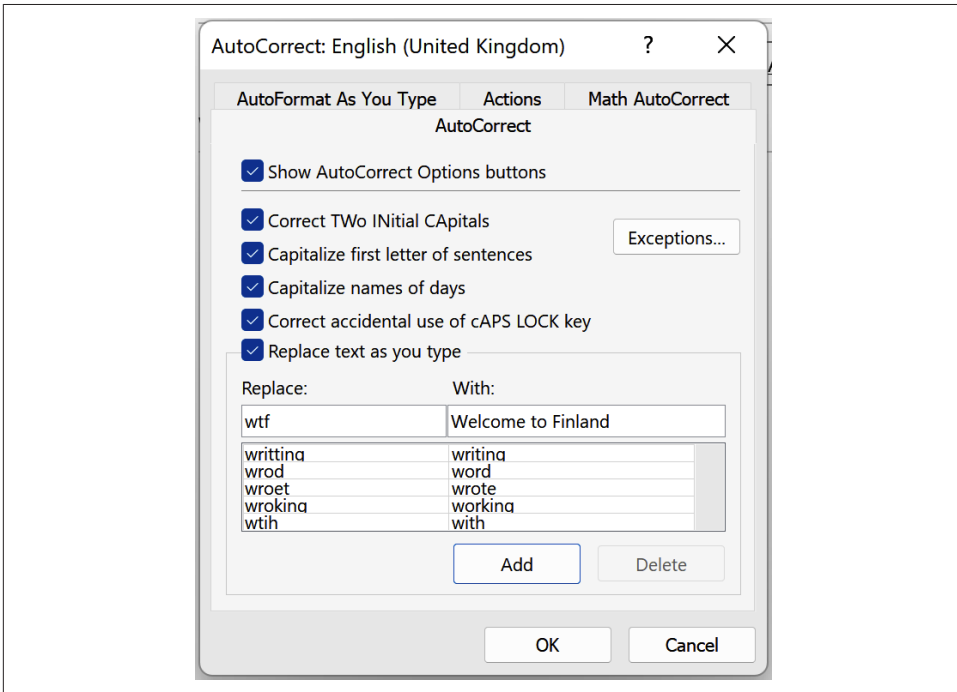
Figure 1-14. The AutoCorrect dialog box

If you want to replace a code with a symbol, first choose Insert ⇒ Symbols ⇒ Symbol, search for the symbol you wish to use, then copy it by pressing Ctrl+C. You can then paste the symbol in the AutoCorrect dialog box's With box by pressing Ctrl+V.

Discussion

This recipe is a handy, time-saving technique for entering the same phrase or symbol many times.

1.16 Using Notes and Comments

Problem

You want to annotate a cell with a note or add a comment others can reply to.

Solution

If you're using Excel 365, you can annotate cells using comments and notes. A *comment* is a threaded annotation that includes a Reply box, while a *note* is an unthreaded annotation without a Reply box (see Figure 1-15).
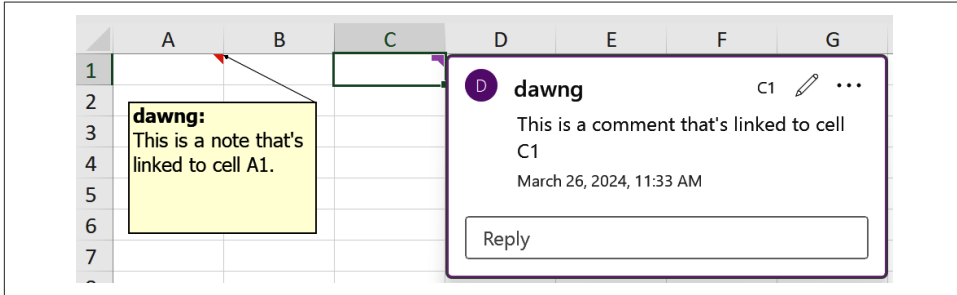
Figure 1-15. Notes versus comments

> Before Excel 365, Excel included only comments, not notes. The old version of comments worked similarly to notes in Excel 365.

To add a comment to a cell, select the cell and choose Review ⇒ Comments ⇒ New Comment, or right-click the cell and choose New Comment. Then type a comment in the Reply or Start a Conversation box and post it. Excel displays the comment when you hover your mouse cursor over the cell, and you can view and manage all the comments in a workbook by choosing Review ⇒ Comments ⇒ Show Comments, which opens Excel's Comments pane.

To add a note, select the cell and choose Review ⇒ Notes ⇒ New Note, or right-click the cell and choose New Note. You then type the text you want to appear in the note. Excel displays the note when you hover your mouse cursor over the cell, select the cell and choose Review ⇒ Notes ⇒ Show/Hide Note, or right-click the cell and select Show/Hide Note. You can see all notes by choosing Review ⇒ Notes ⇒ Show All Notes, and you can convert notes to comments by choosing Review ⇒ Notes ⇒ Convert to Comments.

> If you're using Excel for Mac, you can stop Excel from displaying notes and comments when you hover your mouse cursor over a cell by choosing File ⇒ Options ⇒ Advanced and updating the "For cells with comments" option in the Display section. If you're using Excel for Mac, you can find this option by choosing Excel ⇒ Preferences ⇒ View.

Discussion

Notes and comments are convenient for annotating cells with reminders or enabling discussions. In older versions of Excel, comments are unthreaded and work like notes.

See Also

You can also use data validation to add an input message to cells, see Recipe 2.8.

1.17 Finding and Selecting Cells and Navigation

Problem

You want to quickly find and select cells with specific contents or attributes, such as formulas, notes, or conditional formatting.

Solution

Suppose you have a worksheet and want to find and replace a value, go to a specific cell or range, or find and select cells with conditional formatting. You can do so using the Find & Select menu.

To open the Find & Select menu, choose Home ⇒ Editing ⇒ Find & Select. The menu includes the following options:

Find and Replace
These options let you search for a value and optionally replace it. You can also perform wildcard searches using the ? and * characters as wildcards, where ? is a substitute for a single character and * for any number of characters. Typing **e?t**, for example, finds the text *eat* and *ent*, while typing **e*t** also includes the text *east*. If you want to find text that includes a question mark or asterisk, you can do so by prefixing it with a ~ character.

Go To
This lets you navigate to a particular cell or range reference (for example, Z1000). You can also do this by typing the reference into Excel's Name box, which appears underneath the ribbon to the left of the formula bar.

Go To Special
This opens the Go To Special dialog box, which lets you quickly find and select cells with specific contents or attributes—for example, blank cells or formulas with errors (see Figure 1-16).
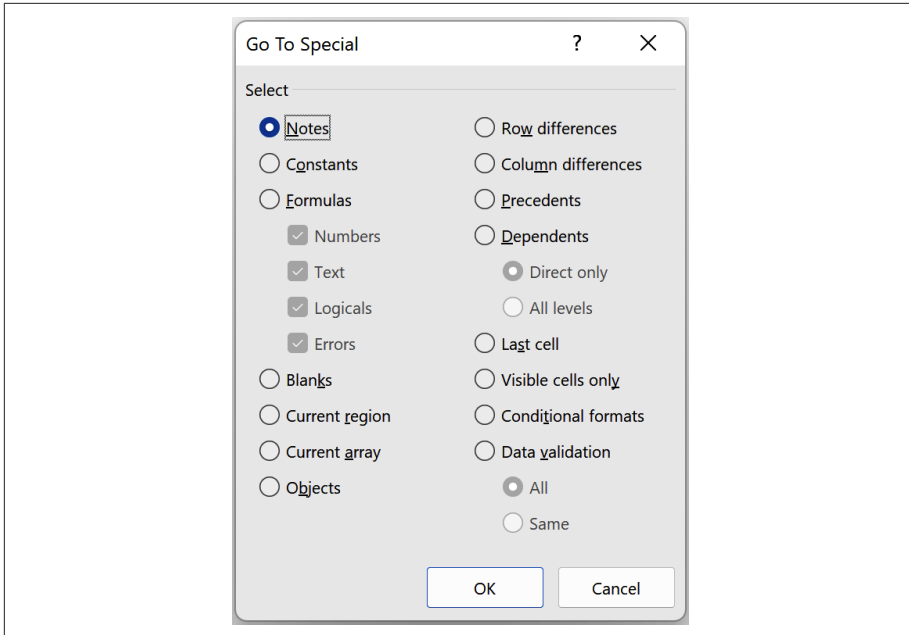
Figure 1-16. The Go To Special dialog box

Formulas, Notes, Conditional Formatting, Constants, and Data Validation

Use these options to find and select cells with the specified contents and attributes.

Select Objects and Selection Pane

These options let you more easily select and manage any objects in your worksheet, such as charts, ink, and shapes.

You can also navigate and select cells using keyboard shortcuts. Here are some of the most useful ones:

Ctrl+Page Up, Ctrl+Page Down

This shortcut navigates to the previous or next worksheet.

Shift+click

This lets you select an adjacent range of cells. Click the first cell, hold down the Shift key, then click the last.

Shift+Arrow keys

This lets you select adjacent rows or columns. Select the first cell, hold down the Shift key, and use the arrow keys to select the adjacent cells.

Ctrl+click or Cmd+click

This lets you select cells that aren't adjacent. Click the first cell, hold down the Ctrl key if you're using Excel for Windows or the Cmd key if you're using Excel for Mac, and then click each cell you want to add to the selection.

Ctrl+A

This selects all the cells in a range, table, PivotTable, or worksheet.

Ctrl+Shift+Arrow keys

This lets you select all the cells in a row or column in a range, table, PivotTable, or worksheet, so Ctrl+Shift+Down, for example, selects the cells in a column.

Discussion

Navigating through spreadsheets and finding cells with specific characteristics can be time-consuming. This recipe helps you work more efficiently using the Find & Select menu options and keyboard shortcuts.

> You can also use hyperlinks to help you navigate to specific locations in your workbook or elsewhere. To insert a link, right-click the cell you want to add it to and choose Link, or select the cell and choose Insert ⇒ Links ⇒ Link.

1.18 Creating a Custom View

Problem

You want to save specific display or print settings—such as hidden rows and columns, filters, or margin widths—so that you can quickly apply them to a worksheet when needed.

Solution

If your workbook doesn't contain a table, you can save settings in a *custom view*: a set of settings that applies to a worksheet.

To create a custom view (see Figure 1-17):

1. Go to the worksheet you want to save a custom view for and change the display and print settings you want to save in the Page Layout menu. To open a dialog box showing the complete set of options, click the launcher icon ↘ for one of the groups in the ribbon.

2. Choose View ⇒ Workbook Views ⇒ Custom Views to open the Custom Views dialog box.

3. Click Add to open the Add View dialog box, type a name for the custom view, select which options you want to include, and click OK to save the view.
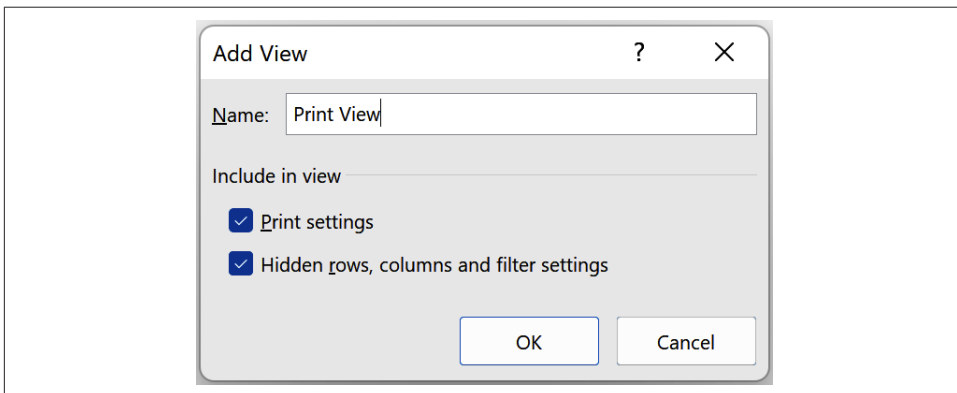
Figure 1-17. The Add View dialog box

To switch to a custom view, go to the worksheet you created the view for, choose View ⇒ Workbook Views ⇒ Custom Views, select the view you wish to display, and click Show.

> Excel disables the Custom Views option if your workbook contains tables, so if you add a table to a workbook containing custom views, you'll no longer be able to show them.

Discussion

Custom views are a handy way of saving specific display or print settings so you can quickly apply them when needed. You can save views with specific filters, for example, or with settings that you want to apply before printing a worksheet.

1.19 Customizing the Ribbon and Ribbon Tabs

Problem

You want to customize the tabs and commands shown in Excel's ribbon.

Solution

You can customize Excel's ribbon using the Customize the Ribbon dialog box. If you're using Excel for Windows, open the dialog box by choosing File ⇒ Options ⇒ Customize Ribbon or right-clicking the ribbon and selecting Customize the Ribbon.

If you're using Excel for Mac, open it by choosing Excel ⇒ Preferences ⇒ Ribbon & Toolbar.

Generally, you use the Customize the Ribbon dialog box to choose which commands to add to which tabs and which tabs to include. By default, Excel shows you a list of the most popular commands and the ribbon's main tabs. However, you can change these options to show commands not in the ribbon and all the tabs (see Figure 1-18).
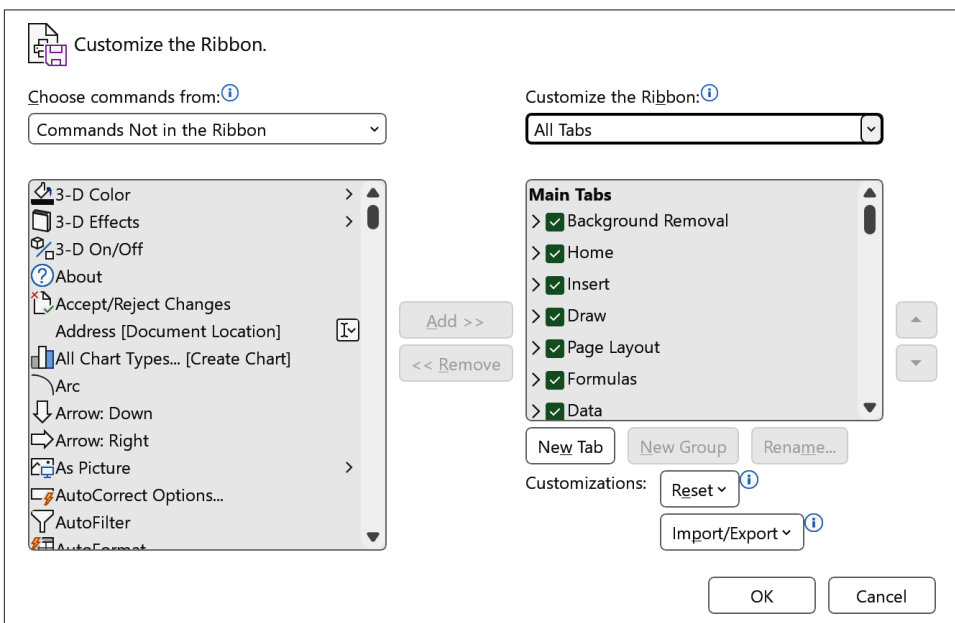
Figure 1-18. The Customize the Ribbon dialog box

Discussion

Customizing Excel's ribbon is vital when you want to use extra tabs, such as the Developer tab, that aren't enabled by default. You can also customize the commands shown or reset the ribbon to delete any customizations you no longer need.

> Many recipes in this book refer to ribbon commands and their default tabs, so over-customizing the ribbon may make it harder to follow these recipes.

See Also

You can also add commands to the Quick Access Toolbar; see Recipe 1.20.

1.20 Using the Quick Access Toolbar

Problem

You want to add frequently used commands to an easily accessed toolbar.

Solution

Use the Quick Access Toolbar: a toolbar that gives you quick access to commands.

Depending on your version of Excel, the Quick Access Toolbar appears above the formula bar or in Excel's title bar. Figure 1-19 shows the Windows version of the toolbar containing AutoSave, Save, Undo, and Redo commands.
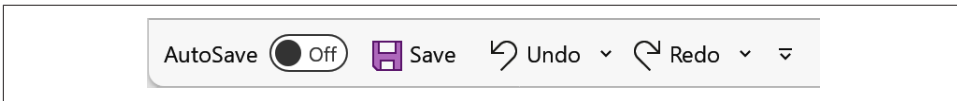
Figure 1-19. The Quick Access Toolbar may appear above the formula bar or in the Excel title bar

> The Quick Access Toolbar may not be displayed by default if you're using Excel for Windows. However, you can show it by right-clicking the ribbon and selecting the Show Quick Access Toolbar.

You can customize the commands shown on the Quick Access Toolbar by clicking the toolbar's Customize Quick Access Toolbar icon ⩔ and selecting the commands you want to display in the menu. To see an exhaustive list of commands, select More Commands from the menu, which opens the Customize the Quick Access Toolbar dialog box; this offers the same list of commands as the Customize the Ribbon dialog box (see Recipe 1.19).

If you're using Excel for Windows, you can also add commands directly from the ribbon by right-clicking the command you want to add and selecting Add to Quick Access Toolbar.

Discussion

The Quick Access Toolbar offers a convenient alternative to Recipe 1.19 since it lets you put all your most commonly used commands in an easily accessible location. You can also use it, for example, to run any macros you've created.

1.21 Using the Accessibility Checker

Problem

You have a workbook and want to check for and correct any accessibility issues.

Solution

Excel includes an Accessibility Checker, which you can use to check a workbook for accessibility issues and correct any the tool finds.

To use the Accessibility Checker, choose Review ⇒ Accessibility ⇒ Check Accessibility ⇒ Check Accessibility to open an Accessibility pane. The pane contains a list of inspection results, such as missing object descriptions; click each one to view a list of recommended actions and decide which to apply.

You can also choose Review ⇒ Accessibility ⇒ Check Accessibility ⇒ Options: Accessibility to see Excel's accessibility options, and Review ⇒ Accessibility ⇒ Check Accessibility ⇒ Accessibility Help to read guidance on making Excel documents more accessible.

Discussion

This recipe introduces you to Excel's Accessibility Checker, which you can use to check for accessibility issues. It also gives a convenient way of addressing each issue it finds, such as adding alternative text to floating or embedded objects, including charts, shapes, and pictures (see Chapters 12 and 13).

References and Structured Data

References are intrinsic to nearly all spreadsheets since most Excel features depend on them. For example, formulas rely on cell references and ranges to perform calculations, tables use structured references that automatically expand and contract as rows are added and deleted, and using absolute, relative, or mixed references can affect how features such as Auto Fill, conditional formatting, and data validation behave.

This chapter covers when and how to use these different types of references and techniques for working with structured data. Recipes include formula-based conditional formatting, custom data validation, using static and dynamic named ranges to create cascading drop-down lists, adding groups and subtotals, and tables.

2.1 Using Relative and Absolute References

Problem

You want to control whether a cell reference in a formula can change when you copy the formula to another cell.

Solution

Relative, absolute, and mixed cell references control how a formula changes when it's copied.

A *relative* cell reference—for example, A1—can change when you copy it to another cell, and it's useful, for example, if you want to perform a calculation for each row or column in a range. So if cell C1 contains the formula =A1*B1 and you use Auto Fill (see Recipe 1.12) to copy it to the range C2:C3, this puts the formula =A2*B2 in C2 and =A3*B3 in C3. Similarly, if cell D1 contains the formula =SUM(A1:B1), copying it to D2:D3 puts =SUM(A2:B2) in D2 and =SUM(A3:B3) in D3.

An *absolute* cell reference doesn't change when you copy the formula, so it's useful, for example, if you want to multiply the sum of each row by a fixed value. An absolute reference has a dollar sign before its column letter and row number—for example, $A$1—so if cell E1 contains the formula =SUM(A1:B1)*$G$1, copying it to E2:E3 puts =SUM(A2:B2)*$G$1 in E2 and =SUM(A3:B3)*$G$1 in E3; the $G$1 reference remains unchanged.

A *mixed* cell reference is one where either the column or row is absolute and has a dollar sign before its column letter or row number, and the other is relative—for example, $A1 or A$1. It's helpful if you want to copy a formula to other cells while fixing any references to a specific row or column. So if you want to calculate the area for various widths and lengths, you can enter widths and lengths in B7:B9 and C6:E6, type the formula **=$B7*C$6** in C7, then copy it down and across to fill C7:E9. Since the formula includes absolute references to the B column and row 6, these don't get updated when you copy the formula (see Figure 2-1).

	A	B	C	D	E	F
5				Length		
6			1	2	3	
7		1	=$B7*C$6	=$B7*D$6	=$B7*E$6	
8	Width	2	=$B8*C$6	=$B8*D$6	=$B8*E$6	
9		3	=$B9*C$6	=$B9*D$6	=$B9*E$6	

Figure 2-1. Using mixed references to calculate area

> In some versions of Excel, you can use the F4 key (or Fn+F4 in Excel for Mac) to toggle between relative, absolute, and mixed references.

Discussion

This recipe gives an overview of using relative, absolute, and mixed cell references and when using each type is appropriate. When deciding which type to use, generally consider whether you want each cell reference's row and/or column to update if you copy the formula.

2.2 Using Relative and Absolute References in Conditional Formatting

Problem

You want to know how to use relative and absolute cell references in a conditional formatting rule that uses a formula.

Solution

Suppose you want to apply conditional formatting (see Recipe 1.9) based on a formula to cells A1:F3. You can do so by selecting the cells, choosing Home ⇒ Styles ⇒ Conditional Formatting ⇒ New Rule, selecting the "Use a formula" option, typing a formula, and specifying a format; Excel applies this format to cells where the formula evaluates to TRUE.

You write the formula relative to the top-left cell in the range—in this example, A1—and, behind the scenes, Excel "copies" it to the other cells in the range. When Excel does so, it keeps any absolute cell references intact but adjusts any relative ones (see Recipe 2.1) so the type of cell reference you use can affect which cells are formatted.

You generally use mixed cell references to compare values in the same row. To format each row where the value in the A column is greater than the value in the B column, for example, you type the formula **=$A1>$B1**; this formats row 1 when A1 is greater than B1, row 2 when A2 is greater than B2, and row 3 when A3 is greater than B3 (see Figure 2-2).

	A	B	C	D	E	F
1	3	3	2	2	1	1
2	2	1	3	1	2	3
3	1	2	1	3	3	2

Figure 2-2. Using the formula =$A1>$B1 with conditional formatting, which compares the values in the A and B columns

To refer to a value in a specific cell, you need to use that cell's absolute cell reference. To format each row where the value in the A column is greater than the value in B3, for example, you type the formula **=$A1>$B$3**; since $B$3 is an absolute reference, it's not adjusted when it's copied to other cells, so Excel formats row 1 when A1 is greater than B3, row 2 when A2 is greater than B3, and row 3 when A3 is greater than B3 (see Figure 2-3).

	A	B	C	D	E	F
1	3	3	2	2	1	1
2	2	1	3	1	2	3
3	1	2	1	3	3	2

Figure 2-3. Using the formula =$A1>$B$3 with conditional formatting, which compares each value in the A column with B3

To apply a conditional format to individual cells, you use relative references. To format each cell whose value is greater than in B2, for example, you type the value **=A1>$B$2**; since A1 is a relative reference, it's adjusted when it's copied to other cells, so Excel applies a format to each cell greater than B2 (see Figure 2-4).

Figure 2-4. Using the formula =A1>$B$2 with conditional formatting, which compares each value with B2

Discussion

This recipe expands on Recipe 1.9 and shows you how to use different types of cell references with conditional formatting. Choosing the correct type of cell reference is crucial since it affects which cells are formatted.

2.3 Using R1C1-Style Cell References

Problem

You want to know how to use R1C1-style cell references and when this can be useful.

Solution

By default, Excel labels columns with letters and rows with numbers, and you use these to refer to cells and ranges, so cell B2, for example, refers to the cell in column B, row 2. This way of referring to cells is called the *A1 reference style*, and it's the one most Excel users are familiar with.

A less common way of referring to cells is using the R1C1 reference style, which uses numbers for both rows and columns. For example, the cell in the second row and the second column has the cell reference R2C2 (see Figure 2-5). This way of referring to cells can be helpful if, for example, you need to perform calculations using column numbers.
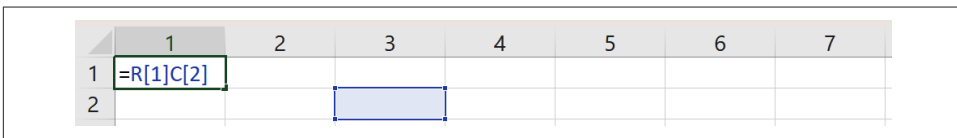
Figure 2-5. The R1C1 reference style uses column numbers instead of letters

You can choose to use either the A1 or R1C1 reference style. First, choose File ⇒ Options ⇒ Formulas if you're using Excel for Windows or Excel ⇒ Preferences ⇒ Calculation if you're using Excel for Mac; then check the R1C1 reference style check box to specify the R1C1 reference style, or leave it unchecked to use A1. When you switch to a different reference style, Excel changes the column labels to numbers or letters depending on the reference style (see Figure 2-5) and converts the workbook's formulas.

Similarly to the A1 reference style, the R1C1 style lets you use absolute, relative, and mixed references.

An R1C1 absolute reference simply specifies the row and column number, so R2C3, for example, is an absolute cell reference that refers to the cell in row two column three; it's equivalent to using $C$2 in the A1 reference style.

An R1C1 relative reference specifies cells using offsets. If you want to refer to cell R2C3 from cell R1C1, for example, you type **=R[1]C[2]** in cell R1C1; this specifies the cell that's one row down and two columns to the right of cell R1C1. Similarly, you specify cell R1C1 from cell R2C2 by typing **=R[-1]C[-1]** since cell R1C1 is one row above and one column to the left of cell R2C2. You can also use =R[1]C to specify the cell one row down in the current column and =RC[1] to specify the cell in the current row one column to the right. See Figure 2-6.

	1	2	3	4	5	6	7
1	=R[1]C[2]						
2							

Figure 2-6. The formula =R[1]C[2] in cell R1C1 refers to cell R2C3

Most recipes in this book assume you're using the default A1 reference style, so using the R1C1 reference style will make it harder to follow the examples.

Discussion

This recipe gives an overview of using the R1C1 reference style. While using the default A1 reference style is preferable in most situations, the R1C1 style can be helpful if, for example, you need to perform calculations using column numbers.

See Also

Relative references in the R1C1 reference style work similarly to using the OFFSET function in the A1 reference style; see Recipe 7.14.

2.4 Referencing Another Worksheet or Workbook

Problem

You want to refer to a cell or range in another worksheet or workbook.

Solution

When you add formulas to cells, you sometimes need to refer to the contents of a cell in another worksheet or workbook.

To refer to a cell or range in another worksheet in the same workbook, you generally use the syntax *worksheet_name*`!cell_or_range`. To refer to cell A1 in a worksheet named Rates, for example, type **=Rates!A1**. If the worksheet name contains characters that aren't letters—such as spaces or hyphens—you must also put the worksheet name in single quotes. To refer to the range A2:B5 in a worksheet named Company Data, you'd use `'Company Data'!A2:B5`.

If you want to refer to a cell or range in another workbook, your syntax depends on whether the workbook is open or closed.

If the workbook is open, use the syntax `[`*workbook_name*`]`*worksheet_name*`!cell_ or_range`. To refer to cell A1 in the Summary worksheet of the open *Starbuzz.xslx* file, use `=[Starbuzz.xslx]Summary!A1`. If the workbook and/or worksheet name contains characters that aren't letters, you must put both in single quotes. To refer to cell A1 in the Company Data worksheet in the open *Starbuzz Accounts.xslx* file, you'd use `'[Starbuzz Accounts.xslx]Company Data'!A1`.

To refer to a cell or range in a workbook that's closed, you also must specify the file path—for example, `='C:\Users\dawng\Desktop\[Starbuzz.xlsx]Summary'!$A$1`. Excel automatically tracks when a linked Excel file is open or closed. If you have a reference to an open workbook that you then close, Excel automatically updates the reference to include the file's path; it removes the path when you reopen the workbook.

> Instead of typing complex cell references, you can get Excel to add them. First, go to the cell where you want to add the reference, start typing the formula, then select the cell or range you want to refer to; when you press Enter/Return, Excel adds the cell reference to the formula. Alternatively, select the cell or range you want to refer to, press Ctrl+C or choose Home ⇒ Clipboard ⇒ Copy to copy it, then paste it in the formula by selecting Home ⇒ Clipboard ⇒ Paste ⇒ Paste Link.

Discussion

This recipe shows you how to refer to cells and ranges in other worksheets or workbooks. Note that it's generally easier for Excel to create the links automatically since it ensures that you use the correct syntax.

2.5 Using 3-D References

Problem

You want to create a formula using the same cell or range in multiple adjacent worksheets.

Solution

Suppose you want to sum the value of cell B1 in three adjacent worksheets: HR, Marketing, and Sales. You can do so using a 3-D reference, which takes the form `first_sheet:last_sheet!cell_or_range`, where `first_sheet` is the name of the first worksheet in the range and `last_sheet` is the name of the last. To get a reference to cell B1 in the worksheets from HR to Sales, you'd use `HR:Sales!B1`, and to sum these values, you'd type **`=SUM(HR:Sales!B1)`**.

> Excel can automatically add 3-D references for you. Go to the cell you want to add the reference to, start typing the formula, and click the first worksheet tab you want to include in the reference. Then hold down the Shift key, click the final worksheet tab you want to include and select the cell or range. When you press Enter/Return, Excel adds the 3-D reference to the formula.

Discussion

3-D references are a handy way of performing calculations with the same cell or range across adjacent worksheets. Not all functions support them, but if you're using Excel 365, you can use `VSTACK` and `HSTACK` (see Recipe 7.4) to form a dynamic array from their values, which you can then use with more functions.

2.6 Naming Cells, Ranges, Constants, and Formulas

Problem

You want to simplify formulas by using names instead of cell or range references, constants, or formulas.

Solution

Naming cells, ranges, constants, and formulas can help make the formulas you add to cells easier to understand since you can use human-friendly names instead of references and values. So instead of using =SUM(Sales!B1:B10)*Rates!B1 to multiply a sum of values by a rate, for example, you can use =SUM(SalesTotals)*Rate.

The quickest way of defining a name for a cell or range is to use the Name box. Select the cell or range you want to name, type its name in the Name Box under the ribbon to the left of the formula bar, and then press Enter/Return to save the name (see Figure 2-7).
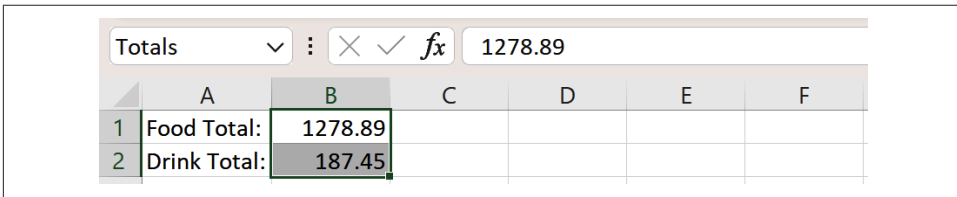
Figure 2-7. Naming a range using the Name box

You can use the Create from Selection option if you have tabular data and want to define a name for each column and/or row. First, select the data you want to define names for, including the row or column labels you want to use for the names; then choose Formulas ⇒ Defined Names ⇒ Create from Selection to open the Create names from dialog box. Next, select one or more options to specify which labels you want to use for the names—top row, left column, bottom row, and right column—then click OK to create the names (see Figure 2-8).
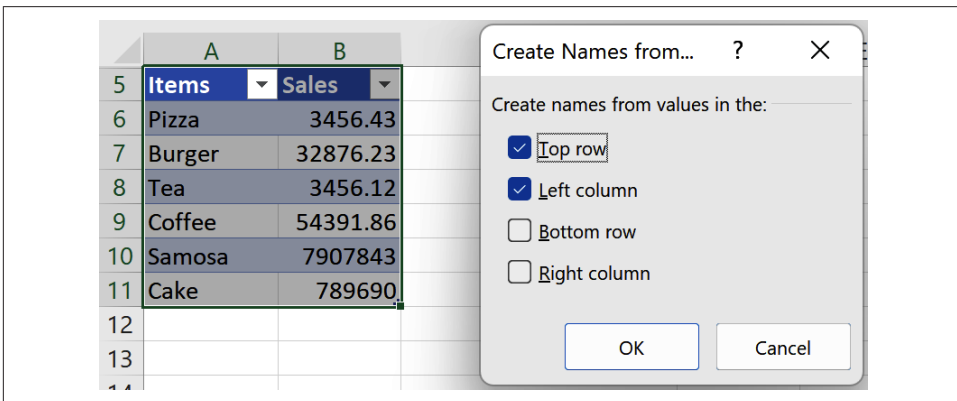
Figure 2-8. Naming rows and columns using the Create from Selection option

When you define a named range based on a column in a table, the range automatically expands to include any new rows you add.

The most flexible way to define names is to use the Define Name option, which lets you define names for cells, ranges, constants, and formulas. Choose Formulas ⇒ Defined Names ⇒ Define Name ⇒ Define Name to open the New Name dialog box, type a name in the Name box, and then use the Refers to box to specify the name's cell, range, constant, or formula. When you click OK, Excel saves the name (see Figure 2-9).
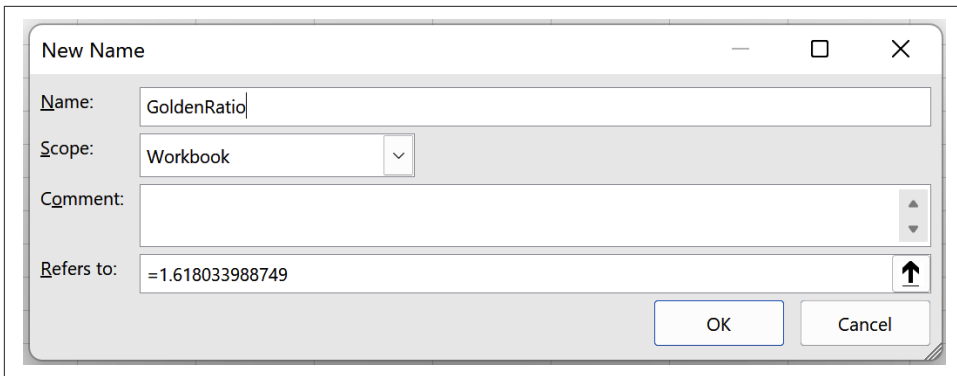
Figure 2-9. Naming a constant using the Define Name option

When you assign a name using the Define Name option, you can also choose a *scope*: the level at which Excel recognizes the name. By default, the scope is Workbook, so you can use the name throughout the workbook; you can, however, change the scope to a specific worksheet. Any names you define using the Name box or Create from Selection have Workbook scope. Note that each name you define must be unique within its scope.

To manage any names you've defined, use Excel's Name Manager, which you open by choosing Formulas ⇒ Defined Names ⇒ Name Manager. The Name Manager lets you edit or delete any existing names or define new ones by opening the new Name dialog box.

You use any names you've defined similarly to using cell references. For names where the scope is the current workbook or worksheet, you use the name in place of any cell or range references—for example, =SUM(B1:B10)*Rate. If the scope is a different worksheet in the same workbook, use the syntax worksheet!name—for example, Sales!Total. You can also refer to a name from a different workbook; if the name's

scope is Workbook, you use *workbook*`!`*name*, and if it's limited to a worksheet, you use
`[`*workbook_name*`]`*worksheet*`!`*name*.

> When you add a defined name, Excel doesn't update any of your
> existing formulas to use it. You can, however, get Excel to replace
> any references by choosing Formulas ⇒ Defined Names ⇒ Define
> Name ⇒ Apply Names.

Discussion

Defining names for cell or range references, constants, and formulas is one of Excel's
most useful features in workbooks that contain many formulas. Defined names make
formulas easier to read, maintain, and reuse, and they're also easier to use than cell or
range references because when you start typing the defined name, Excel shows a list
of matching ones. Using named ranges can also simplify navigation since you can
select the name from the Name box's drop-down list.

See Also

You can also use formulas to define dynamic named ranges; see Recipe 2.7.

If you're using Excel 365, you can use the Name Manager to define a custom function;
see Recipe 17.4.

2.7 Creating Dynamic Named Ranges

Problem

You want to create a named range that automatically expands and contracts when you
add and delete rows.

Solution

Suppose you want to define a named range that includes all the cells in a column that
contain values and accommodates any new rows you add. In this situation, you can
either convert the data to a column in a table and use this for the named range (see
Recipe 2.6) or define a dynamic named range based on a formula.

To create a dynamic named range that uses a formula, you generally choose Formulas
⇒ Defined Names ⇒ Define Name ⇒ Define Name to open the New Name dialog
box, type a name in the Name box, then specify the formula in the Refers to box. Your
formula depends on the data you want the dynamic named range to return.

To return all the cells in a column range that contain values (assuming there are no blank rows), you generally use the formula =*first_cell*:INDEX(*column_range,* COUNTA(*column_range*)), which uses the COUNTA function to count the number of cells in *column_range* containing values, which the INDEX function uses to determine the last cell in the range (see Recipe 7.11). In this formula, *first_cell* is the first cell you want to include in the range, and *column_range* specifies the column containing the values. To define a named range that includes all the values in column A from cell A2, you'd use the formula =$A$2:INDEX($A:$A, COUNTA($A:$A)); see Figure 2-10.
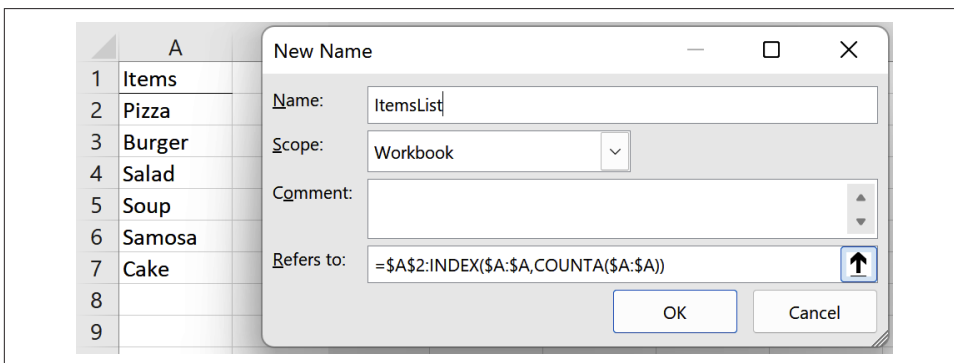
Figure 2-10. Using the INDEX function to create a dynamic named range

If you want the dynamic named range to return the last *n* rows in a column range that contain values (assuming there are no blank rows), use the formula =OFFSET(*first_cell,* COUNTA(*column_range*)-n, 0, n, 1), which uses the OFFSET function (see Recipe 7.14) to determine how many rows to offset *first_cell* by. To return the last three rows in column A from cell A1, you'd use the formula =OFFSET($A$1, COUNTA($A:$A)-3, 0, 3, 1); see Figure 2-11.
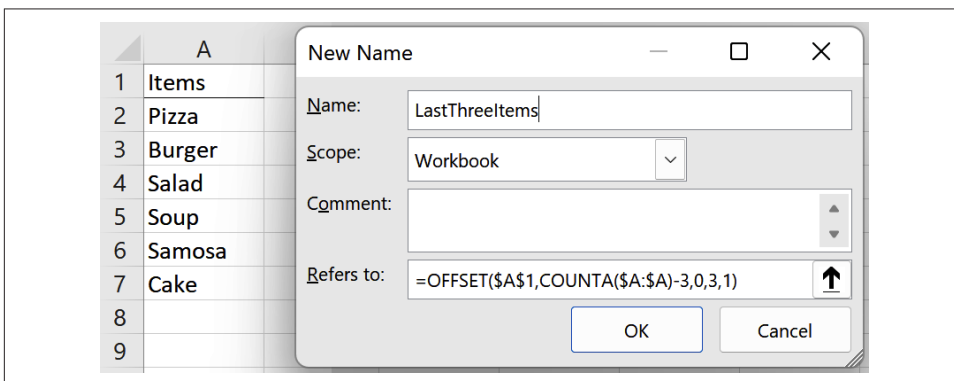
Figure 2-11. Using the OFFSET function to create a dynamic named range

Discussion

If you want to create a dynamic named range that automatically accommodates new rows, basing the range on a table column is the most reliable option; this is because using formulas may not return the results you want if the column contains blank rows or extra values you want to exclude. Using INDEX and OFFSET, however, is handy in situations where the data isn't in a table or you want to exclude part of the data. See Chapter 7 for other functions you might want to use to create a dynamic named range, such as UNIQUE.

2.8 Using Data Validation

Problem

You want to restrict the values you can enter in a cell to a specific range of values.

Solution

Excel's data validation tool lets you define rules that specify which values can be entered in one or more cells—for example, numbers greater than zero. You can also use it to display a message when the user selects a cell.

> When you create a data validation rule, Excel applies it to only any new values you enter, not existing ones. To circle any cells with values that violate validation rules, choose Data ⇒ Data Tools ⇒ Data Validation ⇒ Circle Invalid Data. You can remove these circles by choosing Data ⇒ Data Tools ⇒ Data Validation ⇒ Clear Validation Circles.

To add a data validation rule (see Figure 2-12):

1. Select the cells you want to apply a data validation rule to.
2. Choose Data ⇒ Data Tools ⇒ Data Validation ⇒ Data Validation to open the Data Validation dialog box.
3. In the Settings tab, select the type of data and range of values you want to limit data entry to. To limit data entry to integers greater than zero, for example, you select "Whole number" from the Allow drop-down list, "greater than" from the Data drop-down list, and type **0** in the Minimum box.
4. To display a message when the user selects a cell, click the Input Message tab, check the "Show input message" check box, and specify the message title and text.

5. To customize the message Excel displays when the user enters invalid data, click the Error Alert tab, check the "Show error alert" check box, and specify the message icon, title, and text.
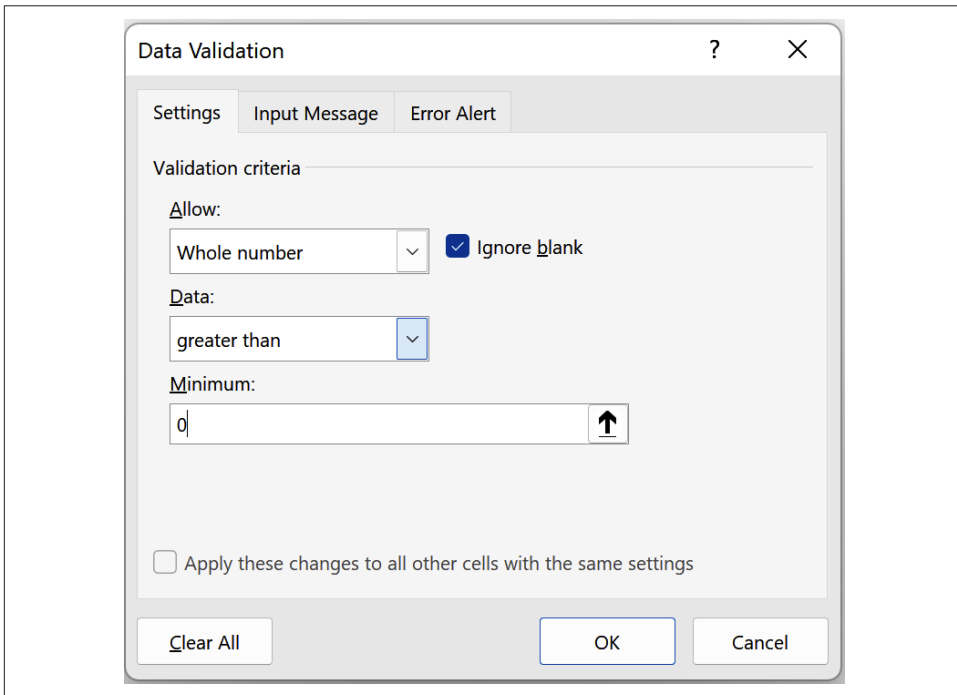
Figure 2-12. The Data Validation dialog box

If you want to edit or remove a data validation rule, you can do so as follows:

1. Select the cells you want to edit the rule for or remove it from.

2. Choose Data ⇒ Data Tools ⇒ Data Validation ⇒ Data Validation to open the Data Validation dialog box.

3. In the Settings tab, check the "Apply these changes to all other cells with the same settings" check box if you want to apply the change to any cells with the same rule you didn't select in step 1.

4. Click Clear All to clear the validation rule and input and error message settings. To remove only the validation rule, select "Any value" from the Allow drop-down list. Then click OK.

Discussion

This recipe gives an overview of using Excel's predefined data validation rules to ensure any values in cells are within a range of values. You can also use it to add a pop-up message to cells instead as an alternative to using notes and comments (see Recipe 1.17).

See Also

See Recipe 1.17 for searching for cells with data validation.

For more information about using the Custom and List data validation options, see Recipes 2.9 and 2.10.

2.9 Creating a Custom Data Validation Rule

Problem

You want to go beyond Excel's predefined data validation rules and use a custom formula.

Solution

Suppose you want to apply a data validation rule limiting data entry to only numbers or weekdays. You can do so by specifying a custom rule.

To define a custom rule, follow the steps in Recipe 2.8 to add a data validation rule, except that in step 3, select Custom from the Allow drop-down list in the Settings tab, type the custom formula in the Formula box, then click OK.

Similarly to Recipe 2.2, you write the formula relative to the top-left cell to which you want to apply the rule. To apply a custom rule to the cells A1:A10, for example, you write the formula relative to cell A1, and, behind the scenes, Excel "copies" it to the other cells.

The formula must evaluate to TRUE or FALSE. To limit data entry to numbers only, for example, use the formula =ISNUMBER(A1), and to restrict it to weekdays, you use =NOT(OR(WEEKDAY(A1)=1, WEEKDAY(A1)=7)).

Discussion

Excel's predefined data validation rules (see Recipe 2.8) let you restrict the values you enter to a range of values. This recipe shows you how to go beyond these rules and write your own custom formulas.

2.10 Entering Data with a Drop-Down List

Problem

You want to enter a value in a cell using a drop-down list of predefined values.

Solution

If you want to limit a cell's values to a predefined list, you can do so by defining a drop-down list. Drop-down lists can make it quicker and easier to enter values in cells and help make data more consistent.

You add a drop-down list using data validation (see Recipe 2.8) as follows:

1. Select the cells you want to add the drop-down list to.
2. Open the Data Validation dialog box by choosing Data ⇒ Data Tools ⇒ Data Validation ⇒ Data Validation.
3. In the Settings tab, select List from the Allow dropdown list and check the In-cell drop-down check box. Then use the Source box to define the list's values and click OK.

You can use several options to define the values:

Use a comma-separated list
> If you want to hardcode static values and it's unlikely they'll change, you can type their values directly in the Source box—for example, **Yes, No, Maybe**. Note that you can modify only the values by updating the data validation rule with this approach.

Use a range reference
> This option is helpful if you want to easily modify the drop-down list's values. Type the values in a range of cells; then enter the absolute reference for this range in the Source box—for example, **=Names!$A$2:$A$7**. Note that if you add or remove values, you must update the range reference in the data validation rule.

Use a table column
> This option is handy for dynamic lists where you want the drop-down list to include new values automatically. First, add the values to a column in a table. Then, reference the column in the Source box using the formula =INDIRECT ("*table_name*[*table_column*]"). If you've added the values to the Items column in a table named Products, you'd type **=INDIRECT("Products[Items]")**; see Recipe 7.12.

Use a named range

This option is the most flexible since you can base the drop-down list on a static or dynamic named range. Define the named range (see Recipes 2.6 and 2.7) and then type its name in the Source box—for example, `=Items` (see Figure 2-13).
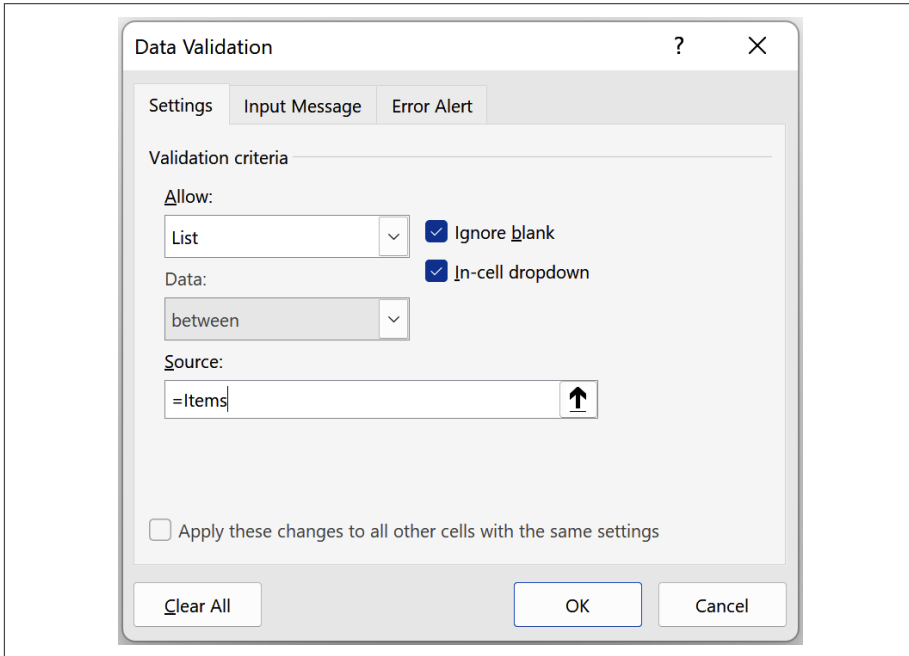
Figure 2-13. Using the Data Validation dialog box to define a drop-down list

Discussion

This recipe uses data validation to create a drop-down list. You can use several options for the list's values, depending on how dynamic you want it to be.

2.11 Defining Dependent or Cascading Drop-Down Lists

Problem

You want to enter data using two drop-down lists, where the values in the second list depend on the value you select in the first.

Solution

Suppose A2:A5 contains a list of food items with a *Food* label in A1, and B2:B4 contains a list of drink items with a *Drink* label in B1. You want to add a drop-down list to the cells in D2:D10, where the list's values are the *Food* and *Drink* labels in A1:B1.

You also want to add a drop-down list to the cells in E2:E10, where the list's values depend on the first, so if you select Food from the first drop-down list, the second list contains the values in A2:A5, and if you select Drink, the list contains the values in B2:B4 (see Figure 2-14).

	A	B	C	D	E	F	G
1	Food	Drink		Category	Item		
2	Burger	Coffee		Drink			
3	Cake	Cola			Coffee		
4	Pizza	Tea			Cola		
5	Salad				Tea		

Figure 2-14. Using dependent drop-down lists

You can solve this problem using a drop-down list with named ranges for the food and drink items. Here are the steps:

1. Create a named range (see Recipe 2.6) with the name *Food* that refers to =$A$2:$A$5 (the food items).

2. Create a named range with the name *Drink* that refers to =$B$2:$B$4 (the drink items).

3. Select D2:D10 and follow the steps in Recipe 2.10 to create a drop-down list with the source =$A$1:$B$1; this populates the list with the Food and Drink labels, which correspond to the named ranges created in steps 1 and 2.

4. Select E2:E10 and follow the steps in Recipe 2.10 to create a drop-down list with the source =INDIRECT(D2); this populates the list with values from the Food or Drink named range, depending on the value selected in the first drop-down list.

Discussion

This recipe offers a convenient way of creating dependent drop-down lists where the values in one list depend on the value you select from another. The key to this recipe is creating named ranges whose names match the values in the first list, since the second drop-down list can then refer to the named ranges for its values.

2.12 Using a Data-Entry Form

Problem

You have a dataset and use a form for data entry without using macros or Visual Basic for Applications (VBA).

Solution

Suppose you have a dataset where each row represents a separate record. You want to be able to create, update, delete, and find records using a quick data-entry form.

If you're using Excel for Windows, you can solve this problem by adding the Form command to the Quick Access Toolbar; follow the instructions in Recipe 1.19 to customize the toolbar, select the All Commands option from the "Choose commands from" drop-down list, then look for the Form command and add it to the Quick Access Toolbar.

To open the data-entry form, select one of the cells in the dataset and click the Form option in the Quick Access Toolbar; this opens the form shown in Figure 2-15, using the dataset's headings for the form labels.
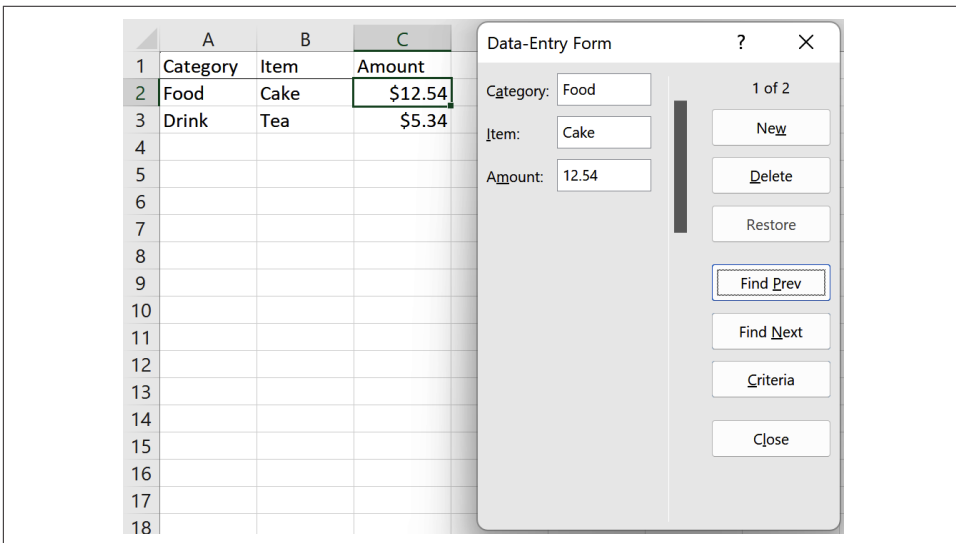
Figure 2-15. Using a quick data-entry form

You can then use the form to enter new records, edit or delete existing ones, go to the previous or next entry, or specify criteria to search for records.

Discussion

Excel's Form command can simplify data entry for tabular datasets. Unfortunately, the command isn't available in the ribbon by default, but you can manually add it to the Quick Access Toolbar, as shown in this recipe.

See Also

To create a custom UserForm whose behavior you program using VBA, see Recipe 18.19.

2.13 Sorting Data by Value, Format, or Custom List

Problem

You have data arranged in columns and want to sort by their values.

Solution

Excel's Sort tools let you sort one or more columns by cell value, format, or using a custom list (see Recipe 1.13).

To sort a column from the lowest to the highest value, select a cell in the column and choose Data ⇒ Sort & Filter ⇒ Sort A to Z or Home ⇒ Editing ⇒ Sort & Filter ⇒ Sort A to Z.

To sort a column from the highest to the lowest value, select a cell in the column and choose Data ⇒ Sort & Filter ⇒ Sort Z to A or Home ⇒ Editing ⇒ Sort & Filter ⇒ Sort Z to A.

To sort multiple columns or sort by cell color, font color, conditional formatting icon, or using a custom list, select a cell in the dataset or the entire range, then choose Data ⇒ Sort & Filter ⇒ Sort or Home ⇒ Editing ⇒ Sort & Filter ⇒ Custom Sort. This option opens the Sort dialog box, which lets you fine-tune how you want to sort the data (see Figure 2-16).
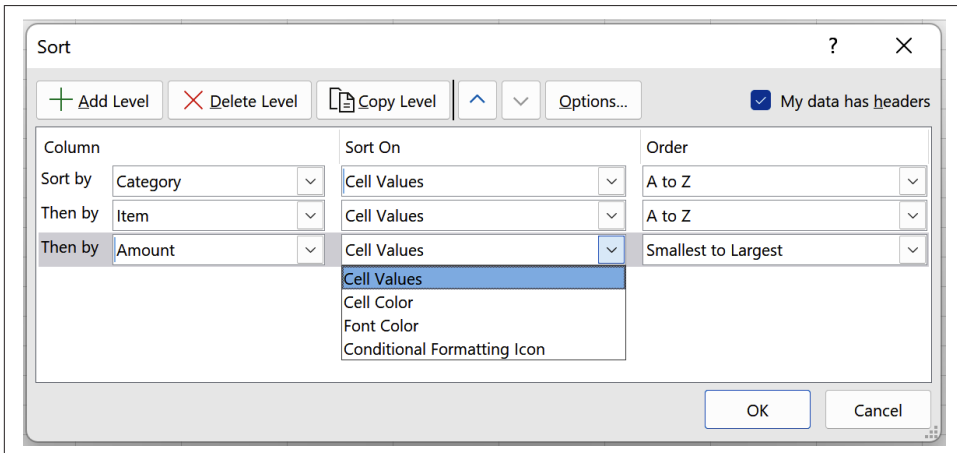
Figure 2-16. The Sort dialog box lets you sort by multiple columns

The Sort tools usually recognize headings (if formatted differently) and keep rows intact, but not always. A more reliable approach is to convert the data to a table before sorting it (see Recipe 2.18).

Discussion

This recipe describes quick ways of sorting data alphabetically or numerically, along with the more flexible features available in the Sort dialog box. Note that you can also sort data using filters (see Recipe 2.14) and functions (see Recipe 7.2).

2.14 Filtering Data

Problem

You have data arranged in columns and want to filter their values.

Solution

Excel's Filter tools let you filter the data in one or more columns by value, by format, and by applying criteria. It also includes a sort facility to sort and filter data simultaneously.

To add a filter, follow these steps (see Figure 2-17):

1. Make sure the data contains headings in the first row.
2. Select a cell or the entire dataset, then choose Data ⇒ Sort & Filter ⇒ Filter or Home ⇒ Editing ⇒ Sort & Filter ⇒ Filter; this adds a filter drop-down arrow to each column heading.
3. Click the drop-down arrow in the columns you want to filter the dataset by; this opens a filter menu showing different options depending on the data type held in that column.
4. Use the options to filter the data; then click OK.

Tables columns automatically include a filter drop-down menu by default, so you don't need to add one manually.
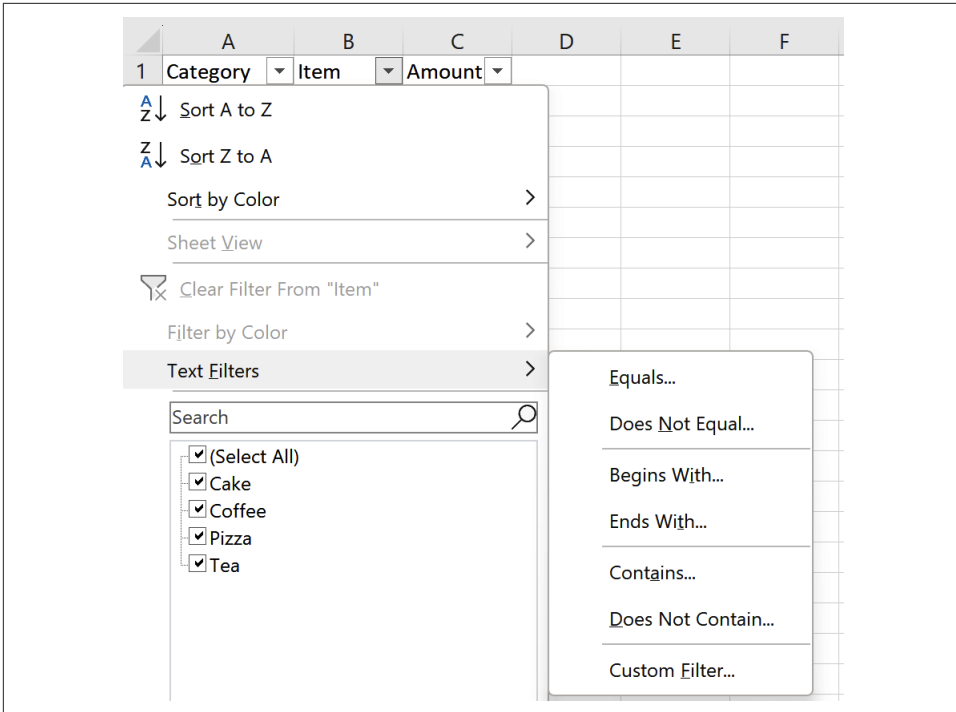
Figure 2-17. The Item column's filter menu

The Filter option lets you apply up to two criteria for each column in the dataset. You can, however, apply more complex criteria using the Advanced Filter option. Generally, you use the following steps (see Figure 2-18):

1. Make sure the dataset contains unique headings in the first row.

2. Set up a criteria range with headings that match the dataset's. So if the dataset has Category, Item, and Amount headings, the criteria range must also have these headings.

3. Specify criteria by adding one or more rows to the criteria range, where criteria in the same row use AND logic, and criteria in separate rows use OR logic. To show data where the Category is Food *and* the Item starts with C, you'd type **="=Food"** in the Category column and `C*` in the Item column in the same row; to show data where the Category is Food or the Item starts with C, you'd type the criteria in separate rows.

4. Choose Data ⇒ Sort & Filter ⇒ Advanced Filter to open the Advanced Filter dialog box.

5. Use the "List range" and "Criteria range" boxes to specify the dataset and criteria ranges; then click OK to apply the filter.
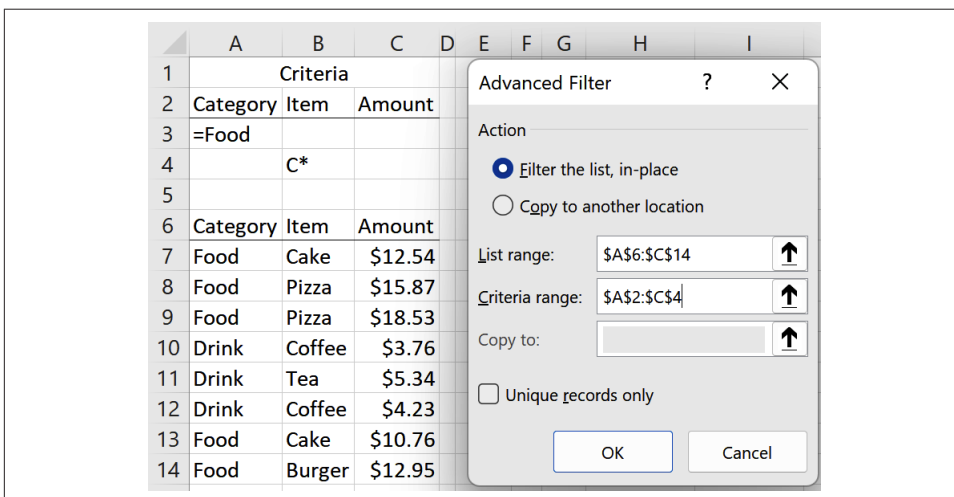
Figure 2-18. The Advanced Filter dialog box

> You can clear any filters you've applied by choosing Data ⇒ Sort & Filter ⇒ Clear. You can also reapply a filter after updating the data by choosing Data ⇒ Sort & Filter ⇒ Reapply.

Discussion

This recipe shows you how to add a filter facility to the columns in a dataset and introduces you to the more complex Advanced Filter option, which you can use for more complex criteria. Note that you can also filter data using the FILTER function (see Recipe 7.3).

2.15 Freezing Panes

Problem

When you scroll a worksheet, you want to stop column headings or other rows and columns from moving.

Solution

When you have a large dataset with column headings, it can be helpful to freeze the first row so that the headings don't move off the screen as you scroll through the data.

To freeze part of the screen, choose View ⇒ Window ⇒ Freeze Panes and select one of the options. For example, you can freeze the top row, the first column, or rows and columns based on the currently selected cell.

Discussion

This recipe is handy when working with large datasets since it lets you freeze any column headers so they don't scroll. Note that you don't need to do this when the data is in a table since tables automatically keep headings visible by default.

2.16 Using AutoSum

Problem

You want to quickly calculate a sum, count, or average of the numbers in a row or column.

Solution

A quick way to insert a sum of the numbers in a row or column is to use AutoSum, which adds a `SUM` formula to the specified cell.

To use AutoSum:

1. Select a cell next to the row or column you want to sum. For example, to sum the numbers in B2:B10, you'd select B11, and to sum the numbers in B2:D2, you'd select E2.
2. Choose Formulas ⇒ Function Library ⇒ AutoSum or Home ⇒ Editing ⇒ Sum; this inserts the formula `=SUM(`*range*`)` into the cell where Excel automatically detects which *range* to use.
3. Adjust the range if necessary; then press Enter/Return.

AutoSum can also calculate the average, count, maximum, or minimum number in a range—click the AutoSum command's drop-down arrow in the ribbon and select the option you want from the menu.

> If you want to find a range's sum, count, and average without inserting a function call, you can use the summary bar at the bottom of the worksheet. Select the cells you want to summarize, and Excel displays the summaries in the summary bar. You can also right-click the summary bar to customize its options.

Discussion

This recipe offers a quick alternative to manually inserting a call to the SUM, COUNT, AVERAGE, MAX, or MIN functions. As with any formula, you can edit the formula Excel creates.
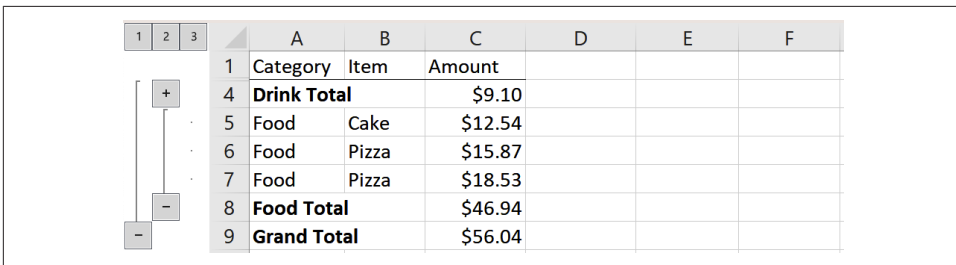
2.17 Using Outlines to Add Subtotals and Groups

Problem

You have a dataset and want to group its rows and include subtotals.

Solution

Suppose you have a dataset with Category, Item, and Amount column headings, and you want to add subtotals and group by each category (see Figure 2-19). You can do so using Excel's Outline tools.

	A	B	C	D	E	F
1	Category	Item	Amount			
4	**Drink Total**		$9.10			
5	Food	Cake	$12.54			
6	Food	Pizza	$15.87			
7	Food	Pizza	$18.53			
8	**Food Total**		$46.94			
9	**Grand Total**		$56.04			

Figure 2-19. Excel has added subtotals and groups to the data

To add subtotals, follow these steps (see Figure 2-20):

1. Make sure there are no blank rows; then sort the data by the column you want to group the data by—in this example, the Category column.

2. Select one of the cells in the dataset or the entire range; then choose Data ⇒ Outline ⇒ Subtotal to open the Subtotal dialog box. Use the dialog box to specify how to summarize the data.

When you click OK, Excel adds subtotals to the dataset and groups the rows; you can collapse and expand these using the - and + buttons on the left or by choosing Data ⇒ Outline ⇒ Show/Hide Detail.

If your data includes subtotals already, you can automatically add groups by choosing Data ⇒ Outline ⇒ Group ⇒ Auto Outline. You can also add groups manually by choosing Data ⇒ Outline ⇒ Group ⇒ Group and remove groups by choosing Data ⇒ Outline ⇒ Ungroup ⇒ Clear Outline.
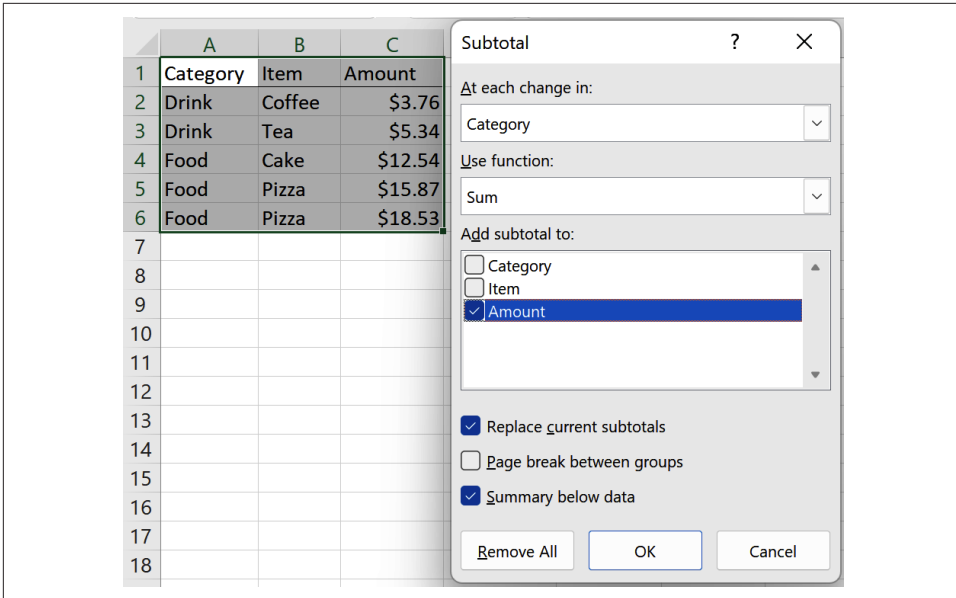
Figure 2-20. The Subtotal dialog box

Discussion

This recipe is handy if you want to automatically show subtotals in a list of data and group rows to make them easier to read. However, if you want to see only summaries of the data, consider using a PivotTable instead (see Chapter 11) since that is a more flexible option.

2.18 Using Tables

Problem

You have related data organized in rows and columns and want to make the data easier to manage and analyze.

Solution

When you have a dataset arranged in rows and columns, and each row is a separate item, it's generally easier to work with if you convert the dataset to a table. A *table* is a named object that automatically expands and contracts when you add or delete records, and autofills formulas, formatting, and data validation. Tables also make working with features such as PivotTables and charts simpler.

To convert a range to a table, select the data (include any headings) and then choose Insert ⇒ Tables ⇒ Table or Home ⇒ Styles ⇒ Format as Table, and pick a style to open the Create Table dialog box. Next, make sure Excel has chosen the correct range for the data, and use the "My table has headers" check box to indicate whether the range includes column headings (see Figure 2-21).
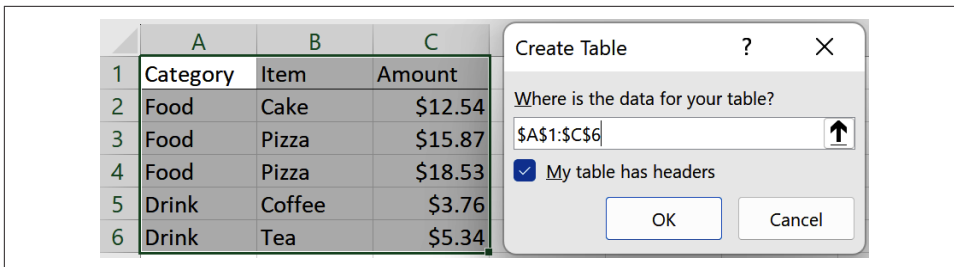
Figure 2-21. The Create Table dialog box

When you click OK, Excel converts the range to a table (see Figure 2-22).
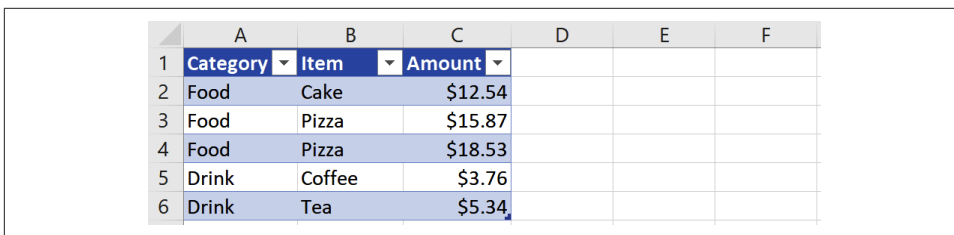
Figure 2-22. The table Excel creates from the data

By default, Excel adds drop-down filter arrows to each column heading, which you can use to sort and filter data (see Recipe 2.14). When you scroll the table data, these headings stay visible, so there's no need to freeze them manually.

An alternative way of filtering table values is to use a *slicer*: a floating object that lets you choose which values to display. To insert a slicer, select a cell in the table, choose Table Design ⇒ Tools ⇒ Insert Slicer, and then select which columns you want to include slicers for. When you click OK, Excel creates the slicer (see Figure 2-23).
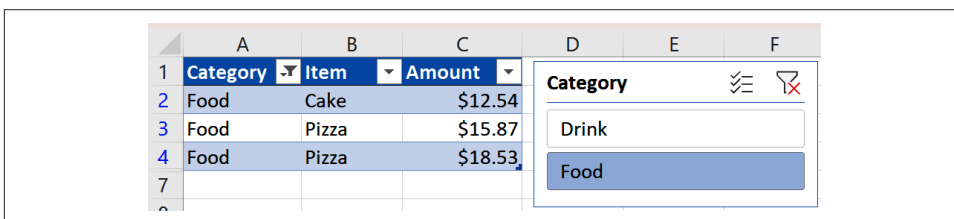
Figure 2-23. The table with a slicer

To use the slicer, either click the value you want the table to display or use the multi-select button to select multiple values. You can also select multiple values by pressing the Ctrl or Cmd key while selecting values.

To calculate column totals and other summaries, you can add a total row to the bottom of the table. First, add a total row by selecting a cell in the table and choosing Table Design ⇒ Table Style Options ⇒ Total Row. Then add, edit, or remove a column total by clicking in the total row cell for that column and selecting an option from its drop-down menu (see Figure 2-24).

	A	B	C	D	E	F
1	Category ▼	Item ▼	Amount ▼			
2	Food	Cake	$12.54			
3	Food	Pizza	$15.87			
4	Food	Pizza	$18.53			
5	Drink	Coffee	$3.76			
6	Drink	Tea	$5.34			
7	Total		$56.04 ▼			

Figure 2-24. A table's total row at the bottom of the table

> The totals in the total row automatically respond to filters, so if you exclude rows from the table, they're also excluded from the totals.

To add a new column to a table, you can usually type a value in the column next to it and Excel automatically adds the column to the table. Alternatively, select a cell in the table and choose Table Design ⇒ Properties ⇒ Resize Table or Home ⇒ Cells ⇒ Insert. You can also quickly remove duplicate values from a table by selecting a cell and choosing Table Design ⇒ Tools ⇒ Remove Duplicates to open the Remove Duplicates dialog box.

> If you want to convert a table back to a normal range, select a cell in the table and choose Table Design ⇒ Tools ⇒ Convert to Range.

Discussion

Tables include many features that make them necessary for records organized in ranges and columns. You can also use them in conjunction with other recipes, such as Recipes 2.8, 2.12, and 2.13.

You can also add formulas to tables in calculated columns and use their data and totals outside the table. See Recipe 2.19 for more details about this.

2.19 Using Structured References

Problem

You want to refer to table data in a calculated column or from outside the table.

Solution

Suppose you have a table with four columns—Salesperson, Jan, Feb, and Mar—which lists the sales per salesperson for each month (see Figure 2-25). You want to add a new column named Total, which sums each row's values in the Jan, Feb, and Mar columns.

	A	B	C	D	E	F	G
1	Salesperson	Jan	Feb	Mar			
2	AA	$15,487.00	$24,587.00	$23,415.00			
3	DE	$18,236.00	$21,459.00	$74,125.00			
4	AC	$12,684.00	$24,387.00	$62,458.00			
5	YR	$17,954.00	$34,158.00	$41,257.00			
6	EF	$32,457.00	$52,748.00	$85,276.00			
7	Total	$96,818.00	$157,339.00	$286,531.00			

Figure 2-25. The sales data in a table

You can solve this problem by adding a calculated column to the table that uses a formula with a *structured reference*: a reference to the table's structure. You create the calculated column like so (see Figure 2-26):

1. Add a new column to the table named Total.

2. Type **=SUM(** in the first cell in the Total column, select the cells in the Jan, Feb, and Mar columns in the first row to add their range to the formula, and then type a closing **)**. The formula should look something like =SUM(SalesTable[@[Jan]: [Mar]]), depending on the name of the table.

3. When you press Enter/Return, Excel autofills the rest of the rows with the formula.

A structured reference differs from a cell or range reference since it refers to parts of the table instead of the cells in which they reside. Each reference typically includes the table name and one or more specifiers; in the reference SalesTable[@[Jan]: [Mar]], for example, the SalesTable part is the name of the table, and [@[Jan]: [Mar]] refers to the range from the Jan to Mar columns in the current row.

Figure 2-26. The sales data, including the Total column

You can find or edit a table's name using the Table Name box in the ribbon. You can find this box by selecting a cell in the table and choosing Table Design ⇒ Properties.

Structured reference specifiers include the following:

[*column_name*]

    This refers to the data in the column named *column_name*, excluding the header and total row.

[[*column_name_1*]:[*column_name_2*]]

    This refers to the range from *column_name_1* to *column_name_2*.

[@*column_name*]

    This is the value in *column_name* in the current row (where the formula is).

[#All]

    This is the entire table, including the data, headers, and total row.

[#Data]

    This refers to the rows with data and excludes the headers and total row.

[#Headers]

    This is the column headers.

[#Totals]

    This refers to the total row.

[[#Totals],[*column_name*]]

    This refers to the *column_name* total row cell.

So SalesTable[Jan] refers to the data in SalesTable's Jan column, and =Sales Table[[#Totals], [Jan]] refers to the Jan column's total row cell.

You can use structured references both in the table and outside it. Typing **=SUM (SalesTable[Jan])** in a cell outside the table, for example, calculates the sum of all the values in SalesTable's Jan column, while typing **=SalesTable[[#Totals], [Jan]]** returns the value in the Jan column's total row; these values may differ if the table is filtered because the totals row responds to filters.

> The [#Data] specifier can sometimes be left out. Typing **=ROWS (SalesTable)** and **=ROWS(SalesTable[#Data])**, for example, both return the number of data rows in the table named SalesTable.

Just like cell references, structured references can be relative or absolute (see Recipe 2.1), which affects their behavior when copied.

To refer to a single column, use *table_name[column_name]* if you want a relative reference and *table_name[@column_name]* if you want a relative reference that refers to the current row. You can also use [*table_name[[column_name]:[column_name]]*] if you want an absolute reference to a single column and *table_name[@[column _name]:[column_name]]* if you want an absolute reference that refers to the cell in the current row.

To refer to a range of columns, you can use *table_name[column_name_1]:table _name[column_name_2]* if you want a relative reference and *table_name[@ column_name_1]:table_name[@column_name_2]* if you want a relative reference that refers to the current row. You can use *table_name[[column_name_1]:[column _name_2]]* if you want an absolute reference and *table_name[@[column_name_1]: [column_name_2]]* if you want an absolute reference that refers to the current row.

Discussion

This recipe details how to use structured references in calculated columns and outside the table. These can be useful if you want to insert a total row or refer to table data elsewhere in your spreadsheet.

> Some features of Excel—for example, named ranges and conditional formatting—don't recognize structured references, so you have to use cell references instead. Excel, however, keeps track of the table range, so the cell references automatically update when you add or delete rows.

Using Formulas

Most spreadsheets rely on formulas to perform calculations and manipulate data using operators, functions, references, and constants. However, as these grow more complex, they can quickly become unwieldy and prone to error.

This chapter contains recipes designed to help you work with formulas more effectively and deal with any problems efficiently. Areas covered include:

- Using arithmetic, comparison, and reference operators, including ones that return a spill range or implicit intersection
- Working with arrays (matrices of values), including array constants, dynamic arrays, and legacy arrays
- Dealing with Excel's error values
- Using formula-auditing tools to analyze and debug formulas, including tracing interdependencies and errors, running error checks, stepping through formulas, and watching cell values and formulas with the Watch window
- Using automatic, manual, and iterative calculations to control when formulas recalculate and resolve circular references

3.1 Using Operators and Order of Precedence

Problem

You want to know what operators are available in Excel, what they do, and their order of precedence.

Solution

Excel includes four types of operators: arithmetic, comparison, text concatenation, and reference.

The *arithmetic operators* let you perform basic mathematical operations and return numbers. They include the following:

+ *and* -
> These are for addition and subtraction (for example, =A1+B1 or =A1-B1). You can also use the minus sign to indicate negative numbers, for example, =-2.

* *and* /
> These are for multiplication and division (for example, =A1*B1 or =A1/B1).

%
> This is used for percentages or dividing by 100 (for example, =20%); this is equivalent to =20/100 or =0.2.

^
> This is the exponentiation operator. The formula =8^2, for example, calculates 8^2, which returns 64.

The *comparison operators* let you compare two values and return TRUE or FALSE. They are as follows:

= *and* <>
> These test whether a value is equal to, or not equal to, another. For example, =A1=B1 returns TRUE if the value in A1 equals that in B1, and =A1<>B1 returns TRUE if their values are not equal.

< *and* <=
> These test whether a value is less than—or less than or equal to—another. For example, =A1<B1 returns TRUE if the value in A1 equals that in B1, and =A1<=B1 returns TRUE if the value in A1 is less than or equal to that in B1.

> *and* >=
> These test whether a value is greater than—or greater than or equal to—another. For example, =A1>B1 returns TRUE if the value in A1 is greater than that of B1, and =A1>=B1 returns TRUE if the value in A1 is greater than or equal to that of B1.

The *text concatenation operator* is &, which you use to join two text strings. See Recipe 5.1 for more details about using this operator.

The *reference operators* combine cells or ranges. They are as follows:

:

This is the range operator, so A1:A10 refers to the cells from A1 to A10, inclusive.

, or ;

Depending on your regional settings (see Recipe 3.2), you can use either `,` or `;` as a union operator, which combines ranges in some functions. The formula `=SMALL((A1:A10, C1:C10), 3)`, for example, returns the third smallest number in the ranges A1:A10 and C1:C10 (see Recipe 8.6). Note that it double-counts any cells that appear in each range, so it's not a strict mathematical union.

Space

This is the intersection operator, which returns the cells common to two references. `=A1:A10 A6:C6`, for example, returns the value in cell A6 because this cell is common to both ranges.

#

This is the spill range operator, which references an entire range in a dynamic array formula (see Recipe 3.5). Excel also uses the # in error values such as `#NAME!`, to indicate when a cell isn't wide enough to display its contents, or when a cell has an invalid format for its value.

@

This is the implicit intersection operator. It's used with structured references to refer to the current row (see Recipe 2.19) and dynamic arrays (see Recipe 3.6).

> Each operator works with specific types, such as numbers or text. If you enter a different type, Excel tries to convert it. So if you use the formula `="3"+"5"`, Excel converts the text strings "3" and "5" to the numbers 3 and 5 because the + operator expects numeric values. The formula `="3+5"*2`, however, returns a `#VALUE!` error because Excel can't convert the text "3+5" to a number.

If a formula contains several operators, Excel executes them in the following order:

1. The `:` range operator
2. The `@` and `#` implicit intersection and spilled range operators
3. The Space intersection operator and `,` or `;` union operators
4. The `-` negation operator when indicating a negative number
5. The `%` percent operator
6. The `^` exponentiation operator

7. The * and / multiplication and division operators

8. The + and - addition and subtraction operators

9. The & text concatenation operator

10. The =, <, >, <=, >=, and <> comparison operators

If the formula includes operators with the same precedence, Excel evaluates them from left to right. You can also use parentheses to override the evaluation order since Excel evaluates expressions in parentheses first; for example, the formula =1+2*3 multiplies 2 by 3 and adds 1 to the result, returning 7, while =(1+2)*3 adds 2 to 1 and multiplies the result by 3, returning 9.

Discussion

This recipe gives an overview of Excel's operators and their order of precedence. Many of these operators are commonly used in Excel formulas, while others, such as # and @, are less well-known.

See Also

You can also use the * and + operators as AND and OR operators in array formulas; see Recipe 7.6.

3.2 Using Excel in Different Regions and Languages

Problem

You want to know how your operating system's region and language settings affect Excel.

Solution

Your operating system's region and language settings can affect Excel's behavior in several ways.

The operating system's language setting determines the names Excel uses for its functions. If the language is English, you'd use a function named SUM to add numbers together, but if the language is German, you'd use SUMME instead. Excel Functions Translator is an add-in that translates function names to different languages, and you can also use websites such as Excel-Translator.

Your region's decimal separator can also affect how you write formulas since it determines which character you use to separate function arguments. If your region uses a full-stop decimal separator, your argument separator is a comma, but if your region

uses a comma as a decimal separator, your argument separator is a semicolon (;) instead (see Recipe 3.1). Having a decimal comma separator also affects how you define array constants (see Recipe 3.3).

> You can override your operating system's decimal separator by choosing File ⇒ Options ⇒ Advanced ⇒ Use System Separators in Excel for Windows or Excel ⇒ Preferences ⇒ Edit ⇒ Use System Separators in Excel for Mac.

Your region also affects how Excel interprets and formats any dates you enter. So if you type **=1/12/2023**, Excel interprets the date as January 12 in some regions and December 1 in others.

Discussion

This recipe provides an overview of how your language and region settings can affect Excel. Note that if you update your settings, Excel automatically applies them to any existing workbooks you open.

3.3 Using Array Constants

Problem

You want to use a hardcoded set of values in a formula or use them to specify values in a column and/or row.

Solution

Suppose you want to find the average of the three largest values in the range A1:A10. You can do so using the formula =AVERAGE(LARGE(A1:A10, {1,2,3})), which is a more compact way of writing =AVERAGE(LARGE(A1:A10,1), LARGE(A1:A10,2), LARGE(A1:A10,3)).

The {1,2,3} in the formula is an *array constant*: a hardcoded list of numeric, text, or logical values enclosed in braces ({ }). So {1,2,3} is an array constant containing the numbers 1, 2, and 3, while {"a","b","c"} is an array constant containing the text values *a*, *b*, and *c*. Note that array constants can't contain functions, formulas, references, or other arrays; for example, the array ={1,2,A1} is invalid since it contains a cell reference.

If your operating system's region uses a comma as a decimal separator, separate each value in an array using a \ instead of a comma. To define an array containing the values 1, 2, and 3, you'd use {1\2\3}. See Recipe 3.2.

Array constants are usually used to pass multiple values to a formula instead of a single value and often return a dynamic array (see Recipe 3.4). For example, the formula =LARGE(A1:A10, {1,2,3}) returns an array containing the three largest numbers in the range A1:A10. However, the formula =AVERAGE(LARGE(A1:A10, {1,2,3})) returns a single value since the AVERAGE function uses the three values to return a single number.

You can also use an array constant to specify values in a horizontal, vertical, or 2-D array. You define a horizontal array using a comma separator, so typing **={1,2,3}** in a cell displays the values in a row. You define a vertical array using a semicolon separator, so typing **={1;2;3}** displays the values in a column. Finally, you define a two-dimensional array using a comma separator for the row values and a semicolon to separate each row, so typing **={1,2,3;4,5,6}** displays two rows with 1, 2, and 3 in the first row and 4, 5, and 6 in the second.

If you're using Excel 365, you can type the array formula—for example, **={1,2,3}**—in a cell and press Enter/Return to see the array's values. For earlier versions of Excel, you need to select the entire output range, type the formula in the top-left cell, and then press Ctrl+Shift+Enter/Return. See Recipe 3.4 for more details.

Discussion

Array constants are a convenient way of hardcoding numeric, text, or logical values without entering them into a range of cells. You can also assign names to any you want to reuse (see Recipe 2.6).

3.4 Using Dynamic and Legacy Array Formulas

Problem

You want to use a formula that returns multiple values in an array.

Solution

Suppose you have a set of numbers in cells A1:A4 and want to multiply each one by 2 and put the results in B1:B4. Instead of typing **=A1*2** in B1, **=A2*2** in B2, and so on, you can use a dynamic or legacy array formula to achieve the same result.

If you're using Excel 2021 or Excel 365, you can use a *dynamic array* formula: a formula that returns multiple values in a resizable array and "spills" outside the cell bounds to populate multiple cells. To populate B1:B4 using a dynamic array formula, for example, you type **=A1:A4*2** in cell B1, and when you press Enter/Return, the formula automatically populates B1:B4—the *spill range*—with the results (see Figure 3-1).

Figure 3-1. Dynamic array formulas automatically spill into neighboring cells

If you're using an older version of Excel, you can't use dynamic array formulas. However, you can achieve similar results using a legacy—or CSE[1]—array formula. *Legacy array formulas* work similarly to dynamic array formulas, except they don't automatically resize or spill, and you need to press Ctrl+Shift+Enter/Return to enter them. To populate B1:B4 using a legacy array formula, for example, use the following steps:

1. Select the range B1:B4.

2. Type the formula **=A1:A4*2** in the formula bar.

3. Press Ctrl+Shift+Enter/Return to enter the formula. When you do so, Excel surrounds the formula with curly braces to become {=A1:A4*2} and populates B1:B4 with the results (see Figure 3-2).

Figure 3-2. Legacy array formulas don't automatically spill into neighboring cells, and you need to press Ctrl+Shift+Enter/Return to execute them

---

1 *CSE* stands for Ctrl+Shift+Enter—the key combination you need to press to enter this type of formula.

Discussion

This recipe gives a handy overview of using dynamic and legacy array formulas. Generally, dynamic array formulas are more straightforward than legacy ones since they resize when you add or remove data from the source range and automatically spill into neighboring cells.

> Dynamic array behavior is native to all functions in Excel 2021 and Excel 365, so any formula that returns more than one value will automatically spill.

See Also

If you're using Excel 365, you can make a function spill that otherwise would not do so by using a LAMBDA helper function. See Chapter 17 for more details about these functions.

3.5 Using Spill Range References

Problem

You have a dynamic array formula and want to refer to its entire spill range.

Solution

Suppose you have a dynamic array formula (see Recipe 3.4) in cell B1 and you want to get a reference to its spill range. You can do so using the # spill range operator.

You use the spill range operator by typing the first cell in the spill range (the one containing the dynamic array formula) followed by a #. To get a reference to the spill range of B1, you'd type **=B1#**, and to add 3 to each value, you'd type **=B1#+3** (see Figure 3-3).

	A	B	C	D	E	F
1	1	2	=B1#+3			
2	2	4	7			
3	3	6	9			
4	4	8	11			

Figure 3-3. Using the # spill range operator to refer to a cell's spill range

Discussion

Dynamic array formulas automatically resize their spill range when data gets added or removed from the source range. Using the # operator is a convenient way of referring to its spill range since it automatically matches the spill range size, so you don't need to manually update any formulas that refer to it.

See Also

You can use the # operator to create dynamic named ranges based on dynamic array formulas; see Recipe 2.7.

3.6 Preventing Dynamic Array Behavior

Problem

You have a dynamic array formula that returns multiple values and want to prevent it from spilling into other cells.

Solution

If you're using a version of Excel that supports dynamic arrays (see Recipe 3.4), any formula that returns multiple values spills its results into neighboring cells by default. You can, however, prevent this behavior by prefixing the function name or range with the @ implicit intersection operator, which reduces multiple values to a single value.

The @ operator works as follows:

- If the formula returns a single result, the operator returns that result. If a dynamic array formula in cell B1 returns a single value, the formula `=@B1#` returns that value.

- If the formula returns a range, the operator returns the value in the same row or column as the formula. So if a dynamic array formula in cell B1 spills to fill the range B1:B4, typing **=@B1#** in cell C1 returns the value in B1, while typing the same formula in cell C3 returns the value in B3.

- If the formula returns an array, the operator returns the first value in the array. So typing **=@{1,2,3}** returns 1 since this is the array's first value.

The @ operator is often used for backward compatibility with older versions of Excel that don't support dynamic array behavior. Typing `=A1:A4*B1:B4` in Excel 365, for example, returns a dynamic array of four values while typing the same formula in an older version of Excel returns a single result. When you open a workbook in Excel 365 that you created in an older version, Excel automatically adds the @ operator to formulas that may now return multiple values to ensure they return a single result.

Discussion

The @ operator is usually used in tables to refer to values in the current row (see Recipe 2.19). You can, however, also use it to prevent Excel's default dynamic array behavior and maintain backward compatibility with older versions of Excel.

3.7 Using the Insert Function or Function Builder Tool

Problem

You want to search for a function or want guidance on using one.

Solution

When you enter a formula in a cell, you generally type it directly in the cell or formula bar. As you type the formula, Excel displays information about each function's arguments, and its AutoComplete feature shows a list of matching functions and defined names.

You can switch off the formula AutoComplete in Excel for Windows by choosing File ⇒ Options ⇒ Formulas and unchecking the Formula AutoComplete check box; if you're using Excel for Mac, you can find this option by choosing Excel ⇒ Preferences ⇒ Auto-Complete.

If you want extra assistance, select the cell you want to add a formula to and then click the Insert Function icon fx next to the formula bar, or choose Formulas ⇒ Function Library ⇒ Insert Function; this opens the Insert Function dialog box in Excel for Windows or the Function Builder pane in Excel for Mac. This feature lets you search for or select a function and helps you enter values for its arguments (see Figure 3-4).
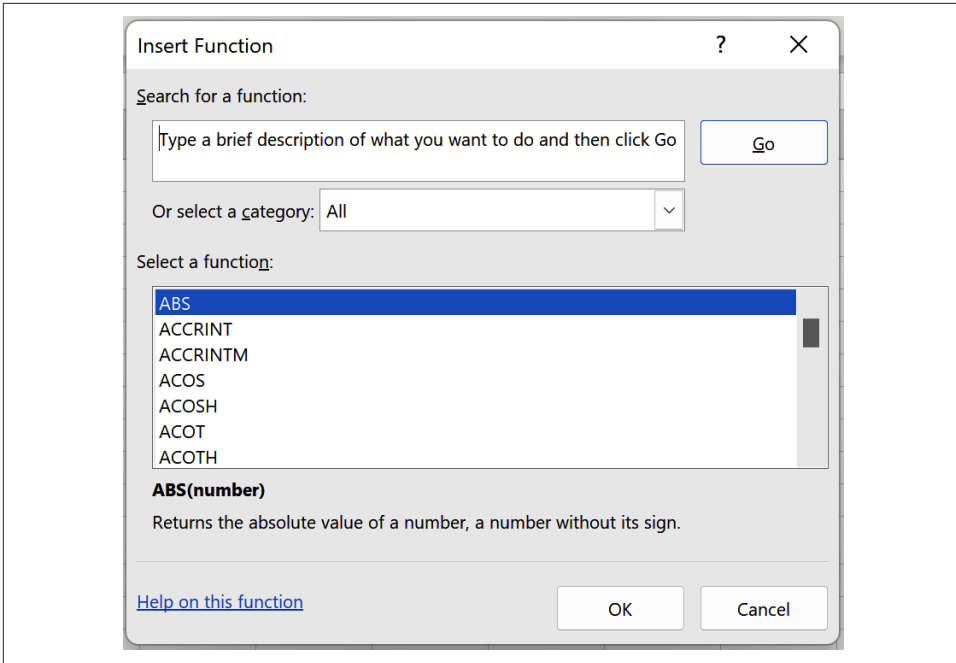
Figure 3-4. The Insert Function dialog box

Discussion

Excel's Insert Function or Function Builder tool is handy when you need more help entering a function. It also lets you browse a list of available functions, excluding secret ones such as `DATEDIF` (see Recipe 6.13).

3.8 Adding Notes to Numeric Formulas

Problem

You have a formula that returns a number and want to add a comment to it.

Solution

If you have a formula that returns a number, you can add a note to it using the `N` function. This function takes the form `=N(`*value*`)` and converts its *value* argument to a number. If *value* is a number, the function returns *value* unchanged, but if it's a text string, the function returns zero; this means you can use the syntax `=`*formula*`+N(`*note*`)` to add a *note* to a numeric *formula* since it simply adds a zero—for example, `=SUM(A1:A3)+N("Sums the first three cells")`.

You can use this technique only with functions that return numbers, including dates and times. Trying to use N with a text function results in a #VALUE! error, and using it with a function that returns a TRUE or FALSE logical value converts TRUE to 1 and FALSE to 0.

Discussion

The N function offers a handy way of adding a comment to a numeric formula. It's more discreet than using Recipe 1.16 and isn't limited to cell formulas.

3.9 Showing Formulas

Problem

You want to display the formula held in one or more cells.

Solution

If you want to see all the formulas in a worksheet instead of their calculated results, choose Formulas ⇒ Formula Auditing ⇒ Show Formulas. This option toggles between seeing the formulas and their values, so select this option again to see the calculated results.

If you want to see the formula held in a single cell, you can use the FORMULATEXT function; this takes the form =FORMULATEXT(*reference*), where *reference* is the cell or range whose formula you want to see. So if cell A1 contains the formula =TODAY(), typing **=FORMULATEXT(A1)** in cell A2 displays the text *=TODAY()*.

If you want to type a formula in a cell without Excel evaluating it, prefix the formula with an apostrophe. For example, typing **'=TODAY()** in cell A3 displays the text *=TODAY()* instead of the current date.

Discussion

This recipe is handy if you want to see a cell's formula without first selecting it. Generally, you use the Show Formulas command to see each formula in a workbook and the FORMULATEXT function to display a single formula.

3.10 Using the Watch Window

Problem

You want to keep track of cell values and formulas, including ones in other worksheets and workbooks.

Solution

If you want to watch a cell's value, you can do so using the Watch window. This tool lets you keep track of cell values, including ones in other worksheets and workbooks.

To add a cell to the Watch window, follow these steps (see Figure 3-5):

1. Select the cell you want to watch.
2. Choose Formulas ⇒ Watch window to open the Watch window.
3. Click Add Watch and then click Add; this adds the cell to the Watch window.
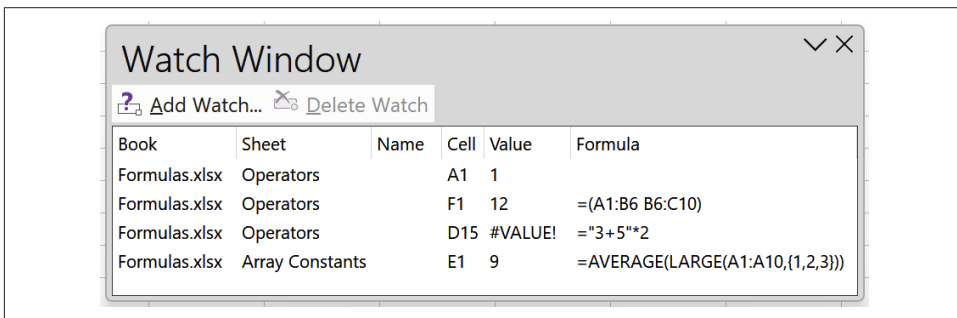
Figure 3-5. The Watch window

The Watch window is a toolbar that you can move or dock. Depending on your version of Excel, you can dock it below the ribbon or move it to the middle of your screen as a floating window.

To remove a cell from the Watch window, select the cell in the Watch window and click Delete Watch.

Discussion

The Watch window is a handy way of watching cell values without having to scroll or navigate to them. It's convenient when you have large spreadsheets since you can watch the cells you're most interested in from a single location and confirm calculations and results.

See Also

You can also monitor cell values by creating a linked picture; see Recipe 13.9.

3.11 Showing Cell Interdependencies

Problem

You want to examine which cells and ranges a formula refers to or which cells refer to a specific cell.

Solution

Suppose you have a formula in a cell and want to see which cell values it uses. You can do so with the Trace Precedents tool, which draws arrows from the cells and ranges the formula uses to the formula itself. To use the tool, select the cell containing the formula and then choose Formulas ⇒ Formula Auditing ⇒ Trace Precedents to see the formula's immediate precedents, click it again to see their immediate precedents, and so on (see Figure 3-6).

Figure 3-6. The Trace Precedents tool draws arrows from any cells that the formula depends on

> Excel generally uses blue arrows to draw the relationships between cells and ranges. A red arrow indicates that a cell or range cascades an error value to another reference. See Recipe 3.13 for more details.

The Trace Dependents tool works similarly to Trace Precedents, except it shows where a specific cell is referred to. To use the tool, select the cell and then choose Formulas ⇒ Formula Auditing ⇒ Trace Dependents to see its immediate dependents, click it again to see their immediate dependents, and so on (see Figure 3-7).
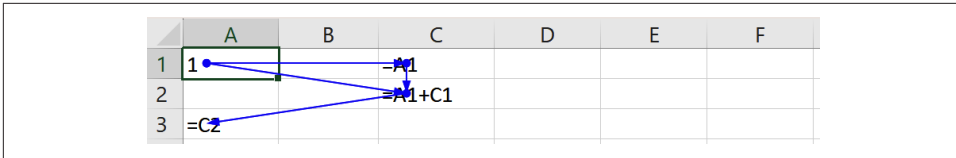
Figure 3-7. The Trace Dependents tool draws arrows to any cells that refer to the cell

When you no longer need to see the tool's arrows, choose Formulas ⇒ Formula Auditing ⇒ Remove Arrows to remove them.

Discussion

This recipe offers a handy technique for visualizing the relationships among cells, or finding out where a cell is used in an unfamiliar workbook. You can use it with a single cell or multiple cells at once, so if you want to see all the interdependencies among a worksheet's cells, select all the cells and choose Trace Precedents or Trace Dependents.

3.12 Performing Background Error Checks

Problem

You want Excel to add an error indicator to cells that may contain errors.

Solution

Excel can perform background error checks to flag cells with potential errors. When it spots one, it adds an error indicator ◤ to the cell's top-left corner—and displays an error button ⚠ when you select the cell.

Clicking the cell's error button opens a menu giving you a brief description of the error and a list of options to help you handle it. These include the following:

Help on this Error
> This gives more information about the error and offers helpful advice on correcting it.

Trace Error
> This uses the Trace Error tool to highlight whether any of the cell's references contains an error value (see Recipe 3.14).

Show Calculation Steps
> This opens the Evaluate Formula dialog box, which lets you evaluate different parts of the formula and step through its calculations.

Ignore Error
> This removes the cell's error indicator.

Edit in Formula Bar
> This closes the menu and moves the focus to the formula bar.

Error Checking Options
> This lets you update Excel's error-checking rules or reset any errors you've told Excel to ignore.

The error-checking rules determine which errors—or potential errors—Excel should highlight and let you switch off background error checks. To customize these rules, choose Error Checking Options from a cell's error menu and check or uncheck the options displayed in the Error checking rules section; you can also find this section by choosing File ⇒ Options ⇒ Formulas if you're using Excel for Windows, or Excel ⇒ Preferences ⇒ Error Checking if you're using Excel for Mac.

Discussion

This recipe gives an overview of Excel's background error checks, which highlight cells with error values and potential issues. You can correct these errors individually or run through them one by one using the Error Checking dialog box (see Recipe 3.13).

3.13 Using Error Checking

Problem

You have worksheet cells with errors and want to address them one by one.

Solution

If your worksheet contains several cells with error values or that have been flagged by Excel's background error check, you can go through them one by one using the Error Checking dialog box. Open the dialog box by choosing Formulas ⇒ Formula Auditing ⇒ Error Checking ⇒ Error Checking (see Figure 3-8).

The Error Checking dialog box includes similar options to the error button menu (see Recipe 3.12) with handy links to other tools. You can also navigate the worksheet's errors using the Next and Previous buttons.
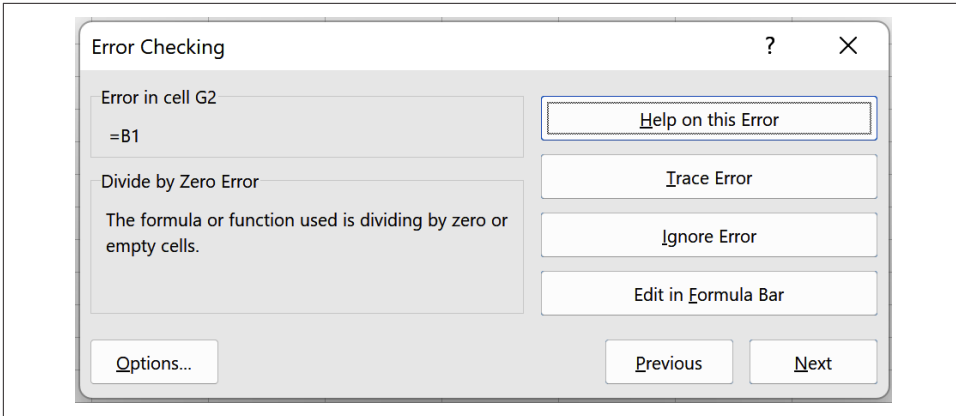
Figure 3-8. The Error Checking dialog box

Discussion

The Error Checking tool provides functionality similar to Excel's background error checks, except that it lets you address any errors individually. Note that it doesn't include any errors you've told Excel to ignore; you can reset these by clicking Options followed by Reset Ignored Errors.

3.14 Tracing Errors

Problem

You have a cell with an error value that may be caused by a reference to another cell.

Solution

When you have a cell with an error value that refers to other cells or ranges, the error may come from one of these references instead of the cell formula itself. For example, if B1 has a #DIV/0! error value and the formula for C2 is =B1, C2 will also have a #DIV/0! error value, which it inherits from B1.

In this situation, you can use the Trace Error tool to help track down the source of the error. It works similarly to the Trace Precedents tool (see Recipe 3.11) except that it works only with cells with error values.

To use the Trace Error tool, select the cell containing the error you want to trace; then choose Formulas ⇒ Formula Auditing ⇒ Error Checking ⇒ Trace Error. When you do so, the tool examines any references the cell's formula depends on and looks for one with an error value. When it finds one, it draws a red arrow from that reference to the formula and repeats the process until it's drawn a hierarchy of error value

dependencies. If no such references exist, the tool draws blue arrows between the cell and its references, and the process stops.

Figure 3-9 shows an example of the Trace Error tool output.

Figure 3-9. Output from the Trace Error tool

You then solve the hierarchy of error values, starting with the root error. So if C2 and B1 have a #DIV/0! error value and C2 depends on B1, you correct the error in B1 and see if it eliminates the error in B1. If it doesn't, remove any remaining arrows by choosing Formulas ⇒ Formula Auditing ⇒ Remove Arrows. Then, use the Trace Error tool again to see if there are any extra error values to solve.

Discussion

This recipe gives an overview of using the Trace Error tool to track down elusive errors in cells that refer to other cells and ranges.

3.15 Correcting Error Values

Problem

You have a cell with an error value and you want to know its meaning and how to correct it.

Solution

When Excel can't evaluate a formula, it uses an error value—for example, #DIV/0!— to indicate the problem. Here's a list of Excel's error values, with suggestions for correcting them:

##
  This usually means that the column isn't wide enough to display the cell's value, so try double-clicking between the column headers to autofit the width. It may also mean that the cell contains a negative date or time value, which is invalid; you can see if this is the case by changing the cell's format to General.

#BLOCKED!
  This occurs when a required resource can't be accessed. It can occur when, for example, you provide an invalid URL to the IMAGE function (see Recipe 13.6) or exceed the image or file size.

#CALC!

This is an unspecified calculation error in an array.

#DIV/0!

This occurs when you try to divide a number by zero or an empty cell. Check the value of the divisor and consider using the IF function to trap any errors—for example, =IF(A2=0, 0, A1/A2); see Recipe 7.5.

#N/A!

This occurs when a value isn't available to the formula and usually happens when a lookup function—for example, XLOOKUP—can't find a match. Consider using the IFNA function to replace it with a friendlier message (see Recipe 7.7).

#NAME?

This means Excel doesn't recognize a name or function. Make sure any names and functions are spelled correctly.

#NULL!

This occurs when you try to use the Space intersection operator (see Recipe 3.1) with two ranges that don't intersect—for example, =SUM(A1:A5 C1:C5). Make sure you're using the correct ranges with the correct operator.

#NUM!

This is when the formula contains invalid numeric values—for example, =SQRT(-2). Check the formula's values and consider trapping the error using the IF function.

#REF!

This occurs when the formula contains an invalid reference, and it usually means the formula refers to a row or column you've subsequently deleted—for example, if cell D1 contains the formula =B1+C1 and you delete the C column. In this situation, try rebuilding the formula.

#SPILL!

This occurs when you're using a dynamic array or spill range, and a populated cell is blocking its output. Try clearing the blocking cell.

#VALUE!

This happens when you try to perform an operation on the wrong data type—for example, using the formula =A6+B6 when A6 or B6 contains text values. You can correct this specific example by changing the formula to =SUM(A6:B6) because the SUM function ignores text values.

Discussion

This recipe offers a handy overview of Excel's error values and strategies for correcting them. To get more help with a specific error, select the Help on this Error option from the cell's error button menu or the Error Checking dialog box (see Recipe 3.13).

3.16 Evaluating Formulas

Problem

You have a formula in a cell and want to step through its calculations.

Solution

If you want to step through a formula's calculations and evaluate different parts, use the Evaluate Formula dialog box. To open the dialog box, select the cell with the formula you want to examine and then select Formulas ⇒ Formula Auditing ⇒ Evaluate, or select Show Calculation Steps from the Error Checking dialog box or the cell's error button menu.

When you open the dialog box, it displays the cell's formula and underlines any expressions that can be evaluated (see Figure 3-10).
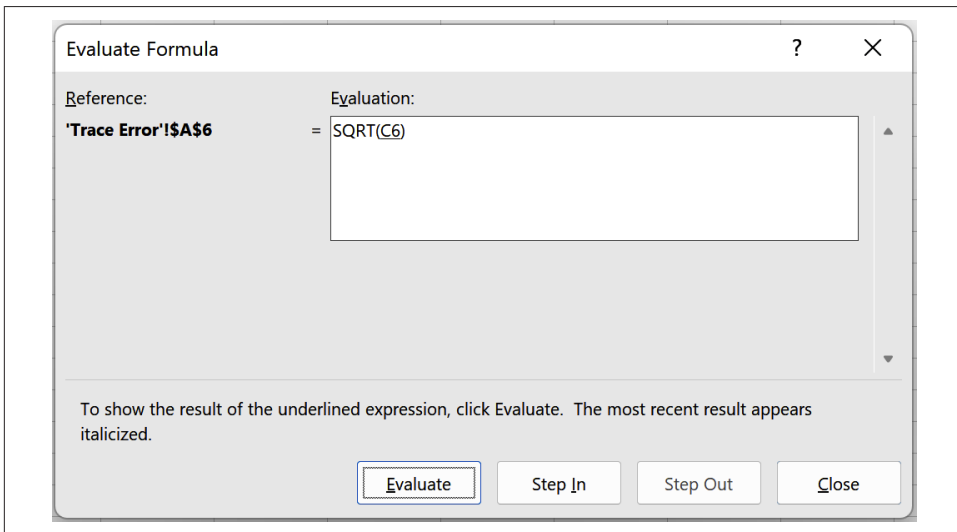
Figure 3-10. The Evaluate Formula dialog box underlines expressions it can evaluate

To evaluate an expression, click the expression and then click Evaluate. Excel returns the results, which can be another expression or a value shown in italics (see Figure 3-11).
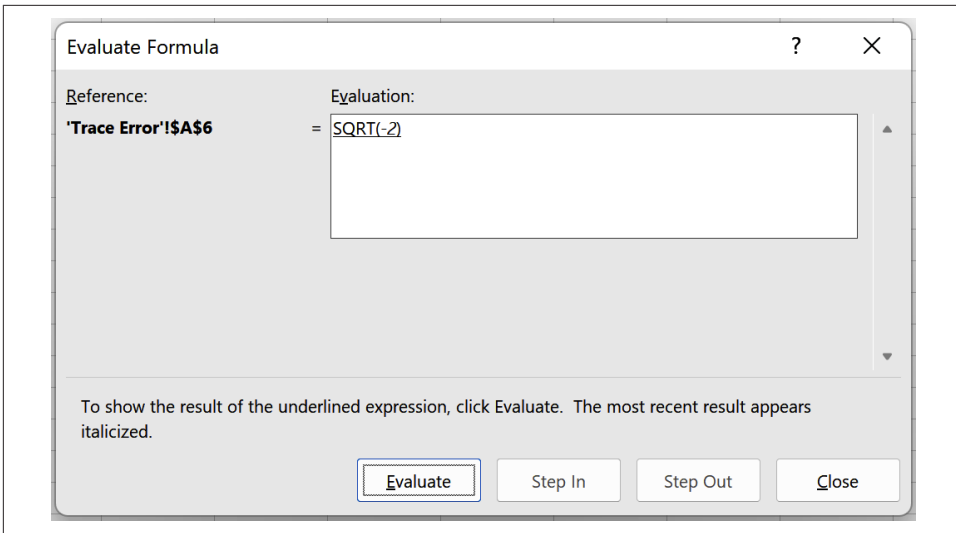
Figure 3-11. The Evaluate Formula dialog box italicizes any values it has evaluated

You can also click Step Into to evaluate an expression's calculations in a separate box (see Figure 3-12). To return to the original formula, click Step Out.
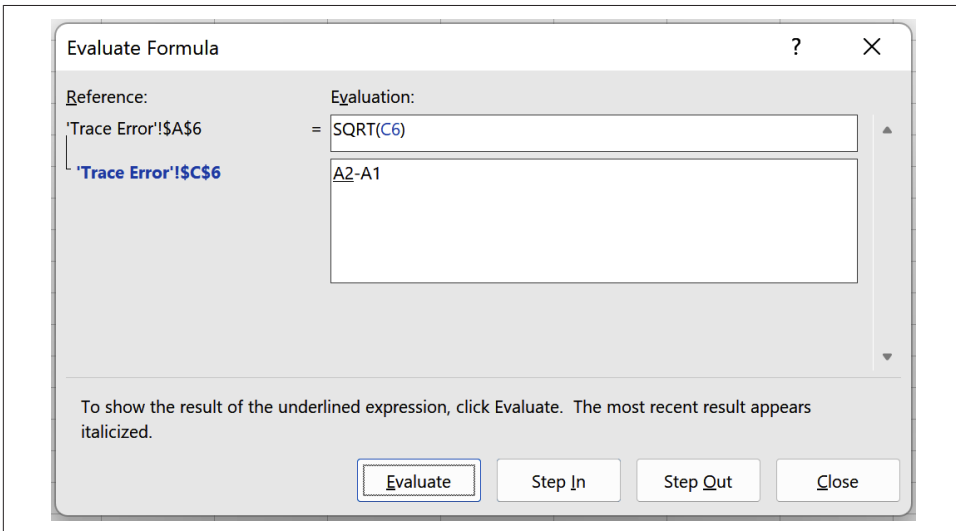
Figure 3-12. Stepping into an expression in the Evaluate Formula dialog box

Once you've finished stepping through the formula, click Restart to go through it again or Close to close the window.

Another way of checking a cell's formula is to select the part you want to evaluate in the cell or formula bar, then press F9 to replace that part with its calculated values. You can then press Esc to undo the change. If you're using Excel 365, selecting the part displays the value in a tool tip.

Discussion

The Evaluate Formula dialog box is a handy way of stepping through complex formulas and ensuring they behave as intended. It's particularly useful for resolving any errors or unexpected results.

3.17 Changing the Calculation Mode

Problem

You have a slow-running workbook and want to speed it up by controlling when Excel calculates formulas.

Solution

By default, when you change a value, formula, or name in a workbook, Excel automatically recalculates any formulas that depend on them. However, if there are many rows and/or formulas, this behavior can sometimes make the workbook sluggish and unresponsive.

You can change Excel's default calculation behavior by choosing Formulas ⇒ Calculation ⇒ Calculations Options and selecting one of the available options:

Automatic
> This default option automatically recalculates any dependent formulas when you change a value, formula, or name.

Automatic Except for Data Tables
> This switches off automatic recalculation for any data tables (see Recipes 14.1 and 14.3).

Manual
> This completely switches off automatic recalculation.

When you choose Manual mode, specify whether Excel should recalculate the workbook before saving it. To control this option in Excel for Windows, choose File ⇒ Options ⇒ Formulas and use the "Recalculate workbook before saving" check box in the Calculation Options section. If you're using Excel for Mac, you can find this check box by choosing Excel ⇒ Preferences ⇒ Calculation.

The calculation mode is a global setting, so changing it in one workbook changes it in all other open workbooks. Each time you save a workbook, Excel saves the current calculation mode.

When you open or create a workbook, Excel decides which calculation mode to use:

- If you open an existing workbook when no other workbooks are open, Excel uses the calculation mode with which it was last saved.

- If you create a new workbook from a template when there are no other open workbooks, Excel uses the calculation mode the template was saved with.

- If you open an existing workbook or create a new one while other workbooks are open, Excel uses the current calculation mode; this is usually the calculation mode of the first document you opened or created unless you changed it.

The order in which you open or create Excel files can affect the calculation mode, so if you have formulas that have stopped automatically recalculating when you want them to, choose Formulas ⇒ Calculation ⇒ Calculations Options and make sure the Automatic calculation option is selected.

If you've switched off automatic calculation, you can use one of the following options to force Excel to recalculate:

Formulas ⇒ Calculation ⇒ Calculate Now, or press F9
This recalculates any formulas that have changed in all open workbooks.

Formulas ⇒ Calculation ⇒ Calculate Sheet, or press Shift+F9
This recalculates changed formulas in the current worksheet.

Press Ctrl+Alt+F9
This recalculates all formulas—including ones that haven't changed—in all open workbooks.

Press Ctrl+Shift+Alt+F9
This checks dependent formulas and then recalculates all formulas in all open workbooks.

Discussion

This recipe gives an overview of Excel's calculation options. Note that the calculation mode is a general setting that applies to all open workbooks.

3.18 Setting Rounding Precision

Problem

You have floating-point rounding errors and want to resolve them.

Solution

When you format a number, Excel retains the underlying value. For example, if cell A1 contains the value 1.64736 and you format it to two decimal places, Excel displays 1.65 but stores the original value; calculating =A1*1000, for example, returns 1647.36 because the formula uses the underlying value.

In some situations, you may want to use the displayed value instead of the stored one in your formulas. There are two general ways of achieving this.

The first—and best—method is to update the formulas so they use one of Excel's rounding functions—for example, =ROUND(A1, 2)*1000 (see Recipe 4.9). This method is preferable since it keeps any underlying values intact.

The second method is to set the value's precision to match its format so Excel stores each value how it's displayed. You can do this in Excel for Windows by choosing File ⇒ Options ⇒ Advanced and checking the "Set precision as displayed" check box in the "When calculating this workbook" section; if you're using Excel for Mac, you can find this setting by choosing Excel ⇒ Preferences ⇒ Calculation. Once you apply this setting, Excel stores the value of A1 as 1.65 instead of 1.64736, so the formula =A1*1000 returns 1650.

> Setting the precision to match the value's format is usually a Very Bad Idea since it can easily lead to permanent data loss. Changing a cell's format, for example, can cause Excel to lose its original value, which can lead to calculation errors. You should, therefore, use this setting only as a last resort and ensure you back up your work before applying it.

Discussion

This recipe considers two ways of working with precision: using rounding functions and setting the precision as displayed. While updating formulas to use rounding functions is more work, it's generally a far better approach than using the "Set precision as displayed" setting, which is inherently ~~evil~~ risky and best avoided.

If your workbook contains unusual rounding errors that don't match manual calculations, check whether the "Set precision as displayed" option is switched on, because it can cause these types of errors.

3.19 Resolving Circular References

Problem

You have a circular reference you want to resolve but don't know how.

Solution

Suppose you want to populate cell A1 with today's date if the cell currently has a value of 0. You can theoretically achieve this by typing the formula `=IF(A1=0, TODAY(), A1)` in cell A1, but there's a problem: the cell contains a reference to its own cell, and by default, Excel can't evaluate its formula.

A formula that, directly or indirectly, refers to its own cell is called a *circular reference*. By default, Excel displays a warning message when it encounters one and may also draw Trace Precedents arrows (see Recipe 3.11).

You can resolve some circular references using *iterative calculations*: calculations that Excel repeatedly evaluates until it converges on a solution or runs out of iterations. To switch on iterative calculations:

1. Choose File ⇒ Options ⇒ Formulas in Excel for Windows or Excel ⇒ Preferences ⇒ Calculation in Excel for Mac.

2. Check the "Enable iterative calculation" check box.

3. Use the Maximum Iterations box to specify the maximum number of times you want calculations to be able to iterate: the higher the number, the more likely Excel is to converge on a solution, but the longer it may take.

4. Use the Maximum Change box to specify how accurate the result needs to be: the smaller the number, the more accurate the result, but the longer it may take. When you click OK, Excel tries to evaluate any circular references. It will either converge on a solution or reach the maximum number of iterations before it can do so.

You can determine whether a workbook includes circular references by switching off iterative calculations and choosing Formulas ⇒ Formula Auditing ⇒ Error Checking ⇒ Circular References. Doing so displays the first cell in the worksheet that contains a circular reference.

Discussion

This recipe shows you how to use iterative calculations to resolve circular references. Generally, circular references can converge to a solution, diverge, or alternate between two possible outcomes.

Iterative calculations are switched off by default since they can slow down your workbook. Generally, you should enable them only when resolving circular references.

Math and Engineering

Math operations are a crucial part of many spreadsheets, and Excel includes an extensive set of functions that saves you the trouble of performing cumbersome manual calculations.

This chapter guides you through Excel's math and engineering formulas, taking you beyond the +, -, *, /, and ^ operators. Recipes include ways of rounding numbers; working with sums, multiples, and divisors; using trigonometry; calculating permutations and combinations; solving problems with matrices; and working with complex numbers.

4.1 Generating Numbers

Problem

You want to generate numbers at random or in a sequence.

Solution

To generate a random decimal number greater than or equal to 0 and less than 1, use the RAND function by typing **=RAND()**. To return a random number greater than or equal to a and less than b, you can tweak this formula to become =RAND()*(b-a)+a. Typing **=RAND()*100**, for example, returns a random decimal number greater than or equal to 0 and less than 100.

To generate a random integer between a and b inclusive, use the RANDBETWEEN function. Generally, use the formula =RANDBETWEEN(a, b), so typing **=RANDBETWEEN(1, 10)** returns a random integer between 1 and 10, inclusive.

To generate a dynamic array (see Recipe 3.4) of random numbers, use the formula =RANDARRAY(*rows, columns, min, max, integer*), where *rows* (optional) is the number of rows (the default is 1), *columns* (optional) is the number of columns (the default is 1), *min* (optional) is the lowest number it's possible to return (the default is 0), *max* (optional) is the highest (the default is 1), and *integer* specifies whether you want to return integer or decimal numbers—use TRUE for integers and FALSE (the default) for decimals. So typing **=RANDARRAY(6)** returns six rows of random decimal numbers between 0 and 1 (inclusive), while typing **=RANDARRAY(5, , 1, 10, TRUE)** returns five rows of random integers between 1 and 10 (inclusive).

To generate a dynamic array of sequential numbers, you use the formula =SEQUENCE (*rows, columns, start, step*), where *rows* (optional) is the number of rows (the default is 1), *columns* (optional) is the number of columns (the default is 1), *start* (optional) is the first number in the sequence (the default is 1), and *step* (optional) is the number you want to increment subsequent numbers by (the default is 1). So typing **=SEQUENCE(6)** returns six rows containing the numbers 1 through 6, and typing **=SEQUENCE(5, , 0, 0.2)** returns five rows containing the numbers 0, 0.2, 0.4, 0.6, and 0.8.

The RANDARRAY and SEQUENCE functions are available only in Excel 2021 and Excel 365.

Discussion

This recipe offers several methods for generating numbers, depending on your version of Excel. You can use them in a wide variety of situations, such as generating text characters (see Recipes 5.3 and 5.4).

See Also

See Recipe 9.9 for other ways of generating random numbers.

4.2 Converting Text or a Boolean to a Number

Problem

You have a value stored as text or a TRUE/FALSE value and want to convert it to a number.

Solution

If a numeric value is stored as text, you can convert it to a number using =*text**1, =*text*+0, or =–*text*. If cell A1 contains the formula =TEXT(123, "00000"), which converts the number 123 to the text *00123* (see Recipe 5.17), typing **=A1*1**, **=A1+0**, or **=–A1** converts the text back to a number, returning 123.

You can convert a Boolean TRUE/FALSE value to a number using a similar technique. So if A1:A5 contains TRUE/FALSE values, you can return their numeric values using the formulas =A1:A5*1, =A1:A5+0, or =–A1:A5. See Recipe 7.6 for an example of this technique.

Discussion

Math functions such as SUM include only numeric values in their calculations, which can lead to unexpected results. If cell A1 contains the formula ="123", for example, which evaluates to the text *123* instead of a number, the formula =SUM(A1) returns 0. This recipe offers a convenient solution to this type of problem.

See Also

You can use the ISNUMBER and ISTEXT functions to check whether a value is stored as a number or text; see Recipe 7.7.

4.3 Getting a Number's Sign and Absolute Value

Problem

You have a number and want to know whether it is positive or negative and its absolute value.

Solution

The SIGN function lets you determine a number's sign using the formula =SIGN(*number*). The function returns 1 if *number* is positive, –1 if it's negative, and 0 if it's zero, so typing **=SIGN(-10)**, for example, returns –1.

To find a number's absolute value—the number without its sign—use the formula =ABS(*number*). For example, if cell A1 contains the number –10, typing **=ABS(A1)** returns 10.

Discussion

This recipe offers a quick way of determining whether a number is positive, negative, or zero and getting its absolute value. They're used, for example, in Recipe 4.10.

4.4 Counting, Summing, and Averaging Cell Values

Problem

You have a range of cells that contain numbers and want to count how many there are and calculate their sum and average.

Solution

Suppose A2:A5 lists the numbers 1 to 4, B2:B5 lists the numbers 11 to 14, C2:C5 lists the numbers 21 to 24, and you want to count how many cells contain numbers, and calculate their sum and average.

To count the cells that contain numbers, you can use the COUNT function. Generally, you use the formula =COUNT(*range*), so typing **=COUNT(A2:C5)** returns 12. You can also pass multiple ranges to COUNT; typing **=COUNT(A2:A5, C2:C5)**, for example, counts the numbers in A2:A5 and C2:C5, returning 8 (see Figure 4-1).

	A	B	C	D	E	F	G
1		Values			Formula	Result		
2	1	11	21	=COUNT(A2:C5)	12		
3	2	12	22	=COUNT(A2:A5,C2:C5)	8		
4	3	13	23	=SUM(A2:A5)	10		
5	4	14	24	=AVERAGE(A2:A5)	2.5		

Figure 4-1. Using COUNT, SUM, and AVERAGE

> You can also use the COUNTA function to count the cells that contain any value (not just numbers) and the COUNTBLANK function to count how many cells are empty.

The SUM and AVERAGE functions work the same way as COUNT except that they return the sum and average of numeric values, so typing **=SUM(A2:A5)** returns 10, and typing **=AVERAGE(A2:A5)** returns 2.5.

Discussion

This recipe uses core math functions—COUNT, SUM, and AVERAGE—to count the numbers in one or more ranges and calculate their sum and average.

Note that if you pass two intersecting ranges to the COUNT, SUM, or AVERAGE functions, they include any numbers in the intersection twice—once for each range. For example, if the range A2:C5 contains numbers, the formula =COUNT(A2:A5, A5:C5) returns 7 instead of 6 because it includes A5 twice. To work around this problem, you can use the formula =COUNT(*range1, range2*)-COUNT(*range1 range2*), where the second COUNT uses the Space operator (see Recipe 3.1) to count the cells in the intersection.

See Also

See Recipe 8.4 for more information on calculating different types of averages.

4.5 Using Criteria to Count, Sum, and Average

Problem

You have a range of cells containing numbers and want to count how many match specified criteria and calculate their sum and average.

Solution

Suppose A2:A8 lists food items, B2:B8 lists their amounts, and you want to find the count, sum, and average of cells that meet specified conditions.

If there's a single condition, you can use the COUNTIF, SUMIF, and AVERAGEIF functions as follows (see Figure 4-2):

COUNTIF

> COUNTIF counts the values that meet a single condition. Generally, you use the formula =COUNTIF(*range, condition*), where *range* is the range of cells you want to apply the *condition* to, so typing **=COUNTIF(B2:B8, ">10")** counts how many amounts are greater than 10 and typing **=COUNTIF(A2:A8, "Pizza")** counts the number of Pizza items.

SUMIF

> SUMIF works similarly to COUNTIF except it sums the values that meet the condition. Generally, you use the formula =SUMIF(*range, condition, sum_range*), where *range* is the range of cells you want to apply the *condition* to, and *sum_range* (optional) is the range of cells you want to sum (if it's different from

range), so typing **=SUMIF(A2:A8, "Pizza", B2:B8)** returns the sum of any Pizza amounts.

AVERAGEIF

AVERAGEIF works the same as SUMIF, except it calculates the average, so typing **=AVERAGEIF(A2:A8, "Pizza", B2:B8)** returns the average amount of any Pizza items.

	A	B	C	D	E
1	Item	Amount	Formula	Result	
2	Pizza	15	=COUNTIF(B2:B8,">10")	3	
3	Salad	5	=COUNTIF(A2:A8,"Pizza")	3	
4	Samosa	6	=SUMIF(A2:A8,"Pizza",B2:B8)	39	
5	Pizza	9	=AVERAGEIF(A2:A8,"Pizza",B2:B8)	13	
6	Burger	12	=COUNTIFS(A2:A8,"Pizza",B2:B8,">10")	2	
7	Burger	9	=SUMIFS(B2:B8,A2:A8,"Pizza",B2:B8,">10")	30	
8	Pizza	15	=AVERAGEIFS(B2:B8,A2:A8,"Pizza",B2:B8,">10")	15	

Figure 4-2. Using criteria to count items and calculate the sum and average

If there are multiple conditions, you can use the COUNTIFS, SUMIFS, and AVERAGEIFS functions as follows (see Figure 4-2):

COUNTIFS

COUNTIFS counts the number of cells meeting multiple conditions. Generally, use =COUNTIFS(*condition_range1, condition1, condition_range2, condition2, …*), where you want to apply *condition1* to *condition_range1*, *condition2* to *condition_range2*, and so on. To count the number of Pizza items with an amount greater than 10, you'd type **=COUNTIFS(A2:A8, "Pizza", B2:B8, ">10")**.

SUMIFS

SUMIFS works similarly to COUNTIFS, except it calculates the sum. Generally, use =SUMIFS(*sum_range, condition_range1, condition1, condition_range2, condition2, …*), where *sum_range* is the range of cells you want to sum, and you want to apply *condition1* to *condition_range1*, *condition2* to *condition_range2*, and so on. For example, **=SUMIFS(B2:B8, A2:A8, "Pizza", B2:B8, ">10")** returns the total amount of any Pizza items with an amount greater than 10.

AVERAGEIFS

AVERAGEIFS works the same as SUMIFS, except it calculates the average, so typing **=AVERAGEIFS(B2:B8, A2:A8, "Pizza", B2:B8, ">10")** calculates the average amount of any Pizza items where the amount is greater than 10.

Discussion

This recipe expands on Recipe 4.4 to calculate the count, sum, and average subject to one or more conditions. An alternative approach would be to add these values to a table (see Recipe 2.18) and apply a filter, or use the FILTER function (see Recipe 7.3).

4.6 Adding and Subtracting Squares of Values

Problem

You have a range of numbers and want to perform calculations using the sum of—or difference between—their squared values.

Solution

Suppose cells A2:A6 list the x values 1 to 5, B2:B6 lists the corresponding y values 11 to 15, and you want to perform calculations using the square of their values. Excel includes several functions you can use as follows (see Figure 4-3):

SUMSQ

SUMSQ calculates Σx^2—the sum of the squares of the x values. You generally use the formula =SUMSQ(*x_values*), so typing **=SUMSQ(A2:A6)** returns 55. You can also pass multiple ranges to the SUMSQ function, so typing **=SUMSQ(A2:A6, B3:B5)**, for example, sums the squares of the values in A2:A6 and B3:B5, returning 564.

SUMX2PY2

SUMX2PY2 calculates $\Sigma(x^2+y^2)$—the sum of the squares of the x and y values. This takes the form =SUMX2PY2(*x_values, y_values*), so typing **=SUMX2PY2(A2:A6, B2:B6)** returns 910.

SUMX2MY2

SUMX2MY2 calculates $\Sigma(x^2-y^2)$—the sum of the differences of the squares between the x and y values. Generally, you use =SUMX2MY2(*x_values, y_values*), so typing **=SUMX2MY2(A2:A6, B2:B6)** returns –800.

SUMXMY2

SUMXMY2 calculates $\Sigma(x-y)^2$—the sum of the squares of differences between the x and y values. Generally, you use =SUMX2MY2(*x_values, y_values*), so typing **=SUMXMY2(A2:A6, B2:B6)** returns 500.

	A	B	C	D	E	F
1	x-values	y-values	Formula	Result		
2	1	11	=SUMSQ(A2:A6)	55		
3	2	12	=SUMSQ(A2:A6,B3:B5)	564		
4	3	13	=SUMX2PY2(A2:A6,B2:B6)	910		
5	4	14	=SUMX2MY2(A2:A6,B2:B6)	-800		
6	5	15	=SUMXMY2(A2:A6,B2:B6)	500		

Figure 4-3. Adding and subtracting sums of squares

> The SUMX2PY2, SUMX2MY2, and SUMXMY2 functions expect the same number of *x* values as *y* values. If this isn't the case, the functions return the #N/A error value.

Discussion

This recipe offers a flexible way of performing different calculations using square values. Note that if you have two ranges that contain the same number of values, the formulas =SUMSQ(*range1, range2*) and =SUMX2PY2(*range1, range2*) return the same value.

4.7 Using Multiplication and Multiples

Problem

You have a range of numbers and want to multiply them, calculate the sum of their products for each row, and find the least or lowest common multiple.

Solution

Suppose cells A2:A4 list the numbers 1, 2, and 6, and B2:B4 list the numbers 4, 8, and 16. You want to find their product, the sum of the products for each row, and the least common multiple.

To multiply all the cells containing numbers, use the PRODUCT function. Generally, you use the formula =PRODUCT(*range*), so typing **=PRODUCT(A2:B4)** returns 6144 (see Figure 4-4).

To calculate the sum of the products of two or more corresponding ranges—for example, two columns—use the SUMPRODUCT function. This function takes the form =SUMPRODUCT(*range1, range1*), where *range1* and *range2* are the two ranges whose values you want to multiply and then add. To multiply the values for each row and

add the results together, for example, you type **=SUMPRODUCT(A2:A4, B2:B4)**, which returns 116.

	A	B	C	D	E	F
1	Values		Formula	Result		
2	1	4	=PRODUCT(A2:B4)	6144		
3	2	8	=SUMPRODUCT(A2:A4,B2:B4)	116		
4	6	16	=LCM(A2:B4)	48		

Figure 4-4. Using PRODUCT, SUMPRODUCT, and LCM

> You can also use the SUMPRODUCT function to perform other operations before summing the values, such as division or subtraction. To do so, replace the comma with the operator you want to use (*, /, +, or -)—for example, =SUMPRODUCT(A2:A4/B2:B4).

To find the least common multiple of a set of integers—the smallest positive integer that's a multiple of each number—use the LCM function. You generally use the formula =LCM(*range*), so typing **=LCM(A2:B4)** returns 48.

Discussion

This recipe offers several functions you can use when multiplying numbers. Note that the SUMPRODUCT function is often used in statistics to calculate weighted averages (see Recipe 8.4 for more information).

4.8 Finding Quotients, Remainders, and Divisors

Problem

You have two numbers and want to find the integer and remainder parts when you divide one number by the other as well as the greatest common divisor for the two numbers.

Solution

Suppose A2 contains the number 26 and A3 contains 6. You want to divide A2 by A3 and split the result into integer and remainder parts.

To find the integer part, use the QUOTIENT function. This takes the form =QUO TIENT(*number, divisor*), where you want to divide the *number* argument by the *divisor*. To find the integer part when you divide A2 by A3, type **=QUOTIENT(A2, A3)**, which returns 4 when A2 is 26 and A3 is 6.

You can also get this result by typing **=TRUNC(A2/A3)**. See Recipe 4.9 for more information about this function.

To find the remainder, use the MOD function. You generally use the formula =MOD(*number, divisor*), where you want to divide the *number* argument by the *divisor*, so to find the remainder when you divide A2 by A3, you type **=MOD(A2,A3)**, which returns 2 when A2 is 26 and A3 is 6.

You can also use the MOD function to return the fractional part of a number using the formula =MOD(*number, 1*).

To find the greatest common divisor—the largest integer that divides the numbers with no remainder—use the GCD function. Generally, you use the formula =GCD(*numbers*), so to find the greatest common divisor for the numbers in cells A2:A3, you type **=GCD(A2:A3)**, which returns 2 when A2 is 26 and A3 is 6.

Discussion

This recipe shows you how to find quotients, remainders, and the greatest common denominator. It also introduces the MOD function, which has many applications—for example, Recipe 6.8.

4.9 Rounding to Decimal Places and Integers

Problem

You have a number and want to round it to a specified number of decimal places or an integer.

Solution

Excel includes various rounding functions you can use, depending on whether you want to round positive and negative numbers up, down, or to the nearest number. Figure 4-5 summarizes the behavior of these functions.

To round to the nearest integer or decimal number, use the ROUND function. You generally use the formula =ROUND(*number, num_digits*), where *num_digits* specifies the number of decimal places you want to round the *number* argument to. If *num_digits* is positive, it specifies the number of decimal places; if it's zero, the function rounds

to the nearest integer; and if it's negative, the function rounds to the left of the decimal point. So typing **=ROUND(5.6, 0)** rounds 5.6 to the nearest integer (returning 6), and typing **=ROUND(-5.673,2)** rounds –5.673 to 2 decimal places (returning –5.67).

		Negative numbers		
		Nearest number	Away from zero	Toward zero
Positive numbers	Nearest number	ROUND		
	Away from zero		ROUNDUP	CEILING.MATH
	Toward zero		FLOOR.MATH INT	ROUNDDOWN TRUNC

Figure 4-5. How different functions round positive and negative numbers

To round positive and negative numbers away from zero, use the ROUNDUP function. This function works similarly to ROUND, except it rounds positive numbers up and negative numbers away from zero; behind the scenes, it rounds up the number's absolute value, increasing its magnitude, and then applies its sign. So typing **=ROUNDUP(5.321, 2)** rounds 5.321 up to 5.33, and typing **=ROUNDUP(-5.3, 0)** rounds –5.3 to –6.

To round positive numbers away from zero and negative numbers toward it, use the CEILING.MATH function. You generally use the formula =CEILING.MATH(*number, multiple*), where *number* is the number you want to round and *multiple* (optional) specifies a multiple you want to round numbers to—it defaults to 1 for positive numbers and –1 for negative. For example, typing **=CEILING.MATH(5.321, 0.1)** rounds 5.321 up to 5.4, and **=CEILING.MATH(-5.3)** rounds –5.3 to –5.

To round positive and negative numbers toward zero, use the ROUNDDOWN function. This function works similarly to the ROUND and ROUNDUP functions, except it rounds positive numbers down and negative numbers toward zero; behind the scenes, it rounds down the number's absolute value, decreasing its magnitude, then applies its sign. So typing **=ROUNDDOWN(5.32, 1)** rounds 5.32 down to 5.3, and typing **=ROUNDDOWN(-5.3, 0)** rounds –5.3 to –5.

> If you want to round a number to an integer, you can use the TRUNC function instead of ROUNDDOWN. TRUNC works the same way as ROUNDDOWN and uses the same arguments, but its *num_digits* argument is optional and defaults to zero. So typing **=TRUNC(5.3)** rounds 5.3 down to 5, and typing **=TRUNC(-5.3)** rounds –5.3 to –5.

To round positive numbers toward zero and negative numbers away from it, use the FLOOR.MATH function, which works similarly to CEILING.MATH. You generally use the

formula =FLOOR.MATH(*number, multiple*), where *number* is the number you want to round and *multiple* (optional) specifies a multiple you want to round numbers to—it defaults to 1 for positive numbers and –1 for negative. So typing **=FLOOR.MATH(5.321, 0.1)** rounds 5.321 down to 5.3, and typing **=FLOOR.MATH (-5.3)** rounds –5.3 to –6.

> If you want to round a number to an integer, you can use the INT function instead of FLOOR.MATH. This function accepts a single argument—the number you want to round to an integer—so typing **=INT(5.3)** rounds 5.3 down to 5, and typing **=INT(-5.3)** rounds –5.3 to –6.

Discussion

As you can see, Excel includes several functions that round numbers in slightly different ways; while the ROUND function rounds values to the nearest number, all the others round positive or negative numbers toward or away from zero. See Figure 4-5 for a comparison of function rounding behavior.

4.10 Rounding to Significant Figures and Multiples

Problem

You have a number and want to round it to a multiple of another number or a specified number of significant figures.

Solution

The ROUND, ROUNDUP, ROUNDDOWN, and TRUNC functions (see Recipe 4.9) let you round a number to a multiple that's a power of 10. Typing **=ROUND(234.5, -1)**, for example, rounds 234.5 to the nearest multiple of 10, returning 230, and typing **=ROUND(234.5, -2)** rounds it to the nearest multiple of 100, returning 200.

You can also use these functions to round a number to a specified number of significant figures. You generally use the formula =ROUND(*number, significant_figures*-LEN(TRUNC(ABS(*number*)))), where you want to round *number* to the specified number of *significant_figures*. So if cell A1 contains the number 23456.789, you'd round it to two significant figures by typing **=ROUND(A1, 2-LEN(TRUNC(ABS(A1))))**, returning 23000.

To round a number to a multiple that's not a power of 10, use the MROUND, CEIL ING.MATH, and FLOOR.MATH functions.

The `MROUND` function rounds a number to the nearest multiple and takes the form =MROUND(*number, multiple*). To round 23456 to the nearest multiple of 5, you'd type **=MROUND(23456, 5)**, which returns 23455, and to round –23456 to the nearest multiple of –5, you'd type **=MROUND(-23456, -5)**, which returns –23455.

> The `MROUND` function's *number* and *multiple* arguments must have the same sign, or it returns a #NUM! error value. If you need to round positive and negative numbers to the nearest multiple, use the formula =MROUND(*number, multiple**SIGN(*number*)) instead.

To round a positive number up to the next nearest multiple, use the `CEILING.MATH` function in the formula =CEILING.MATH(*number, multiple, mode*), where *mode* (optional) specifies how you want to round *number* if it's negative; omit *mode* it to round a negative *number* toward zero, and use 1 to round it away from zero. So typing **=CEILING.MATH(-234, 5)** rounds –234 to the nearest multiple of five toward zero and returns –230, and typing **=CEILING.MATH(-234, 5, 1)** rounds it to the nearest multiple away from zero and returns –235.

To round a positive number down to the next nearest multiple, use the `FLOOR.MATH` function in the formula =FLOOR.MATH(*number, multiple, mode*); in this case, you omit the optional *mode* argument to round a negative *number* away from zero and use –1 to round it toward zero. So typing **=FLOOR.MATH(-234, 5)** rounds –234 to the nearest multiple of five away from zero and returns –235, and typing **=FLOOR.MATH(-234, 5, -1)** rounds it to the nearest multiple toward zero and returns –230.

Finally, to round a number to the nearest even or odd number away from zero, use the `EVEN` and `ODD` functions. Typing **=EVEN(2.3)**, for example, returns 4, while typing **=ODD(2.3)** returns 3.

Discussion

This recipe builds on Recipe 4.9 to round a number to a specified number of significant figures or a multiple of another number.

4.11 Using Powers, Exponents, Square Roots, and Logarithms

Problem

You have a number and want to raise it to a power or find its square root.

Solution

You can use the ^ operator or the POWER function to raise a number to a power. Typing **=2^4** or **=POWER(2,4)**, for example, calculates 2^4 and returns 16.

You can also use the POWER function with ranges in the formula =POWER(*numbers, powers*). If A2:A5 contains the numbers 1 to 4, for example, typing **=POWER(A2:A5, 2)** in Excel 2021 or Excel 365 returns a dynamic array (see Recipe 3.4) containing each number squared, and typing **=POWER(A2:A5, A2:A5)** calculates x^x for each value: 1^1, 2^2, and so on (see Figure 4-6).

	A	B	C	D	E
1	Values	=POWER(A2:A5, 2)	=POWER(A2:A5, A2:A5)		
2	1	1	1		
3	2	4	4		
4	3	9	27		
5	4	16	256		

Figure 4-6. Using the POWER function

To find the square root of a number, use the SQRT function. Typing **=SQRT(4)**, for example, returns 2, and typing **=SQRT(A2:A5)** in Excel 2021 or Excel 365 returns a dynamic array containing the square root of each number in A2:A5.

To raise e to the power of a number, use the formula =EXP(*number*). So typing **=EXP(5)** calculates e^5, returning 148.4132.

The inverse of EXP is the LN function, which returns a number's natural logarithm. You generally use the formula =LN(*number*), which returns the power you need to raise e to so that it equals *number*. So typing **=LN(3)** returns 1.0986—since $e^{1.0986}$ is approximately 3—and typing **=LN(EXP(5))** returns 5.

To find the base-10 logarithm of a number, use the LOG10 function. In general, use the formula =LOG10(*number*), which returns the power you need to raise 10 to so that it equals *number*. So typing **=LOG10(1000)** returns 3 because 10^3 equals 1000.

To find the logarithm of a number for a base you specify, use the LOG function. This function takes the form =LOG(*number,base*) and returns the power you need to raise *base* to so that it equals *number*. So typing **=LOG(16,2)** returns 4 because 2^4 equals 16.

Discussion

This recipe details several functions you can use to work with powers, exponents, and logarithms, including finding a number's square root.

4.12 Summing a Power Series

Problem

You have a power series (for example, $5x^3+4x^2+3x+2$) and want to know its result for a value of x.

Solution

The SERIESSUM function returns the sum of a power series. It takes the form =SERIESSUM(x, n, m, a), which represents the equation $a_1x^n + a_2x^{(n+m)} + a_3x^{(n+2m)} + a_4x^{(n+3m)}$, and so on. The x argument is the value of x, n is the first power of x in the series, m is the step size used to increase n by for each successive power of x, and a is an array of the coefficients used to multiply x. You can calculate $5x^3 + 4x^2 + 3x + 2$, where x is 2, for example, by typing the formula **=SERIESSUM(2, 3, -1, {5,4,3,2})**, which returns 64.

Discussion

This recipe offers a way of calculating the sum of a power series. Instead of typing the formula using the * and ^ operators, you can use the SERIESSUM function to get the same result.

4.13 Using Factorials, Permutations, and Combinations

Problem

You want to find a number's factorial, or the possible number of combinations and permutations when you select r items from n.

Solution

To find a number n's *factorial*—the number of ways of arranging n items, or the product of all positive integers—use the FACT function. You generally use the formula =FACT(n), which returns $n!$—the product of all positive integers from 1 to n— so typing **=FACT(5)** returns 120.

> Excel also has a FACTDOUBLE function, which returns a number's double factorial, or $n!!$—the product of all the integers from 1 to n that have the same parity (odd or even) as n.

To find the number of ways to arrange *n* items, where some are duplicates, use the MULTINOMIAL function. Suppose, for example, that you want to find the number of ways of arranging the letters in the word *Mississippi*, which has only four unique letters. Since there is one letter M, two Ps, and four each of I and S, you can calculate the number of arrangements by typing **=MULTINOMIAL(1, 2, 4, 4)**, which returns 34,650. Generally, you use the formula =MULTINOMIAL(*item_counts*), where *item_counts* specifies how many there are of each item.

To find the number of *permutations* when you select *r* items from *n*—the number of ways of arranging the items when the order matters—and can select each item only once, use the PERMUT function. Generally, you use the formula =PERMUT(*n, r*), so if, for example, you wanted to know how many portfolios you could create by choosing 2 stocks from 20 where you want one stock to have 60% of the allocation and the other stock to have the remaining 40%, you'd type **=PERMUT(20, 2)**, returning 380.

To find the number of *combinations*—where the order doesn't matter—when you select *r* items from *n* and can select each item only once, use the COMBIN function. This function takes the form =COMBIN(*n, r*), so if, for example, you wanted to know how many portfolios you could create by choosing 2 equally weighted stocks from 20, you'd type **=COMBIN(20, 2)**, returning 190.

> The PERMUT and COMBIN functions assume you can select each item only once. If you can select each item more than once, use the PER MUTATIONA and COMBINA functions instead. To find the number of possible five-digit passwords that use numbers from zero to nine, you'd type **=PERMUTATIONA(10, 5)**, returning 100000.

Discussion

This recipe offers an overview of Excel's combinatorics functions, which involve counting the number of possible arrangements. These functions are helpful in areas such as probability and form the basis of the binomial distribution (see Recipe 8.14).

4.14 Using Trigonometry

Problem

You have a trigonometry problem and want to know what functions are available.

Solution

Excel includes the following trigonometry functions:

PI

This returns the constant π (pi) to 15 digits.

RADIANS *and* DEGREES

RADIANS converts an angle from degrees to radians, and DEGREES converts an angle from radians to degrees.

SIN, COS, *and* TAN

These return the sine, cosine, and tangent of an angle specified in radians. To calculate the sine of π/2 radians, you type **=SIN(PI()/2)**, which returns 1.

CSC, SEC, *and* COT

These return the cosecant, secant, and cotangent of an angle specified in radians. To calculate the secant of π/3 radians, type **=SEC(PI()/3)**, which returns 2.

ASIN, ACOS, ATAN, *and* ATAN2

The ASIN, ACOS, and ATAN functions calculate a number's arcsine, arccosine, and arctangent and return an angle in radians; the ATAN2 function returns the arctangent of specified *x* and *y* coordinates.

SINH, COSH, TANH, CSCH, SECH, COTH, ASINH, ACOSH, *and* ATANH

These functions return the hyperbolic sine, cosine, tangent, and so on.

Discussion

As you can see, Excel offers a wealth of functions you can use to solve trigonometry problems. Most of them accept or return an angle in radians, so if needed, use the RADIANS and DEGREES functions to convert between radians and degrees.

4.15 Working with Matrices

Problem

You want to solve a problem in Excel using matrices but don't know how.

Solution

Suppose three customers decide to buy different amounts of two products. You know each product's unit price and weight and want to calculate the total price and weight for each customer. B2:C4 lists each customer's quantities, and F2:G3 lists the unit price and weight.

You can solve this problem by treating the quantities bought and the unit prices and weights as two matrices and multiplying them. You do this using the MMULT function, which multiplies two matrices and returns a dynamic array. In this example, typing

=MMULT(B2:C4, F2:G3) returns a dynamic array that calculates each customer's total price and weight (see Figure 4-7).

	A	B	C	D	E	F	G	H	I
		Plates	Mugs						
1	Customer	quantity	quantity		Product	Price	Weight		
2	Amy	10	6		Plates	4	1.5		
3	Bill	4	8		Mugs	3	0.7		
4	Clara	6	4						
5									
			Weight						
6	Customer	Price total	total						
7	Amy	=MMULT(B2:C4,F2:G3)							
8	Bill	40	11.6						
9	Clara	36	11.8						

Figure 4-7. Using the MMULT function to multiply two matrices

Suppose you know each product's unit price, weight, and totals for each customer. In this case, you can calculate the quantities bought by each customer by multiplying the matrix of customer totals by the inverse of the unit price and weight matrix. You find the inverse of a matrix (if one exists) using the MINVERSE function, so if B2:C4 lists the customer totals and F2:G3 lists the unit price and weight, typing **=MMULT(B2:C4, MIN VERSE(F2:G3))** returns a matrix of the customer purchases (see Figure 4-8).

	A	B	C	D	E	F	G	H	I
			Weight						
1	Customer	Price total	total		Product	Price	Weight		
2	Amy	58	19.2		Plates	4	1.5		
3	Bill	40	11.6		Mugs	3	0.7		
4	Clara	36	11.8						
5									
		Plates	Mugs						
6	Customer	quantity	quantity						
7	Amy	=MMULT(B2:C4, MINVERSE(F2:G3))							
8	Bill	4	8						
9	Clara	6	4						

Figure 4-8. Using the MMULT and MINVERSE functions

Other matrix functions include the following:

MUNIT

This function returns the unit matrix for a specified dimension. For example, the formula =MUNIT(3) returns a unit matrix with three rows and columns.

MDETERM
: This returns the determinant of a matrix.

TRANSPOSE
: This transposes a range of cells that converts the rows to columns and the columns to rows (see Recipe 7.4).

> `MMULT`, `MINVERSE`, `MUNIT`, and `TRANSPOSE` return dynamic arrays (see Recipe 3.4). If you're using a version of Excel earlier than Excel 2021 or Excel 365, select the entire output range before entering the formula, and then press Ctrl+Shift+Enter/Return.

Discussion

This recipe offers a handy overview of using Excel's matrix functions. You can use them, for example, to solve systems of linear equations, as in the examples used in this recipe.

4.16 Converting Between Number Systems

Problem

You have a number and want to convert it to another system (for example, from decimal to binary, octal, or hexadecimal).

Solution

Use the `DEC2BIN`, `DEC2OCT`, `DEC2HEX`, `BIN2DEC`, `BIN2OCT`, `BIN2HEX`, `OCT2DEC`, `OCT2BIN`, `OCT2HEX`, `HEX2DEC`, `HEX2BIN`, and `HEX2OCT` functions to convert numbers from one system to another. To convert the decimal number 9 to binary, for example, you type **=DEC2BIN(9)**, which returns 1001. Similarly, to convert the hexadecimal number A2 to decimal, you type **=HEX2DEC("A2")**, which returns 162.

Discussion

This recipe shows converting numbers between decimal, binary, octal, and hexadecimal numbers; this is useful if, for example, you want to perform bitwise operations on a binary bit field (see Recipe 4.17).

4.17 Performing Bitwise Operations

Problem

You have a binary bit field and want to perform bitwise operations.

Solution

Suppose you have a set of decimal readings from a salt truck's controller system, where each number represents the current state of its four sensors as a binary bit field. The first bit corresponds to the pressure valve sensor, the second to the spreader, the third to the salt feed, and the fourth to the beacon. Each sensor's bit is set to 1 if it's on and 0 if it's off, so if the pressure valve and spreader are on but the salt feed and beacon are off, the controller records the value 0011 in binary, or the decimal number 3.

You can apply the BITAND function to the controller's readings to determine whether a sensor is on or off. This function takes the form =BITAND(*number1, number2*), where *number1* and *number2* are decimals, and returns the bitwise AND of its arguments as a decimal. If the controller records the value 6, for example, you can find out whether the pressure valve—the first sensor—is on by typing **=BITAND(6, 1)>0**, which returns FALSE. Similarly, you can find out whether the spreader—the second sensor—is on by typing **=BITAND(6, BIN2DEC(10))>0**, which returns TRUE. Finally, if cells A2:A6 list a set of readings from the controller and you're using Excel 2021 or Excel 365, you can return a dynamic array (see Recipe 3.4) showing the state of each sensor by typing **=BITAND(A2:A6, BIN2DEC({1,10,100,1000}))>0** (see Figure 4-9).

	A	B	C	D	E	F	G
1	Controller value	Pressure valve	Spinner	Salt feed	Beacon		
2	6	=BITAND(A2:A6,BIN2DEC({1,10,100,1000}))>0					
3	1	TRUE	FALSE	FALSE	FALSE		
4	4	FALSE	FALSE	TRUE	FALSE		
5	11	TRUE	TRUE	FALSE	TRUE		
6	2	FALSE	TRUE	FALSE	FALSE		

Figure 4-9. Using the BITAND function with a binary bit field

Other bitwise functions include the following:

BITOR *and* BITXOR
> These return the bitwise OR and XOR (eXclusive-OR) of their arguments.

BITLSHIFT *and* BITRSHIFT
> These return a number shifted left or right by a specified number of bits.

Discussion

Bitwise operators let you perform operations at a bit level and work with the binary representation of a number instead of the number's value. They're helpful when working with structures such as bit fields, as in the example used by this recipe.

See Also

For more information about converting between decimal and binary number systems, see Recipe 4.16.

4.18 Working with Complex Numbers

Problem

You want to work with complex numbers in Excel and want to know what functions are available.

Solution

A *complex number* is one in the form $a + b$i, where a and b are real numbers, and i is the square root of –1.

To enter a complex number into a cell, you can type it directly or pass real and imaginary coefficients to the COMPLEX function. To enter the imaginary number 6+2i in cell A1, for example, you can either type **6+2i** or the formula **=COMPLEX(6, 2)**.

> Excel also supports using j for complex numbers instead of i. Type either **6+2j** or the formula **=COMPLEX(6, 2, "j")**.

To retrieve a complex number's real and imaginary coefficients, use the IMREAL and IMAGINARY functions. For example, if cell A1 contains the complex number 6 + 2i, typing **=IMREAL(A1)** returns 6, and typing **=IMAGINARY(A1)** returns 2.

To find a complex number's conjugate, argument, and absolute value, use the IMCONJUGATE, IMARGUMENT, and IMABS functions. So if cell A1 contains 6+2i, for example, typing **=IMCONJUGATE(A1)** returns 6-2i.

To perform arithmetic operations with complex numbers, use the IMSUM function to sum them, IMSUB to subtract one from another, IMPRODUCT to calculate the product, and IMDIV to divide one by another. For example, typing **=IMSUM("6+2i", "5-4i")** returns 11-2i, and typing **=IMPRODUCT("6+2i", "5-4i")** returns 38-14i.

Ensure you consistently use i or j to denote a complex number, not both. Typing `=IMSUM("6+2j", "5-4j")`, for example, returns 11-2j, but typing `=IMSUM("6+2i", "5-4j")` returns the `#VALUE!` error value.

Excel also includes complex number versions of many other functions, each prefixed with `IM`. For example, use `IMSQRT` to find the square root of a complex number, `IMLN` to find the natural logarithm, `IMSIN` to find the sine, and so on.

Discussion

This recipe offers an overview of Excel's most important complex number functions. These can be used, for example, with the output from the Analysis ToolPak's Fourier Analysis tool (see Recipe 9.19).

Text Manipulation

Excel isn't a word processor, but in many situations you'll need to manipulate text strings and transform data to fit a particular model.

This chapter shows how to use formulas to perform common text manipulation exercises, including joining text, extracting numeric codes, removing extra spaces and characters, and applying text formats to currency, number, and date/time values.

You can also perform many of the operations in this chapter using Excel's Power Query tool. See Chapter 15 to find out more.

5.1 Concatenating Text

Problem

You have two or more text strings and want to join them.

Solution

Suppose cell A1 contains the text *John*, B1 contains *Doe*, and you want to combine the two text strings.

One method uses the & operator, which concatenates two text strings. Typing the formula **=A1&B1** in cell C1, for example, adds the text in B1 to the end of that in A1 and returns the text *JohnDoe*. If you want to insert a space between the two text strings and return *John Doe* instead, you type the formula **=A1&" "&B1**; this adds a space to the end of the text in A1 before adding the text in B1.

You must surround any static text, such as spaces, with double quotes and omit them for anything you want Excel to evaluate, such as cell references. The formula =A1&B1, for example, joins the contents of cells A1 and B1, while the formula ="A1"&"B1" returns the text *A1B1*.

An alternative approach is to use the CONCAT function. Generally, you use =CON CAT(*text1, text2, …*), where *text1*, *text2*, and so on are the text values you want to concatenate. For example, typing **=CONCAT(A1, " ", B1)** is equivalent to the formula =A1&" "&B1 and returns *John Doe*.

If you're using Excel 2019 or later, you can use the TEXTJOIN function, which lets you specify a delimiter and ignore any empty calls. Generally, you use =TEXT JOIN(*delimiter, ignore_empty, text_values*), where *delimiter* is the delim between each text value, *ignore_empty* specifies whether to ignore any empty cells, and *text_values* refers to the text you want to join together, such as cells or a cell range. To combine the text in cells A1 and B1 separated by a space, for example, you type **=TEXTJOIN(" ", TRUE, A1, B1)** or **=TEXTJOIN(" ", TRUE, A1:B1)**; see Figure 5-1. Similarly, to combine the text in cells A1:A5 with a comma and space between each one, you type **=TEXTJOIN(", ", TRUE, A1:A5)**.

	A	B	C	D	E	F
1	John	Doe	=TEXTJOIN(" ",TRUE,A1,B1)			
2	Jane	Doe	Jane Doe			

Figure 5-1. Using the TEXTJOIN function to concatenate text

Discussion

This recipe offers a convenient way of combining text using the & operator and the CONCAT and TEXTJOIN functions. You can also use Recipe 5.15 to convert an array (such as a range of cells) to a text string, or use Recipe 5.7 to extract part of the text (such as initials).

5.2 Using Character Codes

Problem

You have two or more text strings and want to insert a character—such as a line break—between them.

Solution

Suppose you have text in cells A1:A5, which you want to join using a line break delimiter.

You can solve this problem using the CHAR function, which returns the character associated with a numeric code. CHAR(10), for example, returns a line break on Windows and Excel 365 on a Mac, so to join the text in cells A1:A5 using a line break delimiter, you'd type **=TEXTJOIN(CHAR(10), TRUE, A1:A5)**; for older versions of Excel on a Mac, use CHAR(13) instead of CHAR(10).

> Line breaks are visible only in cells that can wrap text; set this option by selecting the cell and choosing Home ⇒ Alignment ⇒ Wrap Text. To manually insert a line break in a cell, press Alt+Enter in Excel for Windows and Ctrl+Option+Return or Option+Return in Excel for Mac.

You can also use the CODE function to find the numeric code associated with the first character in a text string. For example, to find the code number of the letter A, you use the formula =CODE("A"), which returns 65.

If you want to use Unicode characters, use the UNICHAR function to insert a Unicode character and the UNICODE function to return a character's Unicode number.

Discussion

The numeric codes used by the CHAR and CODE functions correspond to the character set used by your operating system, so it uses the ANSI character set on Windows and the macOS character set on a Mac. The character sets are the same for numeric codes up to 127 inclusive.

Using character codes has a wide range of applications; see Recipes 5.3, 5.4, and 5.14 for some examples.

5.3 Generating a Sequence of Characters

Problem

You want to generate a sequence of characters.

Solution

If you're using Excel 2021 or Excel 365, you can use the CHAR and SEQUENCE functions to generate a sequence of characters.

To generate a list of all the characters returned by the CHAR function, type the formula **=CHAR(SEQUENCE(255))**; this returns a dynamic array (see Recipe 3.4) of 255 characters, starting from CHAR(1).

To generate a list of uppercase letters from A to Z, type the formula **=CHAR(SEQUENCE(26, 1, 65))**; this returns 26 characters, starting from CHAR(65) or A. To generate a list of lowercase letters from a to z instead, type **=CHAR(SEQUENCE(26, 1, 97))**.

Discussion

This recipe offers a convenient way of generating a dynamic array containing a sequence of characters, which you can use with recipes like Recipe 5.7.

5.4 Generating Random Letters

Problem

You want to generate a random letter from A to Z.

Solution

You can generate a random uppercase letter between A and Z inclusive by typing **=CHAR(RANDBETWEEN(65, 90))**. This formula uses the RANDBETWEEN function (see Recipe 4.1) to generate a random integer between 65 and 90 inclusive, which the CHAR function then converts to a letter from A to Z. Similarly, you can generate a random lowercase letter by typing **=CHAR(RANDBETWEEN(97, 122))**.

Discussion

This recipe offers a quick way of generating random letters from the Latin or Roman alphabet. You can use this approach to generate other random characters, too.

5.5 Finding the Length of a Text String

Problem

You have a text string and want to determine its length.

Solution

Suppose you have text in A1 and want to know how many characters it contains. You can determine this by typing **=LEN(A1)**, which returns the length of A1. Generally,

you use the formula =LEN(*text*), so typing **=LEN("Hello everyone!")**, for example, returns 15.

Discussion

The LEN function is a handy way of returning the length of a text string that you can combine with other recipes. For example, using LEN with lets you return all the characters in a text string except for the first.

5.6 Finding Text Position in a Text String

Problem

You have a text string and want to find the location of specific text inside it.

Solution

Suppose you have text in A1 and want to find the location of the first space within it. You can do so by typing **=FIND(" ", A1)**. In general, you use the formula =FIND(*find_text, within_text, start_num*), where *find_text* is the text you want to find, *within_text* is the text string you want to find it in, and *start_num* (optional) is the character number at which you want to start the search—this defaults to 1, so it starts from the beginning of the text string. The formula =FIND(" ", "Hello everyone!"), for example, returns 6—the position of the Space character. If the FIND function can't find the specified text, it returns a #VALUE! error, which you can check for using .

> The FIND function is case-sensitive. The formula =FIND("A", "Aardvark"), for example, returns 1 for the position of the first uppercase letter A, while =FIND("a","Aardvark") returns 2—the position of the first lowercase a.

An alternative to FIND is SEARCH. This function works similarly to FIND, except that it's case-insensitive. The formula =SEARCH("a", "Aardvark"), for example, returns 1.

The SEARCH function can also accept wildcards, which you can use as a substitute for other characters. In general, you use ? as a substitute for a single character and * for multiple characters. The formula =SEARCH("?v", "Hello everyone"), for example, returns 7, which is the position of the character immediately before the v. Similarly, =SEARCH("e*v", "Hello everyone") returns 2, which is the position of the first e; this is different from using =SEARCH("e", "Hello everyone") because it also checks whether the text includes a following v. If you want to search a text string for a

question mark or asterisk, you can do so by prefixing it with a ~ character. The formula =SEARCH("~?", "Hello?"), for example, returns 6—the position of the question mark.

You can also use the FIND and SEARCH functions to search for an array of characters and return the position of the first one. To test whether cell A1 includes a number, for example, you can use the formula =COUNT(FIND({0,1,2,3,4,5,6,7,8,9}, A1))>0, and to find the location of the first number, use =MIN(FIND({0,1,2,3,4,5,6,7,8,9}, A1&"0123456789")); the A1&"0123456789" string ensures the function finds at least one occurrence of each number so it doesn't return an error (see Figure 5-2).

	A	B	C	D	E	F	G
1	NY1234	=MIN(FIND({0,1,2,3,4,5,6,7,8,9},A1&"0123456789"))					
2	LON7854	4					

Figure 5-2. Using the FIND function to find the first number

Discussion

This recipe is a helpful way of finding the location of text in a text string. You can combine this recipe with others. For example, using it with Recipe 5.7 lets you extract part of a substring when you don't know its start position.

5.7 Getting Fixed-Width Text from a Text String

Problem

You have a text string and want to return its first, last, or middle characters.

Solution

Suppose you have text in cell A1 and want to use a formula to return its first, last, or middle characters.

To return the first characters in a text string, you use the formula =LEFT(*text, num_chars*), where *num_chars* is the number of characters—this argument is optional and defaults to 1. To return the first character in cell A1, for example, you type **=LEFT(A1)**, and if A1 and B1 contain someone's first name and last name, you can concatenate their initials by typing **=LEFT(A1)&LEFT(B1)**.

To return the last characters, use the formula =RIGHT(*text, num_chars*) instead. To return the last character in cell A1, for example, type **=RIGHT(A1)**, and to return all the characters except for the first, type **=RIGHT(A1, LEN(A1)-1)**.

To return characters in the middle of a text string, you use the formula =MID(*text, start_num, num_chars*), where *start_num* is the character position you want to start at and *num_chars* is the number of characters you want to return. To extract the second, third, and fourth characters of A1, for example, you type **=MID(A1, 2, 3)**.

If *start_num* is greater than the length of the *text* argument, the MID function returns the empty string "". If the sum of *start_num* and *num_chars* is greater than the length of *text*, the function returns the characters up to the end of the text.

You can also use the MID function with Recipe 5.6 to extract characters based on the text you find in the text string. To return the first two characters in A1 that follow a space, for example, type **=MID(A1, FIND(" ", A1)+1, 2)**; in this example, FIND(" ", A1) returns the position of the first space, so FIND(" ", A1)+1 gets the position of the following character.

Discussion

This recipe is helpful if you want to extract a fixed-width substring from some text. Notice how you can use it with other recipes, such as Recipes 5.5 and 5.6, to tailor your approach.

5.8 Getting Text from a Text String by Delimiter

Problem

You have a text string and want to extract the text before or after a specific character.

Solution

Suppose cell A1 contains someone's first name and last name, separated by a space, and you want to split the text into first name and last name.

To return the text before a delimiter—for example, a space—use the TEXTBEFORE function (if you're using Excel 365). Typing **=TEXTBEFORE(A1, " ")**, for example, returns the text in A1 before the first space. In general, you use =TEXTBEFORE(*text, delimiter, instance_num, match_mode, match_end, if_not_found*), with arguments as follows:

text
    This is the text string you want to extract characters from.

delimiter
    This is the delimiter you want to use.

instance_num (optional)
This is the instance of the delimiter you want to extract text before—the default is 1. Set this to a negative number if you want to start searching for the delimiter from the end of the text.

match_mode (optional)
If the delimiter is text, this argument specifies whether you want the text search to be case-sensitive or case-insensitive. By default it's case-sensitive, and you make it case-insensitive by setting this argument to 1.

match_end (optional)
This specifies whether you want to treat the end of the text as a delimiter. It assumes that you don't by default, and you set it to 1 to override this.

if_not_found (optional)
This is the value you want to return if there's no match, where the default is #N/A.

If there are multiple delimiters, you can specify them using an array. To return the text in A1 before a space or semicolon, for example, use the formula =TEXTBE FORE(A1, {" ",";"}), where {" ",";"} is the array of delimiters.

To return the text after a delimiter, you can use the TEXTAFTER function (if you're using Excel 365); this works like TEXTBEFORE, except it returns the text after a delimiter. Typing **=TEXTAFTER(A1, " ")**, for example, returns the text in A1 after the first space.

> You can also use TEXTBEFORE and TEXTAFTER to return the text between two characters or strings. So if cell A1 contains the text *Start 15:00 End 16:00*, you can return the text between *Start* and *End* using the formula =TEXTBEFORE(TEXTAFTER(A1, "Start"), "End").

To split the text using a delimiter, you can use the TEXTSPLIT function (if you're using Excel 365) to spill the split text across columns (and, optionally, rows). Typing **=TEXTS PLIT(A1, " ")**, for example, splits the text in A1 using a space as a delimiter and adds each substring to a separate column (see Figure 5-3).

	A	B	C	D	E	F
1	John Doe	=TEXTSPLIT(A1," ")				
2	Jane Doe	Jane	Doe			

Figure 5-3. Using the TEXTSPLIT function to split text

In general, you use the formula =TEXTSPLIT(*text, col_delimiter, row_delimiter, ignore_empty, match_mode, pad_with*), with arguments as follows:

text
    This is the text string you want to split.

col_delimiter
    This is the delimiter you want to use to spill text across columns.

row_delimiter (optional)
    This is the delimiter you want to use to spill text across rows.

ignore_empty (optional)
    Set this to TRUE if you want to ignore consecutive delimiters. It defaults to FALSE, which creates an empty cell.

match_mode (optional)
    If the delimiter is text, this argument specifies whether you want the text search to be case-sensitive or case-insensitive. By default it's case-sensitive, and you make it case-insensitive by setting this argument to 1.

pad_width (optional)
    This is the value you want to pad the result by—the default is #N/A.

The TEXTBEFORE, TEXTAFTER, and TEXTSPLIT functions are available only in Excel 365. In earlier versions of Excel, you can use the formula =LEFT(*text,* FIND(*delimiter, text*)-1) instead of TEXTBEFORE, and =RIGHT(*text,* LEN(*text*)-FIND(*delimiter, text*)) instead of TEXTAFTER.

Discussion

This recipe is the reverse of Recipe 5.1; instead of joining text, you use a delimiter to extract parts of the text.

You can also use Power Query for this type of operation instead of using formulas; see Recipes 15.14 and 15.15 for more information.

5.9 Getting Text from a Text String by Digit to Nondigit

Problem

You have a text string with numeric and nonnumeric characters and want to split it into numeric and nonnumeric parts.

Solution

If you're using Excel 365, you can use the TEXTBEFORE and TEXTAFTER functions (see Recipe 5.8) to separate text based on the position of its numeric and nonnumeric characters.

To return the text before the first number, use the formula =TEXTBEFORE(*text,* SEQUENCE(10, 1, 0)). For example, if cell A1 contains the text *NY1234*, typing **=TEXTBEFORE(A1, SEQUENCE(10, 1, 0))** returns *NY*.

To return the text before the first uppercase letter, you can use the formula =TEXT BEFORE(*text,* CHAR(SEQUENCE(26, 1, 65))). So if cell A4 contains the text *1234NY*, typing **=TEXTBEFORE(A4, CHAR(SEQUENCE(26, 1, 65)))** returns *1234*.

To return the text after the last number, you use the formula =TEXTAFTER(*text,* SEQUENCE(10, 1, 0), -1). If cell A4 contains the text *1234NY*, for example, typing **=TEXTAFTER(A4, SEQUENCE(10, 1, 0), -1)** returns *NY*.

To return the text after the last uppercase letter, use the formula =TEXTAFTER(*text,* CHAR(SEQUENCE(26, 1, 65)), -1). If cell A1 contains the text *NY1234*, typing **=TEXT AFTER(A1, CHAR(SEQUENCE(26, 1, 65)), -1)** returns *1234*.

If you're not using Excel 365, you can use the LEFT and RIGHT functions instead. To return the text before the first number, for example, you use the formula LEFT(*text,* MIN(FIND({0,1,2,3,4,5,6,7,8,9}, *text*&"0123456789"))-1), and to return the text after the last number, you use RIGHT(*text,* LEN(*text*)-MIN (FIND({0,1,2,3,4,5,6,7,8,9}, *text*&"0123456789"))+1).

Discussion

This recipe is similar to Recipe 5.8 except that it splits the text by character type instead of a specific delimiter. As you can see, the solution is more straightforward when using Excel 365.

An alternative approach is to use Power Query, which has a built-in facility to split columns using digits and nondigits; see Recipes 15.14 and 15.15 for more details.

5.10 Replacing, Inserting, and Deleting Text

Problem

You have a text string and want to replace one or more characters.

Solution

The `REPLACE` function replaces characters in a text string based on their position and returns the result. In general, you use the formula =REPLACE(*original_text, start_num, num_chars, new_text*), where *original_text* is the original text string, *start_num* is the position of the first character you want to replace, *num_chars* is the number of characters you want to replace, and *new_text* is the text you want to replace the characters with. Typing **=REPLACE("snowballing", 6, 3, "oard")**, for example, replaces the *all* string in *snowballing* with *oard*, returning *snowboarding*.

You can also use the `REPLACE` function to insert or delete characters. Typing **=REPLACE("snowing", 5, 0, "mobil")**, for example, returns *snowmobiling*, and typing **=REPLACE("snowdrifts", 5, 5, "")** returns *snows*.

If you want to replace specific text in a text string, regardless of position, you can use the `SUBSTITUTE` function. In general, you use the formula =SUBSTITUTE (*original_text, old_text, new_text, instance_num*), where *original_text* is the original text string, *old_text* is the text you want to replace, *new_text* is the text you want to replace it with, and *instance_num* (optional) specifies which occurrence of *old_text* you want to replace—by default, it replaces every occurrence. Typing **=SUBSTITUTE("Rome wasn't built in a day", "a d", "Norw")**, for example, returns *Rome wasn't built in Norway*.

> You can nest `SUBSTITUTE` function calls to make multiple substitutions. Typing **=SUBSTITUTE(SUBSTITUTE("Don't go breaking my heart", "re", ""), "he", "t")**, for example, returns *Don't go baking my tart*.

Discussion

The `REPLACE` and `SUBSTITUTE` functions are handy ways of replacing characters in a text string and returning the result. Note that the functions return an updated version of the text string, leaving the original text intact, so typing **=SUBSTITUTE(A1, "mcd", "McD")** in cell B1, for example, doesn't change the text in cell A1. You can, however, update the text in A1 by copying cell B1, selecting cell A1, and choosing Home ⇒ Paste ⇒ Paste Values.

5.11 Removing Extra Characters

Problem

You have a text string and want to remove extra spaces and nonprintable characters.

Solution

If you have a text string that contains extra spaces—for example, leading or trailing spaces or more than one space between words—you can remove them using the TRIM function. In general, use the formula =TRIM(*text*), so typing **=TRIM(A1)**, for example, returns the text in A1 without any extra spaces.

The TRIM function removes only extra instances of the space character CHAR(32) and ignores the nonbreaking space character UNICHAR(160). To remove extra instances of both characters, use the formula =TRIM(SUBSTITUTE(*text,* UNICHAR(160)**,** " ")), which replaces the nonbreaking space character with a space before trimming the text.

If you have a text string that contains nonprintable characters, you can remove most of them using the CLEAN function. In general, you use the formula =CLEAN(*text*), so typing **=CLEAN(A1)**, for example, returns a clean version of the text in A1.

The CLEAN function doesn't remove the Delete character CHAR(127), which might be in text you've imported from elsewhere. Use the formula =CLEAN(SUBSTITUTE(*text,* CHAR(127)**,** "")), to remove this character before cleaning the text.

> You can use the EXACT function to check whether a text string includes extra spaces or nonprintable characters. The function accepts two text strings and returns TRUE if they're the same and FALSE if they're different. So if =EXACT(A1, CLEAN(TRIM(A1))) is FALSE, it means cell A1 includes extra spaces or characters.

Discussion

Text imported from elsewhere, such as web pages, can often include extra nonprintable characters, and this recipe is a convenient way of removing them.

Just as in Recipe 5.10, the TRIM and CLEAN functions return an updated version of the text string, leaving the original text intact, so typing **=TRIM(A1)** in cell B1, for example, doesn't change the text in cell A1. You can, however, update the text in A1 by copying cell B1, selecting cell A1, and choosing Home ⇒ Paste ⇒ Paste Values.

5.12 Counting Words or Specific Characters

Problem

You have text in a cell and want to know how many words it contains and how often a specific character occurs.

Solution

Suppose cell A1 contains some text, and you want to know how many words are in the cell. You can find this out by typing `=LEN(TRIM(A1))-LEN(SUBSTITUTE(A1, " ", ""))+1`, which takes the length of the cell's text trimmed of extra spaces and compares it with the length of the text without any spaces at all.

You can use a similar approach to count the number of times a single character appears in a cell. To count the number of times an uppercase letter E appears in cell A1, for example, you type `=LEN(A1)-LEN(SUBSTITUTE(A1, "E", ""))`.

Discussion

This recipe uses Recipes 5.5, 5.10, and 5.11 to return a cell's word count or how often a specific character occurs.

5.13 Changing Text Case

Problem

You have a text string and want to change its case.

Solution

The `LOWER` function returns the lowercase version of a text string. In general, you use the formula =LOWER(*text*), so typing `=LOWER(A1)`, for example, returns a lowercase version of the text in A1.

The `UPPER` function is similar to the `LOWER` function, except that it returns the uppercase version of a text string. Typing `=UPPER(A1)`, for example, returns an uppercase version of the text in A1.

The `PROPER` function changes a text string's first letter to uppercase along with each letter immediately following a character that's not a letter; it then converts all other characters to lowercase. Typing `=PROPER("old mcdonald had a farm")`, for example, returns the text *Old Mcdonald Had A Farm*.

> Typing `=PROPER("mcdonald")` returns the text *Mcdonald* instead of *McDonald*; to return *McDonald* instead, use `=SUBSTITUTE(PROPER("MCDONALD"), "Mcd", "McD")`, which replaces *Mcd* with *McD* after changing the text case.

Discussion

Excel includes functions that return the lower-, upper-, and proper case versions of a text string. Just as in Recipe 5.10, the `LOWER`, `UPPER`, and `PROPER` functions return an updated version of the text string, leaving the original text intact, so typing `=LOWER(A1)` in cell B1, for example, doesn't change the text in cell A1. You can, however, update the text in A1 by copying cell B1, selecting cell A1, and choosing Home ⇒ Paste ⇒ Paste Values.

5.14 Repeating Characters

Problem

You have a character or text string you want to repeat several times.

Solution

The `REPT` function repeats a given character or text string a specified number of times. Typing `=REPT("-", 10)`, for example, returns 10 dashes. Generally, you use the formula =REPT(*text, num_times*), where *text* is the text you want to repeat and *num_times* is the number of times you want to repeat it.

Discussion

A different use for this recipe is creating dot plots or pictographs, where the number of times you repeat a character reflects an item's frequency (see Recipe 8.1). For example, you can use `UNICHAR(8226)` or `UNICHAR(11044)` to display filled-in circles and `UNICHAR(9606)` to show data bars (see Figure 5-4).

Figure 5-4. Using the REPT function to create a dot plot

5.15 Converting an Array to Text

Problem

You have a range of cells and want to convert their contents to a single text string.

Solution

If you're using Excel 365, you can use the ARRAYTOTEXT function to convert an array—such as a range of cells—to text. In general, use the formula =ARRAYTOTEXT(*array, format*), where *array* is the array you want to convert to text and *format* (optional) is the format you want to apply; set *format* to 0 (the default) to return a concise list of values and set it to 1 to return the data in a strict format you can use in other formulas.

For example, suppose A2:A4 contains the values Fred, Krishna, and Greg, and B2:B4 contains 34, 56, and 27. Typing **=ARRAYTOTEXT(A2:B4)** returns *Fred, 34, Krishna, 56, Greg, 27* and typing **=ARRAYTOTEXT(A2:B3, 1)** returns *{"Fred",34;"Krishna",56;"Greg", 27}*; see Figure 5-5.

	A	B	C	D	E	F
1	Values			Formula	Result	
2	Fred	34		=ARRAYTOTEXT(A1:B3)	Fred, 34, Krishna, 56, Greg, 27	
3	Krishna	56		=ARRAYTOTEXT(A1:B3, 1)	{"Fred",34;"Krishna",56;"Greg",27}	
4	Greg	27				

Figure 5-5. Using the ARRAYTOTEXT function to convert an array to text

Discussion

This recipe offers concise and strict methods for converting an array to a text string. While the concise format is more readable, the strict format returns a text string that formulas can parse.

5.16 Formatting Text as Currency

Problem

You have a numeric value and want to format it in a text string as currency.

Solution

Suppose you have a number in A1 that you want to include in a text string, formatted as currency.

To format the number with the currency symbol for your region, you can use the DOLLAR function. If your region is the United States and A1 contains the number 65,674.7, for example, typing **=DOLLAR(A1)** returns the text string *$65,674.70*. If your region is the United Kingdom, typing the same function returns *£65,674.70*. In general, you use the formula =DOLLAR(*number, decimals*), where *number* is the number

you want to format as currency and *decimals* (optional) is the number of digits to the right of the decimal point—the default is 2.

> You can change the default local currency by changing your region. On Windows, open your Windows Settings and choose Time & Language ⇒ Language & Region ⇒ Regional Format. On Mac, open System Preferences and choose Language & Region ⇒ Region.

To use a different currency symbol or format the currency in another way, you can use the TEXT function. This function lets you specify how you want the text to be formatted, so typing **=TEXT(A1,"€#,##0.00")** returns *€65,674.70* if A1 contains the number 65,674.7, and *–€65,674.70* if it contains –65,674.7. In general, you use the formula =TEXT(*value, format*), where *value* is the value you want to format as a text string and *format* is the format you want to apply (see Recipe 1.5).

Discussion

When you, for example, use an & to join a cell's currency value to a text string, Excel omits the cell's formatting. This recipe lets you format the currency in a way that's appropriate to your situation.

> If you simply want to change the appearance of a cell's currency value, you should apply a format instead of using the DOLLAR and TEXT functions; these convert the currency value to a text string, which functions like SUM and AVERAGE ignore.

5.17 Including Numeric Values in a Text String

Problem

You have a numeric value and want to format it in a text string.

Solution

Suppose you have a number in A1 that you want to include in a text string.

If you don't need to apply any special format to the number, you can append it to a text string using Recipe 5.1. If A1 contains the number 65,674.7, for example, typing **="Number: "&A1** returns the text string *Number: 65674.7*.

If you want to format the number with a fixed number of digits after the decimal point and optionally include thousands separators, you can use the FIXED function. In

general, use the formula =FIXED(*number, decimals, no_commas*), where *number* is the number you want to convert to text, *decimals* (optional) is the number of digits to the right of the decimal point (the default is 2), and *no_commas* is TRUE if you don't want to include thousands separators or FALSE if you want to include them (the default). If A1 contains the number 65,674.7, for example, typing **=FIXED(A1)** returns the text string *65,674.70*, typing **=FIXED(A1, 0)** returns *65,675*, and typing **=FIXED(A1, 0, TRUE)** returns *65675*.

You can apply other formats to numbers using the TEXT function (see Recipe 5.16). To format a number as a percentage, for example, you can use =TEXT(*number*, "0%"), so typing **=TEXT(A1, "0%")**, where A1 contains 0.84, for example, returns the text string *84%*. You can also use it to pad a number with zeros on the left. So if A1 contains the number 656, typing **=TEXT(A1, "00000")** returns the text string *00656*.

> You can also pad a text string with zeros on the right using the LEFT function to strip out extra characters. Typing **=LEFT("NY" &"00000", 5)**, for example, returns the text string *NY000*.

Discussion

Adding a cell's numeric value to a text string omits the cell's formatting. This recipe lets you format the number however you want.

> If you simply want to change the appearance of a cell's numeric value, you should apply a format instead of using the FIXED and TEXT functions; these convert the number to a text string, which functions like SUM and AVERAGE ignore.

5.18 Including Date/Time Values in a Text String

Problem

You have a date/time value and want to format it in a text string.

Solution

Suppose you have a date/time value in A1 that you want to include in a text string.

In this situation, you must format the date/time value using the TEXT function, or the date will be displayed as a number. If A1 contains the date January 14, 2023, for example, typing **=TEXT(A1, "m/d/yyyy")** returns the text string *1/14/2023*, and typing **=TEXT(A1, "dddd, mmmm d, yyyy")** returns *Saturday, January 14, 2023*.

Similarly, if A1 contains the time 8:15:00, typing **=TEXT(A1, "h:mm AM/PM")** returns the text string 8:15 a.m.

Discussion

Behind the scenes, a date/time value is a number (see "Discussion" on page 136). This recipe lets you format the date/time however you want when you include it in a text string. If you simply want to change the appearance of a cell's date/time value, you should apply a format instead of using the TEXT function.

Dates and Times

Dates and times are crucial to many spreadsheet calculations, yet working with them can be surprisingly complex and frustrating.

This chapter takes you through how to use formulas to solve many date and time problems, such as extracting parts of a date, calculating calendar and fiscal quarters, performing date and time calculations, and using working days. Above all, the chapter helps you understand how date/time serial numbers work, taking the mystery out of working with this type of data.

6.1 Returning the Current Date and Time

Problem

You want to enter the current date or time in a cell or use it in a formula.

Solution

If you want to enter the current date in a cell, you can use the keyboard shortcut Ctrl+; which enters a static date that doesn't automatically update. You can also use Ctrl+Shift+; in Excel for Windows or Cmd+; in Excel for Mac to enter the current time. To enter the current date and time in a single cell in Excel for Windows, for example, you'd press Ctrl+;, then press Space, and then Ctrl+Shift+;.

If you want the date and time to dynamically update or to use them in other formulas, you can use the TODAY and NOW functions. For example, typing `=TODAY()` returns the current date, and typing `=NOW()` returns the current date and time.

Discussion

`TODAY` and `NOW` are frequently used with other formulas because they dynamically return the current date and time. Recipe 6.13, for example, uses the `TODAY` function to calculate someone's current age using their date of birth.

6.2 Getting Part of a Date/Time Value

Problem

You have a date/time value and want to get its year, month, day, hours, minutes, or seconds.

Solution

Suppose A2 contains the date/time March 27, 2023 10:45 PM.

To retrieve the date components of A2, type **=YEAR(A2)** to get the year (2023), **=MONTH(A2)** to get the month (3), and **=DAY(A2)** to get the day (27).

To retrieve the time components of A2, type **=HOUR(A2)** to get the hour using a 24-hour clock (22), **=MINUTE(A2)** to get the minutes (45), and **=SECOND(A2)** to get the seconds (0).

Discussion

This recipe lets you split a date into its constituent parts. These functions are commonly used with other recipes to perform date calculations (for example, as in Recipe 6.4).

6.3 Getting the Day of the Week and Week of the Year

Problem

You have a date value and want to retrieve its day of the week and week of the year.

Solution

Suppose A2 contains the date March 27, 2023, and you want to know its day of the week and week of the year.

To get the day of the week as an integer, use the formula =WEEKDAY(*date, return_type*), where *return_type* (optional) is a numeric code specifying how integers are assigned to each day—if you omit it, it uses Sunday as the first day of the

week, so Sunday returns 1, Monday returns 2, and so on. Typing **=WEEKDAY(A2)**, for example, returns 2 because March 27, 2023 was a Monday.

> You can use the TEXT function to get the day's name. The formula =TEXT(*date,* "dddd") returns the full day name—for example, Monday—and =TEXT(*date,* "ddd") returns an abbreviated version—for example, Mon.

To get the week of the year, use the formula =WEEKNUM(*date, return_type*), where *return_type* (optional) is a numeric code that specifies the first day of the week and whether to use system 1 or 2; codes 1, 2, and 11 to 17 use system 1, which interprets the first week as the one containing January 1, while code 21 uses system 2, which interprets the first week as the one containing the first Thursday. By default, the WEEK NUM function uses system 1 with the week beginning on a Monday.

System 2 is sometimes known as the ISO or European week number, which you can also calculate using =ISOWEEKNUM(*date*) instead of =WEEKNUM(*date,* 21).

Discussion

This recipe offers a flexible way of calculating the day of the week and the week of the year using different systems. You can specify which system you want to use when constructing the formulas.

6.4 Getting the Calendar or Fiscal Quarter

Problem

You have a date value and want to find which quarter it's in based on a calendar or fiscal year.

Solution

To return a date's quarter based on a calendar year with a first quarter of January to March, use the formula =ROUNDUP(MONTH(*date*)/3, 0), which divides the date's month by three (since there are three months per quarter) and rounds it up to the nearest integer (see Recipe 4.9). So if cell A2 contains March 27, 2023, typing **=ROUNDUP(MONTH(***date***)/3, 0)** returns 1 (see Figure 6-1).

	A	B	C	D
1	Date	Formula	Calendar Quarter	
2	March 27, 2023	=ROUNDUP(MONTH(A2)/3,0)	1	
3	June 21, 2024	=ROUNDUP(MONTH(A3)/3,0)	2	

Figure 6-1. Calculating a date's calendar quarter

The formula you use to calculate a date's fiscal quarter depends on the start date of the fiscal year. For example, if the fiscal year starts on October 1 so that the first quarter is October to December, you can use the formula =CHOOSE(MONTH(*date*), 2, 2, 2, 3, 3, 3, 4, 4, 4, 1, 1, 1), which chooses the quarter based on the date's month (see Recipe 7.8). Similarly, if the fiscal year starts in April, you can use the formula =CHOOSE(MONTH(*date*), 4, 4, 4, 1, 1, 1, 2, 2, 2, 3, 3, 3) to classify the first quarter as April to June (see Figure 6-2).

	A	B	C
1	Date	Formula	Fiscal Quarter
2	March 27, 2023	=CHOOSE(MONTH(A2),4,4,4,1,1,1,2,2,2,3,3,3)	4
3	June 21, 2024	=CHOOSE(MONTH(A3),4,4,4,1,1,1,2,2,2,3,3,3)	1

Figure 6-2. Calculating a date's fiscal quarter

You can modify the formula to use fiscal quarters that begin partway through a month. For example, if the fiscal year starts April 6 and you want to classify April 6 to June 5 as the first quarter, you can use the formula =CHOOSE(MONTH(*date*-5), 4, 4, 4, 1, 1, 1, 2, 2, 2, 3, 3, 3) to return the fiscal quarters.

> You can use the YEAR and MONTH functions to determine a date's fiscal year. For example, if the fiscal year starts on October 1, you can use the formula =YEAR(*date*)-(MONTH(*date*)<10) & "/" & YEAR(*date*)+(MONTH(*date*)>=10).

Discussion

Excel doesn't have a built-in function to calculate a date's quarter, so this recipe offers several approaches you can use. Calculating the fiscal quarters is particularly handy when using PivotTables to summarize financial data (see Chapter 11).

6.5 Constructing Dates Using Day, Month, and Year

Problem

You have a day, month, and year and want to use them to construct a date.

Solution

You can construct a date value using the DATE function. Typing **=DATE(2023, 3, 27)**, for example, returns the date March 27, 2023 (see Figure 6-3). In general, you use the formula =DATE(*year, month, day*), where *year*, *month*, and *day* are integers and are used as follows:

year

If the *year* argument is 1900 to 9999, Excel uses its value as the year. However, if *year* is 0 to 1899, Excel adds that number of years to 1900 and uses this as the year instead, so it interprets =DATE(1800, 1, 1) as January 1, 3700. This is because Excel can't store date/time values before the year 1900 (see "Discussion" on page 136).

month

If the *month* argument is 1 to 12, Excel uses its value as the month. If *month* is greater than 12, it adds that number of months to the first month of the year, interpreting =DATE(2023, 13, 1) as January 1, 2024. If *month* is 0 or less, it subtracts that number of months plus one from the first month of the year, so it interprets =DATE(2023, -1, 1) as November 1, 2022.

day

If the *day* argument is valid for the month, Excel uses its value for the day. If *day* is greater than the number of days in the month, Excel adds that number to the first day of the month, so it interprets =DATE(2023, 2, 29) as March 1, 2023. If *day* is 0 or less, it subtracts that number of days plus one from the first day of the month, so it interprets =DATE(2023, 2, -1) as January 30, 2023.

Formula	Date
=DATE(2023, 3, 27)	March 27, 2023
=DATE(1800, 1, 1)	January 1, 3700
=DATE(2023, 13, 1)	January 1, 2024
=DATE(2023, 2, 29)	March 1, 2023
=DATE(2023, 2, -1)	January 30, 2023

Figure 6-3. Constructing dates using the DATE function

Discussion

`DATE` is one of Excel's most important date functions because you can use it to perform date calculations and construct dates. It is, however, a common source of errors because Excel can store dates only from January 1, 1900.

> Always use a four-digit year to avoid ambiguity and ensure it's between the years 1900 and 9999. For example, suppose you type the formula **=DATE(15, 1, 1)**. In this case, Excel interprets the year as 1915, while if you type the date **1/1/15** directly into a cell, Excel interprets it as 2015 (depending on your regional settings).

6.6 Constructing Times Using Hours, Minutes, and Seconds

Problem

You have hours, minutes, and seconds and want to use them to construct a time.

Solution

You can construct a time value using the `TIME` function. Typing **=TIME(13,30,0)**, for example, returns the time 1:30 p.m. (see Figure 6-4). In general, you use the formula =TIME(*hours, minutes, seconds*), where *hours*, *minutes*, and *seconds* are integers from 0 to 32767 and are used as follows:

hours

If the *hours* argument is 0 to 23, Excel uses its value as the hours. If *hours* is greater than 23, Excel divides it by 24 and uses the remainder for the hours, so it interprets =TIME(30, 0, 0) as 6:00 a.m.

minutes

If the *minutes* argument is 0 to 59, Excel uses its value as the minutes. If it's more than 59, Excel converts it to hours and minutes, interpreting =TIME(0, 120, 0) as 2:00 a.m.

seconds

If the *seconds* argument is 0 to 59, Excel uses its value as the seconds. If it's greater than 59, Excel converts it to hours, minutes, and seconds, so it interprets =TIME(0, 0, 120) as 12:02 a.m.

Formula	Time
=TIME(13,30,0)	1:30 PM
=TIME(30,0,0)	6:00 AM
=TIME(0,120,0)	2:00 AM
=TIME(0,0,120)	12:02 AM

Figure 6-4. Constructing times using the TIME function

Discussion

The TIME function works similarly to the DATE function (see Recipe 6.5). However, it can't return a time that's more than 24 hours, so it removes any date part.

6.7 Converting a Text Value to a Date/Time Serial Number

Problem

You have a date/time value stored as text and want to filter, sort, or format it as a date/time or use it in date/time calculations.

Solution

Suppose you have the date and time March 27, 2023 6:00 PM formatted as text, and you want to filter or sort it as a date/time, format it differently, or use it in date/time calculations. You can do so by converting it to a serial number that Excel recognizes as a date/time.

To convert the date component to a serial number, use the formula =DATE VALUE(*text*). Typing **=DATEVALUE("27-MAR-2023 6:00 PM")**, for example, returns the serial number 45012, which corresponds to March 27, 2023 (see Figure 6-5 and "Discussion" on page 136).

Formula	Serial Number	Date/Time
=DATEVALUE("27-MAR-2023 6:00 PM")	45012	March 27, 2023
=TIMEVALUE("27-MAR-2023 6:00 PM")	0.75	6:00:00 PM

Figure 6-5. Using the DATEVALUE and TIMEVALUE functions

> The DATEVALUE function uses your system's regional settings to interpret any dates you enter (see Recipe 3.2), so enter dates as unambiguously as possible to avoid surprises. For example, Excel interprets =DATEVALUE("1/12/2023") as January 12 in some regions and as December 1 in others. Similarly, you should use only four-digit years to avoid any ambiguity.

To convert the time component, use the formula =TIMEVALUE(*text*). Typing **=TIMEVALUE("27-MAR-2023 6:00 PM")**, for example, returns 0.75, which corresponds to 6:00 p.m.

To get a serial number that includes the date and time components, use the formula =DATEVALUE(*text*)+TIMEVALUE(*text*). Typing **=DATEVALUE("27-MAR-2023 6:00 PM")+TIMEVALUE("27-MAR-2023 6:00 PM")**, for example, returns the serial number 45012.75, which corresponds to March 27, 2023 6:00 p.m.

> If a cell contains a date/time serial number, you can see the corresponding date and time by changing its format to a date or time.

Discussion

When you type a date or time into a cell, Excel automatically stores it as a serial number and formats it as a date or time. It uses this serial number to perform date/time calculations and sort and filter data. If a worksheet has dates or times in a text format instead, you need to convert them to date/time serial numbers to work correctly.

The date component of the serial number is a sequential integer that increases by one for each day. The earliest date it can store is January 1, 1900, which has a serial number of 1, so March 27, 2023 has a serial number of 45012.

The time component is the serial number's fractional or decimal part, which stores the time as a fraction of a day. It stores the time 00:00 as 0, so 12:00 p.m. is 0.5 since 12 hours is half a day.

Most date/time values have both date and time components, so March 27, 2023 6:00 PM, for example, has a serial number of 45012.75.

> If you try to format a pure time value as a date, Excel shows it as January 0, 1900. Also, Excel incorrectly assumes that 1900 is a leap year, so some functions—for example, WEEKDAY—return incorrect results for dates before March 1, 1900.

6.8 Extracting the Date and Time from a Serial Number

Problem

You have a date/time value and want to extract its date and time components.

Solution

Suppose cell A2 contains the date/time value March 27, 2023 6:00 p.m. (serial number 45012.75; see "Discussion" on page 136), and you want to extract its date and time components. You can do so by splitting the date/time serial number into its integer and fractional parts.

To get the date component, you need to extract the serial number's integer part, which you can do using the INT function. So typing `=INT(A2)` returns the date component of the date/time value in A2—45012—which Excel interprets as March 27, 2023.

You extract the serial number's fractional part to get the time component. For example, typing `=MOD(A2, 1)` returns the time component of the value of A2—0.75—which Excel interprets as 6:00 p.m.

Discussion

This recipe offers a quick way of extracting the date and time components of a date/time value, which you can use in conjunction with other recipes (for example, Recipe 6.13).

6.9 Adding Days, Months, and Years to a Date

Problem

You have a date/time value and want to add a given number of days, months, and years.

Solution

Suppose cell A2 contains the date March 27, 2023, and you want to find the date a specified number of days, months, or years after.

To find the date seven days after the date in A2, you'd type `=A2+7`; this returns the serial number 45019, which Excel interprets as April 3, 2023 (see Figure 6-6 and "Discussion" on page 136). Generally, you use the formula =*date*+*days*, where *days* is the number of days you want to add or subtract.

To find the date a certain number of months after A2, you can use the EDATE function. Generally, you use the formula =EDATE(*date, months*), where *months* specifies the number of months you want to add or subtract. To find the date seven months after the date in A2, for example, you'd type `=EDATE(A2, 7)`, which returns the serial number 45226 (October 27, 2023).

	A	B	C	D
1	Date	Formula	Serial Number	Date
2	March 27, 2023	=A2+7	45019	April 3, 2023
3		=EDATE(A2, 7)	45226	October 27, 2023
4		=DATE(YEAR(A2)+1, MONTH(A2), DAY(A2))	45378	March 27, 2024
5		=DATE(YEAR(A2)+1, MONTH(A2)+6, DAY(A2)+7)	45569	October 4, 2024

Figure 6-6. Formulas for adding days, months, and years to dates

To find the date a given number of years after A2, you can use Recipe 6.2 to get the date's day, month, and year and then use the DATE function to construct a new date (see Recipe 6.5). To add one year to the date in A2, for example, you'd type **=DATE(YEAR(A2)+1, MONTH(A2), DAY(A2))**, which returns the serial number 45378 (March 27, 2024).

You can also use the DATE function to add a combination of years, months, and days. To find the date one year, six months, and seven days after A2, for example, you'd type **=DATE(YEAR(A2)+1, MONTH(A2)+6, DAY(A2)+7)**, which returns the serial number 45569 (October 4, 2024). In general, you use the formula =DATE(YEAR(*date*)+*years,* MONTH(*date*)+*months,* DAY(*date*)+*days*).

Discussion

This recipe offers several ways of adding days, months, and years to a specified date. The most flexible approach is to use the DATE function.

If you want to add working days to a date, excluding weekends and optional holidays, see Recipe 6.14.

6.10 Adding Hours, Minutes, and Seconds to a Time

Problem

You have a date or time and want to add hours, minutes, and seconds.

Solution

The solution to this problem depends on whether you want to add hours, minutes, and seconds to a pure time value or one that includes a date.

If you have a pure time value, you can use Recipe 6.2 to extract its hours, minutes, and seconds and then use the TIME function to construct a new time (see Recipe 6.5). If cell A2 contains the time 06:00 a.m. (serial number 0.25; see "Discussion" on page 136), for example, you'd find the result of adding three hours to it by typing **=TIME(HOUR(A2)+3, MINUTE(A2), SECOND(A2))**, which returns 09:00 a.m. (serial

number 0.375). In general, you use the formula `=TIME(HOUR(`*`time`*`)+`*`hours,`* `MINUTE(`*`time`*`)+`*`minutes,`* `SECOND(`*`time`*`)+`*`seconds`*`)`, where *time* is the original time and *hours*, *minutes*, and *seconds* are the hours, minutes, and seconds you want to add, respectively.

If you want to add time to a value that includes a date, you must ensure you increment the date if the time passes midnight. In this situation, you use the formula `=`*`date_time`*`+(`*`hours`*`/24)+(`*`minutes`*`/1440)+(`*`seconds`*`/86400)`. This approach converts the hours, minutes, and seconds to a decimal and then adds it to the date/time value. So if A2 contains March 27, 2023 6:00 p.m. (serial number 45012.75), you'd find the result of adding 12 hours to it by typing **`=A2+0.5`** (since 12 hours is half a day), which returns March 28, 2023 6:00 a.m. (serial number 45013.24).

Discussion

When using this recipe, consider whether to increment the integer part of the date/ time's serial number when the time passes midnight.

If the date/time value includes a date, you must ensure that any time you add correctly increments the serial number's integer part since the integer part corresponds to the date. For example, adding 24 hours to March 27, 2023 (serial number 45012) should return March 28, 2023 (serial number 45013).

If the value doesn't include a date, you must ensure that its serial number's integer part remains zero when you add time. The time 6:00 PM, for example, has the serial number 0.75, so adding 12 hours (0.5) to it should return the serial number 0.25 (6:00 a.m.) and not 1.25 (January 1, 1900 6:00 a.m).

6.11 Getting the Last Day of the Month

Problem

You have a date value and want to find the last day of the current month or a specified number of months before or after it.

Solution

Suppose you have the date March 27, 2023 in A2 and want to find the last day of the month six months after it. You can do so by typing **`=EOMONTH(A2, 6)`**, which returns the date serial number 45199; you can manually format this as the date September 30, 2023 (see). In general, you use the formula `=EOMONTH(`*`date,`* *`months`*`)`, where months is the number of months before or after the date—a positive value returns a future date, and a negative value returns a past one.

Discussion

This recipe is a convenient way of calculating the last day of the current month or a past or future one. It's also handy for calculating maturity or due dates.

To calculate the last working day of the month, see Recipe 6.14.

6.12 Calculating the Year Fraction

Problem

You have a date value and want to calculate the elapsed and remaining fraction of the year.

Solution

You can calculate the year fraction using the YEARFRAC function. This function accepts two date arguments and calculates the fraction of the year between them as whole days. In general, use the formula =YEARFRAC(*date_1, date_2, basis*), where *basis* (optional) is the type of day count basis—if omitted, it uses 30 days per month. For example, if there are two date values in cells A2 and B2, you'd calculate the fraction of the year between them by typing **=YEARFRAC(A2, B2)**.

To calculate the elapsed fraction of the year, you can use the formula =YEAR FRAC(DATE(YEAR(*date*), 1, 1), *date*); this uses the YEAR and DATE functions to construct a date of January 1 for the date's year, which the YEARFRAC function then uses. So if cell A2 contains the date value March 27, 2023, you'd calculate the elapsed fraction of the year by typing **=YEARFRAC(DATE(YEAR(A2), 1, 1), A2)**, which returns 0.24 or 24% (see Figure 6-7).

	A	B	C
1	Date	Formula	Year Fraction
2	March 27, 2023	=YEARFRAC(DATE(YEAR(A2), 1, 1), A2)	0.24
3		=1-YEARFRAC(DATE(YEAR(A2), 1, 1), A2)	0.76

Figure 6-7. Calculating the elapsed and remaining fraction of the year

To calculate the fraction of the year that remains, you can subtract the elapsed fraction of the year from 1. So if cell A2 contains the date value March 27, 2023, you'd type **=1-YEARFRAC(DATE(YEAR(A2), 1, 1), A2)**, which returns 0.76 or 76%.

Discussion

The YEARFRAC function is a handy way of calculating the fraction of a year between two dates, including the elapsed and remaining fraction of the year.

6.13 Calculating the Difference Between Dates and Times

Problem

You have two dates or times and want to calculate their difference.

Solution

To calculate the difference between two date/time values, you can use the formula =*end_date-start_date*, which returns the difference in days, including the time. For example, if cell A2 contains March 27, 2023 8:00 a.m. and B2 contains September 30, 2023 8:00 p.m., you'd calculate the difference between them by typing **=B2-A2**, which returns 187.5 (see Figure 6-8).

	A	B	C	D
1	Start Date	End Date	Formula	Result
2	March 27, 2023 8:00:00 AM	September 30, 2023 8:00:00 PM	=B2-A2	187.5
3			=DAYS(B2, A2)	187

Figure 6-8. Calculating differences between dates

> To calculate the difference between two times, use the formula =MOD(*time_1-time_2,* 1); see Recipe 6.8. You can then convert the result to hours by multiplying the result by 24.

To calculate the difference in days, you can use the formula =DAYS(*end_date, start_date*). Typing **=DAYS(B2, A2)**, for example, returns 187.

To calculate the difference in days, months, or years, you can use the DATEDIF function. If cell A2 contains someone's date of birth, for example, you can find their current age by typing **=DATEDIF(A2, TODAY(), "y")**. In general, use =DATEDIF(*start_date, end_date, unit*), where *unit* specifies the time unit. Using "d" finds the difference in days; "m" finds the difference in complete months; "y" finds the difference in whole years; "ym" finds the difference in complete months, ignoring the days and years; and "yd" finds the difference in days, ignoring the years.

> The DATEDIF function's *unit* argument also has an "md" option, which returns the difference in days, ignoring the months and years. This option, however, yields inaccurate results.

Discussion

This recipe offers several ways of calculating the difference between two date/time values, depending on whether you want to know the difference in years, months, days, or time.

6.14 Using Working Days

Problem

You want to add working days to a date or find the difference between two dates in working days.

Solution

The WORKDAY function lets you add working days to a date that exclude weekends and optional holidays. In general, you use the formula =WORKDAY(*start_date, days, holidays*), where *start_date* is the date you want to add working days to, *days* is the number of working days you want to add or subtract, and *holidays* (optional) is a range or array containing any holiday dates you want to exclude. So if cell A2 contains March 27, 2023, you'd type **=WORKDAY(A2, 20)** to add 20 working days to it, which returns April 24, 2023. Similarly, if C2:C5 contains the dates March 30 and April 3 to 5, 2023, you'd type **=WORKDAY(A2, 20, C2:C5)** to add 20 working days to A2, excluding the dates in C2:C5; this returns April 28, 2023 (see Figure 6-9).

	A	B	C	D	E
1	Start Date	End Date	Holidays	Formula	Result
2	March 27, 2023	October 31, 2023	March 30, 2023	=WORKDAY(A2, 20)	April 24, 2023
3			April 3, 2023	=WORKDAY(A2, 20, C2:C5)	April 28, 2023
4			April 4, 2023	=NETWORKDAYS(A2, B2)	157
5			April 5, 2023	=NETWORKDAYS(A2, B2, C2:C5)	153

Figure 6-9. Calculating working days

> If the WORKDAY function's *days* argument is positive, the function returns a day in the future, and if *days* is negative, it returns a day in the past. You can use this, for example, to find the last working day of a month using the formula =WORKDAY(EOMONTH(*date*, 0)+1, -1); see Recipe 6.11.

The NETWORKDAYS function lets you calculate the number of whole working days between two dates. In general, you use =NETWORKDAYS(*start_date, end_date, holidays*), where *holidays* is optional, so if cell A2 contains March 27, 2023 and B2 contains October 31, 2023, typing **=NETWORKDAYS(A2, B2)** returns 157. Similarly, if

C2:C5 contains the dates March 30 and April 3 to 5, 2023, typing **=NETWORKDAYS(A2, B2, C2:C5)** returns 153.

> The `WORKDAY` and `NETWORKDAYS` functions count Saturdays and Sundays as weekends. For other working patterns, you can use the `WORKDAY.INTL` and `NETWORKDAYS.INTL` functions, which let you specify the weekend days.

Discussion

This recipe offers handy ways of dealing with working days. You can use the `WORKDAY` function to calculate invoice due dates or expected delivery times, for example, and the `NETWORKDAYS` function to calculate the number of work days left to meet a deadline or employee benefits based on the number of days worked.

If you want to exclude a consecutive sequence of holiday dates, you can use Recipe 6.15 to generate the sequence. For example, the formula =NETWORKDAYS(A2, B2, SEQUENCE(14, , C2)) calculates the number of working days between the dates in A2 and B2, excluding 14 consecutive dates from the date in C2.

6.15 Getting a Sequence of Dates

Problem

You want to get a dynamic array of dates starting from a specific date.

Solution

Suppose you want to generate an array of date serial numbers starting from a specific date. If you're using Microsoft 365, you can do this using the formula =SEQUENCE(*n*, , *date*), where *n* is the number of dates you want the array to contain and *date* is the first date. For example, if cell A2 contains the date March 27, 2023, you could type **=SEQUENCE(10, , A2)** to create a dynamic array of 10 date serial numbers starting from this date, which you can then format as dates.

Discussion

Since date/time values are stored as serial numbers, you can use them with math functions and operations—in this example, the `SEQUENCE` function (see Recipe 4.1). Generating a sequence of dates is handy if, for example, you want to pass sequential holiday dates to the `WORKDAY`, `NETWORKDAYS`, or `NETWORKDAYS.INTL` functions (see Recipe 6.14).

Array, Logic, and Lookup Functions

The recipes in this chapter cover three main areas that many types of spreadsheets require:

- Manipulating arrays or ranges as part of a formula
- Performing logical tests
- Looking up values by name and index

The chapter includes filtering, sorting, and combining arrays using functions instead of menu commands; trapping and handling errors; choosing values to return based on a logical test or matching value; and using `XLOOKUP`, `INDEX`, and `MATCH` to look up values. It also covers ways of working with the `INDIRECT` and `OFFSET` functions to create dynamic references to cells and ranges.

7.1 Getting Unique Values

Problem

You have an array or range and want to retrieve a list of its unique values or the ones that appear exactly once.

Solution

Suppose A2:A6 lists customer first names, B2:B6 lists their last names, and you want to use this data to return two lists: one containing the unique names and another containing names that appear only once.

If you're using Excel 2021 or Excel 365, you can solve this problem using the `UNIQUE` function to return a dynamic array of values. Generally, you use the formula

=UNIQUE(*array, by_column, exactly_once*), where *array* is the range or array whose values you want to examine, *by_column* (optional) specifies whether you want to return unique rows or columns (omit it to return unique rows or set it to TRUE to return unique columns), and *exactly_once* (optional) specifies whether you want to return values that are unique or appear exactly once—omit it to return unique values or set it to TRUE to return values that appear only once. So typing **=UNIQUE(A2:B6)** returns the unique rows of A2:B10, and typing **=UNIQUE(A2:B6, , TRUE)** returns the rows that appear exactly once (see Figure 7-1).

	A	B	C	D	E	F	G	H	I
1	First name	Last name		=UNIQUE(A2:B6)			=UNIQUE(A2:B6, , TRUE)		
2	Amy	Pond		Amy	Pond		Rose	Tyler	
3	Rose	Tyler		Rose	Tyler		Jackie	Tyler	
4	Jackie	Tyler		Jackie	Tyler		Donna	Noble	
5	Donna	Noble		Donna	Noble				
6	Amy	Pond							

Figure 7-1. Using the UNIQUE function

> The UNIQUE function returns unique values across adjacent columns or rows. If they aren't adjacent, you can use the CHOOSECOLS or CHOOSEROWS function to rearrange them (see Recipe 7.4), and then apply the UNIQUE function to the result.

Discussion

This recipe is a handy way of getting a list of an array's unique rows or columns, which you can use to build customer or product lists. You can also use it to get a list of one-time customers, for example, who you might need to contact.

> You can count the number of rows or columns returned by the UNIQUE function using the ROWS or COLUMNS functions, which return the number of rows or columns in an array. For example, typing **=ROWS(UNIQUE(A2:B6))** counts the number of unique rows in A2:B6.

7.2 Sorting an Array

Problem

You have an array or range and want a dynamic array of its values sorted in ascending or descending order.

Solution

Suppose A2:A6 lists customer first names, B2:B6 lists their last names, and you want to return one list that sorts the names by last name and another that sorts them by last name and then first name. If you're using Excel 2021 or Excel 365, you can solve this problem using the SORT and SORTBY functions.

To get a dynamic array that sorts data by a single column or row—for example, the last names—you can use the SORT function. This function takes the form =SORT(*array, sort_index, sort_order, by_column*), where *sort_index* (optional) is the row or column index you want to sort *array* by (the default is 1), *sort_order* (optional) specifies the sort order (omit it for ascending order and set it to –1 for descending order), and *by_column* (optional) specifies whether you want to sort by rows (the default) or columns—set it to TRUE to sort by columns. To get a dynamic array that sorts the names in A2:B6 by the last name, type **=SORT(A2:B6, 2)** (see Figure 7-2).

	A	B	C	D	E	F	G	H	I
1	First name	Last name		=SORT(A2:B6, 2)			=SORTBY(A2:B6, B2:B6, , A2:A6,)		
2	Rory	Pond		Donna	Noble		Donna	Noble	
3	Rose	Tyler		Rory	Pond		Amy	Pond	
4	Jackie	Tyler		Amy	Pond		Rory	Pond	
5	Donna	Noble		Rose	Tyler		Jackie	Tyler	
6	Amy	Pond		Jackie	Tyler		Rose	Tyler	

Figure 7-2. Using the SORT and SORTBY functions

To get a dynamic array that sorts data by multiple columns or rows, use the SORTBY function instead. Generally, you use =SORTBY(*array, by_array1, sort_order1, by_array2, sort_order2,* …), where *array* is the range or array you want to sort, *by_array1*, *by_array2*, and so on are the ranges or arrays you want to sort by, and *sort_order1* and *sort_order2* are their sort orders (optional); omit them to sort in ascending order (the default) or set them to –1 to sort in descending order. To get a dynamic array that sorts the names in A2:B6 by last name and then by first name in ascending order, you type **=SORTBY(A2:B10, B2:B6, , A2:A6,)**.

Discussion

This recipe is a handy way to get a dynamic array of sorted values. It's most useful when combined with recipes such as Recipe 7.1; to get a dynamic array of the unique rows in A2:B6 sorted by last name, for example, type **=SORT(UNIQUE(A2:B6), 2)**.

7.3 Filtering an Array

Problem

You have an array or range and want a dynamic array of values that meet a condition.

Solution

Suppose A2:A6 lists customer first names, B2:B6 lists their last names, and you want to get a list of all the names where the last name is *Tyler*.

If you're using Excel 2021 or Excel 365, you can solve this problem using the FILTER function to return a dynamic array of the data meeting specific criteria. You generally use the formula =FILTER(*array, criteria, if_empty*), where *array* is the array you want to filter, *criteria* is the criteria you want to filter it by, and *if_empty* (optional) is the value you want to return if the filter doesn't return anything—by default, it returns a #CALC! error value. To get a dynamic array that returns only rows where the last name is *Tyler*, you type **=FILTER(A2:B6, B2:B6="Tyler")**; see Figure 7-3.

	A	B	(	D	E	F	G	H	I	J
1	First name	Last name	=FILTER(A2:B6, B2:B6="Tyler")				=FILTER(A2:B6, (B2:B6="Tyler")*(A2:A6="Jackie"))			
2	Amy	Pond		Rose	Tyler		Jackie	Tyler		
3	Rose	Tyler		Jackie	Tyler					
4	Jackie	Tyler								
5	Donna	Noble								
6	Amy	Pond								

Figure 7-3. Using the FILTER function

You can also apply multiple criteria to the FILTER function using the * and + operators, similar to Recipe 7.6. Behind the scenes, the *criteria* argument is an array of TRUE/FALSE Boolean values that's the same width or height as the array you want to filter, so you can use Boolean algebra to control which rows (or columns) the function should return. Generally, you use the * operator if the data must meet all criteria and the + operator if it can meet any, so to get a dynamic array that returns each row where the first name is *Jackie* and the last name is *Tyler*, type **=FILTER(A2:B6, (A2:A6="Jackie")*(B2:B6="Tyler"))**. Similarly, typing **=FILTER(A2:B6, (A2:A6 ="Amy")+(B2:B6="Noble"))** returns all the rows where the first name is *Amy* or the last name is *Noble*.

Discussion

This recipe offers a flexible way of filtering an array or range using specified criteria. Notice how you can use the * and + operators to chain multiple criteria together and build complex conditions.

7.4 Manipulating Arrays

Problem

You have an array or range and want to choose which of its columns to return and their order, append its data to another array, or manipulate the array in other ways.

Solution

Excel 365 includes several new functions you can use to manipulate arrays:

CHOOSECOLS *and* CHOOSEROWS

The CHOOSECOLS function creates a dynamic array that returns some of an array's columns or changes their order. You generally use the formula =CHOOSE COLS(*array, column1, column2, …*), where *array* is the array you want to return columns from, *column1* is the index of the first column, *column2* is the index of the second, and so on. So typing **=CHOOSECOLS(A2:E10, 2, 1, 3)** returns a dynamic array containing the second, first, and third columns from the range A2:E10. The CHOOSEROWS function works similarly to CHOOSECOLS except it returns rows.

> If you're using an older version of Excel, you can replicate the CHOOSECOLS function using =CHOOSE(*array, range_1, range _2, …*); this returns an array containing *range_1* if *array* includes 1, *range_2* if it includes 2, and so on. For example, the formula =CHOOSE({1,2}, B1:B6, A1:A6) returns an array containing B1:B6 and A1:A6 (see Recipe 7.8).

VSTACK *and* HSTACK

The VSTACK function appends one array's rows to the end of another, returning a vertical stack. Generally, you use the formula =VSTACK(*array1, array2, …*), where *array1* is the first array, *array2* is the second, and so on. So typing **=VSTACK(A2:B10, A13:B16)** returns a dynamic array that appends the rows from A13:B16 to the end of the ones from A2:B10. You can also use the VSTACK function with 3-D references to create a dynamic array from values with the same cell reference in adjacent worksheets (see Recipe 2.5). Typing **=VSTACK(HR:Sales! B1)**, for example, returns a dynamic array containing the value of B1 from the

HR and Sales worksheets, and any in between. The `HSTACK` function works similarly to `VSTACK` except it appends one array's columns to the right of another.

DROP

The `DROP` function returns a dynamic array that excludes a specified number of rows and columns from the start or end of an array. You generally use the formula =DROP(*array, rows, columns*), where *array* is the array you want to return data from, and *rows* and *columns* (optional) specify how many rows and columns you want to exclude from the start or end of the array—use a positive number to exclude them from the start and a negative one to exclude them from the end. So typing **=DROP(A2:B10, 2)** returns all the rows in A2:B10 except for the first two, and typing **=DROP(A2:E10, , -3)** returns all the columns in A2:E10 except for the last three.

TAKE

The `TAKE` function returns a dynamic array that includes a specified number of rows and columns from the start or end of an array. This function works similarly to `DROP`, except you specify which rows and columns to include instead of exclude. So typing **=TAKE(A2:B10, 2)** returns the first two rows in A2:B10, and typing **=TAKE(A2:E10, , -3)** returns the last three columns in A2:E10.

TOROW *and* TOCOL

The `TOROW` function returns an array as a single row. You generally use the formula =TOROW(*array, ignore, by_column*), where *array* is the array you want to return as a row, *ignore* (optional) specifies whether to ignore blanks and/or errors (by default, it includes them) and *by_column* (optional) specifies whether to order values by row (the default) or by column—set this argument to `TRUE` to order them by column. So typing **=TOROW(A2:B10)** returns all the values in A2:B10 as a single row. The `TOCOL` function works similarly to `TOROW`, except it returns the array as a column instead of a row. So typing **=TOCOL(A2:B10)** returns all the values in A2:B10 as a single column.

> Older versions of Excel also include a `TRANSPOSE` function. This function transposes an array's rows and columns, converting the rows to columns and the columns to rows. Generally, you use the formula =TRANSPOSE(*array*).

Discussion

Excel 365 includes new functions for manipulating arrays, which you can use in conjunction with Recipes 7.1, 7.2, and 7.3. You can, for example, use the `UNIQUE` and `CHOOSECOLS` functions to return a dynamic array of unique values for columns that aren't

positioned next to each other or use the VSTACK and SORT functions to combine the values in multiple arrays and sort them.

7.5 Using Logical True/False Criteria

Problem

You want to perform a logical test and return one value if it's true and another if it's false.

Solution

Suppose cell A1 contains a number, and you want to return one value if it's greater than 50 and another if it's not.

To solve this problem, you can use the IF function to perform a logical test. You generally use the formula =IF(*logical_test, value_if_true, value_if_false*), where you want to return *value_if_true* if *logical_test* is TRUE or *value_if_false* if it's FALSE. So typing **=IF(A1>50, A1*0.1, 0)** returns A1 multiplied by 0.1 if A1 is greater than 50, and zero if it isn't.

If you're using Excel 2019 or later, you can use the IFS function to perform multiple logical tests and return different values, depending on which ones are true. This function takes the form =IFS(*logical_test1, value_if_true1, logical_test2, value_if_true2, …*), and returns *value_if_true1* if *logical_test1* is TRUE, *value_if_true2* if *logical_test2* is TRUE, and so on. So typing **=IFS(A1>=75, "Excellent", A1>=50, "Good", A1<50, "Could do better")** returns different text values depending on the value in cell A1.

> If you're using an older version of Excel, you can use nested IF functions instead of using IFS; for example, =IF(A1>=75, "Excellent", IF(A1>=50, "Good", "Could do better")).

You can invert a logical test using the NOT function. This takes the form =NOT(*logical_test*), and returns TRUE if *logical_test* is FALSE or FALSE if *logical_test* is TRUE. So typing **=IF(NOT(A1>50), A1*0.1, 0)** returns A1 multiplied by 0.1 if A1 is *not* greater than 50, and zero if it is.

To return one value if multiple conditions are all true and another if they're not, you can use IF with the AND function. The AND function returns TRUE if each argument evaluates to TRUE or FALSE if one or more evaluate to FALSE. For example, typing

`=IF(AND(A1>50, B1>=C1), A1*0.1, 0)` returns A1 multiplied by 0.1 if A1 is greater than 50 *and* B1 is greater than or equal to C1; otherwise, it returns zero.

To return one value if at least one condition is true and another if they're all false, you can use `IF` with the `OR` function. This function works similarly to `AND` except that the function returns `TRUE` if any condition is `TRUE`. So typing `=IF(OR(A1>50, B1>=C1), A1*0.1, 0)` returns A1 multiplied by 0.1 if A1 is greater than 50 *or* B1 is greater than or equal to C1; if both conditions are false, it returns zero.

Discussion

The `IF` function is one of Excel's most commonly used functions. This recipe shows how to combine it with `AND`, `OR`, and `NOT`, and how to use the `IFS` function instead of nested `IF` functions (depending on your version of Excel).

See Also

To return the count, sum, or average of cells that depend on a logical test, see Recipe 4.5.

To use AND and OR conditions with arrays, see Recipe 7.6.

To find out about other types of `IF` formulas, see Recipe 7.7.

7.6 Evaluating AND and OR Conditions in Array Formulas

Problem

You want to use AND and OR logic with arrays.

Solution

Suppose you have a list of items in A2:A4 and a list of names in B2:B4, and you want to find which rows have *Coffee* in the A column and *Joe* in the B column.

One way of solving this problem is to use the `AND` function (see Recipe 7.5) to check the values in each row. So you'd use `=AND(A2="Coffee",B2="Joe")` for the first row, `=AND(A3="Coffee",B3="Joe")` for the second, and so on. This approach, however, isn't ideal since it uses multiple formulas.

Another approach is to use a single formula like so: `=(A2:A4="Coffee")*(B2:B4="Joe")=1`. This formula uses the `*` operator to test AND conditions and returns an array of `TRUE`/`FALSE` values. It works as follows (see Figure 7-4):

1. The formula first evaluates the (A2:A4="Coffee") part, which returns TRUE for each cell in A2:A4 with the value *Coffee* and FALSE for all other cells.

2. It then evaluates the (B2:B4="Joe") part, which returns TRUE for each cell in B2:B4 with the value *Joe* and FALSE for all other cells.

3. The formula then multiplies the results from step 1 and step 2 together. Since the * works with numeric values, it converts TRUE values to 1 and FALSE values to 0 (see Recipe 3.1) and multiplies the values for each row. The multiplication returns 1 for each row where both conditions are TRUE and 0 for all other rows.

4. Finally, the formula tests whether each value in step 3 equals 1 and returns an array of TRUE/FALSE values.

	A	B	C	D
1	Drink	Name	=(A2:A5="Coffee")*(B2:B5="Joe")=1	=(A2:A5="Coffee")+(B2:B5="Joe")<>0
2	Tea	Joe	FALSE	TRUE
3	Coffee	Joe	TRUE	TRUE
4	Coffee	Lucy	FALSE	TRUE
5	Matcha	Lucy	FALSE	FALSE

*Figure 7-4. Using * and + operators in an array formula*

If you want to test for OR conditions, use the + operator instead of *. The formula =(A2:A4="Coffee")+(B2:B4="Joe")<>0, for example, returns an array of TRUE/FALSE values indicating whether the cell in the A column has the value *Coffee* or the cell in the B column contains the value *Joe* for each row.

Discussion

This recipe shows a handy trick for working with AND and OR conditions and returning dynamic arrays. This approach is convenient when used with functions such as SUMPRODUCT (see Recipe 4.7) since you can use it to include only rows that meet specific conditions. Generally, you use * as an AND operator and + as an OR operator.

7.7 Working with Types and Error Values

Problem

You have a value and want to check its type and handle any error values.

Solution

Excel includes many functions you can use to check a value's type. To check whether cell A1 contains a number, for example, you can type the formula **=ISNUMBER(A1)**, which returns TRUE if the value in A1 is a number or FALSE if it isn't. Similar functions include ISTEXT, ISNONTEXT, ISLOGICAL, ISODD, ISEVEN, ISBLANK, ISERR, IS ERROR, ISNA, ISFORMULA, and ISREF, which tests for references.

> The ISERROR function returns TRUE if a value is an error, while ISERR returns TRUE for all error types except #N/A. Note that you can use the ISNA formula separately to check for #N/A errors.

IS functions are often used with the IF function (see Recipe 7.5). So typing **=IF(ISBLANK(A1), "Blank", "Not blank")**, returns *Blank* if A1 is empty and *Not blank* if it isn't. You can also use the IFNA function to handle #N/A values and IFERROR to handle any error, including #N/A. So typing **=IFERROR(A1/B1, "Error")** returns A1 divided by B1 if this calculation is valid and *Error* if it results in an error value.

You can also use the TYPE function to return a numeric code representing the value's type. These codes are 1 for a number, 2 for text, 4 for a logical TRUE/FALSE value, 16 for an error value, 64 for an array, and 128 for compound data. So if cell A1 contains a number, typing **=TYPE(A1)** returns 1.

Discussion

This recipe offers a flexible way of working with different types of value. They let you perform different actions depending on a value's type and trap and handle error values.

7.8 Choosing Values to Return

Problem

You want to choose a value to return based on a matching index or expression.

Solution

If you want to use an index number to return a value from a list of arguments, you can use the CHOOSE function. It takes the form =CHOOSE(*index_number, value1, value2, …*) and returns *value1* if *index_number* is 1, *value2* if *index_number* is 2, and so on. So typing **=CHOOSE(A2, "Car", "Bike", "Truck")** returns *Car* if cell A2 contains 1, *Bike* if it contains 2, and *Truck* if it contains 3. You can also use CHOOSE to

---

return different ranges depending on the value of *index_number*, so you can use the formula =CHOOSE(A2, B2:B4, C2:C4) to return a dynamic array (depending on your version of Excel; see Recipe 3.4) containing the values in B2:B4 if cell A2 contains 1 and the values in C2:C4 if A2 contains 2 (see Figure 7-5).

	A	B	C	D	E	F
1	Index	Vehicles	Animals	=CHOOSE(A2, B2:B4, C2:C4)	=CHOOSE({2,1}, B2:B4, C2:C4)	
2	1	Car	Cat	Car	Cat	Car
3		Bike	Dog	Bike	Dog	Bike
4		Truck	Fish	Truck	Fish	Truck

Figure 7-5. Using CHOOSE

You can also pass the CHOOSE function an array of index numbers to make it return multiple values. For example, typing **=CHOOSE({2,1}, B2:B4, C2:C4)** returns a dynamic array composed of the values in C2:C4 and B2:B4.

If you're using Excel 2019 or later, you can also use the SWITCH function; this is similar to CHOOSE, except it uses an expression to determine which value to return. You generally use the formula =SWITCH(*expression, value1, result1, value2, result2, …, default*), which returns *result1* if *expression* evaluates to *value1*, *result2* if it evaluates to *value2*, and so on; if there are no matching values, the function returns *default*. For example, typing **=SWITCH(A2, "Planes", B2:B4, "Trains", C2:C4, D2:D4)** returns the values in B2:B4 if A2 contains the text *Planes*, C2:C4 if it contains *Trains*, and D2:D4 if it contains neither (see Figure 7-6).

	A	B	C	D	E
1	Expression	Planes	Trains	Other	=SWITCH(A2, "Planes", B2:B4, "Trains", C2:C4, D2:D4)
2	Planes	Jumbo	Sleeper	Car	Jumbo
3		Cargo	Monorail	Truck	Cargo
4		Aerobatic	High-speed	Bus	Aerobatic

Figure 7-6. Using SWITCH

Discussion

The CHOOSE and SWITCH functions can offer convenient alternatives to using IF and IFS (see Recipe 7.5). Note that you can also use CHOOSE as an alternative to the CHOOSECOLS function in older versions of Excel (see Recipe 7.4).

7.9 Looking Up Exact and Nearest Values

Problem

You want to use one value to look up another from a range.

Solution

Suppose you have a customer ID and want to look up the corresponding customer name from A2:C5, where A2:A5 lists the customer IDs, B2:B5 lists the customer first names, and C2:C5 lists the customer last names.

If you're using Excel 2021 or Excel 365, you can look up the customer name using the XLOOKUP function. Generally, you use the formula =XLOOKUP(`lookup_value, lookup_array, return_array, if_not_found, match_mode, search_mode`) with arguments as follows:

`lookup_value`
> This is the value you want to search for.

`lookup_array`
> This is the range or array you want to search.

`return_array`
> This is the range or array that contains the values you want to return.

`if_not_found` *(optional)*
> This specifies what value to return if the function can't find a match; if you omit this argument, the function returns #N/A.

`match_mode` *(optional)*
> This specifies the type of match. If you omit this argument or set it to 0, the function returns `if_not_found` or #N/A if it can't find an exact match. Set `match_mode` to –1 to return the next smallest item, 1 to return the next largest item instead, or 2 to perform a wildcard search that uses *, ?, and ~ as wildcards.

`search_mode` *(optional)*
> This specifies the type of search and the search order. If you omit this argument or set it to 1, the function searches from the first item, and if you set it to –1, the function searches from the last item. If you set `search_mode` to 2, the function performs a binary search that assumes `lookup_array` is sorted in ascending order; setting it to –2 assumes that `lookup_array` is sorted in descending order instead.

To return the first name and last name of the customer whose ID is 2, type **=XLOOKUP(2, A2:A5, B2:C5)**. Similarly, if G2:G5 lists the lower limit of four price

bands and H2:H5 lists the name of each band, you can find which band contains the value 3500 by typing **=XLOOKUP(3500, G2:G5, H2:H5, , -1)**; see Figure 7-7.

	A	B	C	D	E	F	G	H	I
							Lower		
1	ID	First name	Last name	=XLOOKUP(2, A2:A5, B2:C5)			limit	Band	=XLOOKUP(3500, G2:G5, H2:H5, , -1)
2	1	Amy	Pond	Clara	Oswald		0	A	B
3	2	Clara	Oswald				3000	B	
4	3	Rose	Tyler				4000	C	
5	4	Jackie	Tyler				5000	D	

Figure 7-7. Using XLOOKUP

You should use XLOOKUP only to perform a binary search where *lookup_array* has been sorted. Otherwise, the function may return invalid results.

If you're using an older version of Excel, you can use the VLOOKUP function instead. VLOOKUP looks for a value in a column and returns a value in the same row. Generally, you use the formula =VLOOKUP(*lookup_value, lookup_array, index_num, match _type*), where *lookup_value* is the value you want to search for and the first column in *lookup_array*, *index_num* is the index of the column in *lookup_array* you want to return, and *match_type* (optional) specifies the type of match; set it to 0 or FALSE to look for an exact match, and omit it or set it to 1 or TRUE to return an approximate match if *lookup_array*'s first column is sorted in ascending order. So typing **=VLOOKUP(2, A2:C5, 3, FALSE)** looks for *2* in the first column and returns the value in the third column of that row. Similarly, if G2:G5 lists the lower limit of four price bands in ascending order and H2:H5 lists the name of each band, typing **=VLOOKUP(3500, G2:H5, 2)** returns the band name containing the value 3500.

Excel also includes an HLOOKUP function, which looks for a value in a row and returns a value in the same column. This function works similarly to VLOOKUP, so typing **=HLOOKUP("Surname", A1:C7, 3, FALSE)** looks for the "Surname" text in the first row and returns the value in the third row of that column.

Discussion

Looking up values is a crucial aspect of many worksheets, and this recipe shows you how to do this using the XLOOKUP, VLOOKUP, and HLOOKUP functions. XLOOKUP is more flexible than VLOOKUP and HLOOKUP since it doesn't need the value you want to search to be in the first row or column; it is, however, available only in more recent versions of Excel.

You can also look up values using the INDEX function with XMATCH or MATCH (see Recipes 7.10 and 7.11).

7.10 Finding a Matching Value's Index

Problem

You have a value and want to find its relative position in a range or array.

Solution

If you're using Excel 2021 or Excel 365, you can use the XMATCH function to search for an item in a range or array and return its relative position or index. Generally, you use the formula =XMATCH(*lookup_value, lookup_array, match_mode, search_mode*), where you want to search for *lookup_value* in *lookup_array*, and the *match_mode* and *search_mode* arguments are as follows:

match_mode (optional)
> This specifies the type of match. If you omit this argument or set it to 0, the function returns #N/A if it can't find an exact match. Set it to –1 to return the next smallest item, 1 to return the next largest item, or 2 to perform a wildcard search that uses *, ?, and ~ as wildcards.((("asterisk (*) wildcard character")a)

search_mode (optional)
> This specifies the type of search and the search order. If you omit this argument or set it to 1, the function searches from the first item, and if you set it to –1, the function searches from the last item. If you set it to 2, the function performs a binary search that assumes *lookup_array* is sorted in ascending order; setting it to –2 assumes that *lookup_array* is sorted in descending order instead.

For example, if A2:A5 lists food items and B2:B5 lists corresponding amounts, you can find the index of the *Coffee* item by typing **=XMATCH("Coffee", A2:A5)** and the index of the item with an amount of 100 (or the next largest amount) by typing **=XMATCH(100, B2:B5, 1)**; see Figure 7-8.

	A	B	C	D	E	F	G	H
1	Item	Amount		=XMATCH("Coffee", A2:A5)		=XMATCH(100, B2:B5, 1)		
2	Cola	53		2		4		
3	Coffee	65						
4	Soup	356						
5	Salad	143						

Figure 7-8. Using XMATCH

If you're using an older version of Excel, you can use the MATCH function instead of XMATCH. This takes the form =MATCH(*lookup_value, lookup_array, match_mode*), where you want to search for *lookup_value* in *lookup_array*, and *match_mode* specifies the type of match; set *match_mode* to 0 to find the first exact match, –1 if *lookup_array* is sorted in descending order and you want to find the smallest value that's greater than or equal to *lookup_value*, and omit *match_mode* or set it to 1 if *lookup_array* is sorted in ascending order and you want to find the largest value that's smaller than or equal to *lookup_value*. For example, typing **=MATCH("Coffee", A2:A5)** returns the index of the *Coffee* item in A2:A5, and typing **=MATCH(100, B2:B5, -1)** returns the index of the item with an amount of 100 (or the next largest amount) as long as the amounts are arranged in descending order.

Discussion

This recipe offers a flexible approach to finding the index of a matching value. It's often used with Recipe 7.11 as an alternative to using XLOOKUP, VLOOKUP, or HLOOKUP.

7.11 Using an Index to Return a Value

Problem

You have a range or array and want to get the value at a given position.

Solution

The INDEX uses a row and/or column index to return a value from a range or array. Generally, you use the formula =INDEX(*array, row_num, column_num*), where *array* is the range or array you want to return a value from, *row_num* (optional) is the row index, and *column_num* (optional) is the column index. So if A2:A5 lists food items and B2:B5 lists their corresponding amounts, typing **=INDEX(A2:A5, 3)** returns the value in the third row of A2:A6 and typing **=INDEX(A2:B5, 3, 2)** returns the value in the third row and second column of A2:B6 (see Figure 7-9).

	A	B	C	H	I	J
1	Item	Amount		=INDEX(A2:A5, 3)	=INDEX(A2:B5, 3, 2)	
2	Cola	53		Soup	356	
3	Coffee	65				
4	Soup	356				
5	Salad	143				

Figure 7-9. Using INDEX

You can also pass multiple ranges to the INDEX and provide an extra argument specifying which array to return values from. This approach takes the form =INDEX((*array1, array2, …*), *row_num, column_num, array_num*), where *array1, array2*, and so on are the arrays, and *array_num* is the array number to return values from. So typing **=INDEX((A2:A5, A8:A12), 3, , 2)** returns the value in the third row of A8:A12—the second array in (A2:A5, A8:A12).

The INDEX function is often used with XMATCH or MATCH (see Recipe 7.10) to search for and return a value. So if A2:A5 lists food items and B2:B5 lists corresponding amounts, typing **=INDEX(A2:A5, XMATCH(100, B2:B5, 1))** returns the item with an amount of 100 (or the next largest amount) (see Figure 7-10).

	A	B	C	H	I	J
1	Item	Amount		=INDEX(A2:A5, XMATCH(100, B2:B5, 1))		
2	Cola	53		Salad		
3	Coffee	65				
4	Soup	356				
5	Salad	143				

Figure 7-10. Using INDEX with XMATCH

Using INDEX with XMATCH also lets you look up a value for both rows and columns. So if A2:A5 lists food items, B1:D1 lists month names, and B2:D5 lists an amount for each item per month, you can find the March amount for Coffee using the formula =INDEX(A1:D5, XMATCH("Coffee", A1:A5), XMATCH("March", A1:D1)). In this example, XMATCH("Coffee", A1:A5) finds the Coffee row index, XMATCH("March", A1:D1) finds the March column index, and the INDEX function returns the value at the specified row and column index (see Figure 7-11).

	A	B	C	D	E	F
1	Item	January	February	March		=INDEX(A1:D5, XMATCH("Coffee", A1:A5), XMATCH("March", A1:D1))
2	Cola	123	234	346		734
3	Coffee	543	513	734		
4	Soup	756	823	623		
5	Salad	762	714	612		

Figure 7-11. Using INDEX with XMATCH with row and column indexes

You can also use the INDEX function as part of a dynamic range reference to determine the range's first and/or last cell. The formula =A1:INDEX(A:A, 5), for example, returns the range A1:A5, and the formula =A1:INDEX(A:A, COUNTA(A:A)) counts the number of cells containing values in the A column and uses this to determine the range's last cell.

Discussion

The `INDEX` function offers a convenient way of returning a value based on a row and/or column index. It's often used with `XMATCH` and `MATCH`, and it's handy for performing two-way lookups using both rows and columns. You can also use `INDEX` to create dynamic named ranges (see Recipe 2.7).

7.12 Creating Indirect References to Cells and Ranges

Problem

You want to use a text value to dynamically derive a cell or range reference.

Solution

The `INDIRECT` function returns a cell or range reference from a text value so you can create them dynamically. For example, if cell B2 contains the text *A1* and cell A1 contains the number 3, typing **=INDIRECT(B2)** converts the text value of B2 to the cell reference A1, and returns the value in A1 of 3 (see Figure 7-12).

Figure 7-12. Using `INDIRECT` to create a reference to cell A1, returning 3

Generally, you use the formula =INDIRECT(*ref_text, a1*), where *ref_text* is the text you want to translate to a cell reference and *a1* (optional) specifies whether to use A1-style references (the default: set it to `FALSE` to return R1C1-style references; see Recipe 2.3).

> The `INDIRECT` function can interpret R1C1-style references even if these are switched off. So if cell B3 contains the text *R2C1* and cell A2 contains the number 4, typing **=INDIRECT(B3,FALSE)** translates B3's text value *R2C1* as a reference to the cell A2—the second row in the first column—and returns 4.

You can also use the `INDIRECT` function to create cell references by combining text strings. So if cell B2 contains 1 and cell A1 contains 4, typing **=INDIRECT("A"&B2)** concatenates the letter A with the 1 from cell B2, translates this as a reference to cell A1, and returns the value in A1 of 4 (see Figure 7-13).

Figure 7-13. Using INDIRECT to create a reference by combining text strings, returning 4

The INDIRECT function also works with named cells, ranges, and tables. For example, if cell C2 contains the text *SalesTable* and you're using a version of Excel that supports dynamic arrays (see Recipe 3.4), typing **=INDIRECT(C2)** returns a dynamic array of the data in SalesTable. Similarly, if cell C3 contains the text *SalesTable[Amount]*, typing **=SUM(INDIRECT(C3))** returns the sum of SalesTable's Amount column.

> The INDIRECT function doesn't automatically update its cell or range references when you add or delete a row or column. If you type **=SUM(A20:A24)** and insert a new row above A20, Excel automatically changes the formula to =SUM(A21:A25). Typing **=SUM (INDIRECT("A20:A24"))**, however, keeps the cell reference as A20:A24.

Discussion

The INDIRECT function lets you perform tasks in Excel that, at first glance, can seem impossible. Note that in addition to getting references to cells and ranges in the current worksheet, you can also construct references to other worksheets and workbooks.

See Also

See Recipe 7.14 for an example of how to use INDIRECT with other functions.

7.13 Getting a Cell's Address

Problem

You're typing a formula in a cell and want to get the cell's row and column numbers and its address.

Solution

To get a cell's row number, use the ROW function. You generally use the formula =ROW(*reference*), where *reference* is optional and defaults to the cell where you've entered the formula, so typing **=ROW()** in cell C4 returns 4.

The COLUMN function works similarly to the ROW function, except it returns the column number. Typing **=COLUMN()** in cell C5 returns 3.

The ADDRESS function uses row and column numbers to return the cell's address. You generally use the formula =ADDRESS(*row_num, column_num, absolute, a1, sheet_name*), where *row_num* and *column_num* are the row and column numbers, *absolute* (optional) specifies whether to use absolute references prefixed with $'s (the default—set it to 4 to use relative references), *a1* (optional) specifies whether to use A1-style references (the default—set it to FALSE to return R1C1-style references described in Recipe 2.3), and *sheet_name* (optional) specifies the name of the worksheet. So typing **=ADDRESS(2, 4)** returns the text *$D$2*, typing **=ADDRESS(3, 6, 4)** returns *F3*, and typing **=ADDRESS(ROW(), COLUMN())** in C2 returns the text *$C$2*.

> You can also get a cell's location (and other information) using the CELL function. For example, typing **=CELL("address", B1)** returns *$B$1*, typing **=CELL("col", B1)** returns 2, and typing **=CELL("file name", B1)** returns the full path of the file containing the reference.

Discussion

This recipe offers a handy way of returning a cell's row, column, or address, which you can use in other formulas (see Recipe 7.14 for an example).

7.14 Using Offset References

Problem

You have a cell or range of cells and want to get a reference to a cell or range that's a specified number of rows and columns away from it.

Solution

The OFFSET function takes a cell or range as a starting point and offsets it by a specified number of rows and columns. Generally, use the formula =OFFSET(*reference, rows, columns, height, width*), where *reference* is the cell or range you want to start from, *rows* and *columns* are the number of rows and columns you want to offset *reference* by (use positive values to move below or to the right and negative values to move above or to the left), and *height* and *width* (optional) are the height and width of the range you want to return—the default is 1. Typing **=OFFSET(A1, 3, 2)** starts at cell A1, moves three rows down and two columns to the right, and returns the value in cell C4. Similarly, typing **=OFFSET(D3, -2, -3)** starts at cell D3, moves

two rows up and three columns to the left, and returns the value in cell A1 (see Figure 7-14).

Figure 7-14. Using `OFFSET` *to return the value in A1*

You can also use `OFFSET` with other functions, such as `COUNTA` (see Recipe 4.4). For example, typing **=OFFSET(A1, COUNTA(A:A)-1, 0)** returns the last value in the A column (assuming there are no empty cells), and typing **=SUM(OFFSET(A1, COUNTA(A:A)-3, 0, 3))** returns the sum of its last three values. Similarly, if you want to offset the current cell by a given number of rows and columns, you can use `OFFSET` with Recipes 7.12 and 7.13. For example, typing **=OFFSET(INDIRECT(ADDRESS(ROW(), COLUMN())), -3, -4)** in cell E4 uses E4—the current cell—as its starting point, moves two rows up and four columns to the left, and returns the value in cell A1.

Discussion

The `OFFSET` function offers a flexible way of referring to other cells relative to a specified starting point. For example, you can refer to cells a fixed distance away from a given cell or use it with other functions, as demonstrated in this recipe.

Statistical Analysis

Excel includes many statistical functions and features to help you generate statistics and analyze data. For example, you can use it to create frequency distributions, generate statistics such as standard deviation and skewness, work with probability distributions, and more. This chapter guides you through these areas, focusing on using functions, PivotTables, and charts.

8.1 Creating a Frequency Table

Problem

You have a list of text or numeric values and want to count the number of occurrences of each value.

Solution

Suppose you have a list of values in a column and want to find the *frequency* of each unique value: the number of times it occurs in the column. You can do so by creating a frequency table that lists each value and its corresponding frequency.

The method you choose to create the frequency table depends on the type of values for which you want to find frequencies.

For discrete values such as text or integers, you can create the frequency table using a PivotTable as follows:

1. Select the data and insert a PivotTable by choosing Insert ⇒ Tables ⇒ PivotTable.

2. Add the column of values to the PivotTable Fields pane's Rows and Values sections and make sure the Values aggregation is set to Count (see Recipe 11.9). The results are shown in Figure 8-1.

Category ▾		Categories ▾	Frequency	
Pizza		Burger	2	
Burger		Pizza	3	
Salad		Salad	1	
Pizza		**Grand Total**	6	
Pizza				
Burger				

Figure 8-1. A frequency table for discrete data created using a PivotTable[1]

For continuous data such as measurements, you often need to create a grouped frequency table that groups the values into bins or intervals and shows the frequency of each bin. You can do this using a PivotTable or Excel's FREQUENCY function.

To create a grouped frequency table using a PivotTable:

1. Select the data (including any headings) and insert a PivotTable by choosing Insert ⇒ Tables ⇒ PivotTable.

2. Add the field containing the values to the PivotTable Fields pane's Rows and Values sections and make sure the Values aggregation is set to Count.

3. Select one of the row values in the PivotTable, and use Recipe 11.17 to group the values into bins (see Figure 8-2).

Weight ▾		Weight ▾	Frequency	
17.65				
18.55		10-15	1	
17.75		15-20	4	
24.78		20-25	2	
21.22		**Grand Total**	7	
15.69				
10.28				

Figure 8-2. A grouped frequency table created using a PivotTable

---

1 The headings on the PivotTable have been changed.

If you're using Excel 2021 or Excel 365, you can create a grouped frequency table with the FREQUENCY function as follows:

1. Decide on the upper limit you want to use for each bin and type these in a single column (for example, the range C2:C6, as shown in Figure 8-3).

	A	B	C	D	E	F	G
1	Weight ▼		Bin Upper Limits	Frequency			
2	17.65		10	=FREQUENCY(tblWeights[Weight], C2:C6)			
3	18.55		15	1			
4	17.75		20	4			
5	24.78		25	2			
6	21.22		>25	0			
7	15.69						
8	10.28						

Figure 8-3. A grouped frequency table created using the FREQUENCY function

2. Select the cell next to the one that contains the first upper limit (for example, D2).

3. Enter the formula =FREQUENCY(*data_array, bins_array*), where *data_array* is the range listing the values you want to find frequencies for and *bins_array* is the range listing the bin upper limits. For example, to find frequencies of the values in the Weight column of a table named tblWeights, you'd type the formula **=FREQUENCY(tblWeights[Weight], C2:C6)**.

4. Once you've typed the formula, press Enter/Return to return a dynamic array containing the frequency of each bin.

If you're using an older version of Excel, you can use the FREQUENCY function by selecting the entire output range in step 2 (for example, D2:D6), entering the formula in the formula bar, and pressing Ctrl+Shift+Enter/Return (see Recipe 3.4).

Discussion

There are various ways of creating frequency tables, depending on the data type.

Using a PivotTable is often the most convenient method since you can use it for discrete and continuous data. However, using the FREQUENCY function is more flexible because you can have unequal bin sizes, and the results update automatically if the underlying data changes. You do, though, need to take the preliminary step of manually entering the upper limit of each bin.

You can also create frequency tables using the Analysis ToolPak; see Recipe 9.4.

8.2 Showing Cumulative and Percentage Frequencies

Problem

You have a frequency table and want to include a running sum of its frequencies and their percentage values.

Solution

The *cumulative frequency* is a running sum of the frequencies in a frequency table (see Recipe 8.1), and it determines how many values lie below a specified value. The *frequency percentage* shows each frequency's percentage of the overall total, and the *cumulative percentage* is the running sum of these percentages.

If you've created a frequency table by inserting a PivotTable, follow these steps for each statistic you want to create:

1. Add an extra `Count` of the field to the PivotTable Fields pane's Values section.

2. Use Recipe 11.10 to change how the values are displayed. Choose the % of Grand Total option to show each frequency's percentage of the overall total, Running Total In to show the cumulative frequency, and % Running Total In to show the cumulative percentage. Then click OK.

A PivotTable showing all three statistics is shown in Figure 8-4.

Weight	Weight	Frequency	Percentage	Cumulative Frequency	Cumulative Percentage
17.65	10-15	1	14.29%	1	14.29%
18.55	15-20	4	57.14%	5	71.43%
17.75	20-25	2	28.57%	7	100.00%
24.78	**Grand Total**	**7**	**100.00%**		
21.22					
15.69					
10.28					

Figure 8-4. A PivotTable displaying cumulative frequency, percentage, and cumulative percentage

If you've created a frequency table using the `FREQUENCY` function, you can use formulas to include the extra statistics. These formulas are shown in Figure 8-5, where the range D2:D6 lists the frequencies:

1. To add cumulative frequencies to the range E2:E6, type the formula **=D2** in E2, type **=D3+E2** in E3, then fill E4:E6 with the formula in E3.

2. To add percentages to the range F2:F6, type the formula **=D2/SUM($D$2:$D$6)** in F2, fill F3:F6 with the formula in F2, then format the cells as percentages.

3. To add cumulative percentages to the range G2:G6, type the formula **=E2/SUM($D $2:$D$6)** in G2, fill G3:G6 with G2's formula, then format the cells as percentages; this assumes the range E2:E6 lists the cumulative frequency of each row.

	C	D	E	F	G
1	Bin Upper Limits	Frequency	Cumulative Frequency	Percentage	Cumulative Percentage
2	10	=FREQUENCY(tblWeights[Weight],C2:C6)	=D2	=D2/SUM($D$2:$D$6)	=E2/SUM($D$2:$D$6)
3	15		=D3+E2	=D3/SUM($D$2:$D$6)	=E3/SUM($D$2:$D$6)
4	20		=D4+E3	=D4/SUM($D$2:$D$6)	=E4/SUM($D$2:$D$6)
5	25		=D5+E4	=D5/SUM($D$2:$D$6)	=E5/SUM($D$2:$D$6)
6	>25		=D6+E5	=D6/SUM($D$2:$D$6)	=E6/SUM($D$2:$D$6)

Figure 8-5. Formulas for cumulative frequency, percentage, and cumulative percentage

Discussion

This recipe shows you how to generate various statistics relating to frequencies and percentages. They let you determine the percentage of values in each bin and how many lie below a specified value. For example, the grouped frequency table shown in Figure 8-4 shows that the 15-20 bin holds 57.14% of the values. The cumulative frequency for this bin is 5, meaning there are 5 values in this bin and the preceding ones. The cumulative percentage is 71.43%, so these bins hold 71.43% of the values.

See Also

You can also generate these statistics using the Analysis ToolPak; see Recipe 9.4.

If you're using Excel 365, you can also generate running totals using the SCAN function; see Recipe 17.11.

8.3 Using a Histogram or Pareto Chart

Problem

You want to show frequencies or cumulative percentages on a chart.

Solution

Suppose you have a list of values and want to use a chart to show the frequency of each unique value and (optionally) the cumulative percentage. There are several charts you can use for this, depending on the type of data and what you want to use the chart for, including the following:

Column chart

Use this chart type to show frequencies of discrete values such as text (see Figure 8-6). You can find it by choosing Insert ⇒ Charts ⇒ Insert Column or Bar Chart.

Category	Frequency
Burger	2
Pizza	3
Salad	1
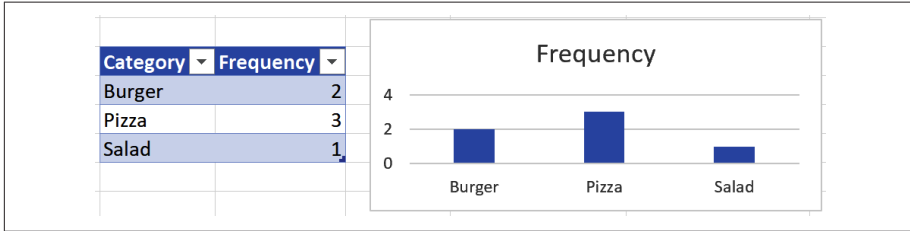
Frequency

Figure 8-6. A column chart showing frequencies of discrete data

Histogram

This chart is like a column chart, except it shows frequencies of continuous values such as measurements (see Figure 8-7). You can find it by choosing Insert ⇒ Charts ⇒ Insert Statistic Chart. Once the histogram is inserted, you may need to format its horizontal axis or data series to show the frequencies correctly (see Recipe 12.13).

Weight
17.65
18.55
17.75
24.78
21.22
15.69
10.28
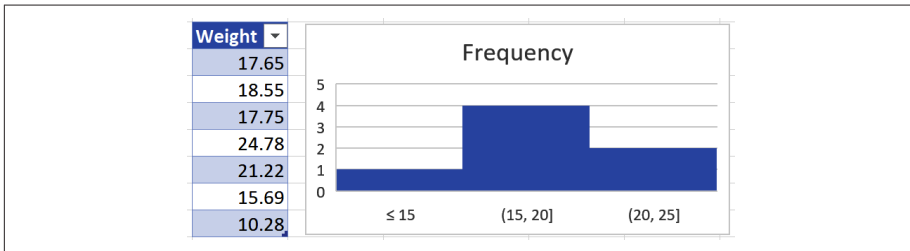
Frequency

Figure 8-7. A histogram showing frequencies of continuous data

Line chart

You can use this chart to show cumulative percentages (see Figure 8-8). You can find it by choosing Insert ⇒ Charts ⇒ Insert Line or Area Chart.

Bin Upper Limits	Cumulative Percentage
10	0.00%
15	14.29%
20	71.43%
25	100.00%
>25	100.00%

Cumulative Percentage
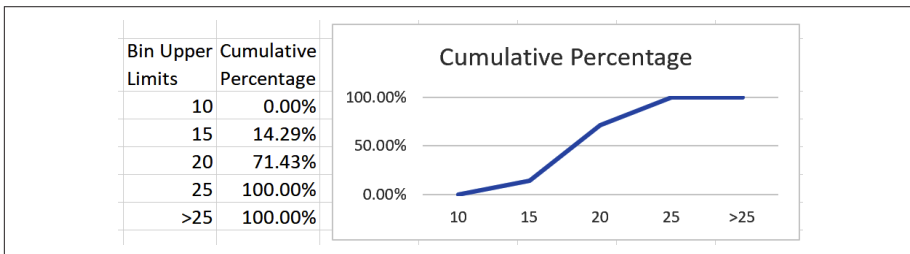
Figure 8-8. A line chart showing cumulative percentages

Pareto chart

This chart is a type of histogram that shows the frequencies in descending order and the cumulative percentages as an extra line (see Figure 8-9). You can find it by choosing Insert ⇒ Charts ⇒ Insert Statistic Chart.
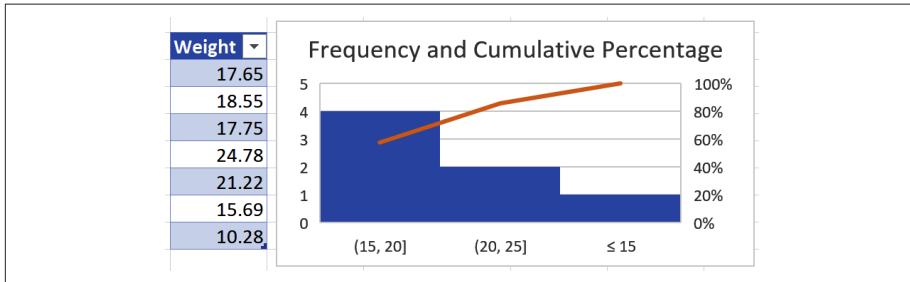
Figure 8-9. A Pareto chart showing frequencies and cumulative percentages

Discussion

The solution to this problem depends on the type of data and the information you want to display.

To display frequencies, you usually use a column chart for discrete data, such as text values or integers, and a histogram for continuous data, such as measurements. The key difference is there are no gaps between a histogram's columns to reflect the continuous nature of the data.

You usually use a line chart or a Pareto chart for cumulative percentages. A Pareto chart combines a histogram and line chart, displaying frequencies and cumulative percentages. You can use it to determine, for example, which bins or categories contain most of the data's values.

See Also

To create a chart based on a PivotTable, see Recipe 12.20.

8.4 Calculating Averages

Problem

You have a dataset and want to know what value is typical.

Solution

Suppose you have a list of numbers and want to find the value of a typical number. To do this, you can use the three averages commonly used in statistical analysis: the mean, the median, and the mode.

The *mean* is the sum of all the values divided by the count. To find the mean of a list of numbers, use the formula =AVERAGE(*numbers*), or use the Average option in a table's total row or a PivotTable. For example, to calculate the mean of numbers in the range A2:A101, you'd type **=AVERAGE(A2:A101)**.

For data that's heavily skewed by extreme values (see Recipes 8.9 and 8.11), you can use =TRIMMEAN(*numbers, percent*) instead. This function returns the mean of the list of *numbers*, excluding a percentage of the highest and lowest values (given by *percent*). So if *percent* is 0.1, the TRIMMEAN function trims 10% of the *numbers*—the highest 5% and the lowest 5%—and returns the mean of the remaining ones.

You can determine the mean for a frequency table that lists the frequencies of discrete numeric values by calculating the weighted average. The formula for this statistic is =SUMPRODUCT(*values, frequencies*)/SUM(*frequencies*), where *values* is the range of the numeric values and *frequencies* is the range of the corresponding frequencies. To find the mean of numeric values in a frequency table where A2:A6 lists the values and B2:B6 lists their frequencies, you'd type **=SUMPRODUCT(A2:A6, B2:B6)/ SUM(B2:B6)**.

The *median* is the value in the middle of a sorted list of numbers, so half the numbers lay above it, and half lay below. To find the median of a list of numbers, use the formula =MEDIAN(*numbers*). To calculate the median of the numbers listed in A2:A101, you'd type **=MEDIAN(A2:A101)**.

The *mode* is the value with the highest frequency: the one that occurs most often. To find the mode of a list of numbers, use =MODE.SNGL(*numbers*). If multiple numbers have the same highest frequency, use =MODE.MULT(*numbers*) instead, which returns an array (see Recipe 3.4) that includes each one. To find all the modes for the numbers listed in A2:A101, for example, you'd type **=MODE.MULT(A2:A101)**.

Unlike for the mean and median, you can also find the mode of a range of text values. If you're using Excel 2021 or Excel 365, you can do this using the formula =INDEX(*values*, MODE.MULT(XMATCH(*values, values*))). To find the most frequent text value listed in A2:A101, for example, you'd type **=INDEX(A2:A101, MODE.MULT (XMATCH(A2:A101, A2:A101)))**.

If you're using Excel 2021 or Excel 365, you can also use the formula =FILTER(*bins, frequencies*=MAX(*frequencies*)) to return a dynamic array of multiple modes from a frequency table, where *bins* refers to the list of bins and *frequencies* is the list of

frequencies. To find all the modes in a frequency table where A2:A6 lists the bins and B2:B6 lists their frequencies, you'd type **=FILTER(A2:A6, B2:B6=MAX(B2:B6))**.

Discussion

You can use various methods to calculate an average, depending on the type of data and the type of average you want to return.

The most commonly used type of average is the mean. This statistic is often used to describe datasets in conjunction with other statistics, such as the standard deviation (see Recipe 8.8). The mean, however, is more sensitive than other types of averages and can be pulled higher or lower by extreme values (see Recipe 8.9).

See Also

You can generate averages and other descriptive statistics using the Analysis ToolPak; see Recipe 9.2.

8.5 Ranking Numeric Data

Problem

You have a list of numbers and want to find the rank and percentage rank of each one when you sort the list in ascending or descending order.

Solution

Suppose you have a list of numbers and you want to find the rank of each number— the order of the numbers when sorted.

To find the rank of a number in the list when the values are sorted in descending order, use =RANK.EQ(*number, list*); this formula returns a rank of 1 for the largest number in the list, 2 for the second largest, and so on. To find the rank of the value in A2 in the list of numbers listed in A2:A101, for example, you'd type **=RANK.EQ(A2, A2:A101)**. To rank the numbers in ascending order, use =RANK.EQ(*number, list,* 1) instead.

> When the list contains tied values, the RANK.EQ function returns their top rank. To return the average rank instead, use the RANK. AVG function.

You can also find the rank of a number as a percentage of the list with =PERCENT
RANK.EXC(*list, number*). This formula returns the percentage rank of the specified
number when the *list* is sorted in ascending order. To find the percentage rank of the
value in A2 in the list of numbers listed in A2:A101, for example, you'd type
=PERCENTRANK.EXC(A2:A101, A2).

> The PERCENTRANK.EXC function calculates the percentage rank of a
> number using the formula $k/(n + 1)$, where k is the number, and n
> is the size of the dataset. This calculation means that it omits per-
> centage ranks of 0% and 100%. To include these percentages, use
> the PERCENTRANK.INC function, which calculates the rank using
> $(k - 1)/(n - 1)$ instead.

Discussion

This recipe is useful for returning the ordinal and percentage rank of the numbers in
a dataset. Note that Excel also has built-in RANK and PERCENTRANK functions. These
return the same values as RANK.EQ and PERCENTRANK.INC, and are there for backward
compatibility with older versions of Excel.

See Also

You can generate some of these statistics using the Analysis ToolPak; see Recipe 9.3.

8.6 Finding the kth Largest or Smallest Value

Problem

You have a list of numbers and want to find the *k*th largest or smallest value.

Solution

Suppose you have a list of numbers. To find the *k*th largest number, use the formula
=LARGE(*numbers, k*), and to find the *k*th smallest number, use the formula
=SMALL(*numbers, k*). You can also find the largest number using =MAX(*numbers*),
and the smallest using =MIN(*numbers*).

Discussion

These functions are helpful if you want to find the numeric value associated with a
given rank. You can also use them to calculate other statistics, such as the range, as in
Recipe 8.8.

8.7 Dividing Data into Quartiles and Percentiles

Problem

You have a list of numbers and want to divide the list into quarters and percentages.

Solution

Suppose you have a list of numbers and want to find its *quartiles*—the values that split the data into quarters when the list is sorted in ascending order (see Figure 8-10):

- The first (or lower) quartile is the number in the middle of the first half of the list.
- The second quartile is the median: the number that splits the list in half.
- The third (or upper) quartile is the middle number of the second half of the list.
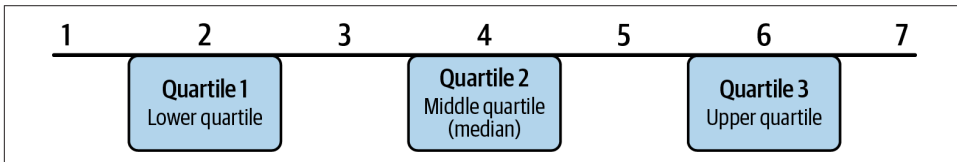
Figure 8-10. The quartiles for seven numeric values

To find the value of each quartile, use the formula =QUARTILE.EXC(*numbers, quartile*), where *numbers* is the list of numbers and *quartile* is a value of 1, 2, or 3 that denotes the quartile. To find the first quartile of the range A2:A101, for example, you'd type **=QUARTILE.EXC(A2:A101, 1)**.

You can also find the *percentiles* for a list of numbers—the values that split the data into percentages. Use the formula =PERCENTILE.EXC(*numbers, k*), where *numbers* refers to the list of numbers and *k* is a number between 0 and 1 that specifies the percentile. To find the 20th percentile for numbers listed in A2:A101, for example, you'd type **=PERCENTILE.EXC(A2:A101, 0.2)** because 0.2 is 20%.

> The .EXC part of the QUARTILE.EXC function means it excludes the median when calculating the first and third quartiles, and the .EXC part of the PERCENTILE.EXC function means it excludes the 0 and 100th percentiles. To include these, use the QUARTILE.INC and PERCENTILE.INC functions instead.

Discussion

Quartiles and percentiles are useful because they let you focus on different parts of the dataset. You can use quartiles, for example, to focus on the middle half of the data between the first and third quartiles. Notice that the first quartile is equivalent to the 25th percentile; both refer to the number that divides the lowest quarter of the data from the rest.

8.8 Calculating Ranges and Variances

Problem

You have a list of numbers and want to measure the spread and variability of the data.

Solution

The *range* is the difference between a dataset's largest and smallest numbers (see Figure 8-11). You can find this using the formula =MAX(*numbers*)-MIN(*numbers*). For example, to find the range of the numbers listed in A1:A101, you'd type **=MAX(A1:A101)-MIN(A1:A101)**.
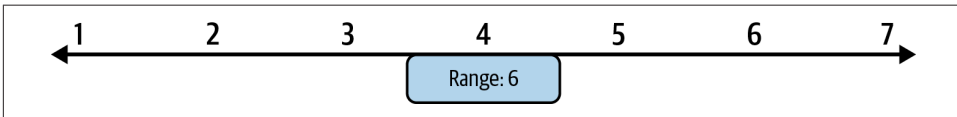
Figure 8-11. The range is the difference between the largest and smallest numbers

The *interquartile range* (see Figure 8-12) is the difference between the dataset's third and first quartiles. You calculate the interquartile range using the formula =QUARTILE.EXC(*numbers*, 3)-QUARTILE.EXC(*numbers*, 1). To find the interquartile range of the numbers listed in A1:A101, for example, you'd type **=QUARTILE. EXC(A1:A101, 3)-QUARTILE.EXC(A1:A101, 1)**.
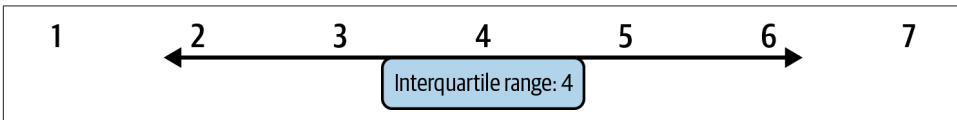
Figure 8-12. The interquartile range is the difference between the third and first quartiles

The variance and standard deviation measure how values scatter from the mean: the higher the statistic, the more the data points spread out (see Figure 8-13).
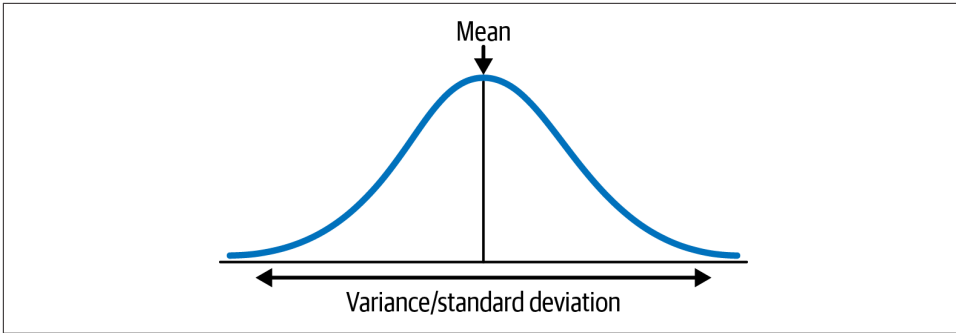
Figure 8-13. The variance and standard deviation measure how values scatter from the mean

The *variance* is the average squared distance of the data from its mean, and the *standard deviation* is the square root of this value. You calculate the variance using the formula =VAR.S(*numbers*) and the standard deviation using =STDEV.S(*numbers*). To find the variance and standard deviation of the numbers listed in A1:A101, for example, you'd type **=VAR.S(A1:A101)** and **=STDEV.S(A1:A101)**, respectively.

> The VAR.S and STDEV.S functions calculate the variance and standard deviation using sample data: a subset of data drawn from a population. If your dataset is the entire population, use VAR.P and STDEV.P instead.

Discussion

You can measure the variation in a list of numbers by calculating the range, interquartile range, variance, and standard deviation. The variance and standard deviation have many applications. For example, if a population follows a normal distribution (see Recipe 8.19), its main characteristics can be defined using the mean and the standard deviation.

See Also

You can also generate the range, variance, standard deviation, and other statistics using the Analysis ToolPak; see Recipe 9.2.

8.9 Finding Outliers

Problem

You have a list of numbers and want to find extreme values that don't represent the rest of the data.

Solution

A dataset's *outliers* are values that are notably lower or higher than the rest of the dataset. You can use two main methods to find these values: using the interquartile range, or using the mean and standard deviation.

The first method classifies any values that are more than 1.5 times the interquartile range away from the first and third quartiles as outliers. You can find these thresholds for a list of numbers using the formulas =2.5*QUARTILE.EXC(*numbers*, 1)-1.5*QUARTILE.EXC(*numbers*, 3) and =2.5*QUARTILE.EXC(*numbers*, 3)-1.5* QUARTILE.EXC(*numbers*, 1).

The second method classifies values more than twice the standard deviation away from the mean as outliers. In other words, the outliers for a set of numbers are values less than =AVERAGE(*numbers*)-2*STDEV.S(*numbers*) or greater than =AVERAGE (*numbers*)+2*STDEV.S(*numbers*).

Discussion

Outliers can significantly affect a dataset's statistics. For example, outliers increase the range and standard deviation and tend to pull the mean toward them. However, other statistics, such as the median and interquartile range, are less sensitive to these effects.

8.10 Using a Box and Whisker Chart

Problem

You have a list of numbers and want to show their range, quartiles, interquartile range, mean, and outliers on a chart.

Solution

Suppose you have a list of numbers and want to insert a chart to help you visualize its range, quartiles, and other statistics. You can do so by inserting a box and whisker chart, or box plot (see Figure 8-14), which you can find by choosing Insert ⇒ Charts ⇒ Insert Statistic Chart.
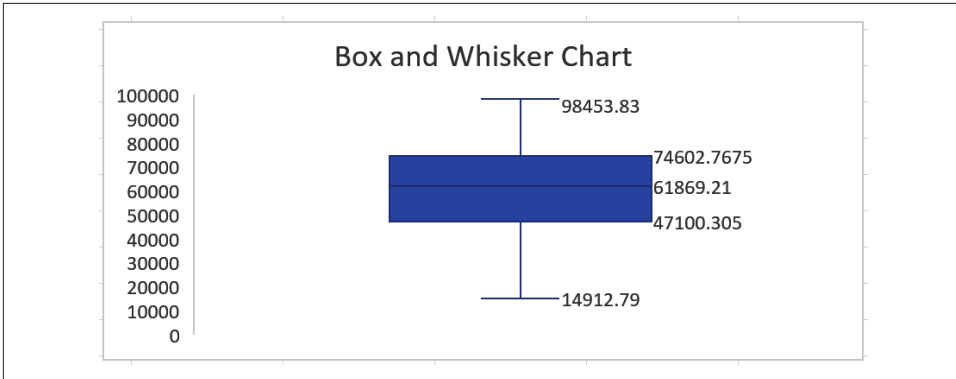
Figure 8-14. A box and whisker chart

A box and whisker chart has the following characteristics:

- A box showing the interquartile range. The bottom of the box shows the first quartile, the top of the box shows the third quartile, and the line in the middle shows the second quartile or median.
- Lines, or whiskers, showing the range, excluding any outliers. The bottom whisker extends below the box and shows the smallest value that's not an outlier, while the top whisker extends above the box and shows the largest value that's not an outlier.
- Optional markers showing the mean. You can display these markers by formatting the chart's data series.
- Optional markers showing outliers, which you can display by formatting the chart's data series.

> The range shown on the box and whisker chart excludes outliers. The chart excludes these outliers irrespective of whether it displays them, so the range shown on the chart may sometimes differ from the actual range.

Discussion

This recipe offers a convenient way of visualizing a dataset's range and interquartile range, along with any outliers.

By default, the box and whisker chart determines the first and third quartiles using the QUARTILE.EXC function, which excludes the median from its calculations. You can change the quartile calculation to include the median by formatting the chart's data series.

The chart classifies values as outliers if they lie more than 1.5 times the interquartile range away from the first or third quartile. See Recipe 8.9 for more details about this method.

8.11 Calculating Skewness

Problem

You have a list of numbers and want to know how asymmetrically they're distributed.

Solution

Suppose you have a list of numbers and want to determine the dataset's degree of asymmetry about the list's mean. You can do so by calculating the *skewness*: a number whose value indicates the degree of asymmetry (see Figure 8-15):

- If the skewness is close to 0, the data is reasonably symmetric about the mean.
- If the skewness is less than –1, there's a high degree of negative (or left) skew. There's a long tail of low values on the left of the distribution.
- If the skewness is greater than 1, there's a high degree of positive (or right) skew. There's a long tail of high values on the right of the distribution.
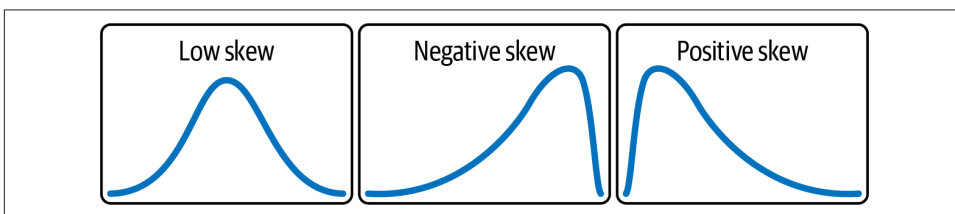
Figure 8-15. Types of skewness

You calculate skewness using the formula `=SKEW(`*numbers*`)`. To determine the skewness of numbers listed in A2:A101, for example, you'd type **=SKEW(A2:A101)**.

> The `SKEW` function calculates skewness using sample data: a subset of data drawn from a population. If your dataset is the entire population, use `SKEW.P` instead.

Discussion

A dataset's skewness can significantly affect some of its other statistics. For example, if a high degree of left skewness exists, this tends to pull the mean to the left so that it's

lower than the median. Conversely, a high degree of right skewness tends to pull the mean to the right so that it's higher than the median.

See Also

You can generate skewness and other descriptive statistics using the Analysis ToolPak; see Recipe 9.2.

8.12 Calculating Probabilities Using a Probability Table

Problem

You have a list of probabilities and want to use them to find the probability of getting a range of outcomes.

Solution

Suppose you have a loaded six-sided die with faces labeled 1 to 6. When you throw the die, the probability of getting a 6 is 0.5, and the probability of getting any other outcome is 0.1.

To work with discrete probabilities like this, you first organize the probabilities in a *probability table*: a table or range of cells that detail each possible outcome and its probability. To create a probability table for the loaded die outcomes, for example, you could list the outcomes in A2:A7 and the corresponding probability for each one in B2:B7 (see Figure 8-16).

	A	B	C	D	E	F
1	Outcome	Probability				
2	1	0.1				
3	2	0.1				
4	3	0.1				
5	4	0.1				
6	5	0.1				
7	6	0.5				

Figure 8-16. A probability table showing probabilities for a loaded die

Once you've created the probability table, you can use it with the PROB function to return the probability of a single outcome or a range of possible outcomes.

To find the probability of a single outcome, you use the formula =PROB(*outcomes, probabilities, outcome*), where *outcomes* is the list of outcomes, *probabilities* is the list of probabilities, and *outcome* is the desired outcome. To return the probability

of an outcome of 3, for example, you'd type **=PROB(A2:A7, B2:B7, 3)**, which returns 0.1.

To find the probability of a range of outcomes, use the formula =PROB(*outcomes, probabilities, lower_limit, upper_limit*), where *outcomes* is the list of outcomes, *probabilities* is the list of probabilities, and *lower_limit* and *upper_limit* are the lower and upper limits of the range of outcomes whose probability you want to find. For example, to find the probability of getting an outcome between 3 and 5 inclusive, you'd type **=PROB(A2:A7, B2:B7, 3, 5)**, which returns 0.3.

You can also use the PROB function to find the probability of not getting an outcome. To do this, subtract the probability of the outcome occurring from 1. For example, to find the probability of not getting an outcome of 6, you'd type **=1-PROB(A2:A7, B2:B7, 6)**, which returns 0.5.

Discussion

A probability table is a helpful way of organizing the probabilities of discrete outcomes, particularly if the outcomes are numeric. For example, in addition to using the PROB function to return probabilities, you can use a chart to plot the probability distribution.

8.13 Calculating Expectation and Variance

Problem

You have a probability table that lists the probability of discrete numeric outcomes and want to find the expected value and amount of variation.

Solution

The *expectation*, or *expected value*, is the outcome you can typically expect. It's the weighted average of the possible outcomes; the value of each outcome is weighted by its probability. You calculate the expectation by using the formula =SUMPRODUCT(*outcomes, probabilities*), where *outcomes* is the list of outcomes, and *probabilities* is the corresponding list of probabilities. If you have a probability table in which A2:A7 lists the outcomes and B2:B7 lists their probabilities, you'd find the expectation by typing =SUMPRODUCT(A2:A7, B2:B7).

The *variance* is the spread of values around the expectation. Calculate the variance using the formula =SUMPRODUCT(*outcomes*^2, *probabilities*)-SUMPRODUCT(*outcomes, probabilities*)^2, where *outcomes* is the list of outcomes and *probabilities* is the corresponding list of probabilities. To find the variance when A2:A7 lists

outcomes and B2:B7 lists their probabilities, for example, you'd type `=SUMPROD`
`UCT(A2:A7^2, B2:B7)-SUMPRODUCT(A2:A7, B2:B7)^2`.

Discussion

The expectation and variance of a probability distribution are like the mean and variance of a dataset. The expectation describes a typical value, and the variance describes the amount of variability around the expectation.

8.14 Using the Binomial Distribution

Problem

You have n independent trials where each trial results in success or failure, and the probability of success remains constant. You want to find the probability that there are x successes.

Solution

Suppose you have a sports team that wins 60% of its games. You want to find the probability that the team wins 15 games or fewer out of the next 20 games.

You can calculate this probability using the *binomial* distribution. This probability distribution applies when you have a fixed number of independent trials, each trial results in success or failure, and the probability of success stays constant. In this situation, for example, there are 20 trials (the number of games), and the probability of success for each trial is 0.6 (because the team wins 60% of its games).

Excel includes three functions you can use to work with the binomial distribution: `BINOM.DIST`, `BINOM.DIST.RANGE`, and `BINOM.INV`.

The formula =BINOM.DIST($x, n, p,$ 1) returns the probability of getting x or fewer successes where n is the number of trials and p is the probability of success. To find the probability that the team wins 15 or fewer games out of 20, for example, you'd type **=BINOM.DIST(15, 20, 0.6, 1)**, which returns 0.949.

To find the probability of exactly x successes, use the formula =BINOM.DIST($x, n, p,$ 0), where the final argument is 0 instead of 1. For example, to find the probability that the team wins exactly 15 out of 20 games, you'd type **=BINOM.DIST(15, 20, 0.6, 0)**, which returns 0.075.

To find the probability of getting between $s1$ and $s2$ successes (inclusive), you use the formula =BINOM.DIST.RANGE($n, p, s1, s2$). To find the probability that the team wins between 10 and 15 games out of 20, for example, you'd type **=BINOM.DIST.RANGE(20, 0.6, 10, 15)**, which returns 0.822.

Finally, you use the formula =BINOM.INV(*n, p, alpha*) to find the lowest value of *x* where the probability of *x* successes is at most greater than or equal to *alpha*. To find how many games the team would need to win at most for the probability of this outcome to be at least 0.2, for example, you'd type **=BINOM.INV(20, 0.6, 0.2)**, which returns 10.

Discussion

Excel offers several solutions for using the binomial probability distribution. Notice that the BINOM.INV function is the inverse of the BINOM.DIST function when used to find the probability of getting *x* or fewer successes.

See Also

Behind the scenes, the binomial distribution derives probabilities by calculating combinations; see Recipe 4.13.

8.15 Using the Negative Binomial Distribution

Problem

You have independent trials that result in success or failure, and the probability of success remains constant. You want to find the probability of getting *f* failures before the *s*th success.

Solution

Suppose you have a sports team that wins 60% of its games. You want to find the probability that it will lose three games before getting its fifth win.

You can solve this problem using the negative binomial distribution. Just like the binomial distribution (see Recipe 8.14), the *negative binomial* distribution applies when you have independent trials that result in success or failure, and the probability of success stays constant. The difference, however, is that the negative binomial distribution models the number of failures before getting the specified number of successes.

To calculate negative binomial probabilities, you can use the NEGBINOM.DIST function.

To find the probability of getting *f* failures before getting the *s*th success, you use the formula =NEGBINOM.DIST(*f, s, p,* 0), where *p* is the probability of success. To find the probability of losing three games before getting the fifth win where the probability of winning is 0.6, for example, you'd type **=NEGBINOM.DIST(3, 5, 0.6, 0)**, which returns 0.174.

To find the probability of getting at most *f* failures before getting the *s*th success, you use the formula =NEGBINOM.DIST(`f, s, p, 1`). To find the probability of losing at most three games before getting the fifth win, for example, you'd type **=NEGBINOM.DIST(3, 5, 0.6, 1)**, which returns 0.594.

Discussion

As you can see, the binomial and negative binomial distributions apply to similar situations. The negative binomial distribution, however, doesn't have a fixed number of trials, and it models the number of failures before getting a certain number of successes.

8.16 Using the Hypergeometric Distribution

Problem

You draw *n* items from a population without putting them back, and each draw results in success or failure. You want to find the probability of getting *x* successes.

Solution

Suppose you have a deck of 52 cards with 13 of each suit. You want to find the probability of getting 4 hearts if you draw 10 cards without putting any back.

You can calculate this probability using the *hypergeometric* distribution. This probability distribution applies when you draw a fixed number of items from a population without putting them back, each item results in success or failure, and the probability of success changes for each item you draw because the population gets smaller. For example, if you draw a heart from a deck of cards, the probability of choosing another heart for the next draw decreases because the deck has fewer hearts.

To calculate hypergeometric probabilities, you can use the HYPGEOM.DIST function.

To find the probability of getting *x* successes when you draw *n* items, and there are *s* successes in the population of size *N*, use the formula =HYPGEOM.DIST(`x, n, s, N, 0`). To find the probability of getting 4 hearts if you draw 10 cards from a deck of 52 cards containing 13 hearts, for example, you'd type **=HYPGEOM.DIST(4, 10, 13, 52, 0)**, which returns 0.147.

To find the probability of getting *x* successes at most when you draw *n* items, you use the formula =HYPGEOM.DIST(`x, n, s, N, 1`) instead. To find the probability of getting at most 4 hearts if you draw 10 cards, for example, you'd type **=HYPGEOM.DIST(4, 10, 13, 52, 1)**, which returns 0.943.

Discussion

The hypergeometric distribution solves similar problems to the binomial distribution (see Recipe 8.14). The main difference is that probability of success remains constant when using the binomial distribution but changes when using the hypergeometric distribution.

8.17 Using the Poisson Distribution

Problem

You have independent events that occur at a constant mean rate and want to find the probability that k such events occur within a given interval.

Solution

Suppose you have a business that receives 20 calls an hour and want to find the probability of getting 25 calls in the next hour.

To solve this problem, you can use the *Poisson* distribution. This probability distribution applies when you know the mean rate at which independent events occur and want to know the probability of getting a certain number of these events in a specified interval.

To calculate Poisson probabilities, use the `POISSON.DIST` function.

To find the probability of k events occurring in an interval, use the formula =POISSON.DIST(*k, mean,* 0), where *mean* is the mean rate at which the event occurs in an interval of that same size. To find the probability of getting 25 calls in the next hour when the mean rate is 20 calls per hour, for example, you'd type **=POISSON.DIST(25, 20, 0)**, which returns 0.045.

To find the probability of k events at most occurring in an interval, you use the formula =POISSON.DIST(*k, mean,* 1). To find the probability of getting at most 25 calls in the next hour when the mean rate is 20 calls per hour, for example, you'd type **=POISSON.DIST(25, 20, 1)**, which returns 0.888.

Discussion

One of the main considerations with the Poisson distribution is to ensure that you use the correct unit for the mean rate. If the mean rate is 20 calls per hour, for example, and you want to find the probability of getting 30 calls at most in the next *2* hours, you'd need to compute the mean rate of calls per 2 hours and type **=POISSON.DIST(30, 2*20, 1)**.

8.18 Using the Exponential Distribution

Problem

You have independent events that occur at a constant mean rate and want to find the probability that the time between events is a given interval or less.

Solution

Suppose you have a business that receives a mean of 20 calls an hour and want to find the probability that there's 1 minute or less between calls.

To solve this problem, you can use the *exponential* distribution. This probability distribution applies when you know the mean rate at which independent events occur and want to know the probability that the time between events is a specified interval or less.

To calculate exponential probabilities, you can use the `EXPON.DIST` function.

To find the probability of an interval of x or less between events, you use the formula `=EXPON.DIST(x, mean, 1)`, where `mean` is the mean rate at which the event occurs in an interval of that same size. To find the probability that there's 1 minute or less between calls when the mean rate is 20 calls per hour, for example, you'd type `=EXPON.DIST(1, 20/60, 1)`, which returns 0.283. In this example, you use `20/60` for the second argument to specify the mean rate of calls per minute. Similarly, to find the probability of there being more than 1 minute between calls, you'd type `=1-EXPON.DIST(1, 20/60, 1)`, which returns 0.717.

Discussion

A strong relationship exists between the Poisson and exponential distributions since both distributions deal with independent events that occur at a constant mean rate. The difference is that the Poisson distribution models the probability of k events occurring per interval, while the exponential distribution models the time between events (see Recipe 8.17).

As with the Poisson distribution, one of the primary considerations is to ensure that you use the same unit for the interval and mean rate. For example, if you specify the interval in minutes, you must pass the `EXPON.DIST` function the mean rate in minutes.

8.19 Using the Normal Distribution

Problem

You want to find probabilities and percentiles for normally distributed data.

Solution

The *normal* distribution is a continuous probability distribution with a bell-shaped curve that's symmetrical around its mean (see Figure 8-17). Most observations are clustered around the mean and taper off away from it. Many phenomena—for example, height, weight, measurement error, and IQ scores—follow a normal distribution.
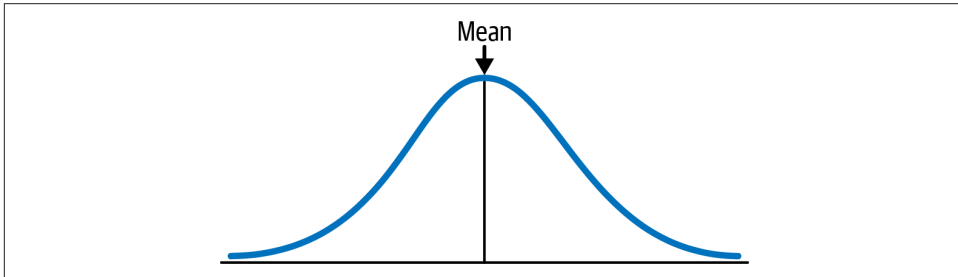
Figure 8-17. The shape of the normal distribution

You use two main functions to work with the normal distribution: `NORM.DIST`, and `NORM.INV`.

The formula =NORM.DIST(*x, mean, standard_deviation,* 1) computes the probability of getting a value of *x* or less from a normal distribution with the specified mean and standard deviation. So if, for example, the weights of bags of flour follow a normal distribution with a mean of 5 pounds and a standard deviation of 0.2 pounds, you'd find the probability that a bag of flour chosen at random weighs 4.8 pounds or less by typing **=NORM.DIST(4.8, 5, 0.2, 1)**, which returns 0.159.

You can also use the `NORM.DlST` function to find the probability of getting a range of values. To calculate the probability that a bag of flour selected at random weighs between 4.8 and 5.2 pounds, for example, you'd use the formula =NORM.DIST(5.2, 5, 0.2, 1)-NORM.DIST(4.8, 5, 0.2, 1), which returns 0.683. This formula first finds the probability that the bag weighs 5.2 pounds at most and then subtracts the probability of weighing 4.8 pounds at most.

The formula =NORM.INV(*p, mean, standard_deviation*) is the inverse of the `NORM.DIST` function; it returns the number *x* for which the probability of getting *x* or less is *p*. To find the weight of a random bag of flour where the probability of getting that weight at most is 0.25, for example, you'd type **=NORM.INV(0.25, 5, 0.2)**, which returns 4.865. Notice that you can also use this formula to find percentiles. To find the 10th percentile, for example, you'd type **=NORM.INV(0.1, 5, 0.2)**, which returns 4.744.

The `NORM.INV` function also lets you generate a list of random numbers drawn from a normal distribution. Type the formula **=NORM.INV(RAND(), *mean, standard_ deviation*)** in each cell, specifying the distribution's mean and standard deviation.

> Excel also has `NORM.S.DIST` and `NORM.S.INV` functions. These are special cases of the `NORM.DIST` and `NORM.INV` functions where the mean is 0 and the standard deviation is 1 (see Recipe 8.20).

Discussion

The normal distribution is one of the most important in statistics. Many natural phenomena follow this distribution, and you can use it to approximate other distributions in some circumstances.

Another use for the normal distribution relates to sampling: if you draw large samples—usually greater than or equal to 30—from a population, the means of those samples will approximately follow a normal distribution. This concept comes from the central limit theorem, and it applies even if the underlying population doesn't follow a normal distribution.

See Also

You can also generate random numbers drawn from a normal distribution using the Analysis ToolPak; see Recipe 9.9.

8.20 Using Z-Scores

Problem

You want to compare the relative position of values drawn from different normal distributions.

Solution

Suppose you have two sales teams that work in different areas. The sales made per person in Team A are normally distributed with a mean of $62,000 and a standard deviation of $15,000; the sales made per person for Team B are also normally distributed with a mean of $5,100 and a standard deviation of $2,100. You want to know whether a salesperson from Team A who made $105,000 performed better, relative to the rest of their team, than a salesperson from Team B who made $11,800.

You can solve this problem by comparing each salesperson's *z-score*, or *standard score*: a standardized version of a data point that uses standard deviations to measure its distance from the mean. Since z-scores are standardized on a common scale, you can use them to analyze the relative position of values from different datasets. If the salesperson from Team A has a higher z-score than the one from Team B, for example, it means that Team A's salesperson performed better relative to the rest of their team.

You can convert a value x to a z-score using the formula `=STANDARDIZE(`x`, mean, standard_deviation)`, specifying the mean and standard deviation. To convert the Team A salesperson's sales figure to a z-score, for example, you'd type `=STANDARDIZE(105000, 62000, 15000)`, which returns 2.867. Similarly, you'd convert the Team B salesperson's sales figure to a z-score by typing `=STANDARD IZE(11800, 5100, 2100)`, which returns 3.19. Since the Team B salesperson has the higher z-score, they performed better, relative to the rest of their team, than the salesperson from Team A.

Discussion

Standardization transforms the points in a normal dataset to follow a normal distribution with a mean of 0 and a standard deviation of 1. A point's z-score is its number of standard deviations distance from the mean. A population follows a normal distribution if it has:

- Approximately 68% of its data points within 1 standard deviation of the mean
- Approximately 95% of its data points within 2 standard deviations of the mean
- Approximately 98% of its data points within 3 standard deviations of the mean

If you know a data point's z-score, you can use the `NORM.S.DIST` function to find the probability of selecting a value with that z-score at most (see Recipe 8.19). You can also use the `NORM.S.INV` to find the z-score associated with a given probability.

8.21 Calculating a Confidence Interval for the Population Mean

Problem

You've drawn a sample from a population and want to construct a confidence interval for the population mean.

Solution

Suppose you've drawn a sample from a population and want to use it to estimate the population mean. In this situation, you can either use the sample mean or construct a

confidence interval: a range of values that you're confident contains the population mean. The confidence interval for the population mean is the sample mean ± a confidence statistic: a number that describes the variation in your estimate with a certain confidence level. For example, a confidence level of 95% means there's a probability of 95% that the confidence interval includes the population mean.

When you know the population's standard deviation, you calculate the confidence statistic using the formula =CONFIDENCE.NORM(*alpha, standard_deviation, sample _size*), where *standard_deviation* is the population's standard deviation, *sample _size* is the size of the sample, and *alpha* is 1 – the confidence level. So if the confidence level is 95% or 0.95, *alpha* is 0.05.

For example, suppose you have sample data in A2:A101 drawn from a population with a known standard deviation of 20. To calculate a 95% confidence interval for the population mean, you'd find its lower limit by typing **=AVERAGE(A2:A101)- CONFIDENCE.NORM(0.05, 20, 100)** and its upper limit by typing **=AVERAGE (A2:A101)+CONFIDENCE.NORM(0.05, 20, 100)**.

When you don't know the population's standard deviation, which is usually the case, you calculate the confidence statistic using the formula =CONFIDENCE.T(*alpha, standard_deviation, sample_size*) instead, where *standard_deviation* is the standard deviation of the sample. To calculate a 95% confidence interval for the population mean from the sample data listed in A2:A101 when you don't know the population's standard deviation, for example, you'd type **=CONFIDENCE.T(0.05, STDEV.S (A2:A101), 100)** to compute the confidence statistic; the 95% confidence interval is =AVERAGE(A2:A101) ± the confidence statistic.

Discussion

The function you use to calculate a confidence interval for the population mean depends on whether you know the value of the population's standard deviation. You usually don't know what this value is, so you use the CONFIDENCE.T function. This function creates a wider confidence interval than using CONFIDENCE.NORM to reflect the extra level of uncertainty.

Behind the scenes, the CONFIDENCE.NORM function calculates the confidence statistic using the normal distribution, and the CONFIDENCE.T function calculates it using the t-distribution. You can work with these distributions directly using the NORM.DIST (see Recipe 8.19) and T.DIST functions.

See Also

You can also use the Analysis ToolPak to calculate a confidence interval for the population mean; see Recipe 9.2.

8.22 Performing a Chi-Squared (χ^2) Test for Independence

Problem

You have frequencies for two categorical variables and want to know if they're related.

Solution

Suppose you want to test whether good or bad weather conditions affect the outcome of a sports team's games. You've gathered data for 50 games listed in B2:C4, with labels in column A and row 1 and totals in column D and row 5 (see Figure 8-18).

	A	B	C	D
1	Observed	Good	Bad	Total
2	Win	11	3	=SUM(B2:C2)
3	Draw	6	7	=SUM(B3:C3)
4	Lose	9	14	=SUM(B4:C4)
5	Total	=SUM(B2:B4)	=SUM(C2:C4)	=SUM(D2:D4)
6				
7	Expected	Good	Bad	Total
8	Win	=D2*B5/$D$5	=D2*C5/$D$5	=SUM(B8:C8)
9	Draw	=D3*B5/$D$5	=D3*C5/$D$5	=SUM(B9:C9)
10	Lose	=D4*B5/$D$5	=D4*C5/$D$5	=SUM(B10:C10)
11	Total	=SUM(B8:B10)	=SUM(C8:C10)	=SUM(D8:D10)
12				
13	Chi Square Test	=CHISQ.TEST(B2:C4,B8:C10)		
14		CHISQ.TEST(actual_range, expected_range)		

Figure 8-18. The data and formulas for the χ^2 test for independence

You can solve this problem using a χ^2 test for independence.

1. Decide on a significance level for the test—for example, 0.05.

2. Create a list of the values you'd expect to get for each match outcome in each weather condition. You can use the observed match results to calculate this. To find the number of games you'd expect the team to win in good weather, for example, you multiply the number of games won (14) by the number played in good weather (26) and divide the result by the total number of games (50). I've listed the formulas for each expected outcome in B8:C10 (Figure 8-18).

3. To perform the χ^2 test for independence, type the formula **=CHISQ.TEST(B2:C4,B8:C10)**, where B2:C4 is the list of observed outcomes and B8:C10 is the list of expected outcomes; this returns 0.059.

If the result of the χ^2 test is less than the significance level you chose in step 1, there's enough evidence to conclude that the weather conditions affect the match outcome. If the result of the χ^2 test is greater than the significance level, as in this example, then there's insufficient evidence to draw this conclusion.

Discussion

This recipe is helpful when you want to find the likelihood that two variables are related. It tests the null hypothesis that the variables are independent against the alternate hypothesis that they're not.

In addition to performing a χ^2 test for independence, you can work directly with the χ^2 distribution's left and right tails using the `CHISQ.DIST` and `CHISQ.DIST.RT` functions. Excel also includes the inverses of these functions: `CHISQ.INV` and `CHISQ.INV.RT`.

See Also

You can perform other hypothesis tests using the Analysis ToolPak; see Recipes 9.13 through 9.17.

8.23 Finding the Line of Best Fit

Problem

You have pairs of values for two variables and want to find the type of line that best fits the data points.

Solution

Suppose you have pairs of values for two variables, and you want to find the type of relationship—if any—that exists between them. One column holds the values for the first variable, and another holds the values for the second (see Figure 8-19):

1. Select the two columns and insert a scatter chart, which you can find by choosing Insert ⇒ Charts ⇒ Insert Scatter (X, Y) or Bubble Chart.
2. Choose Chart Design ⇒ Add Chart Element ⇒ Trendline ⇒ Linear to add a linear trendline.
3. Double-click the trendline to open its Format pane and place a check in the "Display R-squared value on chart" check box: this displays the value of R^2 on the chart—a number between 0 and 1 that indicates how reliably the trendline fits the data points.

4. Change the trendline's type in the Format pane by selecting the Exponential, Linear, Logarithmic, Polynomial, and Power options, and assess which line best fits the data points. You can do this by eye or choosing the one where R^2 is closest to 1.
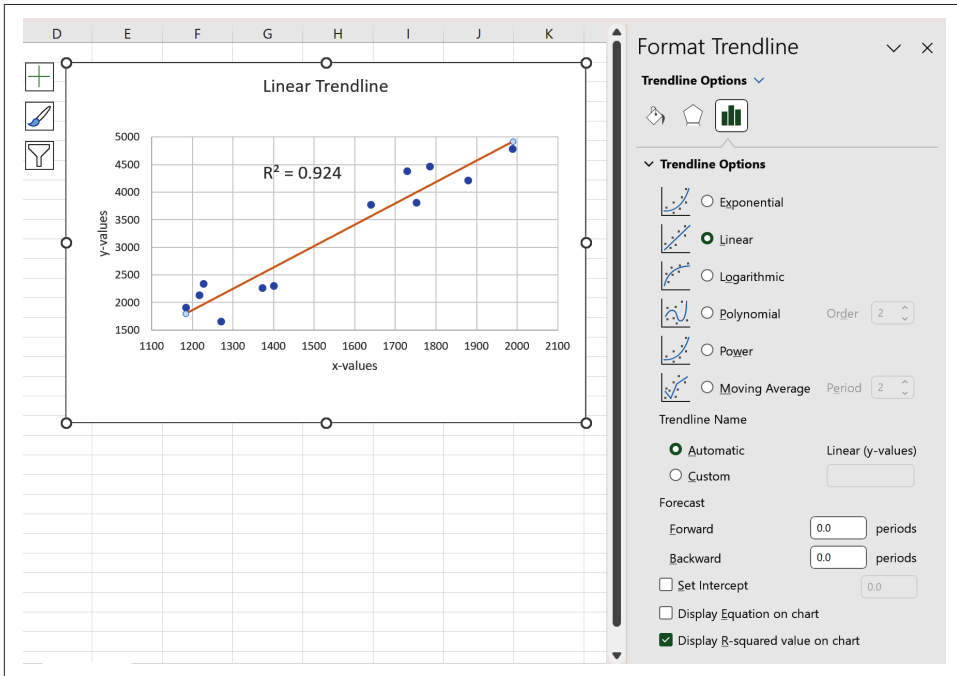
Figure 8-19. Formatting a trendline in the Format pane

Discussion

This recipe is a helpful way of assessing which type of line best fits the data points. This approach uses R^2, which shows how much of the variation is explained by the relationship between the two. For linear trendlines, R^2 is the square of the correlation coefficient (see Recipe 9.10).

See Also

Excel also has a Moving Average trendline; see Recipe 9.5 for more information.

8.24 Getting the Line of Best Fit's Equation

Problem

You have pairs of values and want to find the equation of the line that best fits the data points.

Solution

If there are two variables, the simplest way of getting the line's equation is to add it to the scatter chart. To do this, follow Recipe 8.23, and in step 3, place an additional check in the "Display Equation on chart" check box.

> The chart's equation is accurate only when used with a scatter chart, so don't use it with other charts. Additionally, it shows very few significant figures by default; you can change this by selecting the trendline's equation and using the Format pane to increase the number of decimal places.

An alternative technique is to use formulas to calculate the line's coefficients. This approach works if there are two or more variables.

For a linear equation that takes the form $y = ax + b$, you can find a using the formula =SLOPE(`y_values, x_values`) and b using =INTERCEPT(`y_values, x_values`). If A2:A20 lists the y-values and B2:B20 lists the x-values, for example, you'd find a by typing **=SLOPE(A2:A20, B2:B20)** and b by typing **=INTERCEPT(A2:A20, B2:B20)**.

> If the coefficients you calculate using formulas differ from those shown on the scatter chart's equation, it might be because you're using different columns for your x- and y-values. By default, the chart uses the first column for its x-values and the second for its y-values.

If you have two x variables so that the linear equation takes the form $y = a_1x_1 + a_2x_2 + b$, you can calculate the coefficients using the LINEST function. Find a_1 using =INDEX (LINEST(`y_values, x_values`), 1, 1), a_2 using =INDEX(LINEST(`y_values, x_values`), 1, 2), and b using =INDEX(LINEST(`y_values, x_values`), 1, 3). If A2:A20 lists the y-values and B2:C20 lists the x-values, for example, you'd find a_1 by typing **=INDEX(LINEST(A2:A20, B2:C20), 1, 1)**.

For an exponential equation that takes the form $y = ae^{bx}$, you can find a using =EXP(INDEX(LINEST(LN(`y_values`), `x_values`), 1, 2)) and b using =INDEX (LINEST(LN(`y_values`), `x_values`), 1, 1). If A2:A20 lists the y-values and

B2:B20 lists the x-values, for example, you'd find *a* by typing **=EXP(INDEX (LINEST(LN(A2:A20), B2:B20), 1, 2))**.

For a logarithmic equation that takes the form $y = a\ln(x) + b$, you can find *a* using =INDEX(LINEST(*y_values*, LN(*x_values*)), 1, 1) and *b* using =INDEX(LINEST (*y_values*, LN(*x_values*)), 1, 2). If A2:A20 lists the y-values and B2:B20 lists the x-values, for example, you'd find *a* by typing **=INDEX(LINEST(A2:A20, LN(B2:B20)), 1, 1)**.

For a second-order polynomial equation that takes the form $y = a_1x^2 + a_2x + b$, you can find a_1 using =INDEX (LINEST(*y_values*, *x_values*^{1,2}), 1, 1), a_2 using =INDEX(LINEST(*y_values*, *x_values*^{1,2}), 1, 2), and *b* using =INDEX (LINEST(*y_values*, *x_values*^{1,2}), 1, 3). If A2:A20 lists the y-values and B2:B20 lists the x-values, for example, you'd find a_1 by typing **=INDEX (LINEST(A2:A20, B2:B20^{1,2}), 1, 1)**.

Similarly, for a third-order polynomial equation that takes the form $y = a_1x^3 + a_2x^2 + a_3x + b$, you can find a_1 using =INDEX(LINEST(*y_values*, *x_values*^{1,2,3}), 1, 1), a_2 using =INDEX(LINEST(*y_values*, *x_values*^{1,2,3}), 1, 2), a_3 using =INDEX (LINEST(*y_values*, *x_values*^{1,2,3}), 1, 3), and *b* using =INDEX(LINEST(*y_values*, *x_values*^{1,2,3}), 1, 4).

Finally, for a power equation that takes the form $y = ax^b$, you can find *a* using =EXP(INDEX(LINEST(LN(*y_values*), LN(*x_values*)), 1, 2)) and *b* using =INDEX (LINEST(LN(*y_values*), LN(*x_values*)), 1, 1).

Discussion

The simplest way of finding the line of best fit's equation is to display it on a scatter chart. However, if this isn't possible, you can use the LINEST function to calculate the line's coefficients instead.

See Also

An alternative approach is to use the Analysis ToolPak's Regression tool, which, behind the scenes, uses the LINEST function; see Recipe 9.12.

If you want to estimate future values for linear or exponential data, see Recipe 10.18. See Recipe 10.19 for ways of modeling seasonal or periodic data.

The Analysis ToolPak

As you've seen in Chapter 8, Excel offers a wealth of functions you can use for statistical analysis. Using these, however, can sometimes be time-consuming.

An alternative approach is to use Excel's Analysis ToolPak, which is available for Windows and Mac. This add-in provides data analysis tools that address statistical and engineering tasks such as generating statistics, performing hypothesis tests, and even running Fourier analyses.

This chapter guides you through installing the Analysis ToolPak and using each tool it provides.

9.1 Installing the Analysis ToolPak

Problem

You have some data you want to analyze, and you want to install a suite of tools to help you.

Solution

Load the Analysis ToolPak add-in to Excel. Once it's loaded, you can access the ToolPak from the Data menu by going to the Analyze group and choosing Data Analysis.

To load the Analysis ToolPak in Excel for Windows:

1. Choose File ⇒ Options and select Add-ins.
2. In the Manage box at the bottom of the screen, choose the Excel Add-ins option and click Go.

3. In the Add-ins dialog box, place a check in the Analysis ToolPak check box and click OK.

To load the Analysis ToolPak in Excel for Mac, choose Tools ⇒ Excel add-ins to open the Add-ins dialog box, place a check in the Analysis ToolPak check box, and click OK.

Discussion

The Analysis ToolPak brings extra features to Excel that help make statistical and engineering analyses less time-consuming. This add-in includes tools to generate statistics, generate random numbers and samples, perform hypothesis tests, and more. See the recipes in the rest of this chapter for details on how to use each tool.

9.2 Generating Descriptive Statistics

Problem

You have sample data and want to generate standard statistics from it.

Solution

Suppose you want to generate standard descriptive statistics for the scores of 15 students. A1:A16 lists the scores, including a label in row 1 (see Figure 9-1):

1. Go to the Data menu and choose Data Analysis from the Analyze group. Then select Descriptive Statistics from the list of analysis tools, and click OK to open the tool's dialog box.

2. Choose A1:A16 in the Input Range box for the list of scores, including the label in A1.

3. Choose the Columns option in the Grouped By section because the scores are listed in a column.

4. Place a check in the "Labels in first row" check box because the first row of the input range contains the data's label.

5. Choose one of the "Output options" to specify where you want the Descriptive Statistics tool to output the results (for example, the Output Range C1).

6. Place a check in the Summary Statistics check box.

7. Optionally, place a check in the Confidence Level for Mean check box and enter a confidence level to output the confidence statistic for the population mean. For example, to generate the 95% confidence statistic, you'd type **95**. See Recipe 8.21 for more details of this statistic.

8. Optionally, place checks in the Kth Largest and Kth Smallest check boxes if you want to include these statistics in the output, and type a value for *K*.
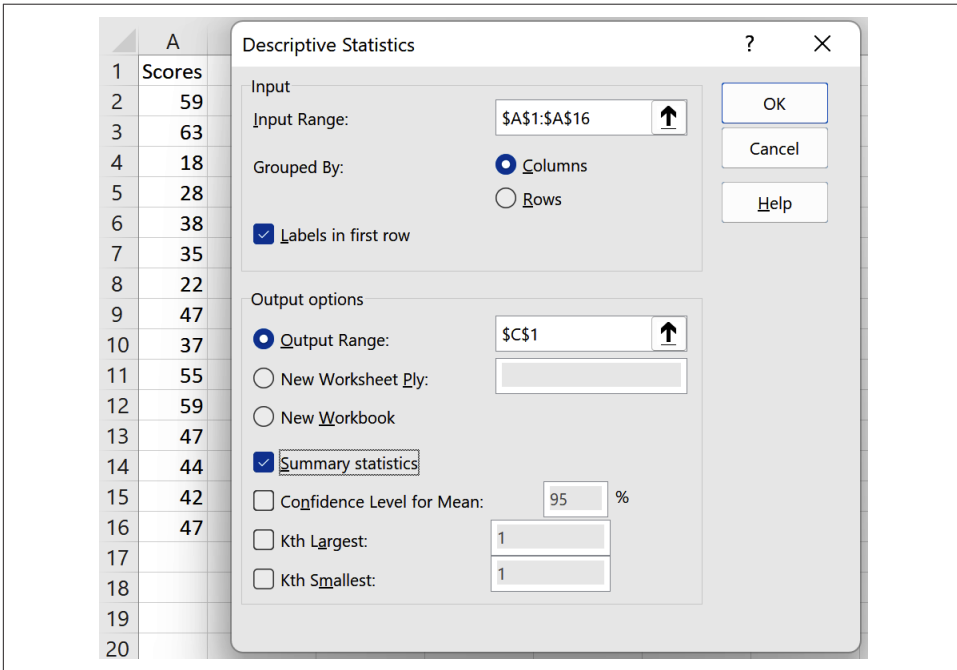
Figure 9-1. The Descriptive Statistics tool's dialog box

When you click OK, the Descriptive Statistics tool generates descriptive statistics for the data (see Figure 9-2).

Figure 9-2. The Descriptive Statistics tool's output

Discussion

The Descriptive Statistics tool offers a quick way of generating standard statistics for a sample dataset. These include measures of central tendency (such as the mean, median, and mode) and measures of variability (such as the sample variance, standard deviation, and skew). You can generate many of these statistics using the recipes in Chapter 8.

This recipe uses the Descriptive Statistics tool to generate statistics for a single dataset. However, you can also generate statistics for multiple datasets at a time by listing each in a separate column and choosing an Input Range that includes the entire range (see Figure 9-3).

Scores A	Scores B		*Scores A*		*Scores B*	
59	76					
63	93		Mean	42.73333	Mean	63.46667
18	28		Standard Error	3.464743	Standard Error	4.675027
28	62		Median	44	Median	62
38	57		Mode	47	Mode	82
35	78		Standard Deviation	13.41889	Standard Deviation	18.1063
22	77		Sample Variance	180.0667	Sample Variance	327.8381
47	82		Kurtosis	-0.571	Kurtosis	-0.52724
37	82		Skewness	-0.30647	Skewness	-0.30691
55	42		Range	45	Range	65
59	66		Minimum	18	Minimum	28
47	53		Maximum	63	Maximum	93
44	54		Sum	641	Sum	952
42	61		Count	15	Count	15
47	41					

Figure 9-3. The Descriptive Statistics tool's output for multiple datasets

Notice that the Descriptive Statistics tool outputs results as values instead of formulas, so you'll need to rerun the tool if the underlying data changes.

9.3 Generating Ordinal and Percentage Rank Statistics

Problem

You have a set of numbers and want to find the rank of each one when sorted in descending order.

Solution

Suppose you have a set of scores from 15 students, and you want to analyze the relative standing of each one. A1:A16 lists the scores, including a label in row 1 (see Figure 9-4):

1. Go to the Data menu and choose Data Analysis from the Analyze group. Then select Rank and Percentile from the list of analysis tools, and click OK to open the tool's dialog box.

2. Choose A1:A16 in the Input Range box for the list of scores, including the label in A1.

3. Choose the Columns option in the Grouped By section because the scores are listed in a column.

4. Place a check in the "Labels in first row" check box because the first row of the input range contains the data's label.

5. Choose one of the "Output options" to specify where you want the Rank and Percentile tool to output the results (for example, the Output Range C1).
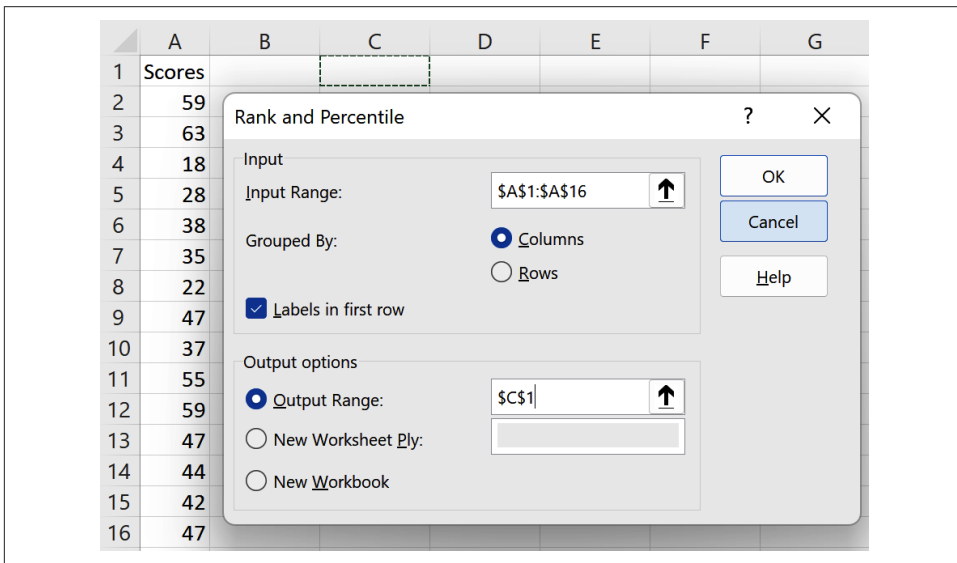

Figure 9-4. The Rank and Percentile tool's dialog box

When you click OK, the Rank and Percentile tool outputs the scores in descending order, along with each one's rank, percentage rank, and position in the original dataset (see Figure 9-5).

Scores		Point	Scores	Rank	Percent	
59		2	63	1	100.00%	
63		1	59	2	85.70%	
18		11	59	2	85.70%	
28		10	55	4	78.50%	
38		8	47	5	57.10%	
35		12	47	5	57.10%	
22		15	47	5	57.10%	
47		13	44	8	50.00%	
37		14	42	9	42.80%	
55		5	38	10	35.70%	
59		9	37	11	28.50%	
47		6	35	12	21.40%	
44		4	28	13	14.20%	
42		7	22	14	7.10%	
47		3	18	15	0.00%	

Figure 9-5. The Rank and Percentile tool's output

Discussion

The Rank and Percentile tool uses the `RANK.EQ` and `PERCENTRANK.INC` functions to return the ordinal and percentage rank of each value in the input range. See Recipe 8.5 to learn more about these functions.

9.4 Generating a Frequency Distribution

Problem

You have a set of numeric data and want to determine how values are distributed by dividing them into intervals and counting how many are in each interval. You optionally want to show a running total for the percentage of values for each bin and display the results on a chart.

Solution

Suppose you have a set of scores for 15 students. You want to group the data into bins and count how many values occur in each bin (the frequency). A2:A16 lists the scores, C2:C11 lists the upper limits for each bin, and the first row contains labels (see Figure 9-6):

1. Go to the Data menu and choose Data Analysis from the Analyze group. Then select Histogram from the list of analysis tools, and click OK to open the tool's dialog box.

2. Choose A1:A16 in the Input Range box for the list of scores, including the label in A1.

3. Choose C1:C11 in the Bin Range box for the list of bin upper limits, including the label in C1.

4. Place a check in the Labels check box because the first row of the input and bin ranges contain labels.

5. Choose one of the "Output options" to specify where you want the Histogram tool to output the results (for example, the Output Range E1).

6. Place a check in the Pareto (sorted histogram) check box (optional) to show extra columns with the bins sorted by frequency.

7. Place a check in the Cumulative Percentage check box (optional) to output a running total for the percentage of values held in each bin.

8. Place a check in the Chart Output check box (optional) to output a histogram or Pareto chart (depending on the other selected options).
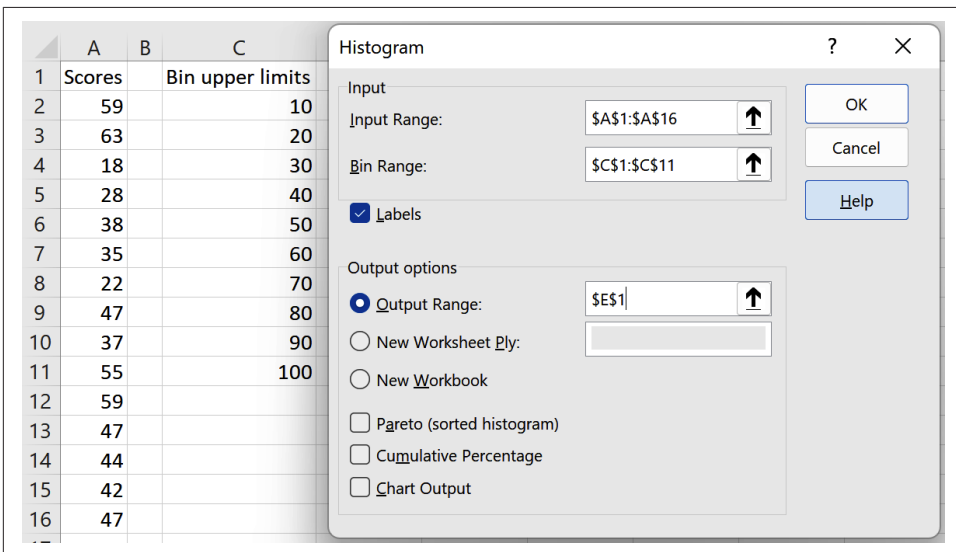
Figure 9-6. The Histogram tool's dialog box

When you click OK, the Histogram tool outputs a table showing the bin upper limits and the frequency of each one. Depending on your selected options, the table may include the cumulative percentage and sort the results by bin frequency (see Figure 9-7). If you selected the Chart Output option, the tool also generates an additional histogram or Pareto chart.

Bin upper limits	Frequency	Cumulative %	Bin upper limits	Frequency	Cumulative %
10	0	0.00%	50	5	33.33%
20	1	6.67%	40	3	53.33%
30	2	20.00%	60	3	73.33%
40	3	40.00%	30	2	86.67%
50	5	73.33%	20	1	93.33%
60	3	93.33%	70	1	100.00%
70	1	100.00%	10	0	100.00%
80	0	100.00%	80	0	100.00%
90	0	100.00%	90	0	100.00%
100	0	100.00%	100	0	100.00%
More	0	100.00%	More	0	100.00%

Figure 9-7. The Histogram tool's output

Discussion

This recipe is a convenient way of generating various statistics relating to frequencies. The primary output shows the frequency of each bin you specify. If you don't provide the tool with a list of bin limits, it will choose them for you.

The cumulative percentage option shows the total percentages for each bin and its predecessors. For example, the output on the left of Figure 9-7 shows that 20% of the score values are in the first 3 bins with upper limits of 10, 20, and 30. If you select the Pareto (sorted histogram) option, the results also show the cumulative percentage sorted by bin frequency. For example, the output shown on the right of Figure 9-7 shows that 53.33% of the score values are in the bins with upper limits of 50 and 40.

Notice that the Histogram tool outputs results as values instead of formulas, so you'll need to rerun the tool if the underlying data changes.

See Also

See Recipes 8.1, 8.2, and 8.3 for other techniques you can use to generate frequencies.

> If you use the Pareto (Sorted Histogram) option, you must ensure that the list of bin upper limits contains values instead of formulas. If it uses formulas, you may encounter errors.

9.5 Generating Moving Averages

Problem

You have a set of data with seasonal variability and want to analyze the general trend.

Solution

Suppose you have sales data indicating that revenue tends to be lower for the third quarter of each year, and you want to analyze the general trend. A2:A20 lists the year and quarter, B2:B20 lists the sales values, and the first row contains labels (see Figure 9-8):

1. Go to the Data menu and choose Data Analysis from the Analyze group. Then select Moving Average from the list of analysis tools, and click OK to open the tool's dialog box.

2. Choose B1:B20 in the Input Range box for the list of sales values, including the label in B1.

3. Place a check in the "Labels in first row" check box because the first row of the input range contains the data's label.

4. Type **4** in the Interval box because the data lists sales for each quarter, and the seasonal pattern repeats every four rows.

5. Choose an Output Range of C2 so that the moving average value for each row is output next to the original sales data.

6. Place a check in the Chart Output check box if you want to generate a line chart with a moving average trendline (optional).

7. Place a check in the Standard Errors check box if you want to generate standard error alongside the moving averages (optional). This option shows the degree of variability between the actual values and the moving averages.
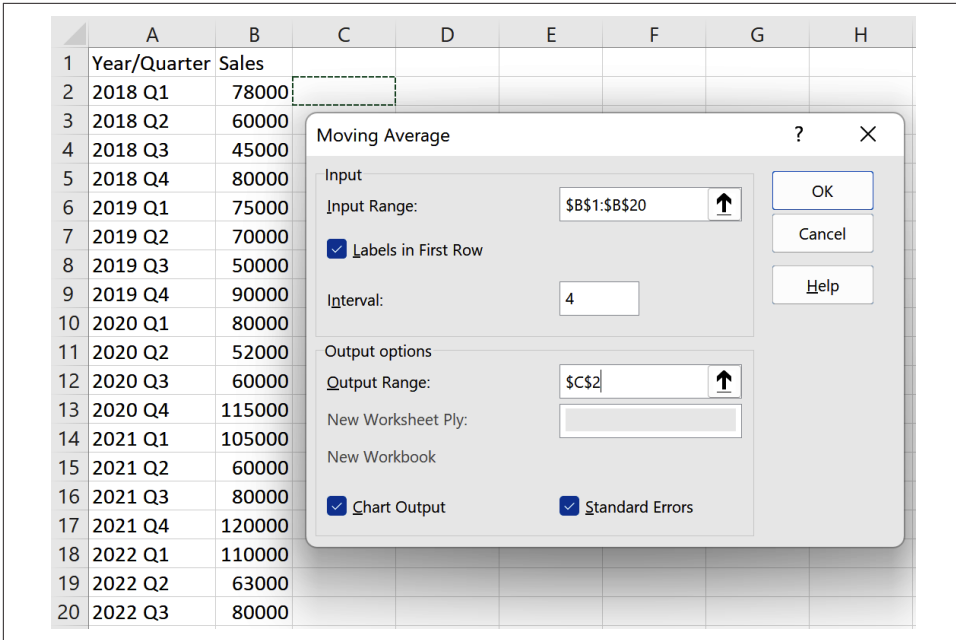
Figure 9-8. *The Moving Average tool's dialog box*

When you click OK, the Moving Average tool outputs the moving average values for the data, along with a chart and the standard errors if you selected these options (see Figure 9-9).
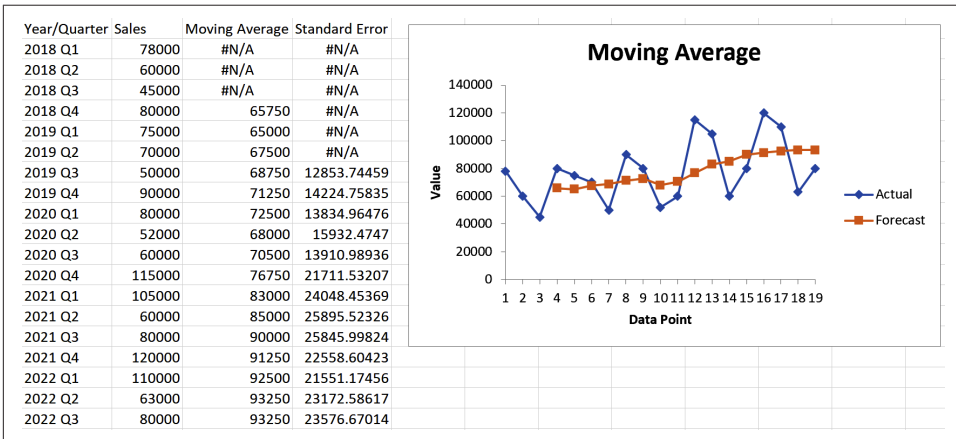
Figure 9-9. *The Moving Average tool's output[1]*

---

1 Extra Moving Average and Standard Error labels have been added to the first row.

Discussion

A *moving average* is a series of averages that "moves" with the data. The interval defines how many data points it uses for these averages. In this recipe, the example uses an interval of 4, which generates the average of four data points: the current data point and the three preceding ones (see Figure 9-10). Using a four-quarter moving average smooths out seasonal variability between quarters because each average contains one data point from each quarter.

	A	B	C
1	Year/Quarter	Sales	Moving Average
2	2018 Q1	78000	#N/A
3	2018 Q2	60000	#N/A
4	2018 Q3	45000	#N/A
5	2018 Q4	80000	=AVERAGE(B2:B5)
6	2019 Q1	75000	=AVERAGE(B3:B6)
19	2022 Q2	63000	=AVERAGE(B16:B19)
20	2022 Q3	80000	=AVERAGE(B17:B20)

Figure 9-10. Formulas generated by the Moving Average tool

9.6 Using Exponential Smoothing

Problem

You have a time series with irregular peaks and troughs and want to smooth out these irregularities to see the general trend.

Solution

Suppose you have a set of sales data for 12 periods that have irregular peaks and troughs, and you want to analyze the general trend. A2:A13 lists the periods, B2:B13 lists the sales, and the first row contains labels (see Figure 9-11):

1. Go to the Data menu and choose Data Analysis from the Analyze group. Then select Exponential Smoothing from the list of analysis tools, and click OK to open the tool's dialog box.

2. Choose B1:B13 in the Input Range box for the list of sales values, including the label in B1.

3. Type a number between 0 and 1 in the Damping Factor box—for example, **0.7** (see "Discussion" on page 209).

4. Place a check in the Labels check box because the first row of the input range contains the data's label.

5. Choose an Output Range of C2 so that the value generated for each row is output next to the original sales data.

6. Place a check in the Chart Output check box if you want to generate a chart showing lines for the actual data and the exponential smoothing output (optional).

7. Place a check in the Standard Errors check box if you want to generate standard error values (optional). These show the degree of variability between the actual and predicted values.
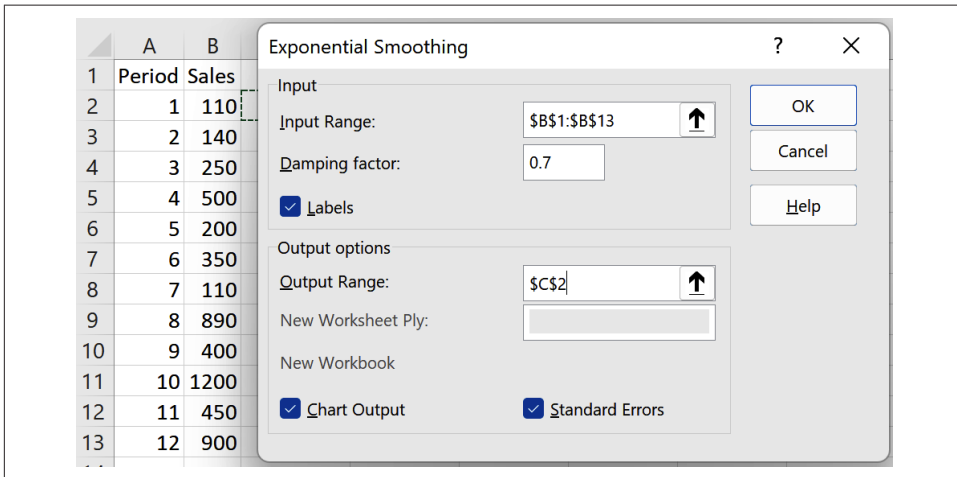
Figure 9-11. The Exponential Smoothing tool's dialog box

When you click OK, the Exponential Smoothing tool outputs exponential smoothing values for the data, along with a chart and the standard errors if you selected these options (see Figure 9-12).
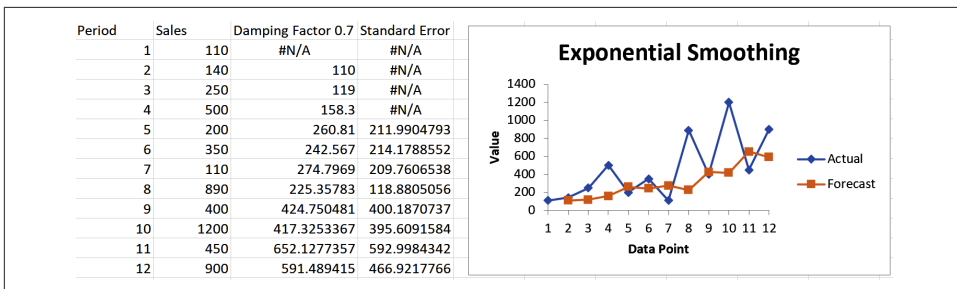
Figure 9-12. The Exponential Smoothing tool's output[2]

---

2 Extra Damping Factor and Standard Error labels have been added to the first row.

Discussion

Exponential smoothing predicts values based on the previous data point and its predicted value.

A *damping factor* is a number between 0 and 1 that adjusts the weighting of the two values. Using a damping factor of 0.7, for example, multiplies the previous data point by 0.3, multiples its previously predicted value by 0.7, and then returns the sum of these results (see Figure 9-13). Values of 0.7 and 0.8 are reasonable damping factors because they adjust the predicted values by 20% to 30% for error.

	A	B	C	
1	Period	Sales	Damping Factor 0.7	
2	1	110	#N/A	
3	2	140	=B2	
4	3	250	=0.3*B3+0.7*C3	
5	4	500	=0.3*B4+0.7*C4	
12	11	450	=0.3*B11+0.7*C11	
13	12	900	=0.3*B12+0.7*C12	

Figure 9-13. Formulas generated by the Exponential Smoothing tool

9.7 Generating a Random Sample

Problem

You have a dataset and want to generate a random sample from it.

Solution

Suppose you have a set of customers, and you want to generate a random sample from this data. A1:E201 lists the data: the first column includes a unique number for each customer, and the first row contains labels (see Figure 9-14):

1. Go to the Data menu and choose Data Analysis from the Analyze group. Then select Sampling from the list of analysis tools, and click OK to open the tool's dialog box.

2. Choose A1:A201 in the Input Range box for the list of unique customer numbers, including the label in A1.

3. Place a check in the Labels check box because the first row of the input range contains the data's label.

4. Choose a Sampling Method of Random.

5. Type the sample size you want to generate in the Number of Samples box (for example, **10**).

6. Choose one of the "Output options" to specify where you want the Sampling tool to output the results (for example, the Output Range G2).
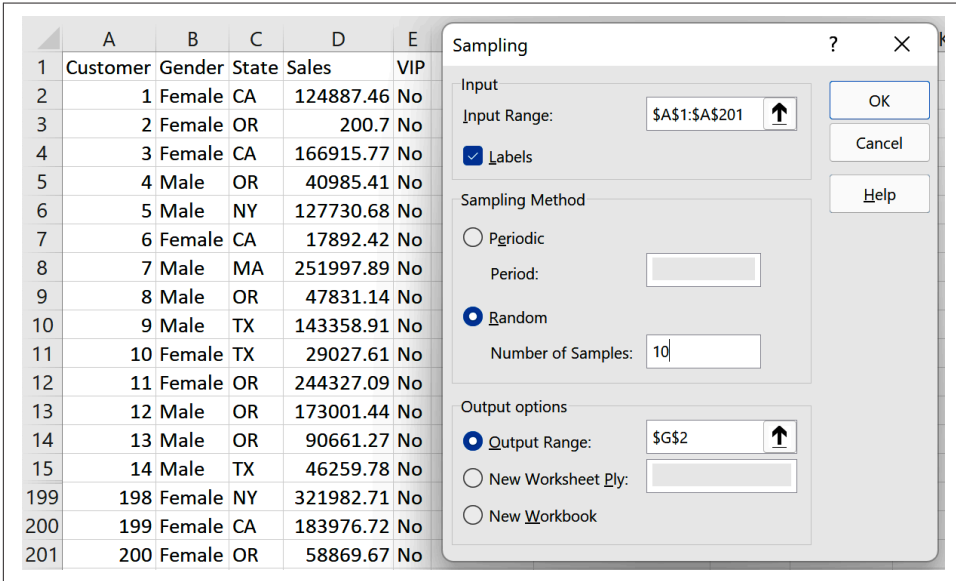
Figure 9-14. The Sampling tool's dialog box for generating a random sample

When you click OK, the Sampling tool outputs a random sample (see Figure 9-15).

Customer	Gender	State	Sales	VIP		Random sample
1	Female	CA	124887.5	No		138
2	Female	OR	200.7	No		73
3	Female	CA	166915.8	No		172
4	Male	OR	40985.41	No		6
5	Male	NY	127730.7	No		196
6	Female	CA	17892.42	No		20
7	Male	MA	251997.9	No		75
8	Male	OR	47831.14	No		4
9	Male	TX	143358.9	No		8
10	Female	TX	29027.61	No		176

Figure 9-15. The Random Sampling tool's output[3]

---

3 An extra label has been added to G1.

Discussion

This recipe describes how to generate a simple random sample from a range of values where every value has an equal probability of being selected.

> Generating a random sample using this tool may return duplicate values where a participant is selected more than once.

9.8 Generating a Periodic Sample

Problem

You have a dataset and want to generate a sample from it by choosing every *n*th item.

Solution

Follow Recipe 9.7, but this time, choose a Sampling Method of Periodic and type a value for *n* (for example, **20**) in the Period box. This option tells the Sampling tool to return every *n*th item.

Discussion

If a dataset is periodic, you can use this recipe to return values from a particular part of the cycle. For quarterly sales figures, for example, you can use a period of 4 to output values from the same quarter each year. Likewise, if you have monthly figures, a period of 12 will return values from the same month.

9.9 Drawing Random Numbers from a Distribution

Problem

You want to create a set of random numbers that follow a specific distribution, such as normal, binomial, or Poisson.

Solution

Suppose you want to generate a set of independent random numbers drawn from a specific distribution—for example, the normal distribution:

1. Go to the Data menu and choose Data Analysis from the Analyze group. Then select Random Number Generation from the list of analysis tools, and click OK to open the tool's dialog box (see Figure 9-16).
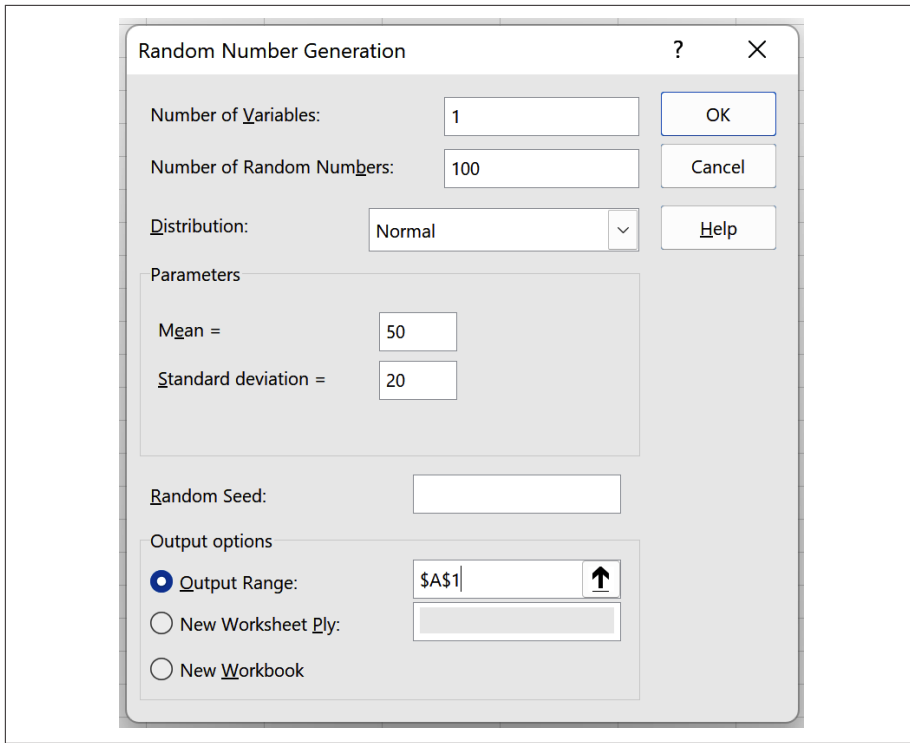
Figure 9-16. The Random Number Generation tool's dialog box

2. Type a number in the Number of Variables box (for example, **1**) to specify how many columns of random numbers you want to return.

3. Type a number in the Number of Random Numbers box (for example, **100**) to specify how many random numbers you want to generate in each column.

4. Choose the type of distribution you want to draw numbers from (for example, Normal).

5. Type parameters for the distribution. The Parameters section varies depending on the type of distribution you've selected. The normal distribution, for example, has Mean and Standard Deviation parameters, which I've set to 50 and 20, respectively.

6. Choose one of the "Output options" to specify where you want the tool to output the random numbers (for example, the Output Range A1).

When you click OK, the Random Number Generation tool outputs independent random numbers drawn from the specified distribution.

Discussion

This recipe offers a flexible way of generating numbers from several distributions. The following options are available:

Uniform
> Generates random numbers between two limits (that you specify), where each number has an equal chance of being chosen (see also Recipe 4.1).

Normal
> Draws random numbers from a normal distribution with the specified mean and standard deviation.

Bernoulli
> Generates 0 or 1 at random using the specified probability of success.

Binomial
> Draws random numbers from a binomial distribution using the specified probability of success and number of trials.

Poisson
> Draws random numbers from a Poisson distribution using the specified lambda value (the expected number of occurrences in an interval).

Patterned
> Repeats a series of numbers in steps between two limits.

Discrete
> Returns values that each have a specific probability of being chosen. This option requires a two-column input range where the first column holds the values and the second column holds the corresponding probability. The sum of the probabilities must equal 100%.

> To generate the same random number sequence multiple times, type an integer between 1 and 32,767 in the Random Seed box. This option specifies the starting value for the Random Number Generation tool's algorithm.

9.10 Generating a Correlation Matrix

Problem

You have pairs of measurements and want to determine if there's a linear relationship, and if so, its strength and direction.

Solution

Suppose you have a set of sales data over 12 months for 3 products—paper, ink cartridges, and computers—and you want to analyze the relationships between the products. A1:D13 lists the data, with the months in column 1 and labels in the first row (see Figure 9-17):

1. Go to the Data menu and choose Data Analysis from the Analyze group. Then select Correlation from the list of analysis tools, and click OK to open the tool's dialog box.

2. Choose B1:D13 in the Input Range box for the list of sales values, including the labels.

3. Place a check in the Labels check box because the first row of each input range contains data labels.

4. Choose one of the "Output options" to specify where you want the Descriptive Statistics tool to output the results (for example, the Output Range F1).
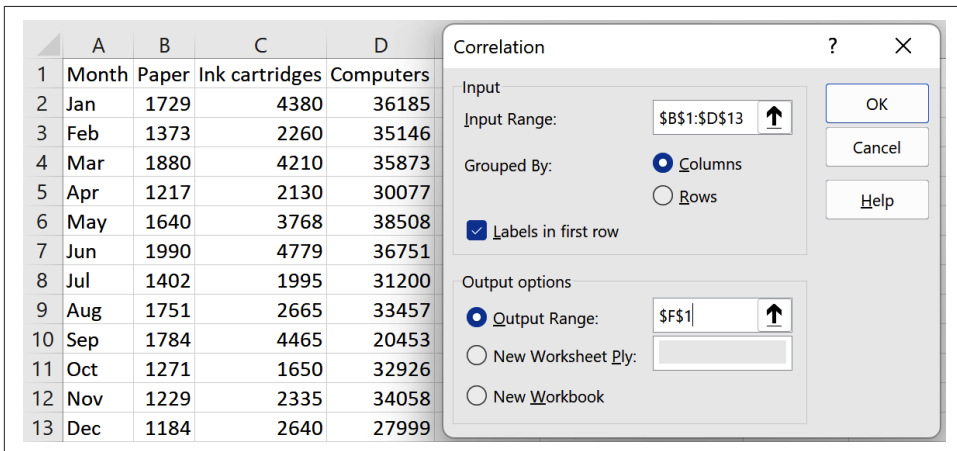
Figure 9-17. The Correlation tool's dialog box

When you click OK, the Correlation tool outputs a correlation matrix showing the correlation coefficient value for each possible pair of measurements (see Figure 9-18).

	Paper	Ink cartridges	Computers	
Paper	1			
Ink cartridges	0.862268599	1		
Computers	0.203414941	0.075400268	1	

Figure 9-18. The Correlation tool's output

Discussion

The *correlation coefficient* measures the strength of the linear relationship between two sets of values. The closer it is to plus or minus 1, the stronger the linear relationship, while a value of 0 indicates no linear relationship.

Behind the scenes, the Correlation tool uses the formula =CORREL(*values_1, values_2*) to find the correlation between each possible pair, where *values_1* and *values_2* are the two sets of values.

Note that the Correlation tool outputs values instead of formulas, so you'll need to rerun the tool if the underlying data changes.

See Also

Correlation is closely related to covariance; see Recipe 9.11.

9.11 Generating a Covariance Matrix

Problem

You have pairs of measurements and want to determine the degree to which their values change.

Solution

Follow Recipe 9.10, but this time choose the Covariance tool (see Figure 9-19). All the other steps remain the same.
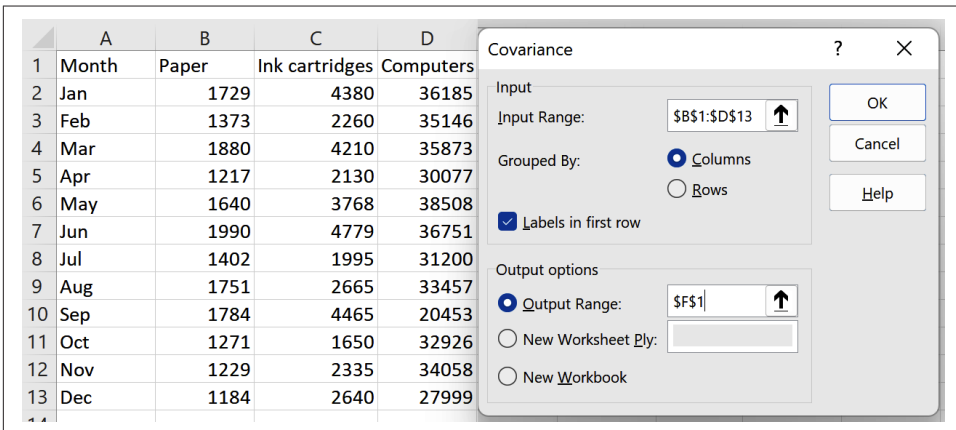
Figure 9-19. The Covariance tool's dialog box

When you click OK, the Covariance tool outputs a covariance matrix showing the covariance value for each possible pair of measurements (see Figure 9-20).

	Paper	Ink cartridges	Computers		
Paper	76240.25				
Ink cartridges	256745.2083	1162880.91			
Computers	262453.9583	379943.1597	21835153.24		

Figure 9-20. The Covariance tool's output

Discussion

Both correlation and covariance measure the relationship between two sets of values. The main difference is that correlation coefficients are scaled to lie between plus or minus 1, while covariances are unscaled.

For example, suppose you have measurements that include the weight in pounds. Then, if you convert the weight to kilograms, the correlation coefficient will stay the same, while the covariance will change.

Behind the scenes, the Covariance tool uses the formula =COVARIANCE.S(*values_1, values_2*) to find the correlation between each possible pair, where *values_1* and *values_2* are the two sets of values.

Note that the Covariance tool outputs values instead of formulas, so you'll need to rerun the tool if the underlying data changes.

9.12 Performing a Linear Regression Analysis

Problem

You have a list of data for a variable and want to analyze how one or more other variables affect it. You also want to find a linear equation for the relationship that you can use to make predictions.

Solution

Suppose you have data showing the number of items sold, product price, and the amount spent on advertising for different periods. You want to use this to predict how many items you'll sell, depending on the price and advertising costs. A2:A8 lists the number of items sold, B2:B8 lists the product price, C2:C8 lists the advertising cost, and the first row contains labels (see Figure 9-21):

1. Go to the Data menu and choose Data Analysis from the Analyze group. Then select Regression from the list of analysis tools, and click OK to open the tool's dialog box.

2. Choose A1:A8 in the Input Y Range box for the range listing the number of items sold.

3. Choose B1:C8 in the Input X Range box for the range listing the price and advertising amount.

4. Place a check in the Labels check box because the first row of the input ranges contains data labels.

5. Choose one of the "Output options" to specify where you want the tool to output the results (for example, the Output Range E1).
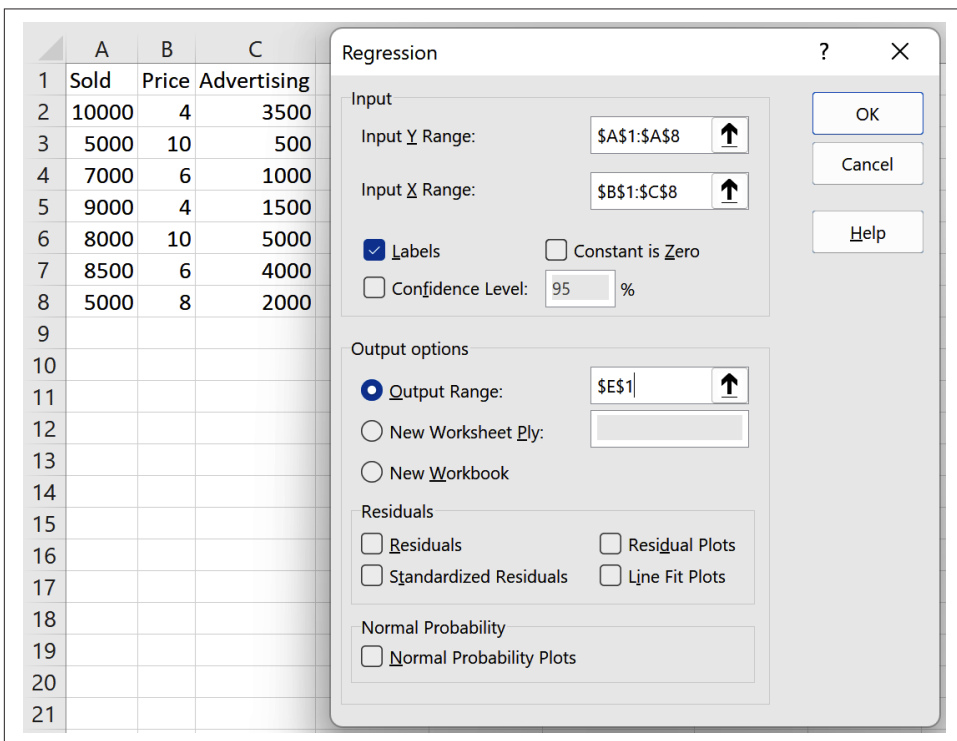
Figure 9-21. The Regression tool's dialog box

When you click OK, the Regression tool outputs three tables.

The first table shows regression statistics (see Figure 9-22). It includes an R-square statistic that indicates how well the analysis fits the data—the closer this value is to 1, the better the fit (see Recipe 8.23). In this example, R-square is 0.838, which means it's a good fit: the price and advertising costs explain 83.8% of the variation in the items sold.

Regression Statistics	
Multiple R	0.91552311
R Square	0.83818256
Adjusted R Square	0.75727384
Standard Error	954.056124
Observations	7

Figure 9-22. Statistics generated by the Regression tool, including R-square

The analysis of variance (ANOVA) table includes a significance F statistic (see Figure 9-23). This statistic shows the statistical significance of the results (see "Discussion" on page 219).

ANOVA					
	df	SS	MS	F	Significance F
Regression	2	18859107.65	9429553.823	10.35960738	0.026184883
Residual	4	3640892.354	910223.0886		
Total	6	22500000			

Figure 9-23. The Regression tool's ANOVA output including significance F

The third table shows the regression analysis results (Figure 9-24). Here, the coefficients describe the straight line that best fits the data, which you can use to make predictions. In this example, the equation for the line is y (items sold) = 9660.3855 − (557.559 × price) + (0.6651505 × advertising).

	Coefficients	Standard Error	t Stat	P-value	Lower 95%	Upper 95%
Intercept	9660.3855	1233.280747	7.8331	0.0014	6236.2492	13084.52
Price	-557.559	153.1677974	-3.64	0.022	-982.821	-132.297
Advertising	0.6651505	0.231567943	2.8724	0.0454	0.0222148	1.308086

Figure 9-24. The Regression analysis output including coefficients

Discussion

The Regression tool performs linear regression analysis using the least squares method. Behind the scenes, it uses the `LINEST` function to fit a straight line through the points in the input ranges you provide (see Recipe 8.24). The Input Y Range box specifies the dependent variable you want to be able to predict, and the Input X Range box specifies the independent variables you want to use in this prediction.

You generally want the Significance F statistic in the ANOVA table to be less than 0.05. If it's greater than 0.05, the results may be unreliable; try rerunning the tool, this time leaving out the variable with the highest P-value in the results table (see Figure 9-24).

The Regression dialog box includes the following additional options:

Constant is Zero
Forces the regression line to pass through the origin so that when the X values are 0, the Y value is also 0.

Confidence Level
The confidence level for the regression analysis.

Residuals
Generates residuals: the differences between the actual and predicted data points.

Normal Probability
Generates an extra table showing probabilities.

Note that the Regression tool outputs values instead of formulas, so you'll need to rerun the tool if the underlying data changes.

9.13 Performing a Two-Sample t-Test

Problem

You have two samples and want to compare the means of the populations they're drawn from when you don't know the population variances.

Solution

Use the t-Test: Two-Sample Assuming Equal Variances tool if you believe the populations have the same variances or the t-Test: Two-Sample Assuming Unequal Variances tool if you think the variances are different.

Suppose you want to test whether two product brands have the same mean weight, and you have a sample of weights for each brand. A2:A16 lists the weights for Brand

A, B2:B16 lists the weights for Brand B, and the first row contains labels (see Figure 9-25).
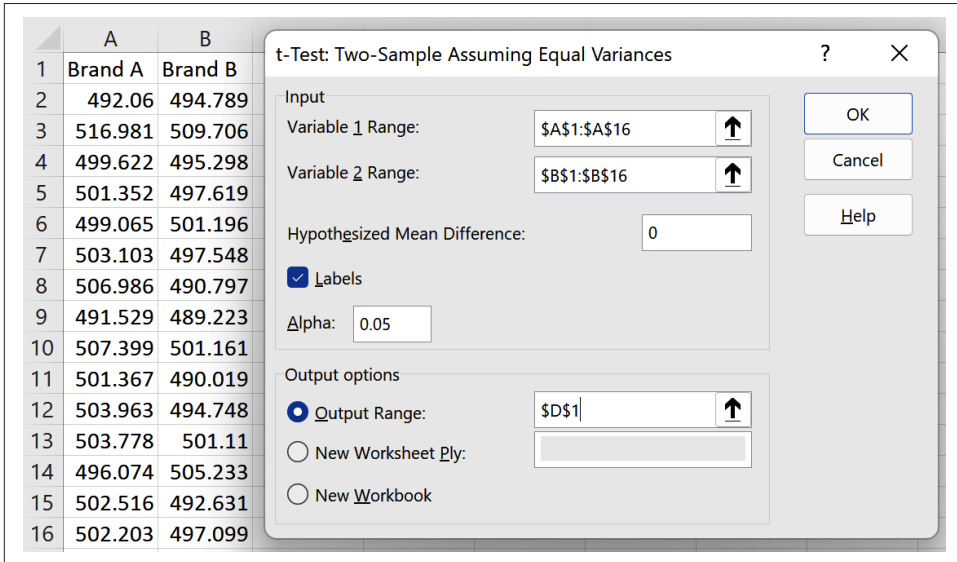
Figure 9-25. *The t-Test: Two-Sample Assuming Equal Variances tool's dialog box*

To solve this problem, you test whether the mean difference between the two populations (the weights for Brand A and Brand B) is 0:

1. Go to the Data menu and choose Data Analysis from the Analyze group.

2. From the list of analysis tools, select t-Test: Two-Sample Assuming Equal Variances or t-Test: Two-Sample Assuming Unequal Variances (this example uses the Equal Variances option). Then click OK to open the tool's dialog box.

3. Choose A1:A16 in the Variable 1 Range box for the list for Brand A.

4. Choose B1:B16 in the Variable 2 Range box for the list for Brand B.

5. Type a value for the Hypothesized Mean Difference. This example uses 0 to test whether the two population means are equal.

6. Place a check in the Labels check box because the first row of the variable ranges contains data labels.

7. Type a value for Alpha, the significance level—for example, **0.05**.

8. Choose one of the "Output options" to specify where you want the tool to output the results (for example, the Output Range D1).

When you click OK, the tool outputs the results (see Figure 9-26).

	Brand A	Brand B
Mean	501.8665333	497.2118
Variance	38.95456455	32.90684489
Observations	15	15
Pooled Variance	35.93070472	
Hypothesized Mean Difference	0	
df	28	
t Stat	2.126633102	
P(T<=t) one-tail	0.021199796	
t Critical one-tail	1.701130934	
P(T<=t) two-tail	0.042399592	
t Critical two-tail	2.048407142	

Figure 9-26. The t-Test: Two-Sample Assuming Equal Variances tool's output

Discussion

This recipe is helpful when you want to find the likelihood that two samples came from populations with equal means. It tests the null hypothesis that the means are the same (or have a hypothesized difference, which you enter in step 5) against the alternate hypothesis that they're not.

The most important statistics generated by the tool are the *p* values for the one-tailed and two-tailed tests. P(T <= t) one-tail gives the *p* value for a one-tailed test: a test that detects differences in the population means in one direction (for example, the mean for Brand A is greater than the mean for Brand B). On the other hand, P(T <= t) two-tail gives the *p* value for a two-tailed test used to detect differences in either direction (for example, the mean for Brand A is greater than or less than the mean for Brand B).

In general, decide whether you need a one-tailed or two-tailed test, and then see if its *p* value is less than the significance level (Alpha) you chose in step 7. If the *p* value is smaller than the significance level, there's enough evidence to reject the null hypothesis. In the example used in this recipe, both *p* values are less than the significance level (0.05), so there's enough evidence to conclude that the mean weight of Brand A is different from the mean weight of Brand B.

Behind the scenes, the tool uses the T.TEST function. This function takes the form =T.TEST(*array1, array2, tails, type*), where *array1* and *array2* are the two datasets, *tails* specifies whether to perform a one-tailed or two-tailed test, and *type* specifies the type of test.

Note that the t-Test tools output values instead of formulas, so you'll need to rerun the tool if the underlying data changes.

See Also

This recipe is for samples where you don't know the value of the population variances. If you know the variances, use Recipe 9.14 instead. If you have more than two samples, use Recipe 9.17 instead.

9.14 Performing a Two-Sample z-Test

Problem

You have two samples and want to compare the means of the populations they're drawn from when you know the population variances.

Solution

Suppose you want to test whether two product brands have the same mean weight, and you have a sample of weights for each brand. A2:A16 lists the weights for Brand A, B2:B16 lists the weights for Brand B, and the first row contains labels (see Figure 9-27). Furthermore, Brand A has a variance of 40, and Brand B has a variance of 31.
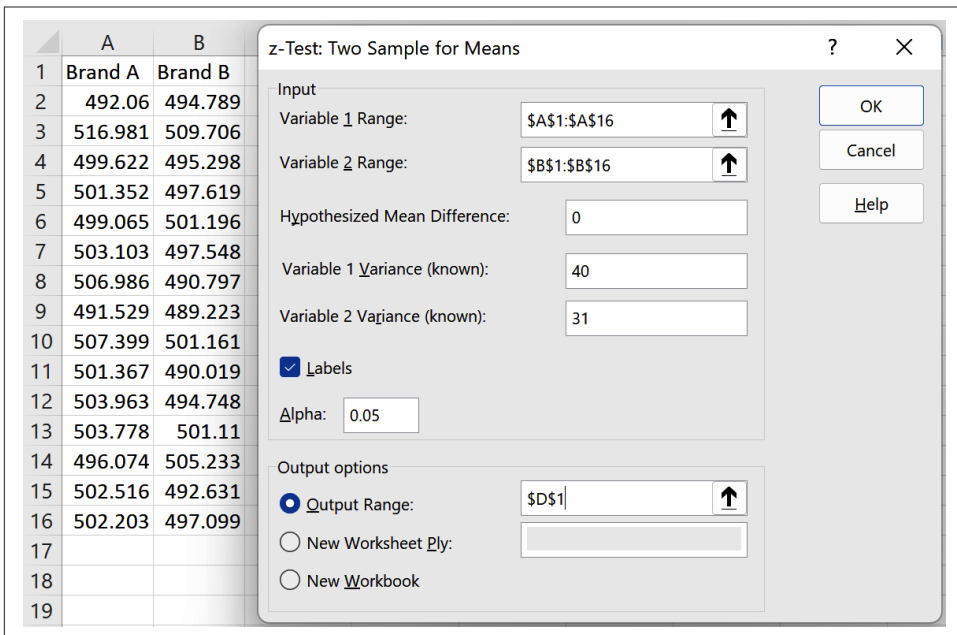
Figure 9-27. The z-Test: Two Sample for Means tool's dialog box

Follow Recipe 9.13, but this time choose the z-Test: Two Sample for Means tool and include these extra steps:

1. Type **40** in the Variable 1 Variance (known) box: this is the variance for Brand A.

2. Type **31** in the Variable 2 Variance (known) box: this is the variance for Brand B.

When you click OK, the tool outputs the results (see Figure 9-28).

	Brand A	Brand B		
Mean	501.8665333	497.2118		
Known Variance	40	31		
Observations	15	15		
Hypothesized Mean Difference	0			
z	2.139494925			
P(Z<=z) one-tail	0.016197803			
z Critical one-tail	1.644853627			
P(Z<=z) two-tail	0.032395606			
z Critical two-tail	1.959963985			

Figure 9-28. The z-Test: Two Sample for Means tool's output

Discussion

This recipe is handy when you want to find the likelihood that two samples came from populations with equal means and you know the value of the population variances. It tests the null hypothesis that the means are the same—or there's a specified difference between them—against the alternate hypothesis that they're not. Note that the z-Test: Two Sample for Means tool outputs values instead of formulas, so you'll need to rerun the tool if the underlying data changes.

The tool outputs p values—P(Z <= z)—for one-tailed and two-tailed tests. The discussion for Recipe 9.13 explains these tests.

> Excel also includes a Z.TEST function; this is useful if you know the population's standard deviation and want to test whether its mean is significantly different from a hypothesized value.

9.15 Performing a Paired Two-Sample t-Test

Problem

You have a sample composed of pairs of measurements and want to test the mean difference between the measurements.

Solution

Suppose you want to test whether a boot camp for athletes makes a difference in how fast they can run. You have a sample of athletes, and you've recorded how long it takes each one to sprint down a track before and after the boot camp. A2:A16 lists the ID of each athlete, B2:B16 lists the times recorded before the boot camp, C2:C16 lists the times recorded after the boot camp, and the first row contains labels (see Figure 9-29).

Figure 9-29. The t-Test: Paired Two Sample for Means tool's dialog box

To solve this problem, you test whether the mean difference between the two times (recorded before and after the boot camp) is 0:

1. Go to the Data menu and choose Data Analysis from the Analyze group. Then select t-Test: Paired Two Sample for Means from the list of analysis tools, and click OK to open the tool's dialog box.

2. Choose B1:B16 in the Variable 1 Range box for the list of times recorded before the boot camp.

3. Choose C1:C16 in the Variable 1 Range box for the list of times recorded after the boot camp.

4. Type a value for the Hypothesized Mean Difference. This example uses 0 to test whether the two means are equal.

5. Place a check in the Labels check box because the first row of the variable ranges contains data labels.

6. Type a value for Alpha, the significance level—for example, **0.05**.

7. Choose one of the "Output options" to specify where you want the tool to output the results (for example, the Output Range E1).

When you click OK, the tool outputs the results (see Figure 9-30).

	Before	After
Mean	93.36533333	101.1846667
Variance	58.14764095	97.92488381
Observations	15	15
Pearson Correlation	0.584586371	
Hypothesized Mean Difference	0	
df	14	
t Stat	-3.676614457	
P(T<=t) one-tail	0.001245007	
t Critical one-tail	1.761310136	
P(T<=t) two-tail	0.002490013	
t Critical two-tail	2.144786688	

Figure 9-30. The t-Test: Paired Two Sample for Means tool's output

Discussion

This recipe is a helpful way of testing whether two measurements taken from the same item or individual differ significantly. You might want to use this test for the following:

- Comparing measurements before and after a treatment
- Comparing the speeds of two car models, where each individual gets to drive each car
- Comparing the effects of two skin lotions, where one is applied to each individual's left arm and the other to their right

The most important statistics generated by the tool are the p values for the one-tailed and two-tailed tests. See the discussion for Recipe 9.13 for an explanation of these statistics.

9.16 Performing a Two-Sample F-Test for Variances

Problem

You have two samples and want to compare the variances of the populations from which they're drawn.

Solution

Suppose you want to test whether the scores for two teams have the same variability, and you have a sample of scores for each team. A2:A16 lists the scores for Team A, B2:B16 lists the scores for Team B, and the first row contains labels (see Figure 9-31).
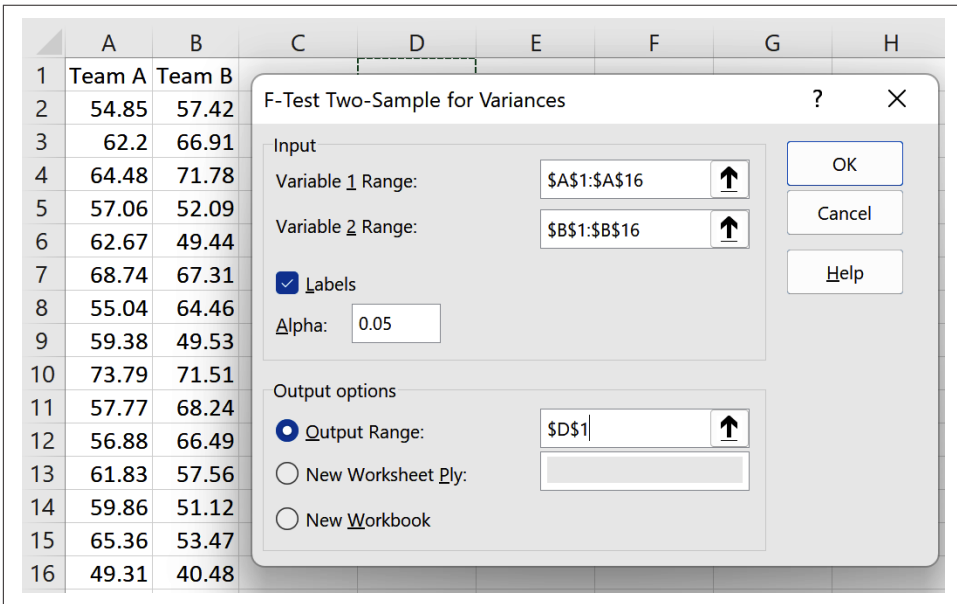
Figure 9-31. The F-Test Two-Sample for Variances tool's dialog box

To solve this problem, you test whether each population (the scores for Team A and Team B) has the same variance:

1. Go to the Data menu and choose Data Analysis from the Analyze group. Then select F-Test Two-Sample for Variances from the list of analysis tools, and click OK to open the tool's dialog box.

2. Choose A1:A16 in the Variable 1 Range box for the list for Team A.

3. Choose B1:B16 in the Variable 1 Range box for the list for Team B.

4. Place a check in the Labels check box because the first row of the variable ranges contains data labels.

5. Type a value for Alpha, the significance level—for example, **0.05**.

6. Choose one of the "Output options" to specify where you want the tool to output the results (for example, the Output Range D1).

When you click OK, the tool outputs the results (see Figure 9-32).

	Team A	Team B
Mean	60.61466667	59.18733333
Variance	36.72552667	92.23384952
Observations	15	15
df	14	14
F	0.3981784	
P(F<=f) one-tail	0.048029033	
F Critical one-tail	0.402620943	

Figure 9-32. The F-Test Two-Sample for Variances tool's output

Discussion

This recipe helps you find the likelihood that two samples came from populations with equal variances. It tests the null hypothesis that the variances are the same against the alternate hypothesis that they're not.

The most important statistic generated by the tool is P(F <= f) one-tail, which is the *p* value for a one-tailed test. If the *p* value is less than the significance level (Alpha) you chose in step 5, there's sufficient evidence to reject the null hypothesis.

Note that the F-Test Two-Sample for Variances tool outputs values instead of formulas, so you'll need to rerun the tool if the underlying data changes.

> To perform a two-tailed F-Test for variances, use Excel's `F.TEST` function instead. This takes the form `=F.TEST(array1, array2)`, where *array1* and *array2* are the ranges of the two datasets.

9.17 Performing a One-Way ANOVA Test

Problem

You have two or more samples that depend on a single factor and want to test whether they're drawn from populations with the same means.

Solution

Suppose you want to test whether three product brands have the same mean weight, and you have a sample of weights for each brand. A2:A11 lists the weights for Brand A, B2:B11 lists the weights for Brand B, C2:C11 lists the weights for Brand C, and the first row contains labels (see Figure 9-33).

	A	B	C
1	Brand A	Brand B	Brand C
2	489.57	497.38	501.15
3	507.8	499	477.48
4	489.29	505.65	483.16
5	519.48	509.09	469.45
6	496.48	505.4	499.93
7	511.76	498.62	474.87
8	485	497.96	480.93
9	514.68	477.68	496.52
10	496.5	496.78	476.21
11	503.23	513.32	504.78

Anova: Single Factor

Input
Input Range: $A$1:$C$11
Grouped By: ● Columns ○ Rows
☑ Labels in first row
Alpha: 0.05

Output options
● Output Range: $E$1
○ New Worksheet Ply:
○ New Workbook

Figure 9-33. The ANOVA: Single Factor tool's dialog box

Since the weights depend on a single factor—the product brand—you can use a one-way ANOVA test to assess whether the samples are drawn from populations with the same means:

1. Go to the Data menu and choose Data Analysis from the Analyze group. Then select ANOVA: Single Factor from the list of analysis tools, and click OK to open the tool's dialog box.

2. Choose A1:C11 in the Input Range box for the list of weights for the three samples, including their label.

3. Choose the Columns option in the Grouped By section because the weights are listed in columns.

4. Place a check in the "Labels in first row" check box because the first row of the input range contains data labels.

5. Type a value for Alpha, the significance level—for example, **0.05**.

6. Choose one of the "Output options" to specify where you want the tool to output the results (for example, the Output Range E1).

When you click OK, the tool outputs the results (see Figure 9-34).

Anova: Single Factor						
SUMMARY						
Groups	*Count*	*Sum*	*Average*	*Variance*		
Brand A	10	5013.79	501.379	139.8506989		
Brand B	10	5000.88	500.088	93.72630667		
Brand C	10	4864.48	486.448	164.9892844		
ANOVA						
Source of Variation	*SS*	*df*	*MS*	*F*	*P-value*	*F crit*
Between Groups	1368.836807	2	684.4184033	5.151602786	0.012730274	3.354130829
Within Groups	3587.09661	27	132.85543			
Total	4955.933417	29				

Figure 9-34. The ANOVA: Single Factor tool's output

Discussion

This recipe uses multiple samples to test whether the means of at least two of the populations they're drawn from are different. It tests the null hypothesis that the means are the same against the alternate hypothesis that they're not.

The Analysis ToolPak includes three separate ANOVA tools. This recipe uses the ANOVA: Single Factor tool, where the sample measurements (in this case, the weights) depend on a single factor (the product brand). It's a more general form of Recipe 9.13, except that you can have more than two samples.

The most important statistic in the tool's output is the *p* value. If this statistic is less than the significance level (Alpha) you chose in step 5, it means there's enough evidence to reject the null hypothesis that the samples are drawn from populations with the same means. In the example used in this recipe, the *p* value (0.0127) is less than the significance level (0.05), so there's enough evidence to conclude that the mean weight of at least one brand differs from the others.

9.18 Performing a Two-Way ANOVA Test

Problem

You have a sample of measurements and want to test whether they're influenced by two factors, either individually or together.

Solution

Suppose you want to test whether the starter culture brand, the yogurt maker model, or some interaction between the two influences the amount of time it takes to make yogurt. You have recorded the time taken in hours, using three yogurt maker models and three brands of starter culture and repeated this four times for each combination. A2:D13 lists the results, with the yogurt maker models arranged in columns, the starter cultures arranged in rows, and data labels in the first row and column (see Figure 9-35).

	A	B	C	D
1		Model A	Model B	Model C
2	Culture 1	8.3	7.3	8.2
3		7.1	6.4	9.4
4		7.8	8.5	9.9
5		6.8	7.9	7.9
6	Culture 2	8.4	8.2	9.2
7		6.7	7.6	8.1
8		6.2	8	8.9
9		7.4	6.9	8.5
10	Culture 3	10.1	10.5	10.1
11		10.7	9.5	10.7
12		9.8	9.1	10.1
13		9.6	10.2	9.8

Anova: Two-Factor With Replication   ?   ✕

Input
Input Range: `$A$1:$D$13` ↑
Rows per sample: `4`
Alpha: `0.05`

Output options
● Output Range: `$F$1|` ↑
○ New Worksheet Ply: `_____`
○ New Workbook

[OK] [Cancel] [Help]
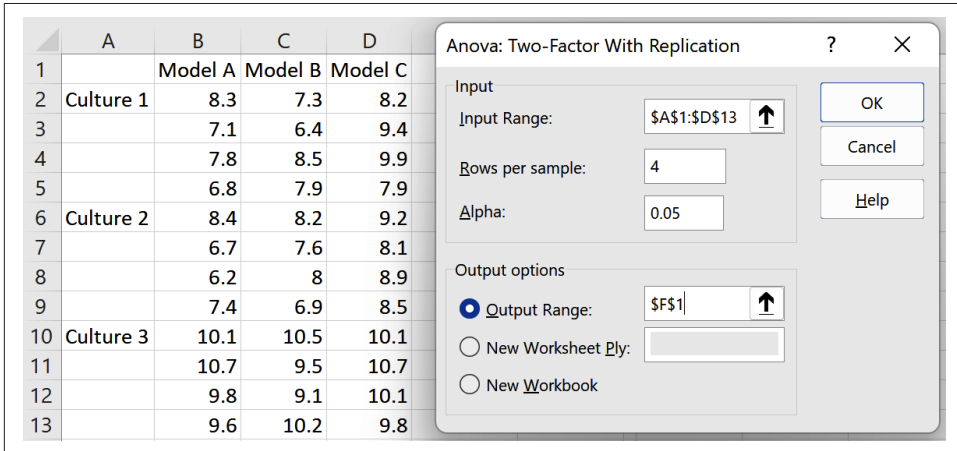
Figure 9-35. The ANOVA: Two-Factor With Replication tool's dialog box

Because the time taken depends on two factors—the starter culture brand and the yogurt maker model—and there are multiple times for each combination, you can use a two-way ANOVA test with replication to analyze the results:

1. Go to the Data menu and choose Data Analysis from the Analyze group. Then select ANOVA: Two-Factor With Replication from the list of analysis tools, and click OK to open the tool's dialog box.

2. Choose A1:D13 in the Input Range box for the entire range of the data, including labels.

3. Type **4** in the "Rows per sample" box because there are four recorded times for each combination, listed in separate rows.

4. Type a value for Alpha, the significance level—for example, **0.05**.

5. Choose one of the "Output options" to specify where you want the tool to output the results (for example, the Output Range F1).

When you click OK, the tool outputs the results in several tables. The most important results are in the ANOVA table (see Figure 9-36).

ANOVA						
Source of Variation	SS	df	MS	F	P-value	F crit
Sample	35.92388889	2	17.96194444	36.56030908	2.07178E-08	3.354130829
Columns	7.153888889	2	3.576944444	7.280625707	0.002958792	3.354130829
Interaction	2.536111111	4	0.634027778	1.290520166	0.298457563	2.727765306
Within	13.265	27	0.491296296			
Total	58.87888889	35				

Figure 9-36. The ANOVA: Two-Factor With Replication tool's output

Discussion

This recipe uses the ANOVA: Two-Factor with Replication tool, where the sample measurements (in this case, the time taken) depend on two factors (the starter culture brand and the yogurt maker model). The term *with replication* in the tool's name means multiple data points exist for each combination of factors.

The tool uses the data to test each factor's influence and how they interact. It tests the null hypothesis that the factors do not influence the outcomes against the alternate hypothesis that they do.

The most important statistics generated by the tool output are the p values. The tool outputs three of these: one for each factor and another for the interaction between the two.

The first p value to consider is the one labeled Interaction. If this statistic is less than the significance level (Alpha) you chose in step 4, the interaction between the two factors significantly influences the results. In the example used in this recipe, the p value for Interaction is 0.298; there's not enough evidence to suggest that interactions between the starter culture brand and the yogurt maker model influence the time taken to culture yogurt.

If the p value for Interaction is higher than the significance level, you next need to consider the p values for Sample and Columns. If either value is less than the significance level, that factor significantly influences the outcomes. In the example used in this recipe, both p values are less than the significance level of 0.05, meaning that the starter culture brand and the yogurt maker model influence the time taken to culture yogurt.

The ANOVA: Two-Factor With Replication tool works with datasets with multiple data points for each combination of factors. If there's a single data point for each combination, use the ANOVA: Two-Factor Without Replication tool instead.

9.19 Running a Fourier Analysis

Problem

You have a periodic dataset and want to transform it to another domain using the Fast Fourier Transform (FFT) method.

Solution

Suppose you've recorded the value of a signal every 0.1 seconds, and you want to convert the results from a time series to the frequency domain. A2:A33 lists the index of each data point, B2:B33 lists the elapsed seconds, C2:C33 lists the signal data, and the first row contains labels (see Figure 9-37):

1. Go to the Data menu and choose Data Analysis from the Analyze group. Then select Fourier Analysis from the list of analysis tools, and click OK to open the tool's dialog box.

2. Choose C1:C33 in the Input Range box for the list of signal data, including the label in C1.

3. Place a check in the "Labels in first row" check box because the first row of the input range contains the data's label.

4. Choose an Output Range of D2 so that the FFT for each row is output next to the original data.
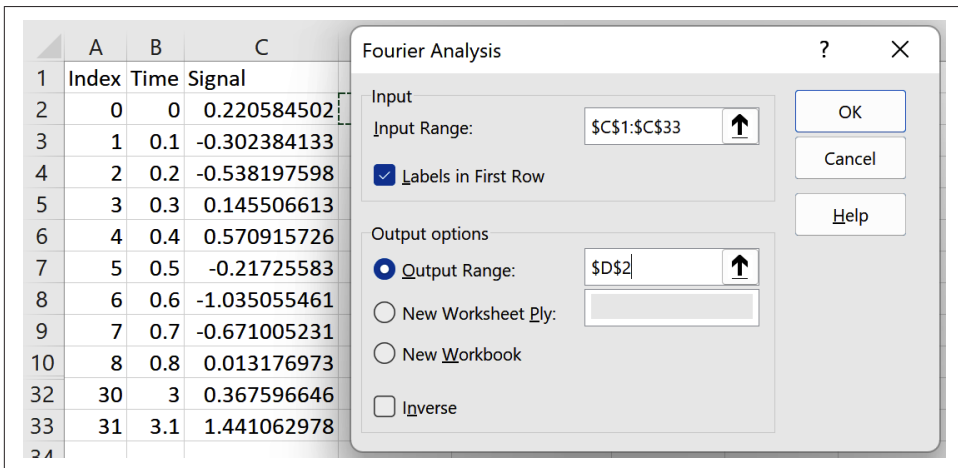
Figure 9-37. The Fourier Analysis tool's dialog box

When you click OK, the tool outputs the FFT values as complex numbers (see Figure 9-38).

	A	B	C	D
1	Index	Time	Signal	Fourier
2	0	0	0.220584502	18.6784862485348
3	1	0.1	-0.302384133	-7.99226886665048+6.19937416958388i
4	2	0.2	-0.538197598	-0.985540594832308+4.57066502113923i
5	3	0.3	0.145506613	8.1843853508566-7.71795940452832i
6	4	0.4	0.570915726	-2.41131761533812+5.95845960634738i
7	5	0.5	-0.21725583	-1.18786774996966+3.38678196583616i
32	30	3	0.367596646	-0.98554059483232-4.57066502113923i
33	31	3.1	1.441062978	-7.9922688666505-6.19937416958386i

Figure 9-38. The Fourier Analysis tool's output[4]

Discussion

The Fourier Analysis tool transforms discrete, periodic data using the FFT method. It's an algorithm used in many areas, including engineering, music, science, and mathematics.

A common use of the FFT method is to transform a time series into the frequency domain and use the output to plot magnitude against frequency (see Figure 9-39):

- To find the frequencies, take the index of each data point; then divide it by the number of data points multiplied by the sample rate.

- To find the magnitudes, use the IMABS function (see Recipe 4.18) to return the absolute value of the FFT output. Then divide each value by the number of data points divided by 2.

	A	B	C	D	E	F
1	Index	Time	Signal	Fourier	Frequency (Hz)	Magnitude
2	0	0	0.22058450:	18.678486248534	=A2/($B$35*$B$36)	=IMABS(D2)/($B$35/2)
3	1	0.1	-0.30238413	-7.992268866650‹	=A3/($B$35*$B$36)	=IMABS(D3)/($B$35/2)
4	2	0.2	-0.53819759	-0.985540594832:	=A4/($B$35*$B$36)	=IMABS(D4)/($B$35/2)
5	3	0.3	0.14550661:	8.1843853508566	=A5/($B$35*$B$36)	=IMABS(D5)/($B$35/2)
6	4	0.4	0.57091572!	-2.411317615338:	=A6/($B$35*$B$36)	=IMABS(D6)/($B$35/2)
7	5	0.5	-0.21725583	-1.187867749969(	=A7/($B$35*$B$36)	=IMABS(D7)/($B$35/2)
32	30	3	0.36759664(	-0.985540594832:	=A32/($B$35*$B$36)	=IMABS(D32)/($B$35/2)
33	31	3.1	1.44106297{	-7.992268866650!	=A33/($B$35*$B$36)	=IMABS(D33)/($B$35/2)
34						
35	Samples	32				
36	Interval	0.1				

Figure 9-39. Formulas calculating the frequency and magnitude of each point

---

4 An extra Fourier label has been added to the first row.

Once you have these results, you can use a scatter chart to plot magnitude against frequency (see Figures 9-40 and 9-41).
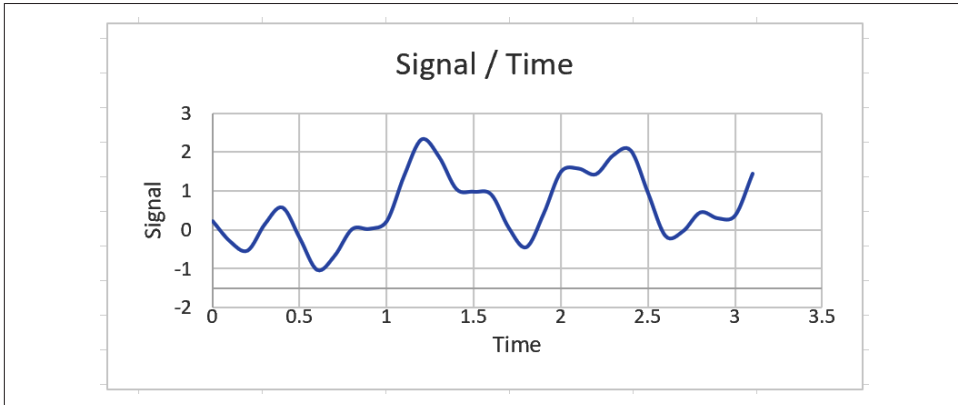
Figure 9-40. A chart showing signal plotted against time
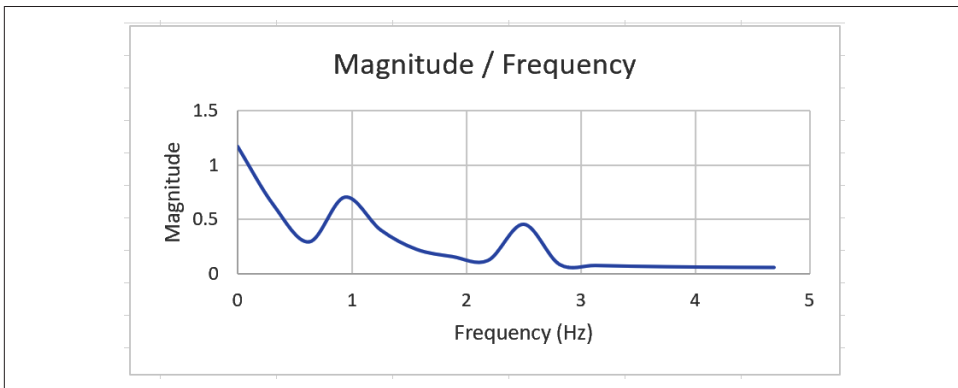
Figure 9-41. A chart showing magnitude plotted against frequency

The Fourier Analysis tool also supports inverse transformations, which transform the FFT values back to the original data. To use this option, place a check in the Inverse check box in the tool's dialog box.

Because of the nature of the FFT algorithm, you must provide the Fourier Analysis tool with a range of values whose size is a power of 2 (2, 4, 8, 16, 32, and so on). If you don't have 2^n values, you must pad the end of the series with zeros until the next power of 2.

Financial Analysis

Excel is an invaluable tool for financial analysis that helps you tackle a wide range of problems. You can use it to calculate loan payments, the projected future value of an investment, an asset's depreciation, and estimated growth under linear, exponential, and seasonal models.

This chapter guides you through these areas and more, using functions, charts, data types, and Excel's Forecast Sheet.

10.1 Calculating Fixed-Rate Loan Payments

Problem

You have a fixed-interest-rate loan and want to know how much you'll pay each period.

Solution

Suppose you're considering taking out a $5,000 loan with an annual interest rate of 10% for 12 months and want to know the monthly payments. B1 contains the interest rate (10%), B2 contains the number of months (12), and B3 contains the loan principal (5000); see Figure 10-1.

	A	B	C	D	E	F
1	Interest Rate (annual):	10%		End of month payments:	=PMT(B1/12, B2, B3)	-$439.58
2	Months:	12		End of month (balloon):	=PMT(B1/12, B2, B3, B4)	-$360.00
3	Loan Amount:	$5,000.00		Start of month (balloon):	=PMT(B1/12, B2, B3, B4, 1)	-$357.02
4	Future Value (balloon):	-$1,000.00				

Figure 10-1. Formulas for calculating monthly loan payments

If you want to make 12 equal payments at the end of each month, you can calculate the amount of each payment using the PMT function; type **=PMT(B1/12, B2, B3)**, which returns –$439.58. This calculation uses the formula =PMT(*rate, nper, pv*), where *rate* is the interest rate per period, *nper* is the total number of periods in the loan term, and *pv* is the principal or present value of the loan.

> Ensure you use the interest rate per period so that the PMT function's *rate* and *nper* arguments use the same units. So, if you're working with monthly periods and have an annual interest rate, you need to divide the annual rate by 12 to get the monthly rate.

If you want to take out a balloon loan with a remaining final balance to pay off at maturity, use the formula =PMT(*rate, nper, pv, fv*), where *fv* refers to the amount that's left at the end of the loan term (the future value). For example, if you want to make 12 equal payments at the end of each month and pay $1,000 at the end of the loan term, you'd type **-1000** in cell B4 for the future value and calculate the amount of each monthly payment by typing **=PMT(B1/12, B2, B3, B4)**, which returns –$360.00.

> Amounts that you receive, such as the loan principal, are positive values. Conversely, the amounts you must pay, such as the monthly payments or the amount left to pay at the end of the loan term, are negative.

If you want to make payments at the start of each period, use the formula =PMT(*rate, nper, pv, fv,* 1). To calculate the amount of each payment made at the start of the month, you'd type **=PMT(B1/12, B2, B3, B4, 1)**, which returns –$357.02.

Discussion

The PMT function offers a flexible way of calculating the payments you need to make on a fixed-rate loan. In addition to specifying the rate and loan term, you can accommodate balloon loans and choose whether payments are made at the start or end of each period.

10.2 Calculating Interest and Principal Loan Payments

Problem

You have a loan and want to know how much the interest charges and principal repayment value will be each period.

Solution

Suppose you're considering taking out a $5,000 loan with an annual interest rate of 10% for 12 months, and you want to know the monthly interest charges and principal repayments. B1 contains the interest rate (10%), B2 contains the number of months (12), and B3 contains the loan principal (5000); see Figure 10-2.

	A	B	C	D	E	F
1	Interest Rate (annual):	10%		Month 6 interest:	=IPMT(B1/12, 6, B2, B3)	-$24.81
2	Months:	12		Month 6 principal:	=PPMT(B1/12, 6, B2, B3)	-$414.77
3	Loan Amount:	$5,000.00		Months 4-6 interest:	=CUMIPMT(B1/12, B2, B3, 4, 6, 0)	-$84.68
4				Months 4-6 principal:	=CUMPRINC(B1/12, B2, B3, 4, 6, 0)	-$1,234.06

Figure 10-2. Formulas for calculating interest charges and principal repayments

To calculate the interest charges for a single period when you make payments at the end of the period, use the formula =IPMT(*rate, period, nper, pv*), where *rate* is the interest rate per period, *period* is the period, *nper* is the total number of periods in the loan term, and *pv* is the principal or present value of the loan. If you wanted to know the interest charge for month 6, you'd type **=IPMT(B1/12, 6, B2, B3)**, which returns –$24.

You can also use the IPMT function to calculate the interest charged for a single period on a balloon loan (see Recipe 10.1) or when you make payments at the start of the period. Use the formula =IPMT(*rate, period, nper, pv, fv, type*), where *fv* refers to the amount that's left at the end of the loan term and *type* refers to when you make payments—omit *type* or set it to 0 if you want to make payments at the end of the period (the default), or set it to 1 if you want to make payments at the start.

> If your loan's principal repayments are the same each period, you can use the ISPMT function to compute the interest payments.

The PPMT function works similarly to the IPMT function, except that you use PPMT to calculate the principal repayment for a single period. In general, you use the formula =PPMT(*rate, period, nper, pv, fv, type*), where *fv* and *type* are optional. To calculate the principal repayment for month 6, for example, you'd type **=PPMT(B1/12, 6, B2, B3)**, which returns –$414.77.

If you want to calculate the interest charged between two periods (inclusive), you can use the formula =CUMIPMT(*rate, nper, pv, start_period, end_period, type*), where *start_period* and *end_period* are the start and end periods, and *type* is 0 if you make payments at the end of the period and 1 if you make them at the start. To

calculate the interest charges between months 4 and 6 (inclusive), you'd type **=CUMIPMT(B1/12, B2, B3, 4, 6, 0)**, which returns –$84.68.

The CUMPRINC function works the same way as the CUMIPMT function, except CUMPRINC calculates the principal repayment between the two periods. To calculate the principal repayment between months 4 and 6 (inclusive) you'd type **=CUMPRINC(B1/12, B2, B3, 4, 6, 0)**, which returns –$1234.06.

Discussion

This recipe is helpful if you want to see the breakdown between interest charges and principal repayment for a given period, or calculate the total interest charges. These values vary through the loan term: the amount paid on interest charges decreases while the principal repayment increases, so you spend more on interest charges toward the start than toward the end.

10.3 Building a Variable Rate Loan Amortization Schedule

Problem

You have a variable-rate loan and want to create an amortization schedule showing the monthly payments, the split between interest charges and principal repayment, and the outstanding balance.

Solution

Suppose you've taken out a $5,000 loan for 12 months with a variable interest rate and want to create an amortization schedule:

1. Create the Month column by typing **Month** in A1, and **0, 1, 2, …, 11, 12** in A2:A14.

2. Create the Annual Rate column by typing **Annual Rate** in B1, and the applicable annual rate for each month in B3:B14.

3. Create the Payment column by typing **Monthly Payment** in C1 and the formula **=PMT(B3/12, $A$14-A2, F2)** in C3 (see "Discussion" on page 239). Then fill C4:C14 with the formula in C3.

4. Create the Interest column by typing **Interest Payment** in D1 and the formula **=IPMT(B3/12, 1, $A$14-A2, F2)** in D3. Then fill D4:D14 with the formula in D3.

5. Create the Principal column by typing **Principal Payment** in E1 and the formula **=PPMT(B3/12, 1, $A$14-A2, F2)** in E3. Then fill E4:E14 with the formula in E3.

6. Create the Balance column by typing **Balance** in F1, **5000** in F2, and the formula **=F2+E3** in F3 to add the principal repayment to the balance for the previous month. Then fill F4:F14 with the formula in F3.

Figure 10-3 shows the schedule, and Figure 10-4 shows the formulas.

	A	B	C	D	E	F
1	Month	Annual Rate	Monthly Payment	Interest Payment	Principal Payment	Balance
2	0					$5,000.00
3	1	12%	-$444.24	-$50.00	-$394.24	$4,605.76
4	2	12%	-$444.24	-$46.06	-$398.19	$4,207.57
5	3	12%	-$444.24	-$42.08	-$402.17	$3,805.40
6	4	6%	-$433.46	-$19.03	-$414.44	$3,390.97
7	5	12%	-$443.17	-$33.91	-$409.26	$2,981.71
8	6	14%	-$446.07	-$34.79	-$411.28	$2,570.43
9	7	10%	-$440.99	-$21.42	-$419.57	$2,150.86
10	8	8%	-$438.81	-$14.34	-$424.47	$1,726.39
11	9	10%	-$440.63	-$14.39	-$426.24	$1,300.15
12	10	7%	-$438.45	-$7.58	-$430.86	$869.28
13	11	7%	-$438.45	-$5.07	-$433.38	$435.91
14	12	7%	-$438.45	-$2.54	-$435.91	$0.00

Figure 10-3. A variable-rate amortization schedule

	A	B	C	D	E	F
1	Month	Annual Rate	Monthly Payment	Interest Payment	Principal Payment	Balance
2	0					5000
3	1	0.12	=PMT(B3/12,$A$14-A2,F2)	=IPMT(B3/12,1,$A$14-A2,F2)	=PPMT(B3/12,1,$A$14-A2,F2)	=F2+E3
4	2	0.12	=PMT(B4/12,$A$14-A3,F3)	=IPMT(B4/12,1,$A$14-A3,F3)	=PPMT(B4/12,1,$A$14-A3,F3)	=F3+E4
5	3	0.12	=PMT(B5/12,$A$14-A4,F4)	=IPMT(B5/12,1,$A$14-A4,F4)	=PPMT(B5/12,1,$A$14-A4,F4)	=F4+E5
6	4	0.06	=PMT(B6/12,$A$14-A5,F5)	=IPMT(B6/12,1,$A$14-A5,F5)	=PPMT(B6/12,1,$A$14-A5,F5)	=F5+E6
7	5	0.12	=PMT(B7/12,$A$14-A6,F6)	=IPMT(B7/12,1,$A$14-A6,F6)	=PPMT(B7/12,1,$A$14-A6,F6)	=F6+E7
8	6	0.14	=PMT(B8/12,$A$14-A7,F7)	=IPMT(B8/12,1,$A$14-A7,F7)	=PPMT(B8/12,1,$A$14-A7,F7)	=F7+E8
9	7	0.1	=PMT(B9/12,$A$14-A8,F8)	=IPMT(B9/12,1,$A$14-A8,F8)	=PPMT(B9/12,1,$A$14-A8,F8)	=F8+E9
10	8	0.08	=PMT(B10/12,$A$14-A9,F9)	=IPMT(B10/12,1,$A$14-A9,F9)	=PPMT(B10/12,1,$A$14-A9,F9)	=F9+E10
11	9	0.1	=PMT(B11/12,$A$14-A10,F10)	=IPMT(B11/12,1,$A$14-A10,F10)	=PPMT(B11/12,1,$A$14-A10,F10)	=F10+E11
12	10	0.07	=PMT(B12/12,$A$14-A11,F11)	=IPMT(B12/12,1,$A$14-A11,F11)	=PPMT(B12/12,1,$A$14-A11,F11)	=F11+E12
13	11	0.07	=PMT(B13/12,$A$14-A12,F12)	=IPMT(B13/12,1,$A$14-A12,F12)	=PPMT(B13/12,1,$A$14-A12,F12)	=F12+E13
14	12	0.07	=PMT(B14/12,$A$14-A13,F13)	=IPMT(B14/12,1,$A$14-A13,F13)	=PPMT(B14/12,1,$A$14-A13,F13)	=F13+E14

Figure 10-4. Formulas for the variable-rate amortization schedule

Discussion

This recipe uses the PMT, IPMT, and PPMT functions to create a loan amortization schedule. The schedule treats each month as the first period of a decreasing loan term, where the principal value is the remaining balance. It achieves this using the following arguments:

- The rate is the applicable annual rate for each month divided by 12.
- The number of periods is the number of months left in the term: the total number of months minus the previous month's number.
- The principal is the remaining balance for the previous month.
- The period for each month—used by the IPMT and PPMT functions—is 1.

10.4 Calculating the Term for a Fixed-Rate Loan

Problem

You want to take out a fixed-rate loan and need to know what the term should be.

Solution

Suppose you want to take out a $5,000 loan with an annual interest rate of 10%. You can afford to pay up to $300 per month at the end of each month and want to know the minimum term. B1 contains the interest rate (10%), B2 contains the monthly payment (–300), and B3 contains the loan principal (5,000) (see Figure 10-5).

	A	B	C	D	E	F
1	Interest Rate (annual):	10%		Number of periods:	=NPER(B1/12, B2, B3)	18.0185
2	Monthly Payment	-$300.00				
3	Loan Amount:	$5,000.00				

Figure 10-5. Formula for calculating the number of periods

You can solve this problem using the NPER function; type **=NPER(B1/12, B2, B3)**, which returns 18.0185. In general, you use the formula =NPER(*rate, pmt, pv*), where *rate* is the interest rate per period, *pmt* is the payment per period, and *pv* is the principal or present value of the loan.

You can add extra arguments to the NPER function to accommodate balloon loans (see Recipe 10.1) or make payments at the start of each period. Use the formula =NPER(*rate, pmt, pv, fv, type*), where *fv* refers to the amount that's left at the end of the loan term, and *type* refers to when you make payments—omit *type* or set it to 0 if you want to make payments at the end of the period (the default), or set it to 1 if you want to make payments at the start.

Discussion

This recipe offers a helpful way of figuring out the minimum loan term you can afford with a fixed interest rate and principal.

If the term is fixed instead of the interest rate, you can use the RATE function to determine the maximum rate per period you can afford. See Recipe 10.10 for more details of this function.

10.5 Calculating the Principal or Present Value

Problem

You want to take out a loan for a fixed rate and term and want to know what principal value you can afford.

Solution

Suppose you want to take out a 12-month loan with a fixed annual interest rate of 10% and pay $500 at the end of each month, and you want to know the maximum principal value. B1 contains the annual interest rate (10%), B2 contains the number of months (12), and B3 contains the monthly payment (–500).

You can solve this problem using the PV function; type **=PV(B1/12, B2, B3)**, which returns $5687.25 (see Figure 10-6). In general, you use the formula =PV(*rate, nper, pmt, fv, type*), where *rate* is the interest rate per period, *nper* is the total number of periods in the loan term, *pmt* is the payment per period, *fv* (optional) refers to any amount that's left at the end of the loan term, and *type* (optional) refers to when you make payments—omit *type* or set it to 0 if you want to make payments at the end of the period (the default), or set it to 1 if you want to make payments at the start.

	A	B	C	D	E	F
1	Interest Rate (annual):	10%		Principal:	=PV(B1/12, B2, B3)	$5,687.25
2	Months:	12				
3	Monthly Payment:	-$500.00				

Figure 10-6. Formula for calculating the principal value

Discussion

The PV function returns the principal or present value. As you can see, this function works similarly to the PMT, IPMT, PPMT, NPER, and RATE functions discussed in earlier recipes.

10.6 Converting Between Nominal and Effective Rates

Problem

You know an investment's nominal annual interest rate and want to find its effective annual rate.

Solution

An investment's *nominal* rate is the annual interest rate before any interest is compounded—before any interest earned is added to the investment. The *effective* rate includes compounding, so it's the annual interest rate the investment actually earns—the interest on the investment's principal and the reinvested interest.

To convert a nominal rate to an effective rate, you use the formula =EFFECT (*nominal_rate, compounding_periods*), where *nominal_rate* is the investment's nominal rate, and *compounding_periods* is the number of *compounding periods* per year: the number of periods when any interest earned is reinvested. If an investment has a nominal interest rate of 10% and compounds each quarter, for example, you calculate the effective interest rate by typing **=EFFECT(0.1, 4)**, which returns 0.1038.

If the investment uses continuous compounding so any interest is continually reinvested, you can convert the nominal rate to an effective rate using the formula =EXP(*nominal_rate*)-1 instead. So if an investment has a nominal interest rate of 10% and uses continuous compounding, you'd calculate the effective interest rate by typing **=EXP(0.1)-1**, which returns 0.1052.

To convert an effective rate to a nominal rate, use the formula =NOMINAL (*effective_rate, compounding_periods*). For example, if an investment has an effective interest rate of 12.36% and compounds every 6 months, you calculate the nominal interest rate by typing **=NOMINAL(0.1236, 2)**, which returns 0.12.

If the investment uses continuous compounding, you can convert the effective rate to a nominal rate using the formula =LN(*effective_rate*+1) instead. If an investment has an effective interest rate of 12.36% and uses continuous compounding, you calculate the nominal interest rate by typing **=LN(0.1236+1)**, which returns 0.1165.

Discussion

This recipe shows how to calculate the interest rate an investment actually earns when it advertises only the nominal rate. You can also use this approach to compare two investments, one that advertises the nominal rate and the other that advertises the effective rate.

10.7 Calculating the Future Value of a Fixed-Rate Lump-Sum Investment

Problem

You've invested a lump sum at a fixed rate and want to know what its value will be in the future.

Solution

Suppose you want to invest $5,000 for 2 years at 7% and want to know its *future value*: the value of the investment at the end of the term. B1 contains the nominal interest rate (7%), B2 contains the number of years for the investment (2), and B3 contains the value of the lump sum deposit (−5000); see Figure 10-7.

	A	B	C	D	E	F
1	Interest Rate (annual):	7%		FV (annual compounding):	=FV(B1 ,B2, 0, B3)	$5,724.50
2	Periods (years):	2		FV (continuous compounding):	=-B3*EXP(B1*B2)	$5,751.37
3	Initial Deposit:	-$5,000.00				

Figure 10-7. Calculating the future value of a fixed-rate lump-sum investment

If the investment compounds annually, you can solve this problem using the FV function; type **=FV(B1, B2, 0, B3)**, which returns $5724.50. In general, you use the formula =FV(*rate, nper,* 0, *pv*), where *rate* is the interest rate per period, *nper* is the number of compounding periods in the term, and *pv* is the initial lump sum deposit. The third argument is set to 0 because it relates to regular deposits, which are covered in Recipe 10.9.

> Ensure that you use the interest rate per period so that the FV function's *rate* and *nper* arguments use the same units. For example, if you have a nominal annual interest rate and the investment compounds monthly, you need to divide the annual rate by 12 to get the monthly rate.

If the investment uses continuous compounding, you can calculate the future value by typing **=-B3*EXP(B1*B2)**, which returns $5751.37. You generally use the formula =-*pv**EXP(*nominal_rate**nper), where *nominal_rate* is the nominal interest rate per period and *nper* is the number of periods.

Discussion

This recipe describes two methods you can use to calculate the future value of a fixed-rate lump-sum investment. In general, use the FV function unless the investment uses continuous compounding, in which case the FV function is inaccurate.

10.8 Calculating the Future Value of a Variable-Rate Lump-Sum Investment

Problem

You've invested a lump sum at a variable rate and want to know what its value will be in the future.

Solution

Suppose you want to invest $5,000 for three years with annual compounding. The nominal annual interest is 7% for year 1, 2% for year 2, and 5% for year 3, and you want to know the investment's future value. B1 contains the value of the lump-sum deposit (–5,000) and B2:B4 lists the interest rates (see Figure 10-8).

	A	B	C	D	E	F
1	Initial Deposit:	-$5,000.00		Future value:	=FVSCHEDULE(-B1,B2:B4)	$5,729.85
2	Year 1 Rate:	7%				
3	Year 2 Rate:	2%				
4	Year 3 Rate:	5%				

Figure 10-8. Calculating the future value using FVSCHEDULE

You can solve this problem using the FVSCHEDULE function; type **=FVSCHEDULE(-B1, B2:B4)**, which returns $5729.85. In general, you use the formula =FVSCHEDULE(-*pv*, *schedule*), where *schedule* is an array of rates for each compounding period of the investment. This formula assumes that *pv* is negative because it's an amount that you're paying.

> Notice that the FVSCHEDULE function uses the opposite sign for its *pv* argument compared with the FV function (see Recipe 10.7).

Discussion

The FVSCHEDULE function is helpful when investing a lump sum at a variable rate.

Notice that the *schedule* argument is an array, so if you want to pass the function a list of interest rates that aren't in a cell range, they must be enclosed in curly braces—for example, =FVSCHEDULE(5000, {0.07,0.02,0.05}).

10.9 Calculating the Future Value of an Investment with Regular Deposits

Problem

You make regular deposits in a fixed-rate investment and want to know its future value.

Solution

Suppose you want to deposit $300 at the start of each month in an investment with a nominal annual interest rate of 7% and want to know the value of your investment after a year. B1 contains the nominal interest rate (7%), B2 contains the number of payment periods, and B3 contains the value of the monthly deposit (−300) (see Figure 10-9).

	A	B	C	D	E	F
1	Annual Rate:	7.00%		Future value:	=FV(B1/12, B2, B3, 0, 1)	$3,739.46
2	Payment Periods:	12				
3	Monthly Deposit:	-$300.00				
4	Compounding Periods:	12				

Figure 10-9. Calculating the future value of regular deposits using the FV function

If the investment's payment and compounding periods are the same (for example, monthly), you can solve this problem using the FV function; type **=FV(B1/12, B2, B3, 0, 1)**, which returns $3739.46. In general, you use the formula =FV(*rate, nper, pmt, pv, type*), where *rate* is the interest rate per period, *nper* is the number of periods, *pmt* is the regular amount invested per period, *pv* (optional) is the initial lump-sum deposit, and *type* (optional) refers to when you make regular deposits—omit it or set it to 0 if you want to make deposits at the end of the period (the default), or set it to 1 if you want to make deposits at the start.

If the investment's payment and compounding periods differ, you can no longer cal-
culate the future value using FV because the function uses the same argument for both
variables. However, you can manually calculate the future value by creating an
investment schedule. For example, Figure 10-10 shows the formulas for an invest-
ment schedule calculating the future value of an investment that compounds annu-
ally, where $300 is deposited at the start of each month. The nominal annual interest
rate is 7%.

	A	B	C	D
1	Annual Rate:	0.07		
2	Payment Periods:	12		
3	Monthly Deposit:	-300		
4	Compounding Periods:	1		
5				
6	Month	Cumulative Deposit	Interest Earned per Month	
7	1	=A7*$B$3	=B7*$B$1/$B$2	
8	2	=A8*$B$3	=B8*$B$1/$B$2	
9	3	=A9*$B$3	=B9*$B$1/$B$2	
18	12	=A18*$B$3	=B18*$B$1/$B$2	
19				
20	Future Value:	=-SUM(C7:C18, B18)		

Figure 10-10. Calculating the future value using an investment schedule

Discussion

This recipe builds on Recipe 10.7 and demonstrates more ways of using the FV func-
tion to calculate the future value of an investment. You can use FV if the investment's
payment and compounding periods are the same; if the two periods differ, FV is inac-
curate and you should create an investment schedule instead.

You can combine Recipes 10.7 and 10.9 to find the future value of a
fixed-rate investment that has an initial lump sum and regular
deposits.

10.10 Meeting Investment Goals

Problem

You have an investment goal and want to know the required term, initial lump sum,
regular deposit amount, or investment term.

Solution

Suppose you have an investment and want its future value to be $40,000. B1 contains an estimate for the annual interest (5%), B2 contains the number of months (60), B3 contains the monthly deposit amount (–500), B4 contains the initial deposit (–1000), and the future value (40000) is in B5. The investment type in B6 is set to 0, meaning regular deposits are made at the end of each month (see Figure 10-11).

	A	B	C	D	E	F
1	Interest Rate (annual):	5%		Number Periods:	=NPER(B1/12, B3, B4, B5, B6)	67.19158
2	Months:	60		Initial Deposit:	=PV(B1/12, B2, B3, B5, B6)	-$4,672.86
3	Monthly Deposit:	-$500.00		Monthly Deposit:	=PMT(B1/12, B2, B4, B5, B6)	-$569.31
4	Initial Deposit:	-$1,000.00		Monthly Rate:	=RATE(B2, B3, B4, B5, B6)	0.81%
5	Future Value:	$40,000.00		Annual Rate:	=12*RATE(B2, B3, B4, B5, B6)	9.67%
6	Type:	0				

Figure 10-11. Formulas for meeting investment goals

To determine the investment term, you can use the NPER function to calculate the number of periods (see Recipe 10.4). To calculate the number of months needed to reach a future value of $40,000 using the estimates in B1:B4, for example, you'd type **=NPER(B1/12, B3, B4, B5, B6)**, which returns 67.19.

To figure out the initial deposit, you can use the PV function (see Recipe 10.5). To calculate the initial deposit needed to reach a future value of $40,000, for example, you type **=PV(B1/12, B2, B3, B5, B6)**, which returns –$4672.

To find out what the regular deposits should be, you can use the PMT function (see Recipe 10.1). To calculate the monthly deposits needed to reach a future value of $40,000, for example, you'd type **=PMT(B1/12, B2, B4, B5, B6)**, which returns –$569.31.

To find out the rate per period, you can use the RATE function. To calculate the monthly interest rate needed to reach a future value of $40,000, for example, you'd type **=RATE(B2, B3, B4, B5, B6)**, which returns 0.81%. To convert this to an annual rate, you'd type **=12*RATE(B2, B3, B4, B5, B6)**, which returns 9.67%.

Discussion

This recipe gives you various options to help meet your investment goals. Decide which option best fits your situation and then use the appropriate formula.

10.11 Calculating Net Present Value

Problem

You want to know if a proposed project is financially viable.

Solution

Suppose you have a proposed project and want to know if it will be profitable before proceeding. You can evaluate this by calculating the *net present value*—the net value of its present and future cash flows, discounted to today's. If this value exceeds 0, the project will likely return a profit.

If the cash flows are periodic and occur at the end of each period, you can calculate the net present value using the NPV function. In general, you use the formula =NPV(*rate, values*), where *rate* is a discount rate (for example, an interest rate), and *values* is the list of cash flows in the order in which they occur. If cell B4 holds a discount rate of 10% and B2:F2 lists cash flows of –$20,000, –$5,000, $2,000, $15,000, and $25,000, you'd calculate the net present value by typing **=NPV(B4, B2:F2)**, which returns $4956.81 (see Figure 10-12).

	A	B	C	D	E	F
1	Year:	1	2	3	4	5
2	Net cash flow:	-$20,000.00	-$5,000.00	$2,000.00	$15,000.00	$25,000.00
3						
4	Discount rate:	10%				
5	Net present value:	=NPV(B4, B2:F2)	$4,956.81			

Figure 10-12. Calculating the net present value with the NPV function

> Don't include any initial outlay made at the start of the project in the values passed to the NPV function, which assumes cash flows occur at the end of each period. Instead, compute the net present value by subtracting the initial outlay from the call to NPV. For example, if the initial outlay is $10,000, you'd calculate the net present value by typing **=NPV(B4, B2:F2)-10000**, which returns –$5043.19.

If the cash flows aren't periodic, you can calculate the net present value using the XNPV function. In general, you use the formula =XNPV(*rate, values, dates*), where *rate* is the discount rate, *values* refers to the cash flows, and *dates* gives the date of each one, with the first date appearing first. If cell B4 holds the discount rate, B2:F2 lists the cash flows, and B1:F1 lists the dates March 28 2023, August 15 2023, April 17 2024, June 30 2024, and May 16 2025, for example, you'd calculate the net present

value by typing **=XNPV(B4, B2:F2, B1:F1)**, which returns $10,683.13 (see Figure 10-13).

	A	B	C	D	E	F
1	Date:	28-Mar-2023	15-Aug-2023	17-Apr-2024	30-Jun-2024	16-May-2025
2	Net cash flow:	-$20,000.00	-$5,000.00	$2,000.00	$15,000.00	$25,000.00
3						
4	Discount rate:	10%				
5	Net present value:	=XNPV(B4, B2:F2, B1:F1)	$10,683.13			

Figure 10-13. Calculating the net present value with the XNPV function

Discussion

Calculating the net present value is a valuable way of evaluating whether a proposed project will likely be profitable. You can also use this technique to compare the profitability of multiple projects.

10.12 Calculating the Internal Rate of Return

Problem

You want to find the discount rate for a series of cash flows that makes the net present value equal to 0.

Solution

The *internal rate of return* is the rate for which the net present value (see Recipe 10.11) is 0.

If the cash flows are periodic and occur at the end of each period, you can calculate the internal rate of return using the IRR function. In general, you use the formula **=IRR(**values, guess**)**, where *values* is the list of cash flows in the order in which they occur, and *guess* is an optional initial guess (see "Discussion" on page 250). So if B2:F2 lists cash flows of –$20,000, –$5,000, $2,000, $15,000, and $25,000, you calculate the internal rate of return for this period by typing **=IRR(B2:F2)**, which returns 16.85% (see Figure 10-14).

	A	B	C	D	E	F
1	Year:	1	2	3	4	5
2	Net cash flow:	-$20,000.00	-$5,000.00	$2,000.00	$15,000.00	$25,000.00
3						
4	Internal rate of return:	=IRR(B2:F2)	16.85%			

Figure 10-14. Calculating internal rate of return with the IRR function

If the cash flows aren't periodic, you can calculate the internal rate of return using the XIRR function. In general, you use the formula =XIRR(*values*, *dates*, *guess*), where *values* refers to the cash flows, *dates* gives the date of each one (with the first date appearing first), and *guess* is an optional initial guess. So if B2:F2 lists the cash flows, and B1:F1 lists the dates, you'd calculate the internal rate of return for this period by typing **=XIRR(B2:F2, B1:F1)**, which returns 25.34% (see Figure 10-15).

	A	B	C	D	E	F
1	Date:	28-Mar-2023	15-Aug-2023	17-Apr-2024	30-Jun-2024	16-May-2025
2	Net cash flow:	-$20,000.00	-$5,000.00	$2,000.00	$15,000.00	$19,000.00
3						
4	Internal rate of return:	=XIRR(B2:F2, B1:F1)	25.34%			

Figure 10-15. Calculating internal rate of return with the XIRR function

IRR and XIRR assume that the interest rate you pay on money used in the cash flows is the same as the rate you receive when you reinvest. If these rates are different, you can use the MIRR function to calculate the modified internal rate of return for a series of periodic cash flows, taking these rates into account. Generally, you use the formula =MIRR(*values*, *finance_rate*, *reinvest_rate*), where *values* is the list of cash flows in the order in which they occur, *finance_rate* is the interest rate you pay on the money you use, and *reinvest_rate* is the rate you receive when you reinvest it. So if cell B4 contains the finance rate (10%), B5 contains the reinvestment rate (12%), and B2:F2 lists cash flows of –$20,000, –$5,000, $2,000, $15,000, and $25,000, you'd calculate the internal rate of return by typing **=MIRR(B2:F2, B4, B5)**, which returns 15.91% (see Figure 10-16).

	A	B	C	D	E	F
1	Year:	1	2	3	4	5
2	Net cash flow:	-$20,000.00	-$5,000.00	$2,000.00	$15,000.00	$25,000.00
3						
4	Finance rate:	10%				
5	Reinvestment rate:	12%				
6	MIRR:	=MIRR(B2:F2, B4,B5)	15.91%			

Figure 10-16. Calculating internal rate of return with the MIRR function

Discussion

This recipe gives three options for calculating the internal rate of return.

The IRR and XIRR functions use iteration to calculate the internal rate of return, starting with the value of the optional *guess* argument (or 0.1 if omitted). If the functions can't find a result within 20 tries, they return the #NUM! error value.

In general, you need to provide a *guess* value only if the function returns #NUM! or if the result isn't close to what you expect.

> You must provide the IRR, XIRR, and MIRR functions with at least one positive and one negative value; if you don't, they won't be able to calculate the internal rate of return.

10.13 Calculating Depreciation

Problem

You have an asset and want to calculate its depreciation.

Solution

Suppose you have an asset that costs $6,000. Its expected life is 5 years, at which point you expect to sell it for $200 (the salvage value), and you want to calculate the asset's depreciation for each year. Cell B1 holds the initial cost, B2 holds the salvage value, and B3 holds the asset's expected life (see Figure 10-17).

	A	B	C	D	E	F
1	Initial cost:	$6,000				
2	Salvage value:	$200				
3	Life (years):	5				
4						
5	Period:	1	2	3	4	5
6	Depreciation (SLN):	=SLN(B1, B2, B3)	=SLN(B1, B2, B3)	=SLN(B1, B2, B3)	=SLN(B1, B2, B3)	=SLN(B1, B2, B3)
7		$1,160.00	$1,160.00	$1,160.00	$1,160.00	$1,160.00

Figure 10-17. Calculating straight-line depreciation with the SLN function

The simplest way of calculating the asset's depreciation is to use straight-line depreciation, which allocates the same depreciation value to each period. You calculate this value using the SLN function; type **=SLN(B1, B2, B3)**, which returns $1,160. Generally, you use the formula =SLN(*cost, salvage, life*), where *cost* is the initial cost, *salvage* is the salvage value you expect to sell it for, and *life* is the asset's expected life.

Excel includes several functions you can use to calculate depreciation using different methods:

The SLN *function*
> This calculates depreciation using the straight-line method.

The SYD *function*

>   This returns the sum-of-years' digits depreciation.

The DB *function*

>   This uses the declining balance method to calculate depreciation.

The DDB *function*

>   This uses the double-declining balance method (or another method you specify).

The VDB *function*

>   This calculates depreciation for any period you specify, including partial periods. It uses the double-declining method by default (or another method you specify) and can switch to the straight-line method when the depreciation exceeds the declining balance calculation.

Discussion

This recipe offers several functions to calculate an asset's depreciation. Decide which type of depreciation applies to your situation and then use the corresponding function.

10.14 Getting Stock and Currency Data

Problem

You want to add stock prices and currency exchange rates to Excel but don't know how.

Solution

If you have an active Microsoft 365 subscription, you can use the Stocks and Currencies data types to get stock prices and currency exchange rates.

> Microsoft warns that the Stocks and Currencies data types provide financial market information as is, so you shouldn't use them for trading purposes or advice.

For example, you can get the price of the S&P 500 Index as follows:

1. Type **Name** in A1 and the stock's name (or part of the name) in A2 (for example, **S&P**).

2. Select A1:A2 and choose Insert ⇒ Table to create a table with headings.

3. Select A2, go to the Data menu, and choose Stocks from the Data Types section.

4. If Excel doesn't find an exact match, it opens the Data Selector pane; you can use this to search for stocks, click one to see more details, or select the stock whose data you want to get. In this example, you want to use data from the S&P 500 Index, so click the Select option next to this stock.

5. Excel updates the text in A2 to reflect the stock name and adds a Stocks data type icon 🏛 to the left of the cell's contents (you can click this icon to see more details about the stock). Excel also adds an Insert Data button 🗐 on the right.

6. Click the Insert Data button and select any fields you want to use—for example, Country/region, Instrument type, Price, 52 week high, and 52 week low. Doing so adds the fields and their values to the table (see Figure 10-18).

	A	B	C	D	E	F	
1	Name	Country/region	Instrument type	Price	52 week high	52 week low	
2	🏛 S&P 500 INDEX US		Index	$4,954.23	$4,975.29	$3,808.86	

Figure 10-18. The price returned by the Stocks data type

You can also use data types to find the exchange rate between two currencies. For example, you can get the exchange rate from US dollars to euros as follows:

1. Type **Name** in A1 and the From/To currency pair in A2 (in this case, **USD/EUR**).

2. Select the cells, and choose Insert ⇒ Table to create a table with headings.

3. Select A2, go to the Data menu, and choose Currencies from the Data Types section.

4. Click the Insert Data button and select any fields you want to use (for example, Price); this adds the current exchange rate to the table.

> By default, the data refreshes every five minutes. You can change this setting by right-clicking the stock or currency name and choosing Data Type ⇒ Refresh Settings. You can also refresh the data manually by choosing Data Type ⇒ Refresh or choosing Data ⇒ Refresh All.

Discussion

This recipe is handy for populating a table with stock and currency data from the Bing servers. While a table isn't essential, it makes using the data types more straightforward since they automatically populate the table headings.

10.15 Getting Historic Stock and Currency Data

Problem

You want to add historical stock prices and currency exchange rates to Excel.

Solution

If you have an active Microsoft 365 subscription, you can use the STOCKHISTORY function to get historical stock prices and currency exchange rates.

> Microsoft warns that the STOCKHISTORY function provides financial market information as is, so you shouldn't use it for trading purposes or advice.

For example, if you want to know the exchange rate from US dollars to euros for the past seven days, you can do so by typing the formula **=STOCKHISTORY("USD/EUR", TODAY()-7, TODAY())**; this returns a dynamic array showing each day's closing exchange rate.

You generally use the formula =STOCKHISTORY(*stock, start_date, end_date, interval, headers, columns*), where *stock* identifies the stock or currency From/To pair, *start_date* is the date from which you want to retrieve data, *end_date* (optional) is the date you want to retrieve data to, *interval* (optional) specifies the interval (use 0 for daily, 1 for weekly, and 2 for monthly), *headers* (optional) specifies whether to return headers, and *columns* (optional) specifies which columns to return—use 0 for Date, 1 for Close, 2 for Open, 3 for High, 4 for Low, and 5 for Volume.

Discussion

The STOCKHISTORY function returns a dynamic array of historical stock prices or currency exchange rates from the Bing servers. The dynamic array includes headings in its first row.

10.16 Using Stock Charts

Problem

You have stock data and want to display it on a chart.

Solution

Excel includes several charts you can use to display stock data, depending on what you want to show.

A line chart can display a single metric, such as closing price over time. You can find this by choosing Insert ⇒ Charts ⇒ Insert Line or Area Chart.

If you want to show a stock's high, low, and closing prices over time, optionally with the opening price and daily volume, you can use one of Excel's stock—or candlestick—charts (see Figure 10-19). Four such charts are available, depending on which options you want to include, and you can find them by choosing Insert ⇒ Charts ⇒ Insert Waterfall, Funnel, Stock, Surface, or Radar Chart.
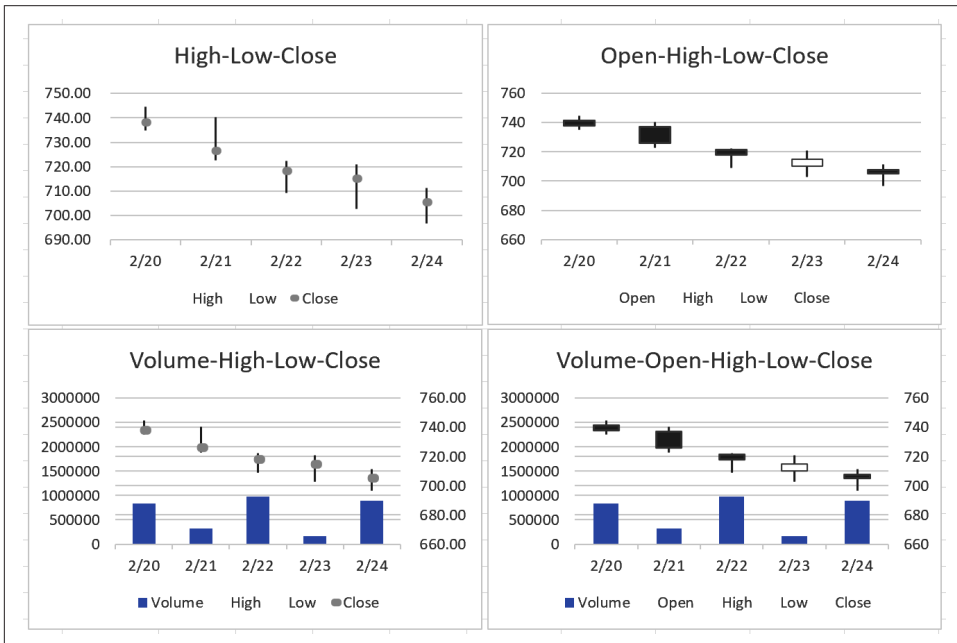
Figure 10-19. The four types of stock charts

Depending on the chart you select, you can show the following data:

High and low prices
   The chart shows the difference between these prices as vertical lines.

Closing price
   The chart displays this metric as a marker. It may not be shown by default if the chart includes the opening price.

Opening price
>   The chart shows the difference between the opening and closing price as an up/down bar, candlestick, or box on the high/low vertical line; if the opening price is higher than the closing price, it's shown as a filled-in down bar, and if the closing price is higher it's shown as an up bar that's not filled in.

Volume
>   This metric, if included, is shown as a column at the bottom of the chart.

Discussion

The solution to this problem depends on the data you want to display; generally, you should choose the chart that best fits your data.

Notice that you can use this recipe with Recipe 10.15 to get historical stock data using the STOCKHISTORY function and display that data on a chart.

10.17 Calculating a Stock's Beta

Problem

You want to determine the volatility of a stock relative to its benchmark.

Solution

Suppose you have historical data for a stock and its benchmark, and you want to calculate the stock's *beta* for that period—its volatility relative to its benchmark.

The first way to calculate the beta is to use the formula =COVARIANCE.S(*stock_values, benchmark_values*)/VAR.S(*benchmark_values*); this calculates the covariance of the two sets of data (see Recipe 9.11), and divides it by the benchmark's variance (see Recipe 8.8).

An alternative method is to use the formula =SLOPE(*stock_values, benchmark_values*); this returns the slope of the linear regression line that best fits the stock and benchmark data points (see Recipe 8.24).

Discussion

A stock's beta indicates its volatility relative to a benchmark—for example, the S&P 500. A beta greater than 1 suggests the stock is more volatile than its benchmark, while a beta less than 1 suggests less volatility.

See Also

If you want to find out how two stocks—or a stock and its benchmark—might be related, see Recipes 8.23 and 8.24.

10.18 Forecasting Linear and Exponential Growth

Problem

You have historic sales values and want to estimate future values, assuming linear or exponential growth.

Solution

Suppose you have the sales figures for a series of months and want to use them to predict the values for future months. B2:B13 lists the sales, A2:A13 lists the months you have sales figures for (numbered 1 to 12), and A14:16 lists the months you want to find sales for (numbered 13 to 16); see Figure 10-20.

	A	B	C	D
		Actual	Linear growth forecast	Exponential growth forecast
1	Month	sales	=TREND(B2:B13,A2:A13,A14:A16)	=GROWTH(B2:B13,A2:A13,A14:A16)
2	1	1000		
3	2	1200		
4	3	990		
5	4	1050		
6	5	1300		
7	6	1500		
8	7	1450		
9	8	1400		
10	9	1600		
11	10	1750		
12	11	1700		
13	12	1800		
14	13		1886.363636	1960.201914
15	14		1961.958042	2072.138018
16	15		2037.552448	2190.466164

Figure 10-20. Linear and exponential growth forecasts using TREND and GROWTH

To generate the forecast values for linear growth, you can use the TREND function as follows:

1. Select the first cell where you want to enter the forecast values (for example, C14).

2. Type the formula **=TREND(B2:B13, A2:A13, A14:A16)**. In general, you use =TREND(*known_ys, known_xs, new_xs*), where *known_ys* is the known y values (the sales figures), *known_xs* is the known x values (the months you have sales figures for), and *new_xs* is the months for which you want to generate future sales figures.

3. Once you've typed the formula, press Enter/Return to generate the linear forecast values.

> TREND returns a dynamic array, so the preceding steps will work if you're using Excel 2021 or Excel 365. For earlier versions of Excel, select the entire output range in step 1 (for example, C14:C16), enter the formula in the top-left cell of this range, and then press Ctrl+Shift+Enter/Return (see Recipe 3.4).

To generate the forecast values for exponential growth, you instead use the GROWTH function. This works similarly to the TREND function, except that you use the formula =GROWTH(*known_ys,known_xs,new_xs*) to generate the exponential forecast values. To generate the forecast sales figures for the months listed in A14:A16, for example, you'd type **=GROWTH(B2:B13, A2:A13, A14:A16)** in cell D14 (see Figure 10-20).

> Like TREND, the GROWTH function returns a dynamic array, so the steps discussed will work if you're using Excel 2021 or Excel 365. For earlier versions of Excel, select the entire output range, enter the formula in the top-left cell of this range, and then press Ctrl +Shift+Enter/Return.

Discussion

This recipe offers a handy technique for generating the predicted future values for linear or exponential growth.

For other types of growth, follow Recipe 8.23 to find the trendline that best fits the data and then use the Forecast Forward option in the trendline's Format pane to extend the trendline. Alternatively, follow Recipe 8.24 to find the equation that best describes the data, which you can then use to calculate the future values.

10.19 Forecasting Seasonal Growth

Problem

You have historic sales values that follow a seasonal or periodic trend and want to estimate future values.

Solution

Suppose the sales figures for a series of quarters follow a seasonal pattern that repeats every four quarters, and you want to use them to predict the values for future quarters. B2:B13 lists the sales, A2:A13 lists the quarters (numbered 1 to 12), and the first row contains labels (see Figure 10-21).

	A	B	C	D	E	F	G
1	Period (quarters)	Sales					
2	1	78000					
3	2	60000					
4	3	45000					
5	4	80000					
6	5	75000					
7	6	70000					
8	7	50000					
9	8	90000					
10	9	80000					
11	10	52000					
12	11	60000					
13	12	115000					

Figure 10-21. Data following a seasonal pattern

If you're using Excel for Windows, you can use Excel's Forecast Sheet to estimate the future values, including confidence intervals for these values. The steps to do so are as follows:

1. Select the data you want to use for the forecast (in this example, A1:B13).

2. Choose Data ⇒ Forecast ⇒ Forecast Sheet.

3. Ensure that the default options are appropriate for your dataset in the Create Forecast Worksheet dialog box (see Figure 10-22). In particular, make sure the Forecast End box displays the last period you want to estimate values for and the Seasonality option is correct for your data; for example, for patterns that repeat every four rows, this needs to be set to 4.
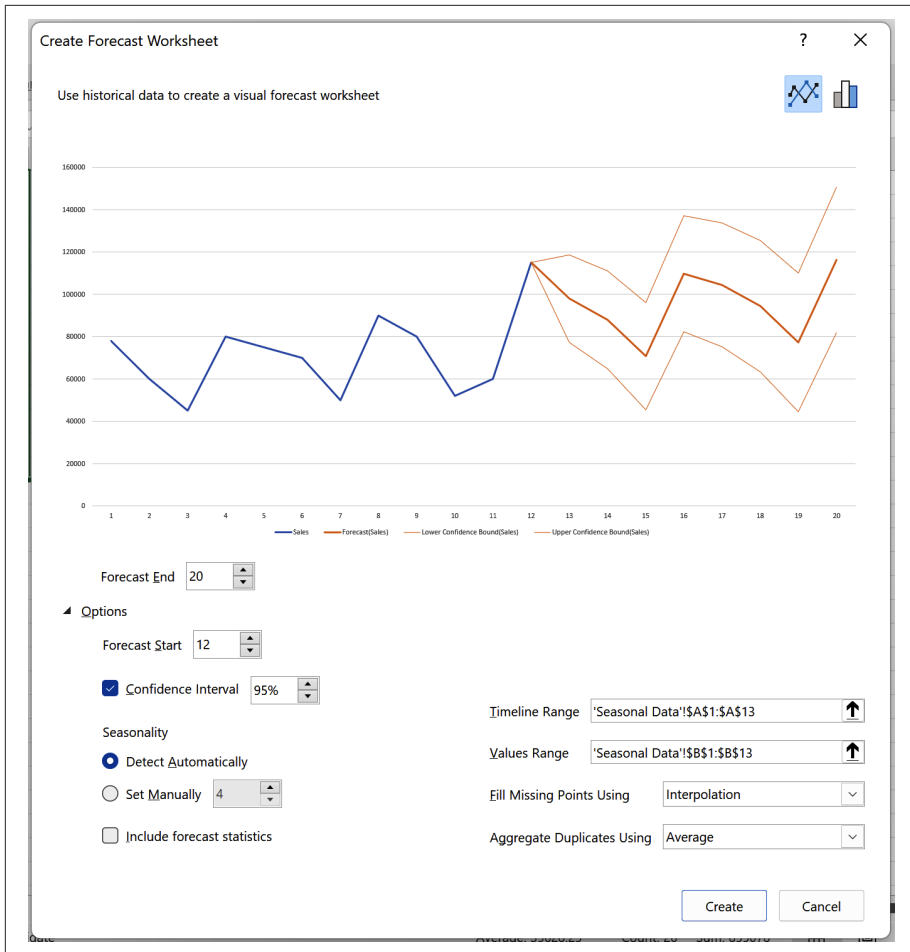
Figure 10-22. The Create Forecast Worksheet dialog box

You can use dates/times or numbers for the periods. However, they should have a consistent step between them, and they can't be zero.

When you click Create, Excel inserts a new worksheet showing the forecast data in a table, with extra rows and columns for the estimated values. Excel also shows this data on a chart (see Figure 10-23).

Period (quarters)	Sales	Forecast(Sales)	Lower Confidence Bound(Sales)	Upper Confidence Bound(Sales)
1	78000			
2	60000			
3	45000			
4	80000			
5	75000			
6	70000			
7	50000			
8	90000			
9	80000			
10	52000			
11	60000			
12	115000	115000	115000.00	115000.00
13		97957.20814	77340.87	118573.54
14		87940.99743	64872.77	111009.23
15		70760.55026	45468.64	96052.46
16		109737.5798	82394.47	137080.69
17		104463.0294	75197.53	133728.53
18		94446.81868	63377.65	125515.99
19		77266.37151	44486.08	110046.67
20		116243.401	81830.71	150656.10

Figure 10-23. The Forecast Sheet output

If you're using Excel for Mac, the Forecast Sheet tool isn't available. You can, however, generate the same forecast using the FORECAST.ETS and FORECAST.ETS.CONFINT functions.

The FORECAST.ETS function generates a forecast value for a target date or period number. To compute the forecast value for quarter 13 where B2:B13 lists the sales and A2:A13 lists quarters 1 to 12, for example, you'd type **=FORECAST.ETS(13, B2:B13, A2:A13)**. In general, you use the formula =FORECAST.ETS(*target_date, values, timeline, seasonality, data_completion, aggregation*), where *target_date* is the date or period number you want to generate the forecast for, *values* refers to the historical values, *timeline* refers to the corresponding dates or numeric periods, *sea sonality* (optional) is the seasonality, which Excel detects automatically by default, *data_completion* (optional) tells Excel how to deal with missing points (it uses the average of neighboring points by default), and *aggregation* (optional) specifies how to aggregate data points with the same timestamp—the default is to use their average.

The FORECAST.ETS.CONFINT function works similarly to the FORECAST.ETS function, except it generates a confidence statistic for a target date or period. You use it to compute a confidence interval for a forecast value, which is the forecast value plus or minus the confidence statistic. To calculate a 95% confidence statistic for quarter 13 where B2:B13 lists the sales and A2:A13 lists quarters 1 to 12, for example, you'd type **=FORECAST.ETS.CONFINT(13, B2:B13, A2:A13)**. In general, you use the formula =FORECAST.ETS.CONFINT(*target_date, values, timeline, confidence,*

seasonality, *data_completion*, *aggregation*), where *confidence* is the confidence level and defaults to 95%.

Discussion

This recipe is a handy way of estimating growth for seasonal data, taking into account any periodic lows and highs.

See Also

See Recipe 8.21 to learn more about confidence intervals.

If you want to analyze seasonal data's general trend, see Recipe 9.5.

PivotTables

PivotTables are one of Excel's most powerful features because they let you interactively analyze, summarize, and explore large amounts of data with just a few mouse clicks. You can slice and dice data in many ways, apply various summaries, and insert custom calculations when you want to go beyond Excel's built-in aggregations.

This chapter contains a collection of recipes designed to help you get the most out of PivotTables and includes the following areas:

- Summarizing data using different aggregations
- Displaying values as percentages, running totals, and more
- Changing a PivotTable's default layout
- Filtering data using slicers and timelines
- Grouping date/time and number fields and manually defining ad hoc groups
- Using calculated fields and items to insert custom formulas
- Using the PivotTable cache to control whether to share groups, calculations, and filters and reinstate deleted tables without using a backup

11.1 Organizing Data for PivotTables

Problem

You want to organize data so you can use it to create a PivotTable.

Solution

Suppose you have a dataset that you want to be able to summarize using a PivotTable. In this situation, the data should have the following structure:

- Unique column headings in the first row, where each column is a unique category (for example, Product, Month, and Amount).
- Each row is a separate record or data item (for example, a sales transaction).
- You should also convert the data to a table because this makes it easier to work with PivotTables.

Figure 11-1 shows two example tables. The leftmost one is appropriately structured for PivotTables because it has unique columns for the Month and Amount categories; the rightmost table is unsuitable because there are separate columns for each month.

Product	Month	Amount		Product	January	February
Coffee	January	1254.15		Coffee	1254.15	4125.41
Tea	January	47862.12		Tea	47862.12	7142.27
Cake	January	1457.57		Cake	1457.57	6543.98
Coffee	February	4125.41		Tea	765.97	
Tea	February	7142.27				
Cake	February	6543.98				
Tea	February	765.97				

Figure 11-1. Two tables containing the same data but with different structures

If you have data that isn't appropriately structured, you can transform it using formulas or Power Query's Unpivot Columns option (see Recipe 15.17).

Discussion

Before creating a PivotTable, it's worth ensuring that its underlying data is appropriately structured. Doing so makes the PivotTable easier to manipulate to return the needed summaries.

11.2 Inserting a PivotTable

Problem

You have a table and want to insert a PivotTable that uses its data.

Solution

Suppose you want to insert a PivotTable based on a table. You can do so by following these steps:

1. Select a single cell in the table and choose Insert ⇒ Tables ⇒ PivotTable or Table Design ⇒ Tools ⇒ Summarize with PivotTable.

2. In the PivotTable dialog box, make sure the Table/Range box displays the table name, choose whether to place the PivotTable in a new worksheet or an existing one, and leave the "Add this data to the Data Model" check box unchecked (see Figure 11-2).
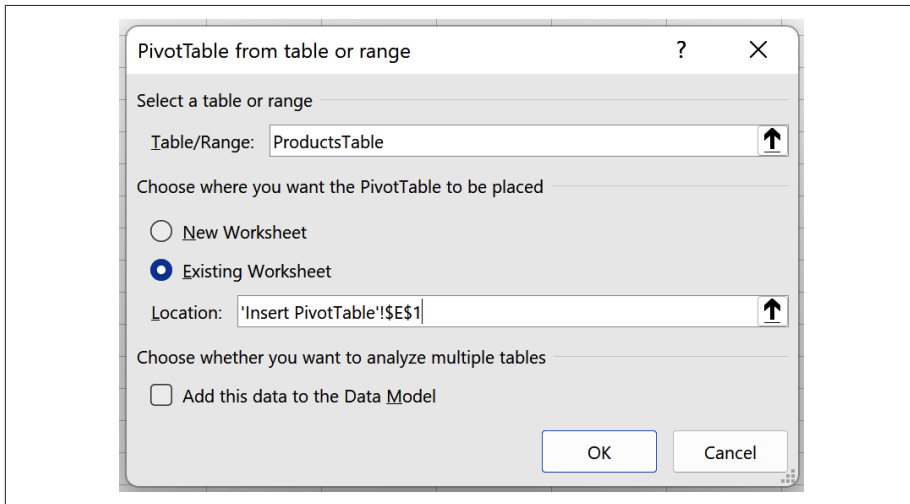
Figure 11-2. Using the PivotTable dialog box to insert a PivotTable

> In addition to using a table as a PivotTable's source, you can use a named or worksheet range. To do so, type or select the name or range in the Table/Range box in step 2.

When you click OK, Excel inserts an empty PivotTable in the specified location and opens the PivotTable Fields pane (see Figure 11-3).
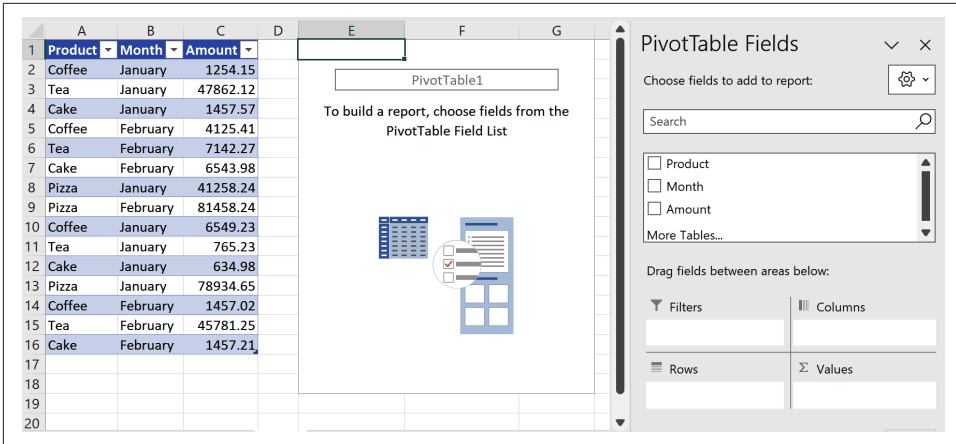
Figure 11-3. An empty PivotTable with the PivotTable Fields pane on the right

Discussion

This recipe outlines how to insert an empty PivotTable based on a table, which you can then populate using Recipe 11.3 and other recipes.

If needed, you can insert a prepopulated PivotTable by selecting a cell in the table and choosing Insert ⇒ Tables ⇒ Recommended PivotTables or Home ⇒ Analysis ⇒ Analyze Data. If you're using Excel 365, you can also use the Analyze Data feature by choosing Home ⇒ Analysis ⇒ Analyze Data to open the Analyze Data pane; select one of the available PivotTables or type a question in the "Ask a question" box.

See Also

By default, you can base a PivotTable on only a single table. However, if you're using Excel for Windows, you can use Excel's data model to use data from multiple related tables. See Chapter 16 to find out more.

11.3 Adding Rows, Columns, and Values

Problem

You have an empty PivotTable and want to add rows, columns, and summary values.

Solution

Suppose you have an empty PivotTable based on a table of sales data with columns named Product, Month, and Amount. To calculate the total amount for each product and month, use the PivotTable Fields pane to specify the PivotTable's contents.

Excel displays the PivotTable Fields pane when you select a cell in the PivotTable. If you've closed the pane, you can reopen it by choosing PivotTable Analyze ⇒ Show ⇒ Field List.

The PivotTable Fields pane includes the following sections:

A list of fields
> This section contains a list of the fields you can use in the PivotTable; these correspond to the column headings in the underlying table.

Filter
> This lets you apply a filter to the PivotTable (see Recipe 11.14).

Columns and Rows
> Use these sections to specify which fields you want to use for the PivotTable's columns and rows.

Values
> This lets you specify which calculations and types of summaries you want the PivotTable to display.

Generally, you populate the PivotTable by adding fields to the Columns, Rows, and Values sections. To calculate the total amount for each product and month, follow these steps:

1. Drag the Product field in the Fields section to the Rows section. This adds a separate row for each unique value in the underlying table's Product column.

2. Drag the Month field to the Columns section. This adds a separate column for each unique value in the underlying table's Month column.

3. Drag the Amount field to the Values section. This calculates the sum of the Amount values for each product and month.

Figure 11-4 shows the fields in the PivotTable Fields pane and the resulting PivotTable.

Since the Amount column contains numeric values, the Values section sums them by default. If you add a text column to the Values section or a numeric column with empty cells, the PivotTable counts them instead. See Recipe 11.9 for how to change these aggregations.

Figure 11-4. A table, its PivotTable, and the PivotTable Fields pane

Discussion

This recipe gives an overview of how you use the PivotTables Fields pane to control the PivotTable's rows, columns, and values. See Recipes 11.4, 11.9, and 11.10 for more ways of using these sections.

11.4 Using Secondary Rows

Problem

You have a PivotTable and want to use two or more fields for its rows.

Solution

Suppose you have an empty PivotTable based on a table of sales data with columns named Category, Product, Month, and Amount. You want to calculate the total amount for each category, product, and month.

You can solve this problem using secondary rows. You add secondary rows by following these steps:

1. Drag the Category field in the Fields section to the Rows section; then drag the Product field to the Rows section, placing it underneath the Category field. This adds secondary rows to the PivotTable so each Product value appears under the relevant Category.

2. Drag the Month field to the Columns section and the Amount field to the Values section; this calculates the sum of the amount values for each category, product, and month.

Figure 11-5 shows the resulting PivotTable and its PivotTable Fields pane.

	F	G	H	I
Sum of Amount	Column Labels			
Row Labels	January		February	Grand Total
⊟Drink		56430.73	12724.7	69155.43
Coffee		7803.38	5582.43	13385.81
Tea		48627.35	7142.27	55769.62
⊟Food		122285.44	135240.68	257526.12
Cake		2092.55	8001.19	10093.74
Pizza		120192.89	81458.24	201651.13
Tea			45781.25	45781.25
Grand Total		178716.17	147965.38	326681.55

PivotTable Fields ∨ ✕

Choose fields to add to report: ⚙ ▾

Search 🔍

☑ **Category**
☑ **Product**
☑ **Month**
☑ **Amount**

Drag fields between areas below:

▼ Filters   ▥ Columns
                   | Month ▾ |

≡ Rows   Σ Values
| Category ▾ | | Sum of Amount ▾ |
| Product ▾ |

Figure 11-5. Adding secondary rows to a PivotTable

You can also add secondary rows by drilling down into the data. Double-click a row or column item in the PivotTable; then use the Show Detail dialog box to specify which data you want to display.

Discussion

Secondary rows let you slice, dice, and group your PivotTable data and include summary values for different combinations. You can also include secondary columns by dragging extra fields to the Columns section.

11.5 Refreshing a PivotTable's Data

Problem

You have a PivotTable based on a table and want to refresh the PivotTable when you update the table's data.

Solution

A PivotTable provides a snapshot of the table's data, so if you update the underlying table, you need to manually refresh the PivotTable to update its summaries. You can refresh a PivotTable using the following methods:

- Right-click the PivotTable and choose Refresh.
- Select a cell in the PivotTable, choose PivotTable Analyze ⇒ Data ⇒ Refresh ⇒ Refresh or Refresh All.
- Choose Data ⇒ Queries & Connections ⇒ Refresh ⇒ Refresh or Refresh All.

Generally, choose the Refresh option to refresh the selected PivotTable and Refresh All to refresh all the data in the workbook.

To automatically refresh a PivotTable when you open the workbook, select a cell in the PivotTable and choose PivotTable Analyze ⇒ PivotTable ⇒ Options to open the PivotTable Options dialog box. Then select the Data tab and place a check in the "Refresh data when opening the file" check box.

Discussion

This recipe shows you how to refresh a PivotTable when you update the underlying data source. Selecting the Refresh option is more efficient than Refresh All but may lead to out-of-date data in other PivotTables.

Behind the scenes, refreshing a PivotTable's data updates its cache; see Recipe 11.28 for more details.

11.6 Moving a PivotTable

Problem

You have a PivotTable and want to move it elsewhere.

Solution

To move a PivotTable, select one of its cells and then choose PivotTable Analyze ⇒ Actions ⇒ Move PivotTable to open the Move PivotTable dialog box. Select a new location for the PivotTable; then click OK.

Alternatively, select the entire PivotTable, press Ctrl+X to cut it, select its new location, and press Ctrl+V to paste it.

Discussion

This recipe provides two quick ways of moving a PivotTable to a new location. Which method you use is mainly down to personal preference.

11.7 Changing a PivotTable's Appearance

Problem

You have a PivotTable and want to tweak its style or appearance.

Solution

You can make general changes to a PivotTable's appearance by selecting a cell in the PivotTable, choosing the Design menu, and using the available options. These options include the following:

Subtotals and Grand Totals
These options let you switch the PivotTables grand totals and subtotals on or off.

Report Layout
This lets you choose a different layout for the PivotTable and repeat item labels.

Blank Rows
If you're using secondary rows, you can use this option to separate each group of rows with a blank row.

Row Headers and Column Headers
These options let you change the format of the PivotTable's row and column headers.

Banded Rows and Banded Columns
These let you shade alternate rows or columns to make the PivotTables values easier to digest.

PivotTable Style
This provides a style gallery so you can quickly change the PivotTable's style and color scheme.

Discussion

This recipe gives an overview of changing a PivotTable's general appearance. Try experimenting with the options until the PivotTable looks the way you want.

See Also

You can also apply conditional formatting to PivotTables; see Recipe 1.9.

11.8 Changing the Default Layout

Problem

You want to override the default style Excel uses for new PivotTables.

Solution

If you're using Excel 2019 or later on Windows, you can change the default style Excel applies to any new PivotTables you create. To do so, choose File ⇒ Options ⇒ Data; then click the Edit Default Layout button to open the Edit Default Layout dialog box (see Figure 11-6).
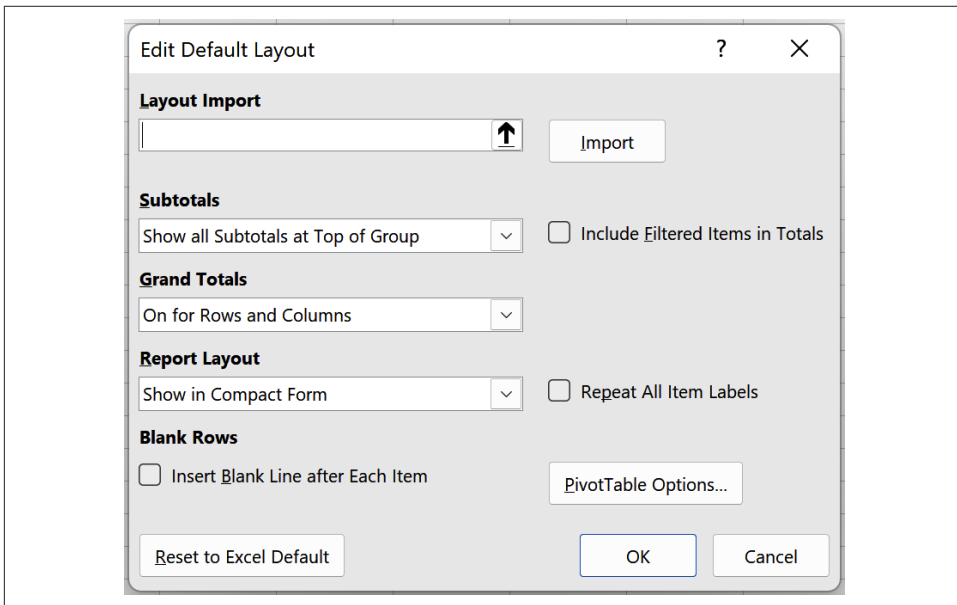
Figure 11-6. The Edit Default Layout dialog box

The dialog box includes a series of options that correspond to those used by Recipe 11.7, along with these additional options:

Layout Import
Use this option if you have an existing PivotTable whose style you want to make the default. Select a cell from the PivotTable and then click the Import button to import its settings.

PivotTable Options
This launches the PivotTable Options dialog box.

Reset to Excel Default
Use this option to restore Excel's default PivotTable settings.

Discussion

This time-saving recipe is handy if you want any new PivotTables you create to have a specific style. This facility, however, is available only if you're using Excel 2019 or later on Windows.

11.9 Changing Value Aggregations

Problem

You have a PivotTable and want to change the summary value aggregations it displays.

Solution

By default, Excel sums any numeric fields you add to the Values section of its Pivot-Table Fields pane. You can, however, change this aggregation to display the average, highest value, smallest value, and more.

To change the value aggregation, follow these steps (see Figure 11-7):

1. In the PivotTable Fields pane's Values section, click or right-click the field (depending on your version of Excel) to open its shortcut menu. Then choose the Value Field Settings option to open the Value Field Settings dialog box.

2. In the Summarize Values By tab, select the type of calculation you want to use (for example, Average).

3. Click OK to close the dialog box and update the PivotTable.
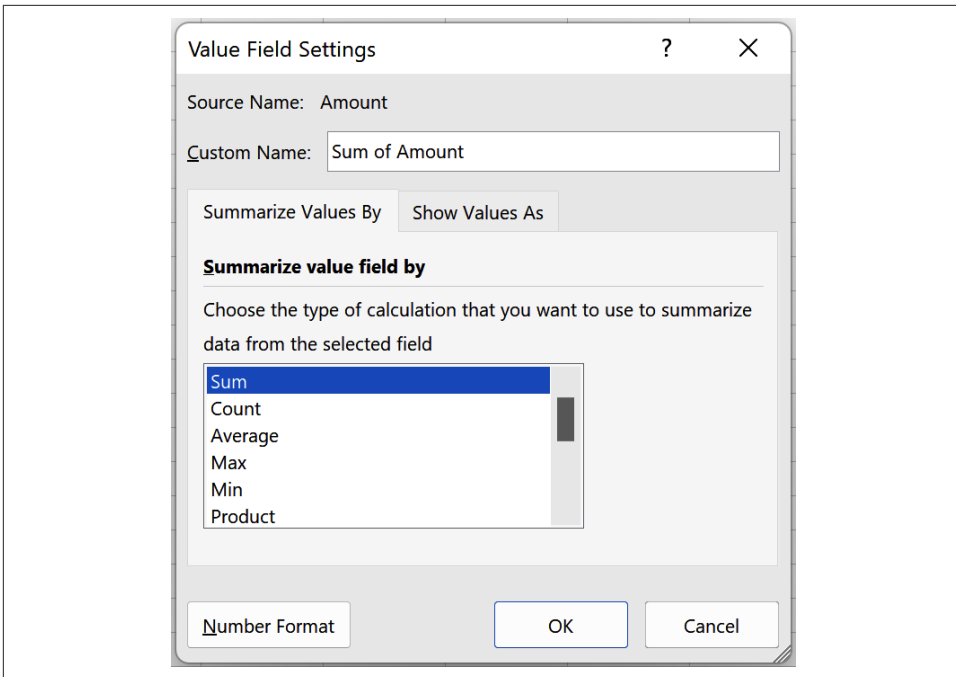
Figure 11-7. The Value Field Settings dialog box

You can also use the Value Field Settings dialog box to change the aggregation's name and format its numbers. In the dialog box, use the Custom Name box to change the default name and click the Number Format button to open the Format Cells dialog box and format the values (see Recipe 1.3).

Discussion

This recipe offers a handy way of changing Excel's default calculations for PivotTable values. Note that you can also use it to add multiple aggregations of the same field: add the field to the PivotTable Fields pane's Values section multiple times and then use the Value Field Settings dialog box to apply a different calculation.

11.10 Showing Different Value Calculations

Problem

You have a PivotTable and want to display its values as percentages, running totals, and more.

Solution

Follow the steps in Recipe 11.9 to open the Value Field Settings dialog box and select an aggregation; then select the Show Values As tab and choose one of the available options. These options are as follows:

No Calculation
> This displays the value of the aggregation without any additional calculations.

% of Grand Total
> This displays values as a percentage of their grand total.

% of Column Total and % of Row Total
> These display the values in each column or row as a percentage of the column or row total.

% Of
> This displays values as a percentage of the value of the Base Item in the Base Field (which you select).

% of Parent Row Total and % of Parent Column Total
> These are handy if your PivotTable uses secondary rows or columns because they display the value as a percentage of the parent row or column total. Behind the scenes, they use the calculation (`item value`)/(`parent row or column total`).

% of Parent Total
> Use this option with secondary rows or columns to display the value as a percentage of the parent item in the Base Field (which you select).

Difference From and % Difference From
> These options display the difference and percentage difference between the value and the Base Item in the Base Field (which you select).

Running Total in and % Running Total in
> These display running totals and percentages for items in the Base Field (which you select); see Recipe 8.2.

Rank Smallest to Largest and Rank Largest to Smallest
> These display the rank of each value, starting with a rank of 1 (see Recipe 8.5).

Index
> This option calculates an index for each value using the formula (`value * (Grand Total)`)/((`Row Grand Total`) * (`Column Grand Total`)).

Discussion

This recipe tweaks the way how Excel displays values in PivotTables to show values as percentages, running totals, and more.

11.11 Creating Custom Subtotals

Problem

You have a PivotTable with secondary rows or columns and want to show a custom subtotal with a different aggregation from that of the PivotTable's values.

Solution

Suppose you have a PivotTable that uses Category and Product fields for its rows and the sum of the Amount field for its values to show the total amount for each category and product. You want to adjust the Category subtotal so that it calculates the count of its values while keeping the Product sums intact.

You can solve this problem by adjusting the Category field's settings as follows:

1. In the PivotTable Fields pane's Rows section, click or right-click the Category field (depending on your version of Excel) to open its shortcut menu. Then choose the Field Settings option to open the Field Settings dialog box.

2. In the Subtotals & Filters tab, select the Custom Subtotals option; then choose the type of calculation you want to apply—for example, Count.

3. Click OK to close the dialog box and update the PivotTable.

Figure 11-8 shows the Field Settings dialog box and the resulting PivotTable, including its PivotTable Fields pane.

 Some users may find it confusing to see subtotals that use a different aggregation from that of the PivotTable's values. Consider using Recipe 11.9 instead to add multiple aggregations to the PivotTable Fields pane's Values section.
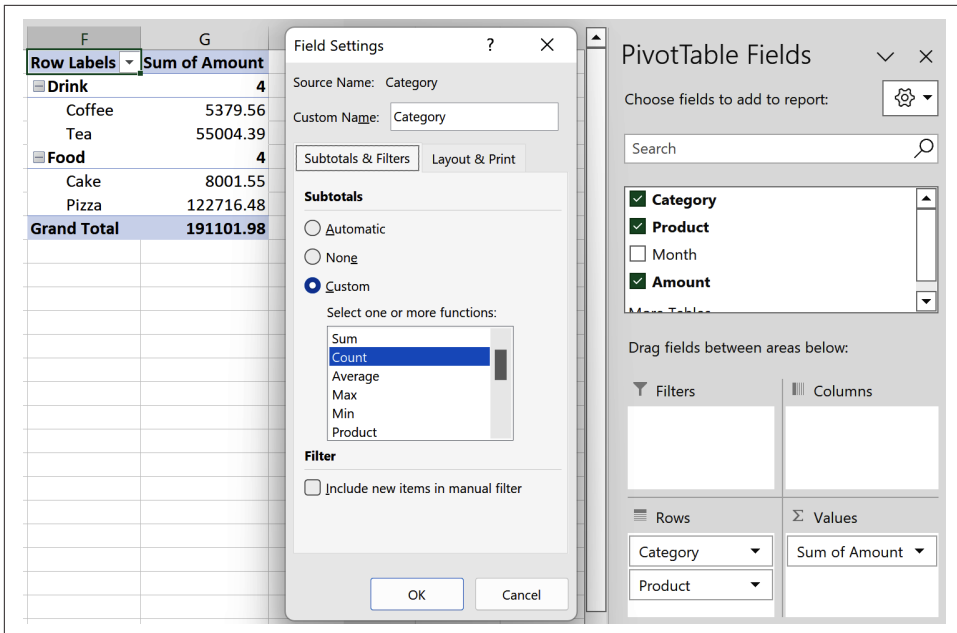
Figure 11-8. Using the Field Settings dialog box to calculate the Count for the Category field, while still using Sum for Product

Discussion

This little-known technique demonstrates creating custom subtotals with aggregations that differ from the PivotTable's values. Note that you'll see these subtotals only in PivotTables that use secondary rows or columns.

11.12 Sorting Data

Problem

You have a PivotTable and want to sort its data.

Solution

By default, Excel adds sort and filter drop-down arrows for the PivotTable's row and column labels, and you can use these to specify how to sort the data.

If the PivotTable's sort and filter arrows are missing, you can add them to the headers by selecting a cell in the PivotTable and choosing PivotTable Analyze ⇒ Show ⇒ Field Headers.

To sort a row or column's items, open the sort and filter menu by clicking the drop-down arrow, select a field (if you're using secondary rows or columns), and then use the available options to specify the sort order. You can sort items in ascending or descending order, choose More Sort Options to sort by the PivotTable's value calculations, or define a custom sort order.

When you sort a PivotTable's items in ascending or descending order, Excel first checks whether they form part of a custom list (see Recipe 1.13). So if a PivotTable contains the items Jan and Feb, and there's a custom list of months that includes these, the PivotTable will use the list to sort them.

Discussion

Excel automatically sorts PivotTables by default. This recipe, however, lets you override the default sort order so you can sort values in whichever order you prefer.

11.13 Moving Items Manually

Problem

You have a PivotTable and want to manually move its items instead of sorting them.

Solution

To manually move an item in the PivotTable's rows or columns, right-click the item and use the Move options to reposition it. Depending on the item's current position, you can generally move it up, down, or to the beginning or end of its group.

Discussion

This recipe is handy if you want to manually sort a PivotTable in a specific way. For example, you can use it to move a calculated item (see Recipe 11.23) to the top or bottom of its group.

11.14 Filtering Data
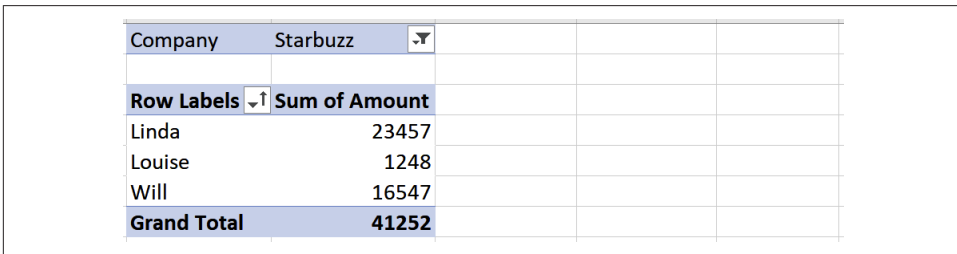
Problem

You have a PivotTable and want to filter its data.

Solution

The most comprehensive way of filtering a PivotTable is to use the sort and filter drop-down arrows Excel adds to its row and column labels (see Recipe 11.12). To apply a filter using this method, open the sort and filter menu by clicking the drop-down arrow, select a field (if you're using secondary rows or columns), and use the available options to filter the data's labels or values.

A second way of filtering a PivotTable is to use the PivotTable Fields pane's Filter section. When you add a field to this section, Excel adds a separate filter above the Pivot-Table, which you can use to filter the data (see Figure 11-9).

Company	Starbuzz ▼	
Row Labels ▾↑	Sum of Amount	
Linda	23457	
Louise	1248	
Will	16547	
Grand Total	**41252**	

Figure 11-9. A PivotTable with the Company field added to the PivotTable Fields pane's Filter section

> You can't add a field to the Filter section and include it in the Rows or Columns sections. If you try to do so, Excel automatically removes the field from the other sections.

Another way of filtering the data is to use a *slicer*: a floating object that lets you choose which values to display. To insert a slicer, select a cell in the PivotTable, choose PivotTable Analyze ⇒ Filter ⇒ Insert Slicer, select which fields you want to include slicers for, and click OK. To use the slicer, click the value you want the PivotTable to display or use the multiselect button to select multiple values. You can also select multiple values by pressing the Ctrl or Cmd key while selecting values (see Figure 11-10).

Sum of Amount	Column Labels ▼		
Row Labels ▼↑	CatChat	Starbuzz	Grand Total
Linda	67643	23457	91100
Louise	6742	1248	7990
Will	690324	16547	706871
Grand Total	**764709**	**41252**	**805961**

Company ▣ ▽✗
- CatChat
- Manic Mango
- Starbuzz

Figure 11-10. A PivotTable filtered using a Company slicer

If the PivotTable contains any date fields, you can use a *timeline* to filter their values: a floating object that works similarly to a slicer but filters dates by year, month, calendar quarter, or day. To insert a timeline, select a cell in the PivotTable, choose Pivot-Table Analyze ⇒ Filter ⇒ Insert Timeline, select which date fields you want to include timelines for, and click OK. To use the timeline, select the type of period you want to filter the PivotTable by and then select the period (see Figure 11-11).

Order date ▽✗

Q1 2023   MONTHS ▼

2023

JAN   FEB   MAR   APR   MAY   JUN   JUL   AUG   SE

◄   ►
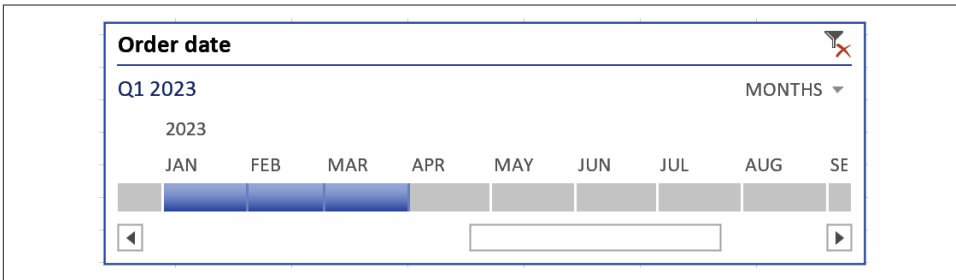
Figure 11-11. A timeline filtering a PivotTable by month

A timeline can filter a PivotTable by adjacent periods but not non-adjacent ones. If you need to filter by nonadjacent periods, consider grouping the PivotTable by date (see Recipe 11.16) and use slicers to filter the groups.

Discussion

This recipe describes several methods to filter a PivotTable's data. The primary considerations are how you want to filter the PivotTable and the intended audience; slicers and timelines are generally easier to use and interpret, but the options in the sort and filter drop-down menus are more powerful.

Timelines can filter data only by calendar quarters, not fiscal quarters. To filter by fiscal quarters, use Recipe 6.4 to derive the quarter, add it to the underlying table as a calculated column, then filter the PivotTable by this field. Alternatively, see Recipe 16.13 (if you're using Excel for Windows).

11.15 Using a Filter to Create Multiple PivotTables

Problem

You have a PivotTable and want to quickly create separate PivotTables for each of a field's values.

Solution

Suppose you have a PivotTable with fields named Company, Product, Month, and Amount and want to create a separate PivotTable for each Company value. You can do so by following these steps:

1. In the PivotTable Fields pane, drag the field you want to create new PivotTables for to the Filters section—in this example, the Company field.

2. Choose PivotTable Analyze ⇒ PivotTable, click the Options button's drop-down arrow, then select Show Report Filter Pages; this opens the Show Report Filters dialog box.

3. The Show Report Filters dialog box shows a list of any fields in the PivotTables Fields pane's Filters section, and you use this list to specify which one you want to create new PivotTables for. In this example, you want to create separate PivotTables for each Company value, so place a check in the Company check box.

4. When you click OK, Excel creates a new PivotTable for each value, placing each in a separate worksheet.

Discussion

This recipe offers a quick way of creating separate PivotTables for each field item. If you add a new value to the field, Excel doesn't automatically create a new PivotTable, so you must do this manually.

11.16 Grouping by Date/Time

Problem

You have a PivotTable with a date field and want to choose whether to group calculations by year, quarter, month, and so on.

Solution

By default, Excel automatically groups any date or time fields you use for its rows and columns. For example, if you have a PivotTable with a date field and you add the field to the PivotTable Fields pane's Rows section, Excel automatically groups its values by year, quarter, and month. Similarly, if you add a time field to the Rows section, Excel automatically groups its values by hour.

> Turn off automatic grouping for date/time fields in Excel for Windows by choosing File ⇒ Options ⇒ Data and unchecking the "Disable automatic grouping of Date/Time columns in PivotTables" option. If you're using Excel for Mac, you can find this option by choosing Excel ⇒ Preferences ⇒ Tables & Filters.

Once you've added a date/time field to the PivotTable, you can manually adjust how it's grouped. Select one of the date/time cells, choose PivotTable Analyze ⇒ Group ⇒ Group Selection to open the date/time Grouping dialog box, and then select how you want to group the data by clicking—or holding down the Ctrl key and clicking—the relevant options. When you click OK, Excel applies the grouping (see Figure 11-12).
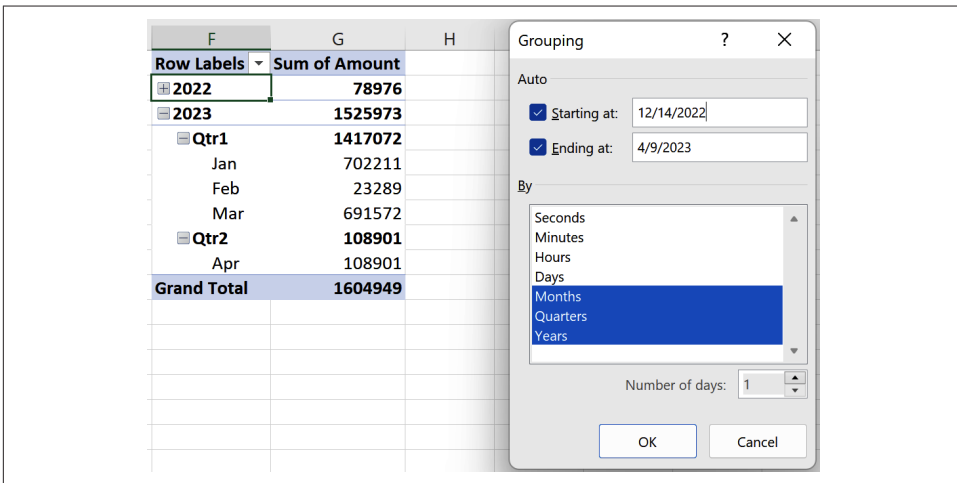
Figure 11-12. The Grouping dialog box for date/time fields and the corresponding Pivot-Table

You can also use the date/time Grouping dialog box to specify the first and last dates the PivotTable should display and group results by a certain number of days. To organize values in three-day groups, select the Days option and type **3** in the "Number of days" box.

Finally, you can ungroup the dates by selecting a cell containing one of the group's items and then choosing PivotTable Analyze ⇒ Group ⇒ Ungroup.

Discussion

This recipe gives a helpful overview of how to group a PivotTable using date/time fields. Note that you can use this recipe with Recipe 11.14 to filter any groups you've added.

See Also

Excel adds any groups you create to every PivotTable sharing the same cache. See Recipe 11.28 for more details.

11.17 Grouping by Number

Problem

You have a PivotTable with a number field and want to group summary values by numeric ranges.

Solution

Suppose you have a PivotTable with Student and Score fields, where Score contains numeric values and Student contains text values, and you want to count how many students scored 1 to 10, 11 to 20, 21 to 30, and so on. You can do so by following these steps:

1. In the PivotTable Fields pane, drag the Score field to the Rows section and the Student field to the Values section.

2. Select one of the Score cells in the PivotTable; then choose PivotTable Analyze ⇒ Group ⇒ Group Selection to open the number Grouping dialog box.

3. In the Grouping dialog box, use the Starting At and Ending At boxes to specify the lower and upper limits of the groups and the By box to specify the interval for each group. To group values from 1 to 100 where the interval for each group is 10, you'd type **1** in the Starting at box, **100** in the Ending At box, and **10** in the By box.

4. Click OK to apply the changes to the PivotTable (see Figure 11-13).
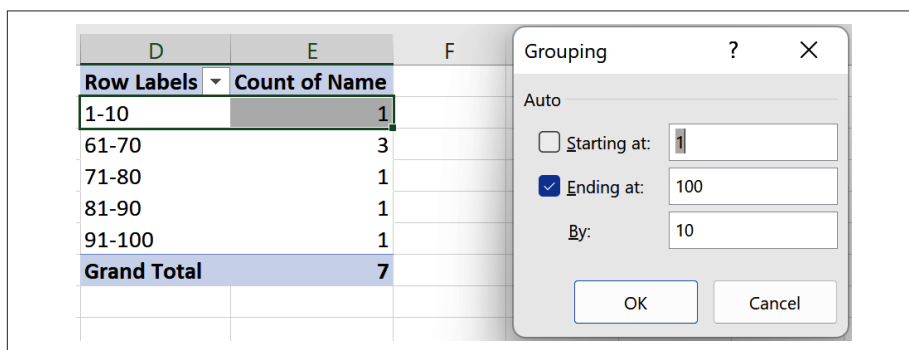
Figure 11-13. The Grouping dialog box for number fields and the corresponding PivotTable

To remove the groups, select a cell containing one of the group's items, then choose PivotTable Analyze ⇒ Group ⇒ Ungroup.

Discussion

This recipe is similar to Recipe 11.16 except that it shows you how to group a Pivot-Table using number fields.

See Also

Excel adds any groups you create to every PivotTable sharing the same cache. See Recipe 11.28 for more details.

11.18 Manually Grouping by Text Values

Problem

You have a PivotTable with a text field and want to manually group calculations by selected text values.

Solution

Suppose you have a PivotTable with Product and Amount fields where Product contains text values, Amount contains numeric values, and you want to create an ad hoc group for some of the products and show the sum of their amount. You can do so by following these steps:

1. In the PivotTable Fields pane, drag the Product field to the Rows section and the Amount field to the Values section.

2. In the PivotTable, select the cells containing the Product items you want to appear in the first group. Then choose PivotTable Analyze ⇒ Group ⇒ Group Selection to group them.

3. Repeat step 2 to create groups for the remaining Product items.

4. Name each group by selecting its group label and typing its name in the cell or formula bar (see Figure 11-14).

Category	Product	Month	Amount		Row Labels	Sum of Amount
Drink	Coffee	January	1254.15		Cream Tea	63005.94
Drink	Tea	January	47862.12		Tea	55004.39
Food	Scone	January	1457.57		Scone	8001.55
Drink	Coffee	February	4125.41		Other	128096.04
Drink	Tea	February	7142.27		Coffee	5379.56
Food	Scone	February	6543.98		Pizza	122716.48
Food	Pizza	January	41258.24		Grand Total	191101.98
Food	Pizza	February	81458.24			

Figure 11-14. A PivotTable showing two manual groups, along with the table the PivotTable is based on

To remove the groups, select the groups you want to ungroup, then choose PivotTable Analyze ⇒ Group ⇒ Ungroup.

Discussion

This recipe is handy if you want to produce similar results to using secondary rows (see Recipe 11.4) but can't use another field's values to derive the groups. Instead, you can manually add ad hoc groups.

See Also

Excel adds any groups you create to every PivotTable sharing the same cache. See Recipe 11.28 for more details.

11.19 Including Groups with Missing Data

Problem

You have a PivotTable whose rows are grouped and want the PivotTable to display every group, including ones with no data.

Solution

Suppose you have a PivotTable with Student and Score fields, and you've used Recipe 11.17 to add a group of the Score field to the rows and a count of the Student field to the Values. You want the PivotTable to display all the Score groups, including any without data.

By default, PivotTables display only groups that contain values. However, you can override this behavior by following these steps:

1. In the PivotTable Fields pane's Rows section, click or right-click the Score field (depending on your version of Excel) to open its shortcut menu. Then choose the Field Settings option to open the Field Settings dialog box.

2. In the Layout & Print tab, place a check in the "Show items with no data" check box.

3. Click OK to close the dialog box and update the PivotTable.

Discussion

This recipe offers a convenient way of ensuring that PivotTables show rows for all groups, including ones with no data. Note that by default, any rows with no data display their values as empty cells instead of zeros, but you can override this behavior using Recipe 11.20.

11.20 Changing the Format of Empty Cells

Problem

You have a PivotTable with empty cells and want to change their format to display zeros or N/A.

Solution

If a PivotTable has no value for a particular cell, that cell remains empty by default. However, you can change the default format to replace empty cells with a specified number or text value by following these steps:

1. Select a cell in the PivotTable and choose PivotTable Analyze ⇒ PivotTable ⇒ Options to open the PivotTable Options dialog box.

2. In the Layout & Format tab, ensure that the "For empty cells show" check box is checked; then specify the number or text you want the PivotTable to display in any empty cells. To display a zero, for example, you'd type **0**.

3. Click OK to close the dialog box and update the PivotTable.

Discussion

This recipe provides a quick way of changing how a PivotTable displays cells with no values. It's convenient when combined with Recipe 11.19 because you can format any group's values with no available data.

11.21 Using Calculated Fields

Problem

You have a PivotTable and want to add a custom formula that refers to one or more fields.

Solution

Suppose you have a table with columns named Salesperson, Units, and Amount. You want to create a PivotTable showing each salesperson's bonus, which is 2% of their total amount if they sell more than 5,000 units. To solve this problem, you can create a PivotTable that uses a *calculated field*: a custom formula that uses one or more of the PivotTable's fields.

You create the PivotTable, including the calculated field, as follows (see Figure 11-15):

1. Create a PivotTable and add the Salesperson field to the Rows section.
2. Select a cell in the PivotTable and choose PivotTable Analyze ⇒ Calculations ⇒ Fields, Items, & Sets ⇒ Calculated Field to open the Insert Calculated Field dialog box.
3. Type the calculated field's name in the Name box (for example, **Bonus**).
4. Add the custom formula in the Formula box. For example, to return 2% of the amount if more than 1,000 units are sold, you would use the formula =IF (Units>5000, Amount*2%, 0).

> You can add a field name to a calculated field formula by double-clicking it in the dialog box's Fields list instead of typing it. This helps avoid any typing errors, and it's particularly helpful for field names containing spaces, which need to be put in single quotes, such as 'Unit Price'.
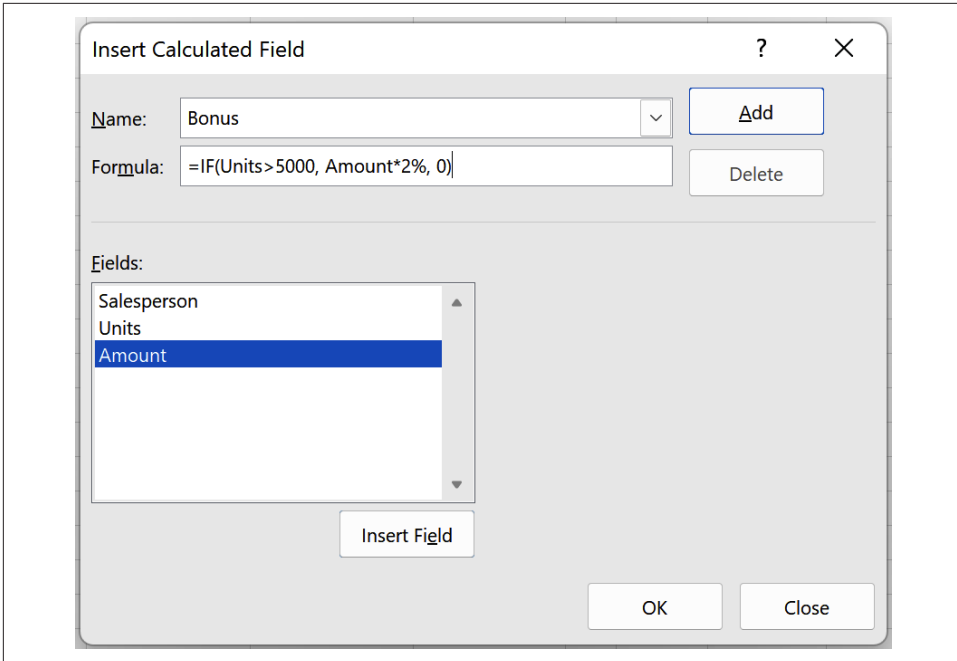
Figure 11-15. The Insert Calculated Field dialog box

When you click the OK button, Excel closes the Insert Calculated Field dialog box, adds the new calculated field to the PivotTable Fields pane's field list, adds it to the Values section (as *Sum of Bonus*), and displays the results in the PivotTable (see Figure 11-16).
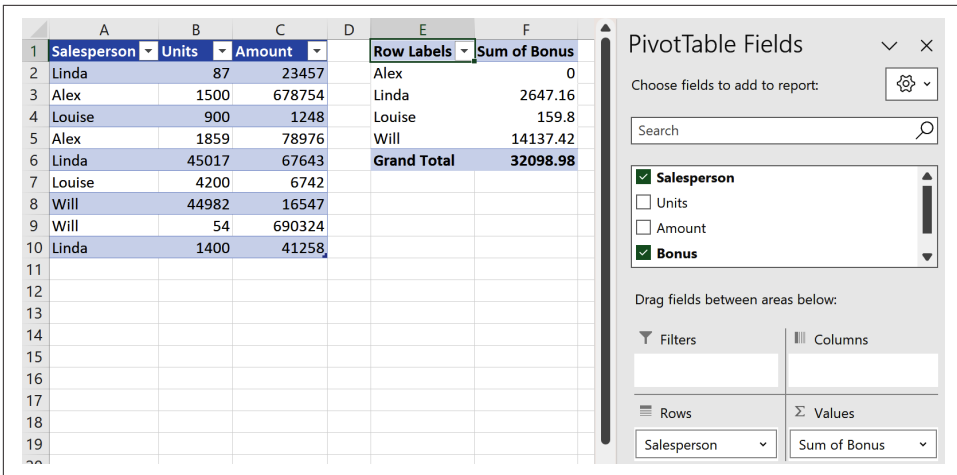
Figure 11-16. The table and PivotTable, including the Bonus calculated Field

To remove a calculated field from the PivotTable, remove it from the PivotTable Fields pane's Values section. You can also modify or permanently delete it as follows:

1. Select a cell in the PivotTable and choose PivotTable Analyze ⇒ Calculations ⇒ Fields, Items, & Sets ⇒ Calculated Field to open the Insert Calculated Field dialog box.

2. Select the calculated field from the Name box drop-down list.

3. Either change the formula and click Modify or delete the formula by clicking Delete. Then click Close to close the dialog box.

> Excel adds any calculated fields you create to the PivotTable cache (see Recipe 11.28) so they're automatically available to all other PivotTables sharing the same cache. So if you modify or permanently delete a calculated field in one PivotTable, it affects the other PivotTables.

Calculated fields have the following limitations:

- You can add them only to the PivotTable Fields pane's Values section, not the Rows, Columns, or Filters sections.

- They can't refer to worksheet cells or any PivotTable totals or subtotals.

- Their formulas use the sum of any field they refer to, and any functions or formulas you include use these sums. So a calculated field with the formula `=Amount` implicitly returns the calculation `=SUM(Amount)`, and the calculated field formula `=IF(Units>5000, Amount*2%, 0)` implicitly uses `=IF(SUM(Units)>1000, SUM(Amount)*2%, 0)`.

- Changing a calculated field's aggregation in the PivotTable Fields pane's Values section doesn't affect its return value. For example, if you change the aggregation from Sum to Count, the calculated field stays unchanged.

> Calculated fields are easily misunderstood because they implicitly sum any fields they reference instead of using the underlying table values. So if a PivotTable contains fields named Quantity and UnitPrice, the calculated field formula `=Quantity*UnitPrice` implicitly adds the calculation `=SUM(Quantity)*SUM(UnitPrice)` to the PivotTable instead of `=SUM(Quantity*UnitPrice)`.

Discussion

This recipe gives an overview of using calculated fields to add custom formulas to a PivotTable that refer to one or more fields. They work similarly to calculated columns in a table (see Recipe 2.19), except they implicitly sum any fields they refer to, so may return different results.

See Also

Another way of adding custom formulas is to use calculated items; see Recipe 11.23.

11.22 Using Calculated Fields to Count Items

Problem

You have a PivotTable and want to add a custom formula that refers to the count of one or more fields.

Solution

A calculated field (see Recipe 11.21) implicitly uses the sum of each field in its formula, so the formula =Amount in a calculated field implicitly calculates =SUM(Amount). If you want to use the count of a field's items instead of its sum, you can use the following workaround:

1. Add a new column named, for example, CountItems, to the PivotTable's underlying table. Then type **=1** in the column's first row and press Enter/Return to insert 1 in each row.

2. Refresh the data for any PivotTables that refer to the table (see Recipe 11.5) to add the new column to the PivotTable's fields list as a new field.

3. Create a calculated field whose formula refers to the new field when you need to count the number of items.

To return a count of the table's items, you'd create a calculated field with the formula =CountItems, which implicitly calculates =SUM(CountItems); since the CountItems field is 1 for each item, =SUM(CountItems) returns the number of items. Similarly, the calculated field formula =IF(CountItems>5, Amount*2%, 0) implicitly uses the calculation =IF(SUM(CountItems)>5, SUM(Amount)*2%, 0) and returns 2% of the total amount if there are more than five items.

Discussion

This recipe offers a helpful workaround if you need to add a custom formula that counts a field's items instead of summing them.

11.23 Using Calculated Items

Problem

You have a PivotTable and want to add a custom formula that refers to one or more items in a single field.

Solution

Suppose you have a table with fields named Amount and Status, and the Status value can be *Pending*, *Shipped*, *Returned*, or *Canceled*. You want to create a PivotTable showing the total amount for each status, and include an extra row showing the sum of any refunds—amounts with the status *Returned* or *Canceled*—which are then subtracted from the grand total. To achieve this, you can create a PivotTable with a *calculated item*: a custom formula that refers to one or more items in a single PivotTable's field.

You create the PivotTable (including the calculated item) as follows (see Figure 11-17):

1. Create the PivotTable, and then add the Status field to the Rows section and the Amount field to the Values section.

2. In the PivotTable, select one of the Status field's values.

3. Choose PivotTable Analyze ⇒ Calculations ⇒ Fields, Items, & Sets ⇒ Calculated Item to open the Insert Calculated Item dialog box for that field.

4. Type the calculated item's name in the Name box (for example, **Refunded**); this is the name Excel displays in the PivotTable.

5. Type the custom formula in the Formula box—in this example, `=-(Canceled + Returned)`. You can also add items to the formula by selecting the Region field in the dialog box's Fields list, then either double-clicking the item in the Items list or selecting it and clicking Insert Item .
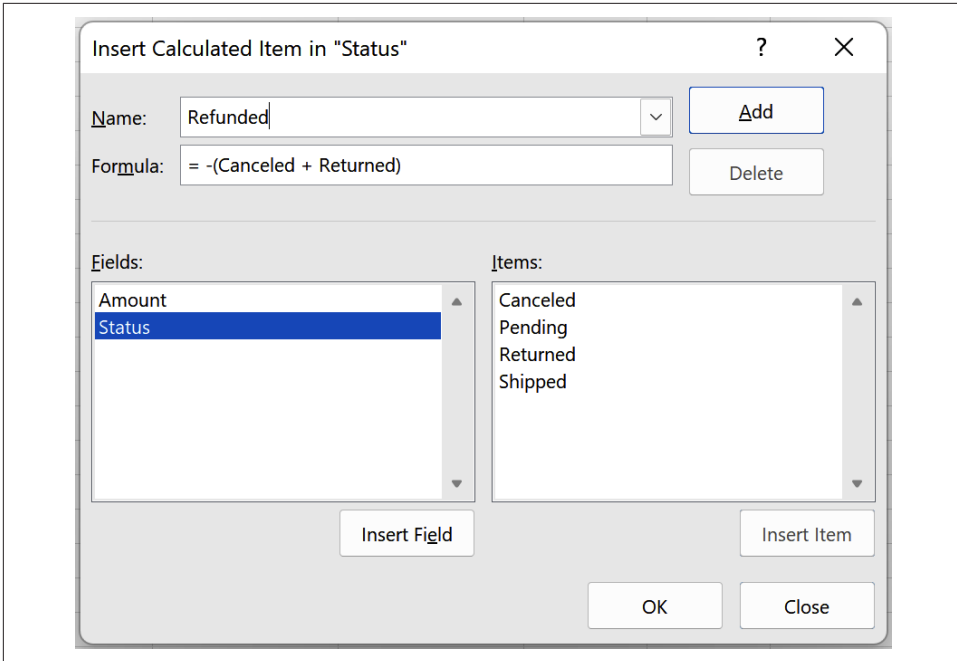
Figure 11-17. The Insert Calculated Item dialog box

When you click the OK button, Excel adds the item to the field as a new value and displays it to the PivotTable as a new row. Its value is included in the grand total, so in this example, any refunds are deducted from it (see Figure 11-18).
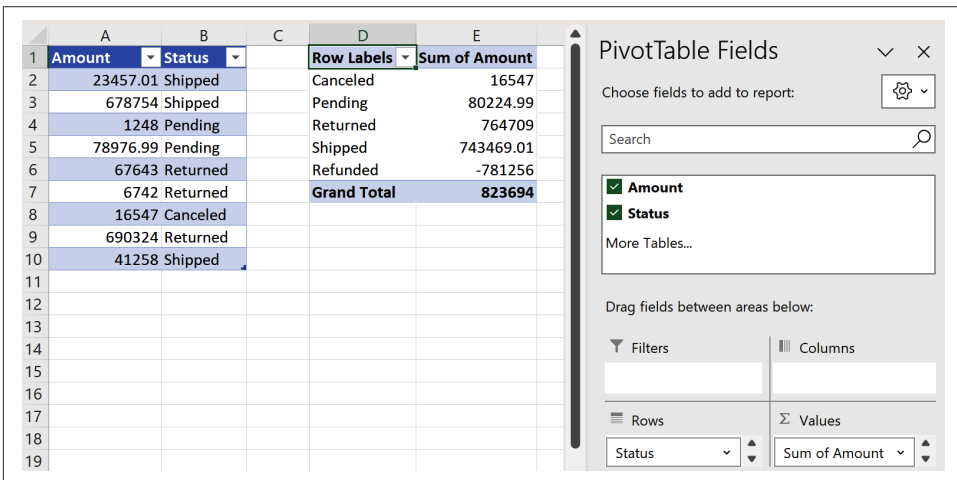
Figure 11-18. The PivotTable, including the Refunded calculated item

If you have the same value in different fields, Excel may display an error message if you try to refer to the value in the calculated item's formula. In this situation, you can specify which one you're referring to by including the field name. For example, use `Status[Returned]` to refer to the Returned value in the Status field.

You can remove a calculated item from a PivotTable by applying a filter that excludes it. You can also modify or permanently delete it as follows:

1. Select the calculated item in the PivotTable and choose PivotTable Analyze ⇒ Calculations ⇒ Fields, Items, & Sets ⇒ Calculated Item to open the Insert Calculated Item dialog box.
2. Select the calculated item from the Name box drop-down list.
3. Either change the formula and click Modify or delete the formula by clicking Delete. Then click Close to close the dialog box.

You can also modify the calculated item's formula in the PivotTable by selecting a cell containing one of its calculated values, then modifying the formula in the formula bar.

Excel adds any calculated items you create to the PivotTable cache (see Recipe 11.28) so any other PivotTables sharing the same cache inherit them. So deleting a calculated item from one PivotTable affects the other PivotTables.

Calculated items have the following limitations:

- They can't refer to worksheet cells or any PivotTable totals or subtotals.
- If a field includes a calculated item, you can't add the field to the PivotTable Fields pane's Filters section.
- If you add the field to the Values section, the PivotTable doesn't include the calculated item.
- You can add the field to only one section of the PivotTable. If the PivotTable includes a calculated item, you can't add two instances of the same field to the Values section to display different aggregations of it.

Unlike calculated fields (see Recipe 11.21), the aggregation you use in the PivotTable Fields pane's Values section affects the calculation. For example, if you have a calculated item in the Status field where the formula is `=-(Canceled + Returned)` and you specify a Count of Amount in the Values section, the calculated item implicitly computes `=COUNT(`*`Amount where Status is Canceled`*`) + COUNT(`*`Amount where Status is Returned`*`)`.

Discussion

This recipe shows you how to add custom formulas using calculated items. Unlike calculated fields whose formulas refer to one or more fields (see Recipe 11.21), a calculated item can refer to one or more items in a single field.

See Also

If you have multiple fields with calculated items, you may need to specify the order in which Excel should evaluate them. See Recipe 11.25 for further details.

11.24 Referring to Position in a Calculated Item Formula

Problem

You want to add a calculated item whose formula refers to an item by position.

Solution

Suppose you have a PivotTable with fields named Region and Amount, and you want to insert a calculated item to calculate 10% of the amount for the first region. To do so, add a calculated item for the Region field (see Recipe 11.23) and type the formula **`=Region[1]*10%`** in its Formula box, where `Region[1]` refers to the Region field's first item in the PivotTable. Generally, you refer to an item by position using the syntax *`field`*`[`*`position`*`]`, where *`field`* is the calculated item's field and *`position`* is its position in the PivotTable.

When you refer to an item by its position, the item being referred to may change if you adjust the PivotTable's sort order or apply a filter. You also risk creating a circular reference, which occurs if the calculated item's formula refers to itself.

You can also refer to items by position relative to the calculated item. For example, the formula `=Region[-1]*10%` refers to the item one row above the calculated item

and multiplies it by 10%. In contrast, the formula `=Region[+2]*10%` uses the item two rows below the calculated item instead.

> If you refer to an item above the first row or below the last one, the calculated item can't evaluate the formula and displays a `#REF!` error value.

Discussion

In some situations, you may need to refer to an item by its position in the PivotTable—for example, if you need to refer to the most recent date. This recipe shows how to achieve this.

11.25 Changing the Calculated Item Solve Order

Problem

You have a PivotTable with calculated items in its rows and columns and want to specify which formula to use when conflicts exist.

Solution

When you add calculated items to a PivotTable's rows and columns, conflicts may arise when the PivotTable calculates cell values using multiple formulas. For example, suppose you have a sales table containing Country, Amount, and Status columns, where the Country column includes the values *Australia*, *Canada*, and *USA*, and the Status column includes *Shipped* and *Returned*. You want to create a PivotTable showing the return rate—the number of sales returned divided by the number of sales—for each country and also calculate the return rate for North America: *Canada* and *USA*. You can define the PivotTable as follows:

1. Create a new PivotTable based on the table and add the Country field to the Rows section, the Status field to the Columns section, and the Amount field to the Values section. Change the Amount aggregation to Count (see Recipe 11.9).

2. Select the PivotTable and choose Design ⇒ Grand Totals ⇒ Off for Rows and Columns to hide the grand totals.

3. Select one of the Status labels in the column header—for example, *Returned*—and create a new calculated item named *Return Rate* with the formula `=Returned/ (Returned+Shipped)`.

4. Select one of the Country labels in the first column—for example, *Canada*—and create a new calculated item named *North America* with the formula `=Canada +USA`.

The PivotTable calculates the North America return rate using both the *North America* and *Return Rate* calculated items and returns a different result depending on which it evaluates first. For example, using the preceding steps, the PivotTable first uses the *Return Rate* formula to calculate the return rate for *Canada* and *USA* and then uses the *North America* formula to add them together. So if the return rates for *Canada* and *USA* are 0.67 and 0.33, respectively, the *North America* return rate is 1, or 100% (see Figure 11-19).

Country ▾	Amount ▾	Status ▾		Count of Amount	Column Labels ▾			
Australia	678754	Shipped		Row Labels ▾	Returned		Shipped	Return Rate
USA	1248	Shipped		Australia			1	0
Canada	78976.99	Shipped		Canada	2		1	0.666666667
Canada	67643	Returned		USA	1		2	0.333333333
USA	6742	Returned		North America	3		3	1
USA	16547	Shipped						
Canada	690324	Returned						

Figure 11-19. The PivotTable using the Return Rate formula first

To correct this, you can modify the solve order by selecting a cell in the PivotTable and choosing PivotTable Analyze ⇒ Calculations ⇒ Fields, Items, & Sets ⇒ Solve Order to open the Calculated Item Solve Order dialog box; this displays a list of the PivotTable's calculated items in the order in which it evaluates them. Then select the *North America* calculated item and click Move Up to move it up the hierarchy (see Figure 11-20).
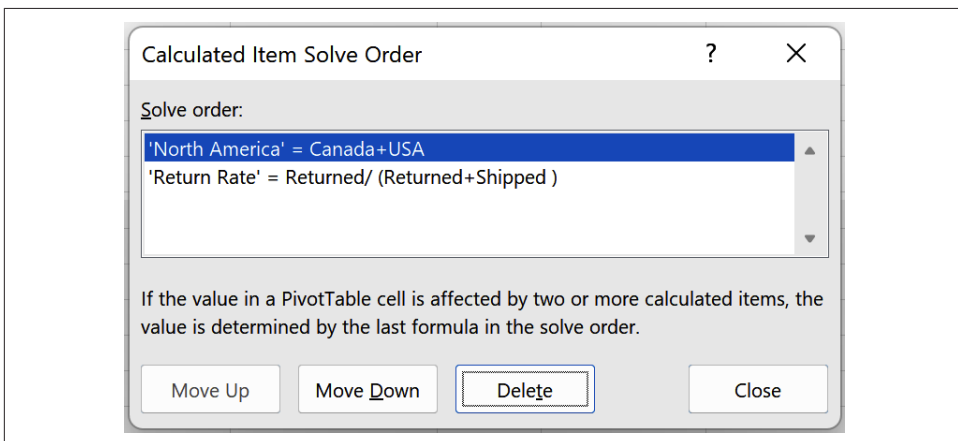
Figure 11-20. The Calculated Item Solve Order dialog box

When you click Close, Excel closes the dialog box and updates the PivotTable. This time, it first uses the *North America* formula to calculate the number of shipped and returned items, and then uses the *Return Rate* formula to calculate the return rate. For the data shown in Figure 11-21, this is now 0.5, or 50%.

Country	Amount	Status		Count of Amount	Column Labels			
Australia	678754	Shipped		Row Labels	Returned		Shipped	Return Rate
USA	1248	Shipped		Australia			1	0
Canada	78976.99	Shipped		Canada	2		1	0.666666667
Canada	67643	Returned		USA	1		2	0.333333333
USA	6742	Returned		North America	3		3	0.5
USA	16547	Shipped						
Canada	690324	Returned						

Figure 11-21. The PivotTable using the North America formula first

Discussion

This recipe shows you how to view and modify the solve order for PivotTables with two or more calculated items. Doing so lets you specify which calculations take precedence so you can control which one to apply if conflicts arise.

11.26 Generating a List of Custom Formulas

Problem

You have a PivotTable containing calculated fields and items and want to generate a list of each.

Solution

To output a list of every custom formula in a PivotTable, select a cell in the PivotTable and choose PivotTable Analyze ⇒ Calculations ⇒ Fields, Items, & Sets ⇒ List Formulas; this creates a new worksheet listing every calculated field and item, including its name, formula, and solve order.

Discussion

Managing multiple custom formulas in a PivotTable can quickly become unwieldy. This recipe shows you how to output each formula so you can see every calculation in a single place.

11.27 Changing a PivotTable's Data Source

Problem

You have a PivotTable and want to change it to use a different data source.

Solution

Suppose you have a PivotTable based on a table and want to change its data source to another. You can do so by following these steps:

1. Select a cell in the PivotTable and choose PivotTable Analyze ⇒ Data ⇒ Change Data Source to open the Change PivotTable Data Source dialog box.

2. In the Table/Range box, type or select the new source.

3. Click OK to update the PivotTable.

> If the PivotTable uses field names in the old source that aren't available in the new source, Excel automatically removes them from the PivotTable.

Discussion

This recipe outlines a quick way of changing a PivotTable's source to base it on a different table or range. This approach works best if the new source has the same column headings as the old one.

11.28 Using the PivotTable Cache

Problem

You have two PivotTables based on the same table and want to choose whether to share the same groups, custom formulas, slicers, and timelines.

Solution

When you create a new PivotTable, Excel copies the data it's based on and saves it in the *PivotTable cache*—an extra storage area Excel saves with the workbook. The PivotTable uses the data in its cache as the basis for its summaries, so if you update the original table, the PivotTable doesn't immediately include your changes; you first need to refresh the PivotTable (see Recipe 11.5), which updates the PivotTable cache (see Figure 11-22).
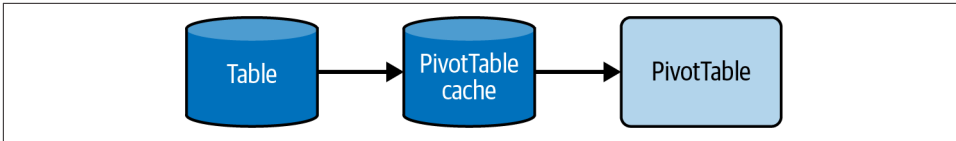
Figure 11-22. The PivotTable gets its data from the PivotTable cache—a copy of the source data

Excel automatically reuses the PivotTable cache for any new PivotTables you create based on the same data source. So if you have two PivotTables based on the same table, they automatically share the same cache (see Figure 11-23).
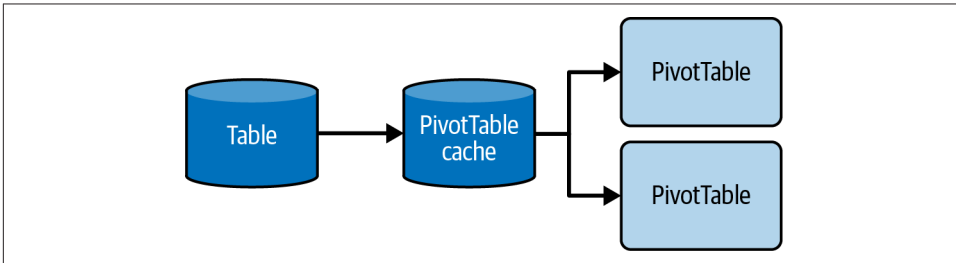
Figure 11-23. Two PivotTables sharing the same PivotTable cache

Reusing the cache for multiple PivotTables can help make the file size smaller because the PivotTable cache isn't duplicated; however, it also has the following consequences:

- When you refresh a PivotTable, all other PivotTables sharing the same cache are also refreshed.
- Any slicers and timelines you insert can be shared with any other PivotTables using the same PivotTable cache (see Recipe 11.29).
- When you group a field in one PivotTable, the same grouping is applied to all other PivotTables with the same cache.
- Excel adds any custom formulas you create to the PivotTable cache so they're shared by every PivotTable that uses it. For example, adding a calculated item to a field in one PivotTable automatically adds it to the others.

If you want to create two PivotTables based on the same table but don't want them to use the same cache so that any groups and custom formulas you add aren't shared, you can use the following workaround:

1. Select a cell in the table and insert the first PivotTable, using the table name as its source. Excel copies and stores the table data in a new PivotTable cache.

2. Select the table and choose Table Design ⇒ Tools ⇒ Convert to Range to convert the table to a worksheet range. Excel automatically changes the first PivotTable's source to the worksheet range.

3. Select Insert ⇒ Tables ⇒ Table to convert the range back to a table.

4. Insert a second PivotTable based on the table, using the new table name as its source. Excel creates a new PivotTable cache for the second PivotTable.

5. Repeat steps 2 to 4 for every new PivotTable you wish to create with a new cache.

Once you've followed these steps, any groups or custom formulas you add to one PivotTable are no longer shared with the other because they use separate caches (see Figure 11-24).
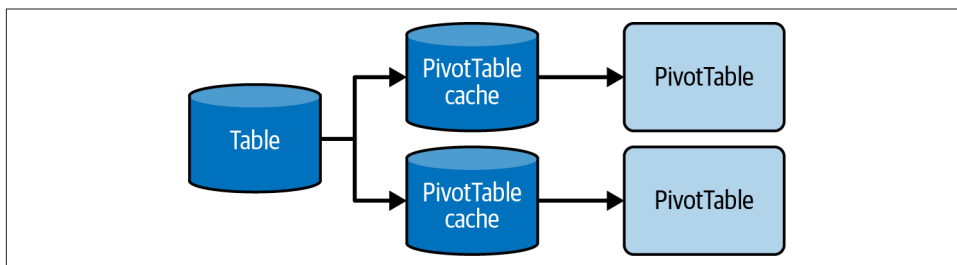
Figure 11-24. Two PivotTables based on the same table but using separate caches

Discussion

This recipe gives an overview of the PivotTable cache and its consequences. If needed, you can use the steps in this recipe to create multiple PivotTables based on the same table with separate caches, so if you add a group or custom formula to one PivotTable, Excel doesn't automatically add it to the others.

11.29 Filtering Multiple PivotTables That Share a Cache

Problem

You have multiple PivotTables based on the same table and want to filter them using a single set of slicers and timelines.

Solution

Suppose you have two PivotTables that share the same PivotTable cache (see Recipe 11.28). You want to insert one or more slicers and/or timelines and use them to filter both PivotTables.

Follow these steps to solve this problem (see Figure 11-25):

1. Select a cell in the first PivotTable and use Recipe 11.14 to insert the slicers and/or timelines.

2. Select a cell in the second PivotTable, then choose PivotTable Analyze ⇒ Filter ⇒ Filter Connections to open the Filter Connections dialog box.

3. The Filter Connections dialog box lists the slicers and timelines used by the first PivotTable because they share the same PivotTable cache. Place a check against any you also want to apply to the second PivotTable, then click OK.

Filter Connections (PivotTable6)		? ✕
Select filters to connect to this PivotTable		
Caption	**Name**	**Sheet**
☑ Company	Company	Filtering
☐ Order date	Order date	Filtering
		OK Cancel
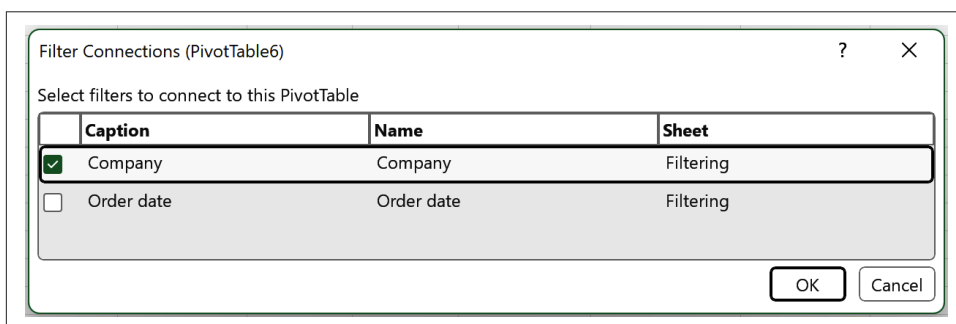
Figure 11-25. The Filter Connections dialog box

You can also select a slicer or timeline and specify which PivotTables to apply them to by choosing Slicer ⇒ Slicer ⇒ Report Connections, or Timeline ⇒ Timeline ⇒ Report Connections.

Discussion

This recipe shows you how to filter multiple PivotTables using the same slicers and/or timelines as long as they share the same PivotTable cache; this is usually the case if you've based the PivotTables on the same data source.

11.30 Reducing the Workbook File Size

Problem

You have a workbook with PivotTables and want to shrink its file size.

Solution

When you insert a PivotTable, Excel automatically copies its source data and adds it to the PivotTable's cache (see Recipe 11.28). Doing so makes the PivotTable more efficient but increases the workbook file's size.

If you need to reduce the file size, you can choose to not save the PivotTable cache but re-create it each time you open the workbook and use it in memory. You can do so by following these steps:

1. Select a cell in the PivotTable and choose PivotTable Analyze ⇒ PivotTable ⇒ Options to open the PivotTable Options dialog box.

2. Select the Data tab, then in the PivotTable Data section, uncheck the "Save source data with file" check box, and check the "Refresh data when opening the file" check box.

3. Click OK to apply the changes.

Discussion

This recipe is handy if you have a workbook that includes a PivotTable and want to reduce its file size. Note that doing so can make the workbook slower to open because it needs to re-create the PivotTable cache each time you open it.

An alternative approach is to save the PivotTable cache but delete the PivotTable's underlying table if this is no longer required. You can learn more about this option in Recipe 11.31.

11.31 Reinstating a PivotTable's Source Data

Problem

Suppose you have a PivotTable based on a table that you have deleted. You want to reinstate the table without resorting to backups.

Solution

Imagine you have a PivotTable that is based on a table and that saves its cache with the workbook file (see Recipe 11.30). If you don't use the table for anything other than the PivotTable and don't expect to make any further updates, you can delete the table, and the PivotTable will continue to work; this is because the PivotTable saves a copy of the table in its cache.

If you need to reinstate the table, ensure that the PivotTable is unfiltered and double-click the cell showing the row and column grand total (see Figure 11-26); this pastes the original table into a new worksheet.

Sum of Amount	Column Labels ▼			
Row Labels ▼	CatChat	Manic Mango	Starbuzz	Grand Total
Alex		757730		757730
Linda	67643	41258	23457	132358
Louise	6742		1248	7990
Will	690324		16547	706871
Grand Total	764709	798988	41252	1604949

Figure 11-26. Double-click the selected grand total to paste the original table into a new worksheet

Discussion

This little-known trick is helpful if you've deleted a PivotTable's data source and need to get it back. The only requirement is that the PivotTable must have saved a copy of the data in its cache.

11.32 Referring to PivotTable Values

Problem

You have a PivotTable and want to refer to one of its values from elsewhere.

Solution

Suppose you want to add a formula to a worksheet cell that refers to a value in a PivotTable. To do so, start typing the formula, then click the PivotTable cell when you want to refer to it.

When you click the cell, Excel inserts a call to the `GETPIVOTDATA` function that returns the PivotTable's calculation. So if the PivotTable starts in cell F1 and you click its grand total of the Sum of Amount value, Excel adds `GETPIVOTDATA("Sum of Amount", $F$1)` to the formula. Similarly, if you click the Sum of Amount value for the Pending item in the Status field, Excel adds `GETPIVOTDATA("Sum of Amount", $F$1, "Status", "Pending")`.

Referring to a value using `GETPIVOTDATA` returns the PivotTable's calculation instead of a specific cell reference, so if you change the sort order, Excel returns the same calculation. If you need to refer to the cell itself, manually type the cell reference (for example, $G$6).

You can use GETPIVOTDATA in a function only by selecting a cell in a PivotTable; if you try to add or modify it manually, Excel displays an error message. If you subsequently delete the calculation from the PivotTable, Excel displays a #REF! error value.

Discussion

This recipe gives an overview of referring to PivotTable values from elsewhere. Instead of using cell references to refer to a value's location, this approach uses the GETPIVOTDATA function to return the calculation.

You can switch GETPIVOTDATA off for all workbooks by selecting a PivotTable cell and choosing PivotTable Analyze ⇒ PivotTable, clicking the Options drop-down arrow, then unchecking the Generate GetPivotData option. Excel will then return a cell reference when you click a PivotTable's cell instead of using GETPIVOTDATA.

Charts

Charts can make tables of numeric data more accessible and help make it easier to see patterns, relationships, and trends. These features make charts helpful in analyzing data and communicating messages about that data to others.

Excel includes a range of predefined charts, and the recipes in this chapter help you work with them more effectively. Areas covered include the following:

- When to use each chart type and how to insert them based on a table, worksheet range, dynamic named range, or PivotTable
- How to customize charts by adding and formatting elements, dynamically changing titles, customizing data labels, changing legend entries, and more
- How to create bespoke chart types and save them as chart templates

12.1 Using Different Chart Types

Problem

You want to know what types of charts Excel includes and when to use them.

Solution

Excel includes a wide variety of charts, each one designed to serve a particular purpose. Here's a list of the main ones, organized by category:

Pie charts
Pie charts show the relative size of each value as a percentage of a whole. Excel includes 2-D and 3-D Pie charts and a *Doughnut* chart—a pie chart with a hole in

the middle (see Figure 12-1). There are also *Pie of Pie* and *Bar of Pie* charts, which let you show some values on a separate mini-chart; use these to highlight selected values or make smaller percentages more readable (see Recipe 12.12).
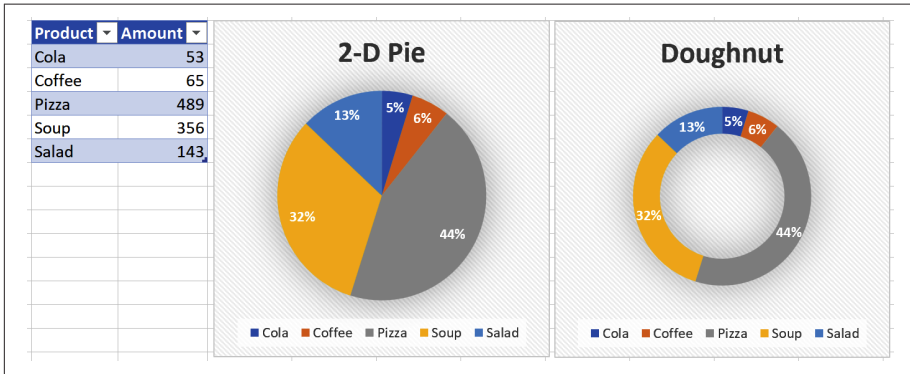
Figure 12-1. A 2-D Pie and Doughnut chart

> Pie charts generally work best when you have a small number of data points with positive values. If the data includes a negative value, Excel displays it on the chart as though it's positive since it charts only absolute values. You can, however, use data labels to highlight any negative values.

Column or Bar charts

These let you compare values across categories in one or more data series. Use a *Clustered* Column or Bar chart to compare values, a *Stacked* Column or Bar chart to compare the total values across categories, and a *100% Stacked* Column or Bar chart to compare relative percentages within each category (see Figure 12-2).
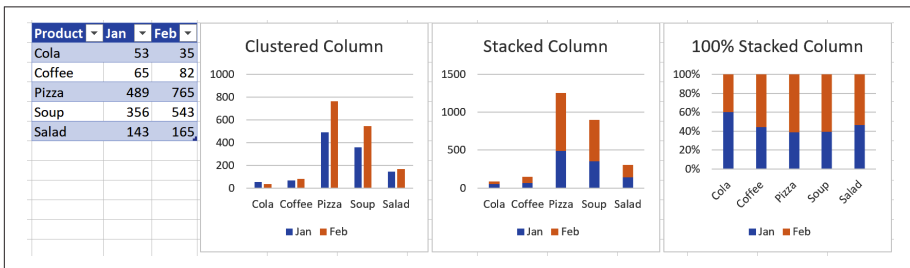
Figure 12-2. A Clustered Column, Stacked Column, and 100% Stacked Column chart

Line or Area charts

Use these charts to show trends or variations over time. Use a *Line* or *Area* chart to compare trends, a *Stacked* Line or Area chart to compare the trends of any totals, and a *100% Stacked* Line or Area chart to compare relative percentages (see Figure 12-3).
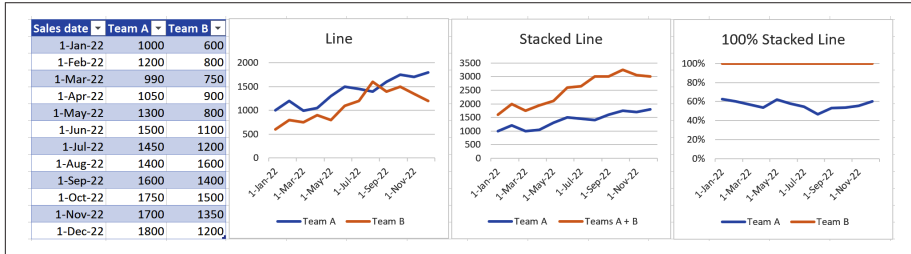
Figure 12-3. *A Line, Stacked Line, and 100% Stacked Line chart*

Stacked Line or Area charts stack values on top of one another, which can give casual users a misleading view of the data. You can mitigate this by updating the chart title and series names to clarify the chart's purpose.

Hierarchy charts

These are handy when you want to compare values of data organized in a hierarchy. Use a *Treemap* chart to display a proportionally sized rectangle for each value or a *Sunburst* chart to show proportions within concentric rings (see Figure 12-4). Alternatively, use a PivotChart (see Recipe 12.20) to show this type of data.
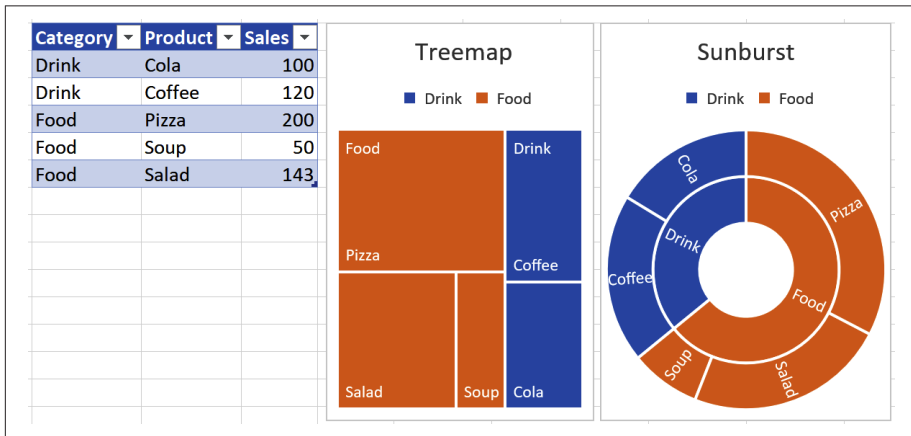
Figure 12-4. *A Treemap and Sunburst chart*

Statistic charts

These show a statistical analysis of your data and include Histogram, Pareto, and Box and Whisker charts. See Recipes 8.3, 8.10, and 12.13 for more details of these chart types.

Scatter (XY) or Bubble charts

A Scatter (XY) chart lets you plot XY points and can show the relationship between two things. A Bubble chart is like a Scatter chart, except it shows each point as a bubble; you can use bubble size to represent a third measure (see Figure 12-5). See Recipes 9.10, 9.12, and 8.23 for other ways of determining the relationship between two or more variables.
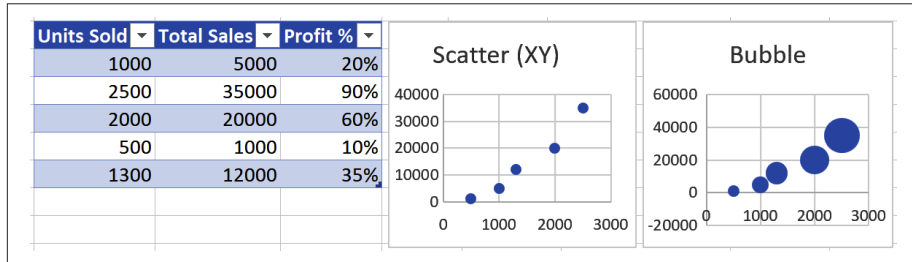
Figure 12-5. A Scatter (XY) and Bubble chart

Waterfall chart

Use this chart to show running totals and the cumulative effect of positive and negative numbers (see Figure 12-6).
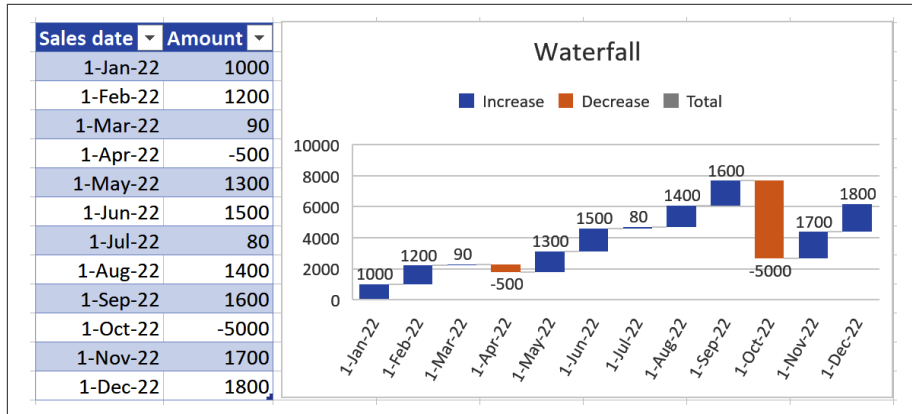
Figure 12-6. A Waterfall chart

Funnel chart

This chart shows the relative value of stages in a process (see Figure 12-7).
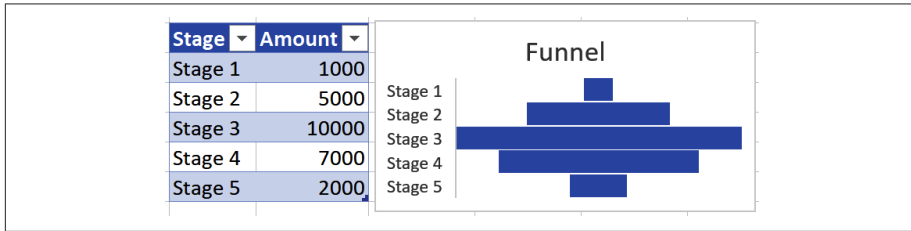
Figure 12-7. A Funnel chart

Stock charts

Use these to show a stock's performance over time (see Recipe 10.16).

Surface charts

These display two or more data series on a 3-D surface and work similarly to topographic maps (see Figure 12-8).
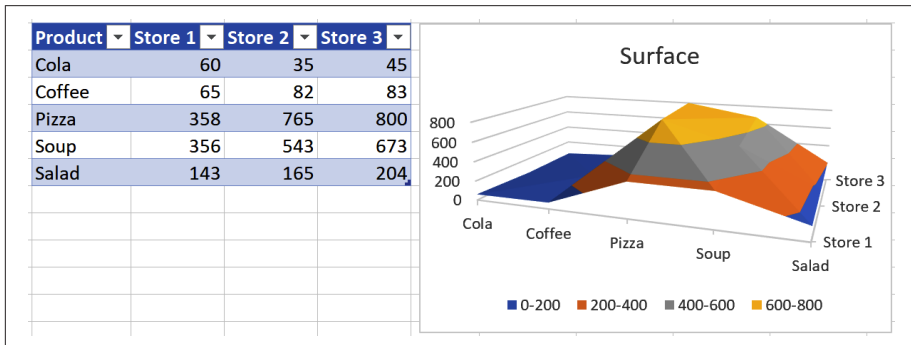
Figure 12-8. A Surface chart

Radar charts

Radar charts use a separate axis for each category; the axes radiate outward from the center. They can be helpful for data organized in categories that aren't directly comparable, such as performance or survey data (see Figure 12-9).

Event	Score
Vault	8
Floor	13
Pommel Horse	15
Parallel Bars	10
Still Rings	2
Horizontal Bar	9
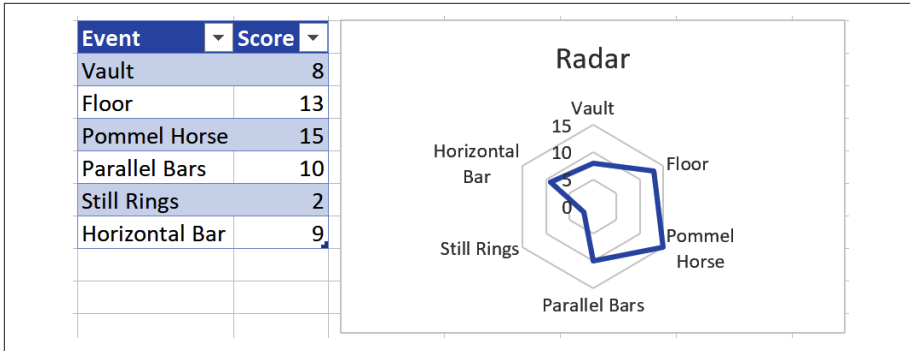
Figure 12-9. A Radar chart

Combination charts

These let you display two or more types of charts—for example, column and line charts—on a single chart. They can be helpful when you have mixed types of data or data series with wildly different ranges of values (see Recipe 12.14).

Filled Map chart

This chart lets you compare values across geographic regions, such as countries, states, and cities (see Figure 12-10). If you're using Excel for Windows, you can also plot geographic data using the 3D Maps tool (see Recipe 13.12).

State	Sales
New York	5000
Maine	3000
California	8000
Oregon	4000
Texas	7000
Iowa	6000
Alabama	2000
Colorado	5000
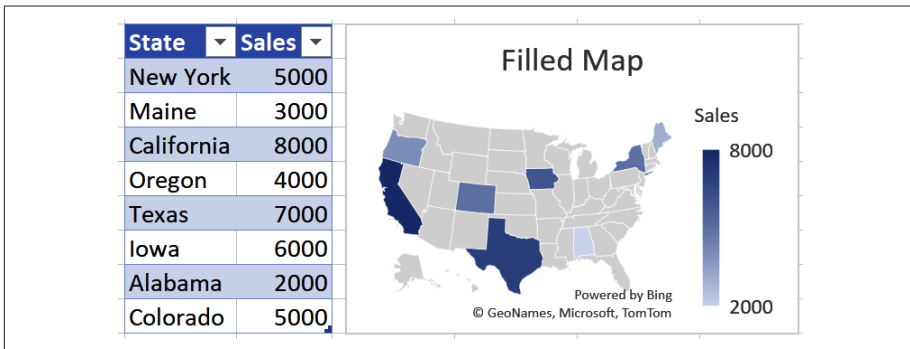
Figure 12-10. A Filled Map chart

PivotChart

A PivotChart is a chart that displays PivotTable data (see Recipe 12.20). Use it to slice and dice data and display summaries of hierarchical data instead of using a Treemap or Sunburst chart.

Discussion

This recipe provides an overview of Excel's available chart types and when to use them. You can also use them to create other custom charts. See Recipe 12.21 for an example.

See Also

To create a dot plot or pictograph, see Recipes 5.14 and 12.11.

To create a chart displaying a seasonal or cyclical forecast, see Recipe 10.19.

12.2 Inserting a Chart

Problem

You have a table that includes numeric data and want to visualize it using a chart.

Solution

Suppose you have a table with two columns named Product and Amount, and you want to use a pie chart to compare the relative sizes of each product's amount (see Figure 12-1).

To insert a chart, you generally select a cell in the table and then select the type of chart you want to insert from the Insert menu. So, to insert a 2-D Pie chart, you choose Insert ⇒ Charts ⇒ Insert Pie or Doughnut Chart and select the Pie option from the 2-D Pie section. Alternatively, choose Insert ⇒ Charts ⇒ Recommended Charts to insert a chart Excel recommends for the data.

> If you're using Excel 365, you can also create a chart using the Analyze Data feature. Select a cell in the table, choose Home ⇒ Analysis ⇒ Analyze Data to open the Analyze Data pane, then select one of the available charts or use the "Ask a question" box to ask a question about the data.

Once you've selected a chart, Excel adds it to the worksheet as a floating or embedded object that you can move or resize by clicking and dragging the chart or its corners. You can also move the chart to a different worksheet by selecting the chart and choosing Chart Design ⇒ Location ⇒ Move Chart to open the Move Chart dialog box; select Object In to move the chart to an existing worksheet and New Sheet to create a new *chart sheet*—a separate worksheet that contains only a chart.

Once you've created a chart, you can change it to a different chart type. Select the chart and choose Chart Design ⇒ Type ⇒ Change Chart Type; then select the type you want to use.

Discussion

This recipe shows you how to insert a chart based on table data. Basing a chart on a table is generally better than basing it on a worksheet range since it automatically accommodates any rows you add or delete. If you base the chart on a worksheet range and add or delete rows, you need to manually adjust the chart's data selection, which you can do by selecting the chart to show its data range, then clicking and dragging the range's corners to resize it.

See Also

To use a sparkline to insert a line or column chart in a single cell, see Recipe 13.10.

12.3 Filtering a Chart

Problem

You have a chart and want to exclude some of its data.

Solution

Suppose you have a chart and want to filter the data it displays.

One approach is to filter the table on which the chart is based since, by default, filtering the table also filters the chart. If needed, you can stop this default behavior (so the chart shows all the table's data) by selecting the chart, choosing Chart Design ⇒ Data ⇒ Select Data to open the Select Data Source dialog box, clicking Hidden and Empty Cells, and then checking the "Show data in hidden rows and columns" check box.

Depending on where you place the chart relative to its table, filtering the table may move and/or resize the chart. You can control this behavior by opening the chart area Format pane (see Recipe 12.6), clicking the Size & Properties tab, and then selecting one of the move and size options in the Properties section; choose from "Move and size with cells" (the default), "Move but don't size with cells," and "Don't move and size with cells."

If you're using Excel for Windows, you can also apply a filter to the chart without filtering the table by selecting the chart and clicking the Chart Filters button next to it.

Doing so opens the Chart Filters pane, which you can use to filter the chart by data series and category (see Figure 12-11).
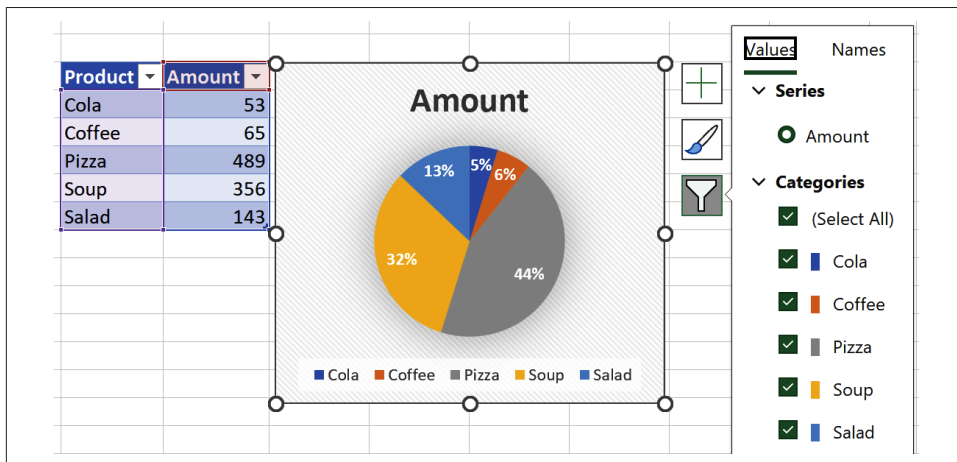
Figure 12-11. Filtering a chart in Excel for Windows

Discussion

This recipe shows two ways of filtering a chart, depending on your version of Excel.

By default, filtering a table will filter every chart based on it. This behavior is convenient if you add a slicer to the table (see Recipe 2.18) because it can help you build a dashboard that presents different views of the same data.

12.4 Tweaking a Chart's Appearance

Problem

You have a chart and want to quickly change its layout, style, and color scheme.

Solution

Excel includes several predefined layouts, styles, and color palettes, which you can select to customize a chart's appearance.

To change the layout, select the chart, choose Chart Design ⇒ Chart Layouts ⇒ Quick Layout, then click the layout you wish to apply.

To change the style, select the chart, choose Chart Design ⇒ Chart Styles, then click the style you want to use from the Quick Styles gallery.

To change the colors, select the chart, choose Chart Design ⇒ Chart Styles ⇒ Change Colors, then click the color palette you want to use.

If you're using Excel for Windows, you can also pick a style or color palette by clicking the Chart Styles paintbrush button next to the chart and selecting the Style or Color option.

Discussion

This recipe gives an overview of how to quickly tweak a chart to change its layout, style, and color palette. You can also hover your mouse cursor over each one to see a preview of its appearance.

12.5 Adding and Removing Chart Elements

Problem

You have a chart and want to add or remove elements like axes, titles, or trendlines.

Solution

Once you've inserted a chart, you can customize it by adding extra chart elements or deleting existing ones.

To add an element to a chart, select the chart, choose Chart Design ⇒ Chart Layouts ⇒ Add Chart Element, and select the element from those available; the list of available elements varies from chart to chart because you can't add all elements to every chart type. If you're using Excel for Windows, you can also add a chart element by selecting the chart and clicking the Chart Elements plus button next to it.

To remove an existing element, select the element in the chart and press the Delete key. Alternatively, choose Chart Design ⇒ Chart Layouts ⇒ Add Chart Element and select the None option for the chart element you wish to remove, or deselect it from the chart's Chart Elements button if you're using Excel for Windows.

Discussion

This recipe shows you how to customize a chart by adding extra elements or removing existing ones. See Recipe 12.6 for a list of chart elements and their uses.

12.6 Formatting Chart Elements

Problem

You have a chart and want to change the format of one or more of its elements.

Solution

To make basic changes to a chart element, such as its style, colors, and size, select the element in the chart, choose Format, and use the available options. You can also right-click the element to display a mini toolbar.

To make more advanced or specific changes, such as changing the type of trendline to display, double-click the element—or select the element and choose Format ⇒ Current Selection ⇒ Format Selection—to open the Format pane. Then, use the available options to format the element.

> To switch to formatting a different element, select it in the chart, choose Format ⇒ Current Selection, and select the element from the Chart Elements drop-down list, or select it from the Chart Options drop-down list in the chart's Format pane.

Here are some useful chart elements, what they are for, and useful ways to format them (see Figure 12-12):

Chart area
    This rectangular box contains all the chart's elements. Select this element to make chart-wide changes to text size, font, and style without changing each element separately. You can also use it to control whether the chart moves or resizes with the worksheet's cells; see Recipe 12.3.

Plot area
    This rectangular box contains the chart plot. Excel often resizes it automatically when you add or remove chart elements. You can also resize it manually to accommodate any shapes (see Recipe 13.3) you want to add to the chart.

Chart title
    This short caption specifies the chart's purpose. You can change its location, edit its text, or follow the steps in Recipe 12.7 to create a dynamic chart title based on a cell's contents.

> You can add descriptive text to the chart by inserting a text box; select the chart and choose Insert ⇒ Text ⇒ Text Box, or choose Format ⇒ Insert Shapes and select the Text Box option from the Shapes gallery.

Data series
    These are the chart's visual elements that represent the data: a pie chart's slices, a line chart's lines, a column chart's columns, and so on. By formatting the data series, you can rotate a pie chart by changing the angle of its first slice, adjust the

spacing between columns in a column chart by changing the series overlap and gap width, smooth or add arrows to lines in a line chart, and more.

Legend

This rectangular box contains keys and text identifying each item or data series. You can format the legend as a whole or select and format individual entries. To change the legend entry text, see Recipe 12.17.

Data labels

These add labels to the chart's data points. To specify what the labels should display, click the Label Options button in the Format pane and choose from the available options. Depending on the chart type, these include each data point's name, value, percentage, and more. You can also link the data label text to cell values (see Recipe 12.8) or select an individual data label to hardcode its text or dynamically update it based on a single cell's contents (see Recipe 12.7).

> You can also choose to see each data point's value when you hover your mouse pointer over it. To control this behavior, choose File ⇒ Options ⇒ Advanced if you're using Excel for Windows; then check or uncheck the "Show data point values on hover" check box in the Chart section. If you're using Excel for Mac, you can find this option by choosing Excel ⇒ Preferences ⇒ Chart.
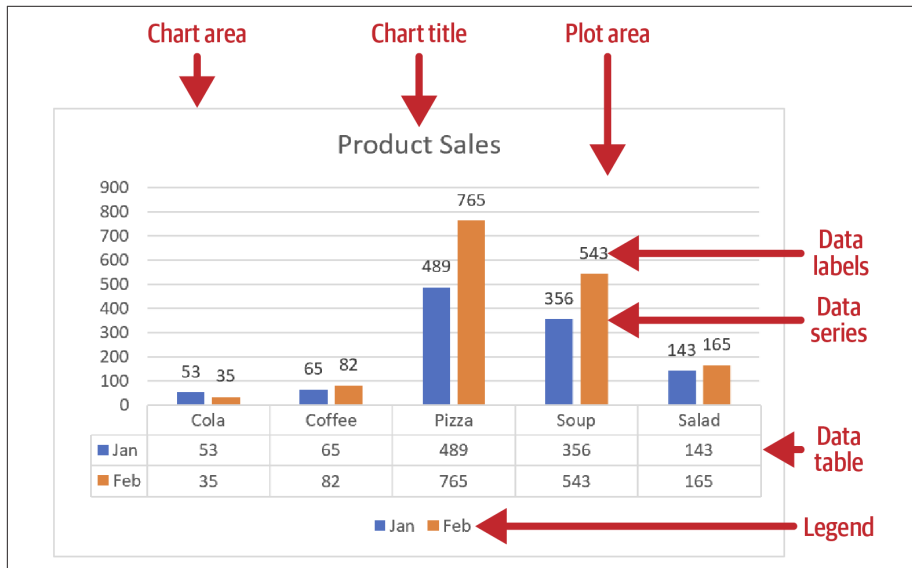
Figure 12-12. A chart showing a selection of chart elements

Data table

This element is a table containing the chart's data. You can use this as an alternative to a legend and/or data labels.

Axes

Most charts include one or more axes (see Figure 12-13). Use this option to specify their bounds, the values they display, the position of any tick marks, and more (see Recipe 12.9).

Axis titles

These let you specify what each axis represents (see Figure 12-13). You can either hardcode the axis title text or follow the steps in Recipe 12.7 to dynamically change the title based on a cell's contents.

Gridlines

These are lines spaced at regular intervals that help you read a data point's value from the chart's axis (see Figure 12-13). Charts can include major and minor horizontal and vertical gridlines, and you control their positions by formatting the axes (see Recipe 12.9).

Trendline

This element shows the line best fitting the series data points (see Recipe 8.23). You can display an exponential, linear, logarithmic, polynomial, power, or moving average trendline, display its equation (see Recipe 8.24), and use the Forecast options to extend the trendline to make short-term forecasts (see Recipe 10.18). Figure 12-13 shows an example trendline.
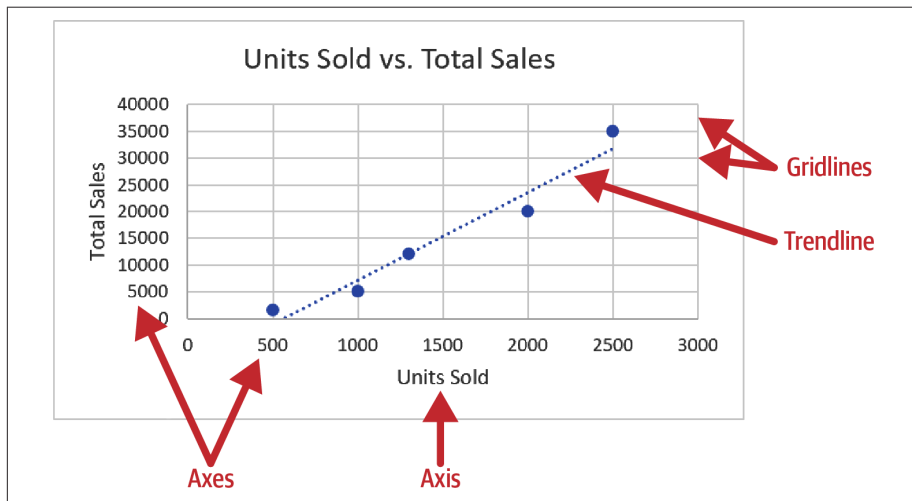
Figure 12-13. A chart showing more chart elements

Error bars

You can add bars showing the data point's value plus and/or minus another value, such as a percentage, standard deviation, standard error, or a fixed or custom value (see Figure 12-14).
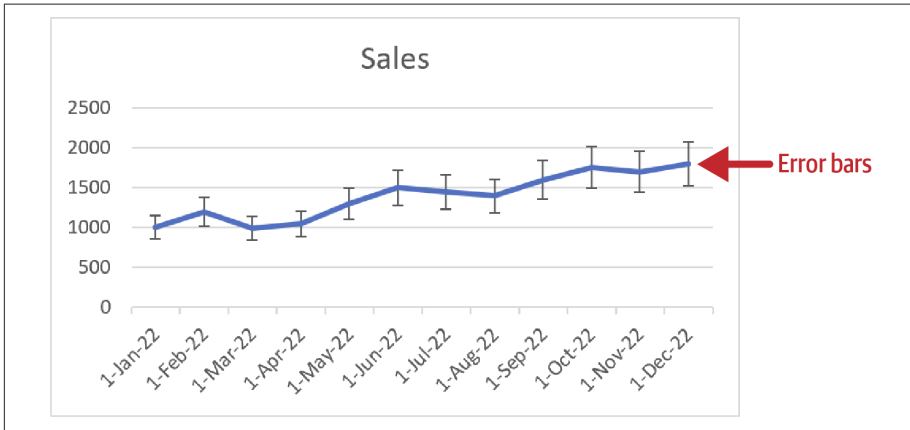
Figure 12-14. A chart showing error bars

Up/down bars

If you have a line chart with two or more series, these let you compare the series at each point (see Figure 12-15).
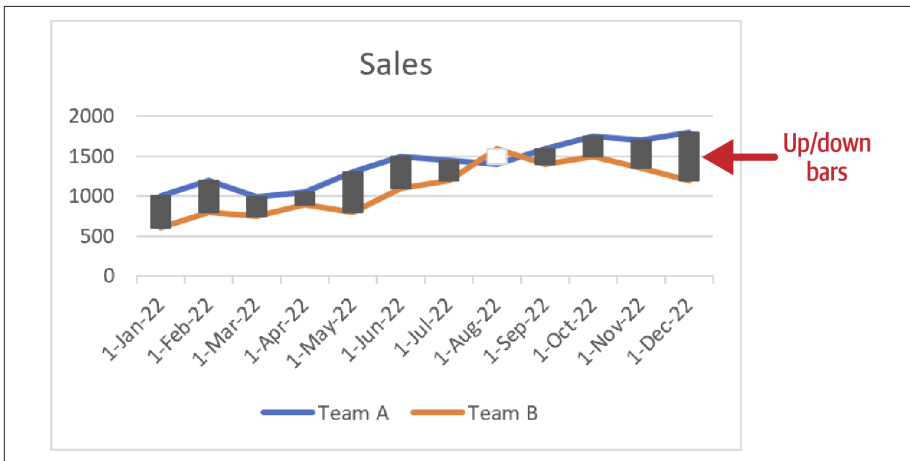
Figure 12-15. A chart showing up/down bars

Lines

You can add high-low or drop lines to a line chart. High-low lines serve a similar purpose to up/down bars and show the difference between two or more series at

each point (see Figure 12-16), while drop lines add a vertical line from each data point to the horizontal axis (see Figure 12-17).
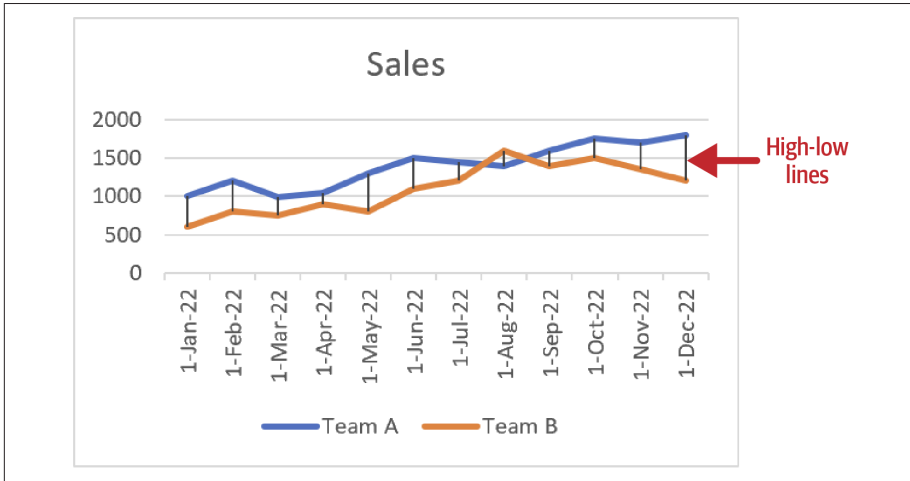
Figure 12-16. A chart showing high-low lines
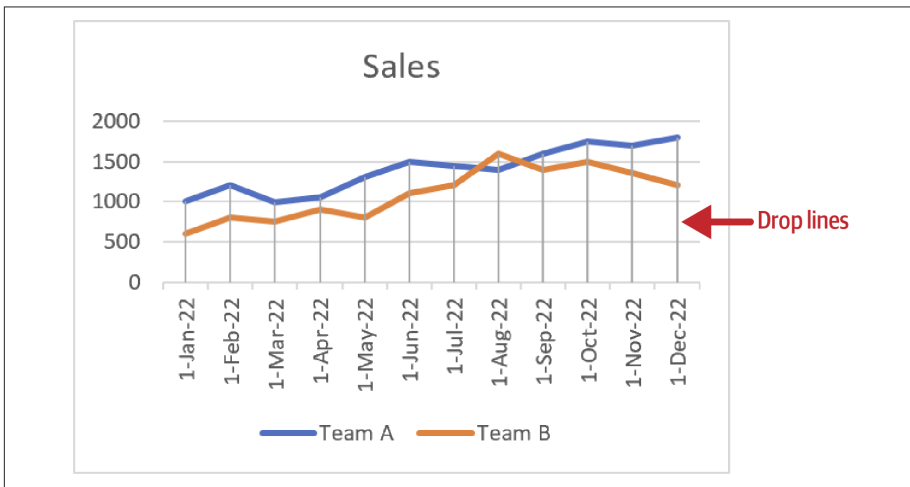
Figure 12-17. A chart showing drop lines

You can choose to see a chart element's name when you hover your mouse pointer over it. To control this behavior, choose File ⇒ Options ⇒ Advanced if you're using Excel for Windows, and in the Chart section, check or uncheck the "Show chart element names on hover" check box. If you're using Excel for Mac, you can find this option by choosing Excel ⇒ Preferences ⇒ Chart.

Discussion

This recipe gives an overview of Excel's chart elements, their purpose, and ways in which you can format them. Try experimenting with the options to see the effects they have.

12.7 Creating Dynamic Titles and Labels

Problem

You have a chart and want to base its chart or axis titles on cell values.

Solution

By default, Excel hardcodes the chart's title so it doesn't dynamically update. You can, however, link it to a cell value by selecting the chart title and typing **=*cell_reference*** in the formula bar, where ***cell_reference*** is a reference to the cell, including the worksheet name. To use the value of cell A1 in the Sales worksheet for a chart title, for example, you'd select the chart title and type **=Sales!A1** in the formula bar.

You can use a similar approach with other titles, such as axis titles and the labels of individual data points.

Discussion

This recipe shows you how to use a cell value for a chart or axis title so it dynamically updates when the cell's value changes. You can use a similar approach to dynamically update the chart's legend entries (see Recipe 12.17).

12.8 Customizing Data Label Text

Problem

You want to customize a chart's data labels to display something other than the data point value or name.

Solution

Suppose you want to insert a Stacked Column chart whose data labels show the total of each column. While this option isn't available by default, you can customize the data labels to achieve the desired results. The steps are as follows:

1. Add a new column to the table on which you want to base the chart, populated with the value you want to display in each row's data label. So, to add data labels

that display totals, you'd add a new calculated column that calculates the total of each row (see Recipe 2.19).

2. Select the table columns on which you want to base the chart, excluding the column for the data labels you created in step 1. Then insert the chart—a Stacked Column chart in this example—and add data labels to it.

3. In the chart's Format pane (see Recipe 12.6), format the data labels by clicking the Label Options button and then checking the Value From Cells check box.

4. When prompted, select the range you want to use for the data labels—the column values you added in step 1. Then click OK to add the values to the data labels.

5. Uncheck any label options you don't want the data labels to display, such as Value. You may also want to delete any extra data labels Excel may have created for you in step 2.

Figure 12-18 shows an example chart created using this technique and the table data on which it's based.
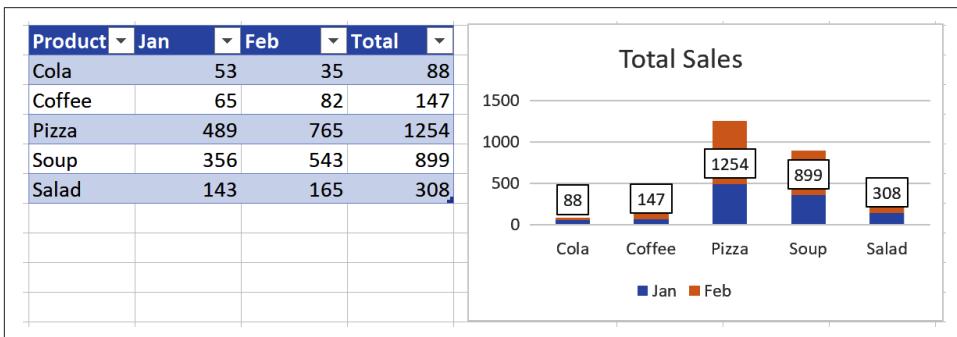
Figure 12-18. A chart whose data labels are based on the table's Total column

Discussion

The data labels Value From Cells format option offers a flexible way of customizing the text displayed for each data point. You can also use it, for example, to apply a custom number format (see Recipe 1.5) or include symbols.

12.9 Controlling Chart Axes and Gridlines

Problem

You have a chart with an axis and want to control its bounds, gridlines, tick marks, and more.

Solution

Suppose you have a chart with one or more axes—for example, a scatter chart—and you want to tweak an axis to adjust its bounds, scale, or where tick marks appear. You can generally do so by opening the chart's Format pane for the axis (see Recipe 12.6), clicking the Axis Options button, and using the available options.

You can change the range of numeric values the chart can display by adjusting the axis bounds. In the Bounds section, use the Minimum and Maximum boxes to specify the smallest and largest value the chart can display on that axis, respectively.

To change the interval at which numbers appear on the axis, type the interval in the Major box in the Units section. So typing **1000** in the Major box displays the numbers 0, 1,000, 2,000, and so on, depending on the axis bounds. You can also adjust the units shown by choosing an option from the "Display units" drop-down list and applying a number format—similar to Recipes 1.4 and 1.5—using the options in the Number section.

> If you have one or more data series whose values use a different scale, you can plot them on a secondary axis, depending on the chart type. To do so, select the data series, click the Series Options button in the Format pane, then select the Secondary Axis option. Alternatively, consider using a combination chart (see Recipe 12.14).

Charts can include major and minor gridlines, and the Major box also controls the interval at which the chart displays any major gridlines. To specify the interval for any minor gridlines instead, type a value in the Minor box.

If you want to display values in reverse order, check the "Values in reverse order" check box. For example, checking this option for a column chart's vertical axis results in a chart whose columns descend from the top of the chart.

> You can quickly switch a chart's horizontal and vertical axes by selecting the chart and choosing Chart Design ⇒ Data ⇒ Switch Row/Column.

Some chart axes—for example, the horizontal axis of a column or line chart—include extra options that let you control the axis type and position relative to its tick marks.

The Axis Type option controls how the chart spaces its axis values. When you insert, for example, a column chart, Excel automatically detects whether the values for the horizontal axis are dates or times and classifies the axis type as Date if they are or Text

if they're not. If the axis type is Text, the chart spaces the values—and therefore its columns—an equal distance apart. However, if the axis type is Date, the chart spaces the values on a date scale, which can result in narrow, unevenly spaced columns. Changing the Axis Type option to Text corrects this problem.

The Axis Position option lets you control where the axis values appear relative to the chart's tick marks. You can choose the "On tick marks" option to place values on the tick marks or the "Between tick marks" to place values between them. Generally, using the "Between tick marks" option (the default) works best for column charts, while the "On tick marks" option works best for line charts.

> When you insert a line chart, Excel uses the "Between tick marks" option by default for its horizontal axis, making the chart harder to interpret because the tick marks don't align with the chart's data points or values. To correct this problem, select the "On tick marks" option instead.

Discussion

This recipe presents some of the most useful options to control a chart's axes and gridlines. Note that you can format only one axis at a time, so if you want to make the same changes to multiple axes, you must manually apply them to each one.

12.10 Displaying Negative Values

Problem

You have a chart and want to control how positive and negative values are displayed.

Solution

Depending on the chart type, you can control how to display positive and negative values by formatting the data series.

If you're using a column or bar chart, you can use different colors for positive and negative values; open the Format pane for the data series (see Recipe 12.6), click the Fill & Line button, then check the "Invert if negative" check box. You can then use the Color option to specify the colors for the positive and negative values.

If you're using a bubble chart, by default, it doesn't display any bubbles with a negative size. To show these bubbles, open the Format pane for the data series, click the Series Options button, then check the "Show negative bubbles" check box.

Discussion

This recipe shows you how to apply different formats to data points depending on whether their value is positive or negative. The available options depend on the type of chart you're using.

You can also manually format an individual data point by selecting it in the chart and changing its format. However, if the underlying value changes, you may need to manually update the chart to reflect this.

12.11 Using Pictures in Column Charts

Problem

You have a column chart and want to use a picture for its columns.

Solution

Suppose you have a column chart and want to fill its columns with a picture or stack pictures on top of one another. You can do so by formatting the data series.

Open the Format pane for the data series (see Recipe 12.6), click the Fill & Line button, and select the "Picture or texture fill" option from the Fill section. Then click the Insert button to choose a picture and use the Stretch and Stack options to specify whether to stretch the picture to fill each column or stack pictures on top of one another (see Figure 12-19).
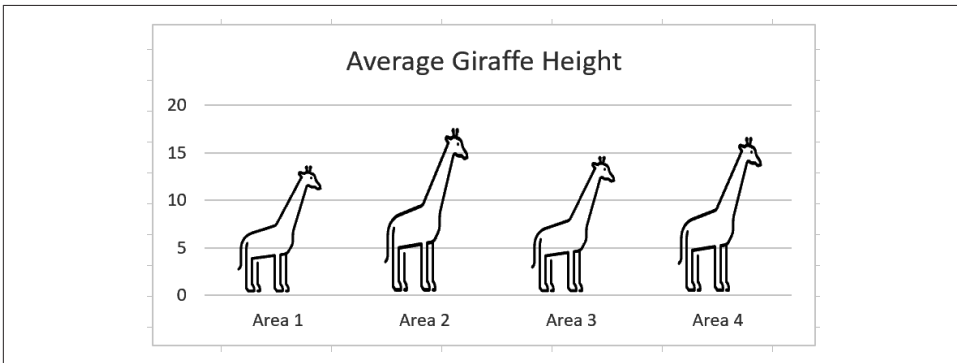
Figure 12-19. Using a picture to fill a column chart's columns

Discussion

This recipe shows you how to use a picture for a column chart's columns instead of a plain color. You can also try experimenting with other Fill options, such as using a texture, pattern, or gradient.

You can use this recipe to create a dot plot or pictograph, depending on your chosen picture. Another way of achieving this is with Recipe 5.14.

> While using a picture fill can be appropriate in some circumstances, ensure that doing so doesn't detract from the chart's overall message and purpose. For example, accurately reading a data point's value from the chart's axis may be more challenging with a picture fill.

12.12 Formatting Pie of Pie and Bar of Pie Charts

Problem

You have a Pie of Pie or Bar of Pie chart and want to control which data points appear on which chart.

Solution

As described in Recipe 12.1, a Pie of Pie—or Bar of Pie—chart is a type of pie chart that displays some of its data points in a second plot. You can specify which data points to display on the second plot by formatting the chart's data series.

In the chart's Format pane for the data series (see Recipe 12.6), click the Series Options button, then choose one of the available options from the Split Series By drop-down list; these are Position, Value, Percentage Value, and Custom. To display all data points with a percentage value less than 10% on the second plot (see Figure 12-20), you'd select Percentage Value from the drop-down list, then type **10%** in the "Values less than" box that appears when you select this option.
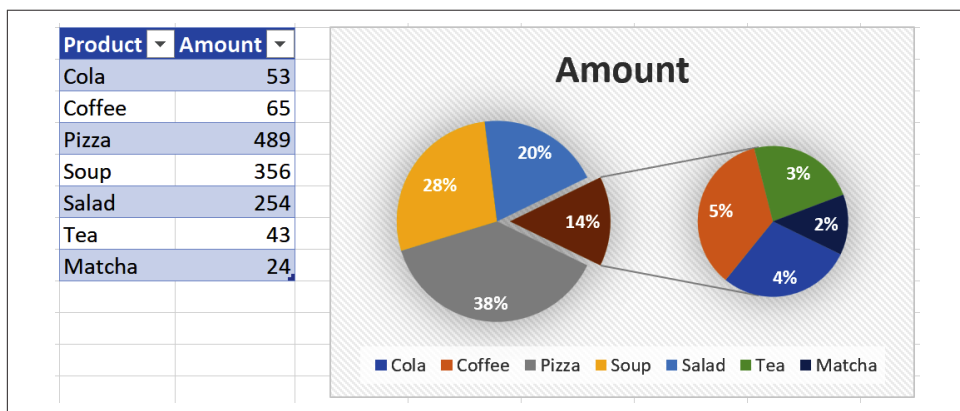
Figure 12-20. A Pie of Pie chart, where values less than 10% appear on a second plot

Discussion

Pie of Pie and Bar of Pie charts are helpful if you want to use a pie chart and have several data points with smaller values. This recipe shows you how to control which points the chart displays on a second plot, which helps make the smaller values more readable.

12.13 Formatting a Histogram Chart

Problem

You have a Histogram chart and want to control the width and number of its bins.

Solution

By default, Excel automatically determines the width and number of its *bins*—a histogram's columns—based on your provided data. You can, however, override these settings.

If you're using Excel for Windows, select the chart's horizontal axis; then in the Format pane (see Recipe 12.6), click the Axis Options button and select one of the Bins options in the Axis Options section; if you're using Excel for Mac, you can find these options by formatting the data series instead. The available options are as follows:

By Category
This option uses text or category values for the horizontal axis instead of numeric values, so it's handy for displaying the frequencies of discrete values (see Recipe 12.6).

Bin width
This controls how bins are organized by specifying how wide they should be; Excel then calculates how many bins to display.

Number of bins
This specifies how many bins to display; Excel then calculates each one's width.

Automatic
With this option (the default), Excel automatically decides how many bins to display and their width.

When the chart uses numeric values for its horizontal axis, Excel automatically determines the interval for each bin. So, if you have numeric values ranging from 5 to 100 and specify 5 bins, Excel may use an interval of 5 to 24 for the first bin, 24 to 43 for the second, and so on. You can control these intervals using the "Overflow bin" and "Underflow bin" options. For example, if you check the "Underflow bin" option and set it to 20, the chart will include values less than or equal to 20 in its first bin and

adjust the remaining bin intervals. Similarly, checking the "Overflow bin" option and setting it to 80 updates the chart's final bin to include values greater than 80 (see Figure 12-21).
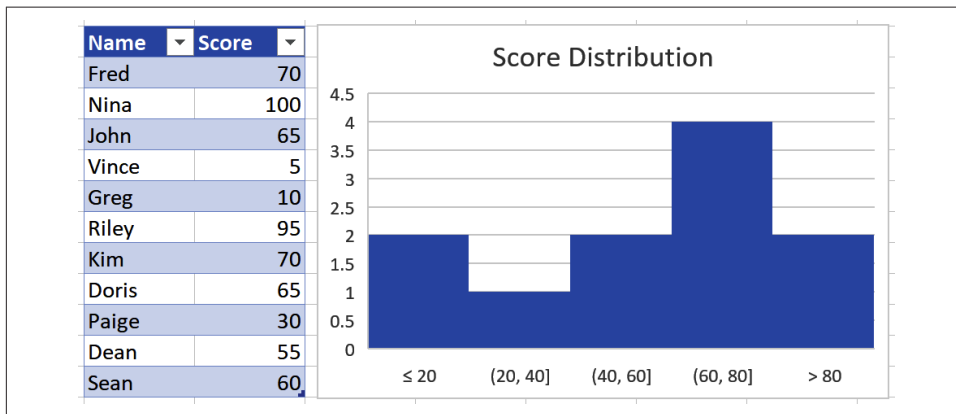
Figure 12-21. A Histogram chart with 5 columns, an underflow of 20, and an overflow of 80

By default, Excel sets the gap width between each bin to 0%, so there are no gaps between each bin. If the chart uses discrete category values for its horizontal axis, you should consider adjusting this setting. If you're using Excel for Windows, you can select the data series, click the Series Options button in the Format pane, and adjust the Gap Width option; if you're using Excel for Mac, you can find this option by formatting the horizontal axis.

Discussion

Controlling a Histogram chart's bins can be surprisingly tricky, so this recipe provides an overview of how to do so. Notice how you have to format the chart's horizontal axis or data series, depending on your version of Excel.

12.14 Specifying a Combination Chart's Chart Types

Problem

You want to use a combination chart and specify which type of chart to use for each series.

Solution

If you're using Excel for Windows, you can specify which chart types to include in a combination chart. Here are the steps to insert the chart (see Figure 12-22):

1. Select the data on which you want to base the chart; then choose Insert ⇒ Charts ⇒ Combo and select the Create Custom Combo Chart option to open the Insert Chart dialog box.

2. Make sure the Combo option is selected, then use the Chart Type option to specify the chart type for each data series and the Secondary Axis check box to indicate whether any series should use a secondary axis.

3. Click OK to insert the chart.
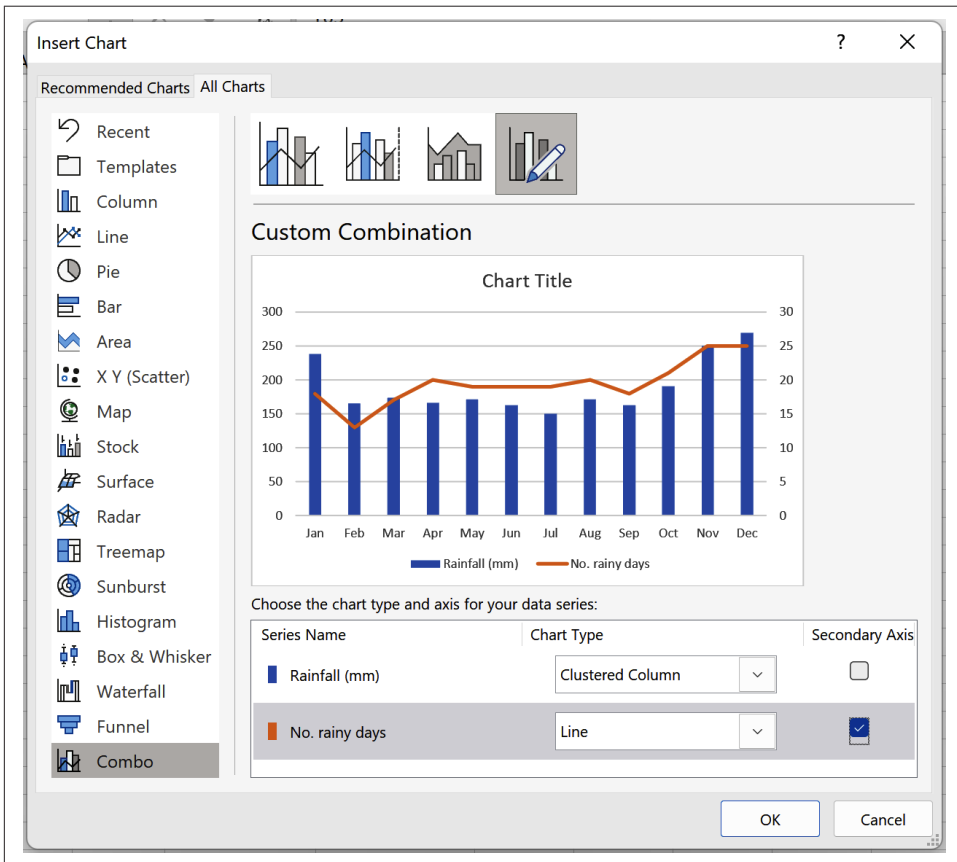
Figure 12-22. Inserting a combination chart where one uses a secondary axis

To make further adjustments to the chart types, select the chart and choose Chart Design ⇒ Type ⇒ Change Chart Type to open the Change Chart Type dialog box.

Discussion

This recipe shows you how to insert a combination chart that uses different chart types for different series. When choosing the chart types, consider the overall purpose, how each type contributes to this, and how well they work together. For example, combining clustered column and line chart types works well, while combining clustered column and pie chart types doesn't.

12.15 Handling Empty Cells

Problem

You have a chart and want to control how to display any missing data points.

Solution

Suppose you have a line chart whose underlying table includes empty cells for some data points. By default, Excel displays the missing data points as gaps. However, you can override this by selecting the chart, choosing Chart Design ⇒ Data ⇒ Select Data to open the Select Data Source dialog box, clicking the Hidden and Empty Cells button, and selecting one of the "Show empty cells as" options. These options are Gaps (the default, which leaves gaps), Zero (which treats empty cells as zero), and "Connect data points with line" (which draws a line between the two data points on either side of the missing value). You can also specify whether to treat the #N/A error value as an empty cell and show or hide data in hidden rows and columns (see Recipe 12.3).

Discussion

This recipe lets you control how a chart interprets missing data points. However, not all options are available for every chart type, so you can't, for example, apply the "Connect data points with line" option to a pie chart.

12.16 Basing a Chart on Noncontiguous Data

Problem

You want to insert a chart based on table columns that aren't adjacent or are in separate tables.

Solution

To create a chart based on multiple data ranges, first select all the data you want to base the chart on; you can do so by selecting the first range, pressing the Ctrl key (or

the Cmd key in Excel for Mac), and then selecting any other ranges. Then insert the chart by choosing Insert ⇒ Charts and selecting a chart type (see Recipe 12.2).

Discussion

This recipe shows you how to insert a new chart based on nonadjacent columns. To add an extra data series to an existing chart, see Recipe 12.17.

12.17 Changing a Data Series Name and Legend Entry

Problem

You have a chart and want to change a data series name and legend entry.

Solution

When you add a legend to a chart, Excel uses the names of each data series for its legend entries. To change one of the entries, you must change the name of the corresponding data series. To do so, follow these steps (see Figure 12-23):

1. Select the chart and choose Chart Design ⇒ Data ⇒ Select Data to open the Select Data Source dialog box.

2. In the Legend Entries (Series) section, select the data series whose name you want to change; then click the Edit button to open the Edit Series dialog box.

3. Type the new name of the series in the Series Name box, or select a cell whose value you want to use for it.

4. Click OK to close the Edit Series dialog box, then click OK again to close the Select Data Source dialog box.
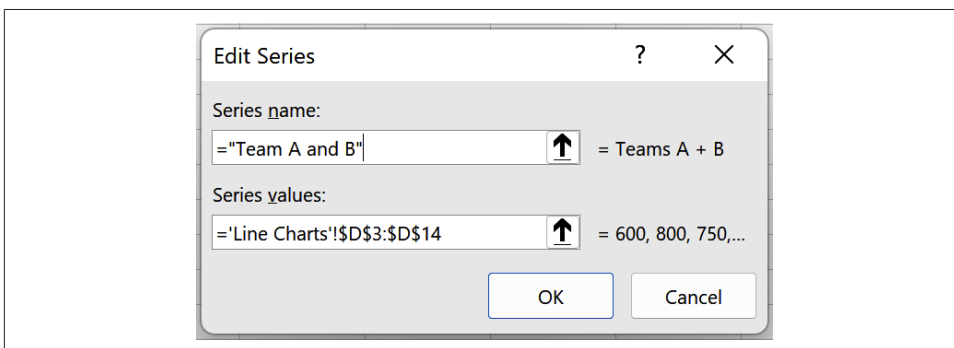
Figure 12-23. Editing a data series to change the series name

Discussion

This recipe shows you how to change the name of a chart's data series, which Excel displays in the legend. You can either hardcode the series name by typing it directly in the Series Name box or make the name dynamically update using a cell reference (similar to Recipe 12.7).

12.18 Adding a Series or Changing the Data Source

Problem

You have a chart and want to change one or more of the ranges it uses for its data or add a new data series.

Solution

Suppose you want to change how a chart uses its data or provide it with new data. You can use several methods, depending on what you want to achieve.

The first option is to manually adjust the data range. To do this, select the chart to show its data range; then click and drag the selected area's corners to resize it.

The second option is to switch the chart's horizontal and vertical axes; select the chart, then choose Chart Design ⇒ Data ⇒ Switch Row/Column.

The final option is to edit the chart's data source. Select the chart and choose Chart Design ⇒ Data ⇒ Select Data to open the Select Data Source dialog box. Then do one or more of the following:

Enter a new range in the "Chart data range" box.
> Doing this replaces the chart's data range and automatically updates the ranges it uses for its series and horizontal axis.

Edit a data series.
> Select the series you want to update in the Legend Entries (Series) section and click the Edit button to open the Edit Series dialog box. Then, use the available options—which depend on the chart type—to update the series data ranges.

Add a new series.
> Click the Add button in the Legend Entries (Series) section to open the Edit Series dialog box; then complete the various options.

Delete a series.
> Select the series you want to delete in the Legend Entries (Series) section and click the Remove button to delete it. Alternatively, select the data series in the chart and press the Delete key.

Change the values or labels in the horizontal or category axis.

In the Horizontal (Category) Axis Labels section, click the Edit button to open the Axis Labels section; then replace the range in the "Axis label range" box.

Discussion

This recipe is handy if you have an existing chart and want to tweak its data ranges or add a new range. For example, you may want to base a chart on a dynamic named range instead of a table (see Recipe 12.19) and fine-tune the data series or horizontal axis labels. The most flexible option is the Select Data Source dialog box because this determines which range the chart uses for each purpose.

12.19 Basing a Chart on a Dynamic Named Range

Problem

You want to insert a chart based on a dynamic named range.

Solution

Suppose you have a worksheet named Sales and want to insert a line chart showing the last five values in its A column. You can do so by basing the chart on a dynamic named range as follows:

1. Create a dynamic named range for the data series by choosing Formulas ⇒ Define Name to open the New Name dialog box and entering a name and formula for the dynamic named range (see Recipe 2.7). To return the last five values in the Sales worksheet's A column, you'd type **LastFiveSales** in the Name box, **=OFFSET(Sales!$A$1,COUNTA(Sales!$A:$A)-5,0,5,1)** in the Refers To box, leave the Scope as `Workbook`, then click OK (see Figure 12-24).
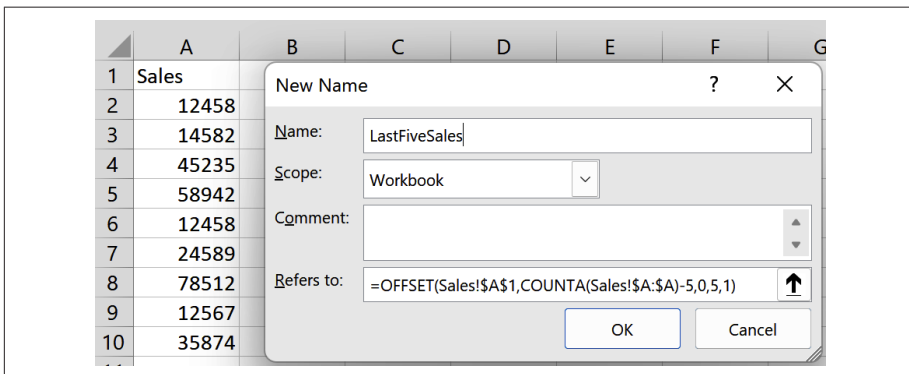
Figure 12-24. Defining the `LastFiveSales` *dynamic named range*

2. Insert the chart by selecting an empty cell, then choosing Insert ⇒ Charts ⇒ Insert Line or Area Chart, and selecting the 2-D Line option.

3. Select the chart and choose Chart Design ⇒ Data ⇒ Select Data to open the Select Data Source dialog box.

4. In the Legend Entries (Series) section, click Add to open the Edit Series dialog box.

5. Enter a reference to the dynamic range in the "Series values" box, including the name of a worksheet with access to it. So, to enter the dynamic named range you created in step 1, you'd type **=Sales!LastFiveSales**, where `Sales!` refers to a worksheet with access to the `LastFiveSales` dynamic named range (see Figure 12-25).
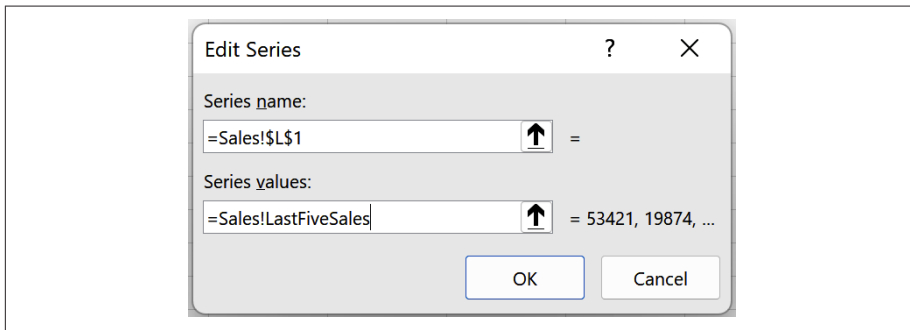
Figure 12-25. Adding the `LastFiveSales` dynamic named range to the Edit Series dialog box

6. Click OK to close the Edit Series dialog box and click OK again to close the Select Data Source dialog box.

> If you omit the worksheet name in step 5, Excel displays an error message because it doesn't recognize the name of the dynamic named range. To avoid this error, include a worksheet name when referencing the dynamic named range.

Discussion

Basing a chart on a dynamic named range is more complex than basing it on a table, but it's necessary in some situations. This recipe shows you how.

12.20 Inserting a PivotChart

Problem

You want to display PivotTable data in a chart.

Solution

You have a PivotTable (see Chapter 11) and you want to display its data in a chart. You can do so by inserting a *PivotChart*, a graphical representation of PivotTable data.

To insert the PivotChart, select a cell in the PivotTable and choose PivotTable Analyze ⇒ Tools ⇒ PivotChart, or Insert ⇒ Charts ⇒ PivotChart. If you're using Excel for Windows, you can then select the type of PivotChart you want to create, such as a clustered column chart (see Figure 12-26).
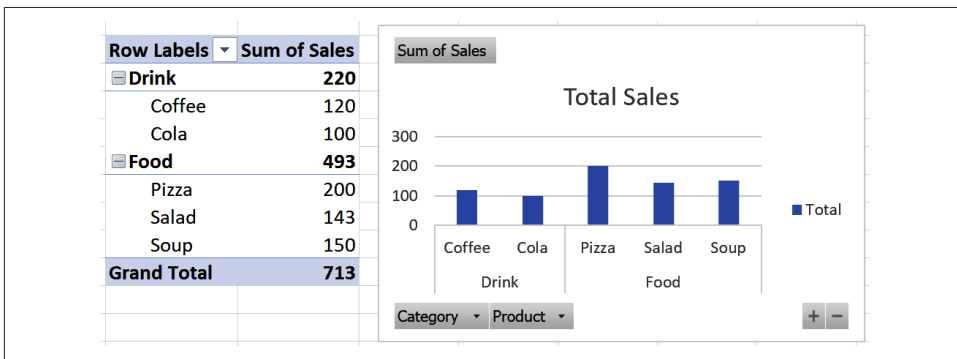
Figure 12-26. A PivotTable and PivotChart

If you haven't already created a PivotTable, you can insert a PivotChart (with or without an accompanying PivotTable) by selecting a cell in the table you want to base it on, choosing Insert ⇒ Charts, opening the PivotChart menu, and selecting the PivotChart or PivotChart & PivotTable option.

Once you've inserted a PivotChart, you can use its PivotChart Fields pane to control which data to display on the chart. The PivotChart Fields pane works similarly to the PivotTable Fields pane (see Recipe 11.3), except that it includes Legend (Series) and Axis (Categories) sections instead of Columns and Rows. Adding a field to one of these sections adds its data to the PivotChart.

There's a two-way link between a PivotChart and its accompanying PivotTable—if there is one—so adding a field to the PivotTable's Columns section also adds it to the PivotChart's Legend (Series) section, and vice versa. If you add multiple PivotCharts to a Pivot-Table, they will all show the same data.

Similarly to a PivotTable, you can add slicers and timelines to a PivotChart to filter its data (see Recipe 11.14), which can be helpful when creating dashboards. If you're using Excel for Windows, the PivotChart also includes field buttons, which you can use to filter the chart's data. You can show or hide these buttons by selecting the PivotChart and choosing PivotChart Analyze ⇒ Show/Hide ⇒ Field Buttons.

Discussion

A PivotChart provides an effective way of visualizing PivotTable summaries and the insights they provide and combines the features of both PivotTables and charts. This recipe gives an overview of how to use and work with them.

12.21 Creating a Gantt Chart

Problem

You want to display a project schedule in a Gantt chart.

Solution

By default, Excel doesn't include a predefined Gantt chart. However, you can approximate one using a 2-D Stacked Bar chart as follows (see Figure 12-27):

1. Create a table with three columns: Task Name, Start Day, and Duration. Populate the table with the date you want to display in the chart.

2. Select the table and insert a 2-D Stacked Bar chart by choosing Insert ⇒ Charts ⇒ Insert Column or Bar Chart and selecting the Stacked Bar option from the 2-D Bar section.

3. Select the Start Day data series; then set its fill to No Fill and its border to No Line (see Recipe 12.6).

4. Select the vertical axis containing the Task Name labels and reverse its order by opening the chart's Format pane, clicking the Axis Options button, and checking the "Categories in reverse order" check box.

5. Change the chart title to reflect its purpose and delete the legend.
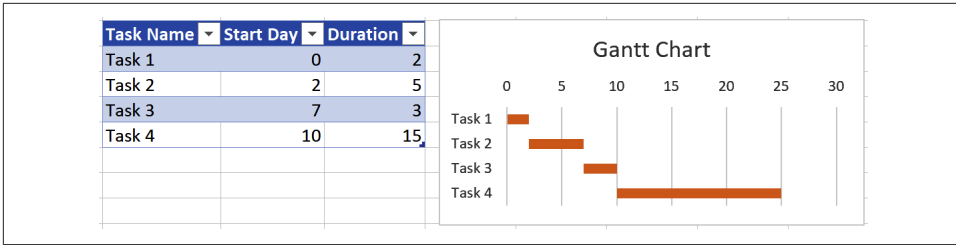
Figure 12-27. A Gantt chart created from a 2-D Stacked Bar chart

Discussion

This recipe shows how to format a 2-D stacked bar chart like a Gantt chart. You can use a similar technique to create other charts too; select the chart that's most like the one you want to create, then customize its format.

Once you've created the chart and formatted it to your liking, you can save it as a chart template to reuse it elsewhere (see Recipe 12.22).

12.22 Creating and Using Chart Templates

Problem

You've created a chart and want to be able to use its format and settings elsewhere.

Solution

Suppose you've used Recipe 12.21 to create a custom Gantt chart and want to reuse it without re-creating it from scratch. You can do so by saving the chart as a template.

To save the chart as a template, right-click it and choose the Save As Template option. Enter a name for the template, then click Save to save it in the folder Excel uses for chart templates.

To create a new chart based on the template, if you're using Excel for Windows, choose Insert ⇒ Charts ⇒ Recommended Charts to open the Insert Chart dialog box, click All Charts, then select the template from the Templates section. If you're using Excel for Mac, choose Insert (from the Mac top menu bar) ⇒ Chart ⇒ Templates and select the template.

To apply the template to an existing chart, select the chart and choose Chart Design ⇒ Type ⇒ Change Chart Type ⇒ Templates; then select the template.

Discussion

This recipe shows you how to add to Excel's predefined chart options by creating a chart template. This feature lets you apply formats and settings similarly to using a workbook template (see Recipe 1.7).

Graphics, Sparklines, and 3D Maps

Graphics can help bring spreadsheets to life and communicate your message more effectively than text and numbers alone. While Chapter 12 focused solely on charts, this chapter covers other areas—from using symbols and shapes to creating 3-D data visualizations.

The recipes in this chapter include areas such as the following:

- Inserting symbols, shapes, sketches, and pictures
- Creating SmartArt diagrams
- Using the Camera tool to create linked pictures and monitor parts of your workbooks
- Working with sparklines
- Using 3D Maps to explore geographic and temporal data and create virtual tours and movies

13.1 Inserting Symbols

Problem

You want to add symbols or special characters to a cell or custom number format.

Solution

Suppose you want to add a copyright character to a cell or up and down arrows to a custom number format. You can do so by inserting a symbol or special character.

You can add a symbol or special character to a cell as follows (see Figure 13-1):

1. Select the cell and put the cursor where you want to insert the symbol or special character.

2. Choose Insert ⇒ Symbols ⇒ Symbol to open the Symbol dialog box.

3. Click Symbols or Special Characters, depending on the type of character you wish to insert, then browse the available options. If you've clicked the Symbols option, you can use the Subset drop-down list to see all the available characters in a specific category, such as arrows or geographic characters.

4. Select the symbol you wish to insert; then click the Insert button to insert it.

5. Repeat step 4 for each symbol you wish to insert, then close the dialog box.
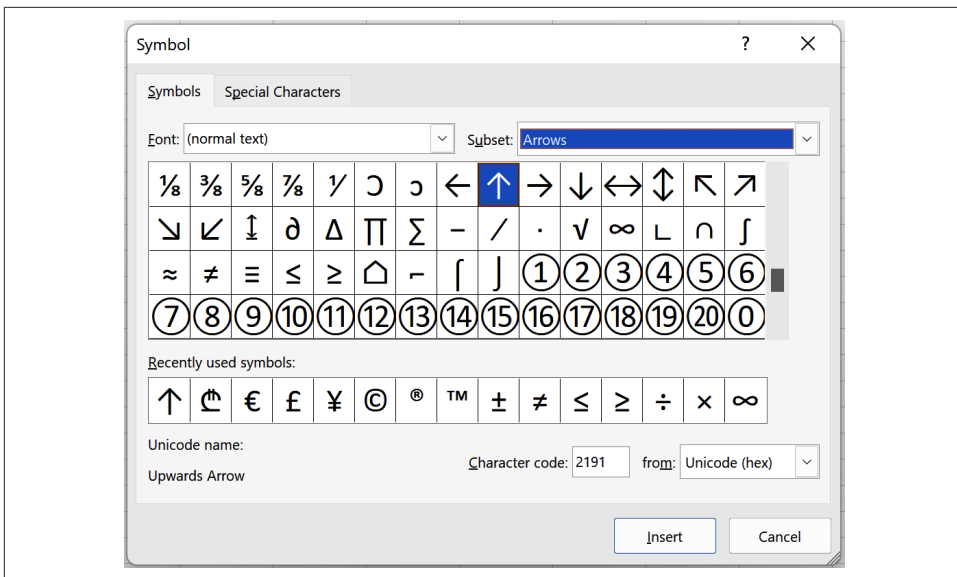
Figure 13-1. The Symbol dialog box

To add a symbol or special character to a custom number format, insert it in a cell, copy it by selecting the character and pressing Ctrl+C, then paste it into the custom number format by pressing Ctrl+V.

Discussion

This recipe shows you how to insert symbols and special characters using the Symbol dialog box. You can also enter these characters using the `UNICHAR` function (see Recipe 5.2) or by customizing AutoCorrect to replace a short typed code with a symbol (see Recipe 1.15).

13.2 Inserting Equations

Problem

You want to add an equation to a worksheet.

Solution

To insert an equation, choose Insert ⇒ Symbols ⇒ Equation to create an equation from scratch. Alternatively, open the Equation drop-down menu and select one of the predefined equations—for example, Area of Circle or Binomial Theorem.

When you insert an equation, Excel adds it to the worksheet as a floating or embedded object. You can then edit it by selecting it and using the available options in the Equation menu.

Discussion

This recipe shows you a handy way of inserting an equation as a floating or embedded object, which you can then move and edit.

13.3 Inserting Shapes

Problem

You want to add a text box, arrow, rectangle, or other shape to a worksheet.

Solution

Suppose you have a worksheet and want to add a shape, such as a text box, callout, or arrow. You can do so by choosing Insert ⇒ Illustrations ⇒ Shapes, selecting one of the available options, and clicking where to place it.

Once you've added the shape, you can move it by clicking and dragging it, resize it by clicking and dragging its corners and edges, rotate it by clicking and dragging its rotate icon ⟳, or format it using the available options in the Shape Format menu; you can also right-click the shape and choose Format Shape to open the Format Shape pane.

You can add text to most shapes by double-clicking the shape and typing the text you want to include. You can also link the text to a cell's value so the text dynamically changes when you update the cell: select the shape and then type `=cell_reference` in the formula bar. So, to add the contents of cell A1 to a text box, you'd insert the text box, select it, and then type `=$A$1` in the formula bar.

Discussion

This recipe gives an overview of how to add a shape to a worksheet so you can, for example, insert a callout or arrow. You can also add a shape to a chart by selecting the chart and choosing Format ⇒ Insert Shapes (see Recipe 12.6).

See Also

You can add shapes by drawing them with the Draw tool and converting the ink to shapes; see Recipe 13.4.

To learn how to add more complex shapes, see Recipe 13.5.

13.4 Using the Draw Tool

Problem

You want to insert a quick sketch or shape by drawing it on a worksheet.

Solution

Suppose you have a worksheet and want to draw a quick sketch to illustrate a point. You can do so by choosing Draw ⇒ Drawing Tools, selecting a pen, and then drawing on the worksheet.

> If your ribbon doesn't include the Draw tab, you can use Recipe 1.19 to enable the tab and add it to the ribbon.

You can customize a pen by selecting it, clicking its drop-down arrow, and choosing from the available options. You can also convert your ink marks to one or more shapes by choosing Draw ⇒ Drawing Tools ⇒ Lasso Select, circling the ink, then choosing Draw ⇒ Convert ⇒ Ink to Shape.

Once you've finished drawing, choose Draw ⇒ Drawing Tools ⇒ Select to deactivate the pen.

To remove or hide any ink you've added, you can either select the ink and press the Delete key or choose Review ⇒ Ink ⇒ Hide Ink and choose from the available options (Hide Ink, Remove All Ink on Sheet, and Remove All Ink in Workbook).

Discussion

Drawing on a worksheet can be handy if you're sharing your screen and want to draw a quick sketch or if you want to insert shapes by drawing them instead of using Recipe 13.3. This recipe provides an overview of how to do so.

13.5 Using SmartArt

Problem

You want to insert a diagram, such as an organization chart or Venn diagram.

Solution

Excel includes a variety of predefined SmartArt graphics, including a basic Venn diagram, organization charts, process diagrams, and cycle diagrams. You can insert one of these by choosing Insert ⇒ Illustrations ⇒ SmartArt, selecting one of the options, and clicking OK (see Figure 13-2).
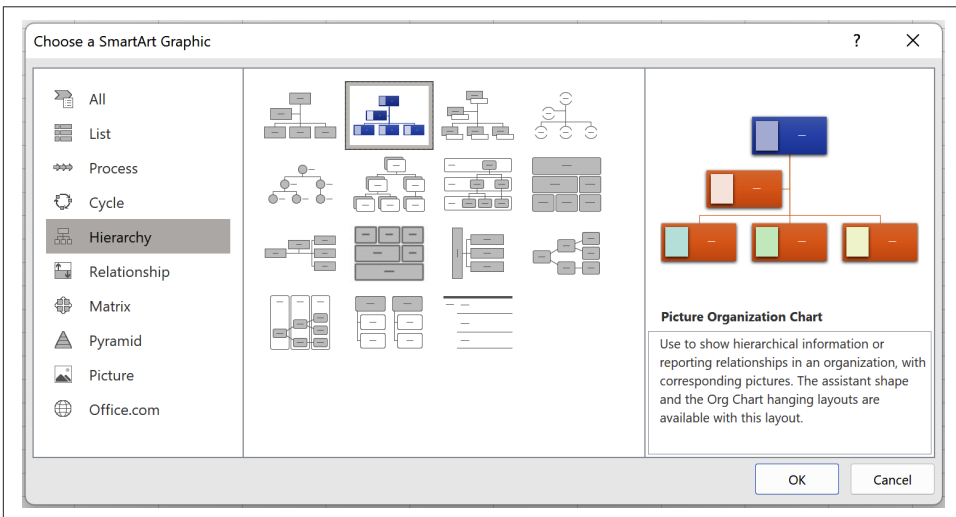
Figure 13-2. Some of the available SmartArt graphics

Once you've inserted the graphic, you can move it by clicking and dragging it, and change its appearance by choosing SmartArt Design or Format and using the available options; for example, you can change its colors or style. You can also right-click the graphic and choose Format Object to open the Format Shape pane.

To change the text, you can either type directly in the graphic or use its *text pane*—a text editing pane on the left of the graphic. To open the text pane, click the arrow on the graphic's left edge or choose SmartArt Design ⇒ Create Graphic ⇒ Text Pane. To add more shapes or rearrange the text, choose SmartArt Design ⇒ Create Graphic and use the available options.

SmartArt graphics can contain only static text, so you can't link the text to a cell's value. However, you can convert the graphic to shapes by choosing SmartArt Design ⇒ Reset ⇒ Convert to Shapes, and then use Recipe 13.3 to set each shape's text to a cell reference.

Discussion

This recipe gives an overview of using SmartArt graphics to insert diagrams. Try browsing the available options to see what's available.

13.6 Inserting Pictures

Problem

You want to add a picture to a worksheet, cell, header, or footer.

Solution

To add a picture to a worksheet, choose Insert ⇒ Illustrations ⇒ Pictures (or Insert ⇒ Illustrations ⇒ Pictures ⇒ Place Over Cells in Excel 365), and select This Device, Stock Images, or Online Pictures, depending on the picture location. Then select a picture—or use the Ctrl key to select multiple pictures—and click Insert.

You can also copy and paste a picture from another worksheet, workbook, or application by selecting the picture, pressing Ctrl+C, selecting where you want to place the picture, and pressing Ctrl+V.

If you're using Excel 365, you can add a picture to a cell by selecting the cell, choosing Insert ⇒ Illustrations ⇒ Pictures ⇒ Place In Cell, and then selecting the picture. In other versions of Excel, you can achieve similar results using the following steps:

1. Add the picture to the worksheet, move it to the cell, and resize it to fit inside. If necessary, you can resize the cell's rows and columns to make the cell bigger or use Recipe 1.6 to merge several cells.

2. Select the picture, right-click it, and choose the Size & Properties option to open the Format pane. If the Format pane is already open, click the Size & Properties button to open the appropriate section.

3. Select the "Move and size with cells" option from the Properties section; doing so ensures that the picture moves with the cell when you resize the rows and columns and isn't displayed if you hide the cell or apply a filter that excludes it.

If you're using Excel 365, you can also add a BMP, JPG/JPEG, GIF, TIFF, PNG, ICO, or WEBP picture to a cell using the IMAGE function. Generally, you use the formula =IMAGE(*source, alt_text, sizing, height, width*), with arguments as follows:

source
> This is the image file's URL. It must use an HTTPS protocol and be 255 characters or less. If the URL is longer than 255 characters, try pasting it in a cell and using the formula =IMAGE(*cell_reference*) instead. If the URL you pass to the IMAGE function needs authentication, Excel doesn't render the image, and it also blocks any redirected URLs due to security concerns.

alt_text
> Use this to provide alternative text describing the image for accessibility (optional); see Recipe 1.21.

sizing
> Use this to specify the image dimensions (optional). Use 0 to fit the image in the cell and maintain its aspect ratio, 1 to fill the cell with the image and ignore its aspect ratio, 2 to keep the original image size, and 3 to specify the image size using the *height* and *width* arguments.

height and width
> These specify the image height and width in pixels (optional). If you provide only one of these, the image maintains its aspect ratio. You must provide one of these arguments if the *sizing* argument is 3, and you must omit them if *sizing* is 0, 1, or 2.

Finally, you can add a picture to a worksheet's header or footer. Choose Insert ⇒ Text ⇒ Header & Footer to add the header or footer; then select the header or footer and choose Header & Footer ⇒ Header & Footer Elements ⇒ Picture to insert a picture.

Discussion

This recipe describes adding pictures to your workbook using menu commands and the IMAGE function. Since the IMAGE function uses a URL, you can dynamically change the image by updating the URL.

See Also

See Recipe 13.8 for more details about moving and sizing objects—including pictures—with cells.

13.7 Grouping Objects

Problem

You have several floating or embedded objects in a worksheet and want to keep them together when you move them.

Solution

Suppose you have two or more floating or embedded objects—such as shapes, pictures, SmartArt graphics, or charts—in a worksheet, and you want to move them together easily. You can do so by grouping the objects as follows:

1. Select the first object, press the Ctrl key, and select any others you want to group.
2. Choose Shape Format ⇒ Arrange ⇒ Group Objects ⇒ Group.

Once you've grouped the objects, clicking one of them selects the group so you can move them all together; clicking the object again selects the object itself so you can move it within the group.

To ungroup the objects, select the group and choose Shape Format ⇒ Arrange ⇒ Group Objects ⇒ Ungroup.

You can also use the options in the Arrange group to align, rotate, and flip objects. Select the object or objects; then choose the appropriate option.

Discussion

Trying to keep multiple objects together when you move them can take time and effort, and objects can often go astray. This recipe shows you how to group the objects to quickly and easily move them together.

13.8 Moving and Sizing Objects with Cells

Problem

You have a floating or embedded object and want to control its position and size when you resize, insert, delete, hide, or filter rows and columns.

Solution

Suppose you have a floating or embedded object—a shape, picture, or SmartArt graphic—and you want to control whether it moves or changes size with the worksheet's cells, such as when you resize, insert, delete, or hide rows or columns or apply a filter. You can do so by selecting the object, opening its Format pane (see Recipe 13.3), clicking the Size & Properties button, and selecting one of the move and size options in the Properties section. These options include the following:

Move and size with cells
> This option (the default) repositions and resizes the object when you make changes to the rows and columns.

Move but don't size with cells
> This repositions the object but keeps its size intact.

Don't move or size with cells
> This maintains the object's position and size so it stays fixed.

Discussion

This recipe gives an overview of how to control whether a floating or embedded object moves or resizes when you make changes to the worksheet's rows and columns. Doing so can be helpful if, for example, you want to insert a picture in a cell, so hiding or filtering the row also hides the picture.

You can use a similar technique to ensure that a chart doesn't resize when you filter the rows it's based on (see Recipe 12.3).

13.9 Inserting a Linked Picture

Problem

You want to copy part of a worksheet and display it as a static or dynamic picture.

Solution

Suppose you're working with formulas in several worksheets or workbooks, and you want to be able to monitor cell ranges from each one. You can solve this problem by creating a *linked picture*: a picture that's like a live feed of part of a worksheet that automatically mirrors any changes you make.

To create the linked picture, select the range you want to be able to monitor, press Ctrl+C to copy it, then go to where you want to place the linked picture and choose Home ⇒ Clipboard ⇒ Paste ⇒ Linked Picture. Doing so pastes the linked picture as a floating or embedded object, so if you make any changes to the cell range you've

copied—such as changing cell values or inserting a chart—the linked picture automatically displays them.

You can also insert a linked picture using Excel's little-known Camera tool. Use Recipe 1.20 to add the Camera command to the Quick Access Toolbar, select the range you want to create a linked picture of, click the Camera option in the Quick Access Toolbar, then click where you want to insert the picture. Once you've added the linked picture, you can quickly navigate to its source by double-clicking it.

To insert a static snapshot of a cell range that doesn't update, select the cell range, press Ctrl+C to copy it, select where you want to insert it, and choose Home ⇒ Clipboard ⇒ Paste ⇒ Picture. This option is generally available across other Office applications, so you can also use it, for example, to paste a static picture of a cell range into Word.

> If you want to add a picture of a cell range to non-Office applications, select the cell range, then copy it by choosing Home ⇒ Clipboard ⇒ Copy ⇒ Copy as Picture; this opens the Copy Picture dialog box, which you can use to specify the picture's format.

Discussion

Linked pictures are handy when you want to see part of another worksheet or workbook from elsewhere. For example, you can use them to monitor calculations instead of using the Watch window (see Recipe 3.10).

You can also use linked pictures in dashboards to mirror any cells, charts, diagrams, and so on that you want to share.

13.10 Using Sparklines

Problem

You want to visualize a data series and highlight key characteristics by inserting a tiny chart in a worksheet cell.

Solution

Suppose you have a row or column of numeric data and you want to see its trend and high and low points in a tiny chart. In this situation, you can insert a *sparkline*: a simple chart-like graphic that fits in a single cell.

Excel includes the following sparkline types:

Line
> This option resembles a line chart, so it's handy for displaying the data's general trend.

Column
> This option is like a column chart, so you can use it to compare relative values and highlight high and low points.

Win/Loss
> This option indicates whether each value is positive or negative, so you can use it, for example, to scan a data series for any negative values. It displays each positive value as a block above the horizontal axis, each negative value as a block below it, and leaves any zero values blank.

Figure 13-3 shows the three types of sparkline.
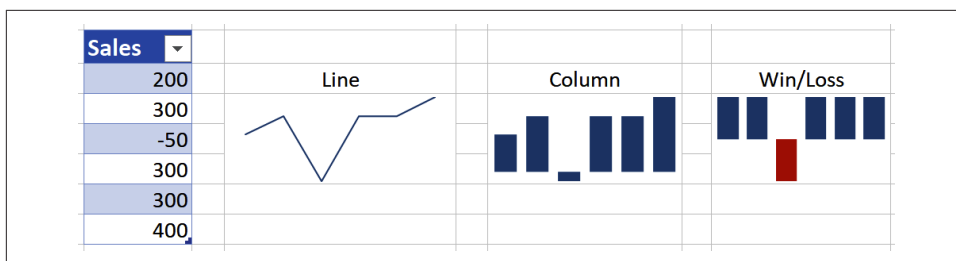
Figure 13-3. The three types of sparkline: Line, Column, and Win/Loss

> Sparklines don't display axis values or data labels, so you can't use them to find the values or specific data points. They instead show the data series' general trend and the relative value of each data point.

You insert a sparkline as follows:

1. Select the data range you want to base the sparkline on. Make sure you exclude any headings.

2. Choose Insert ⇒ Sparklines and select the type of sparkline you want to insert. So, to insert a sparkline that resembles a line chart, you'd select the Line option.

3. In the Create Sparklines dialog box, ensure that the data range entered in the Data Range box is correct; then add a cell reference to the Location Range box to specify where to insert the sparkline.

4. Click OK to create the sparkline.

You can also base a sparkline on a dynamic named range. To do so, use Recipe 2.7 to create the named range, then type its name in the Data Range box in step 3. For example, to create a sparkline that uses a dynamic named range called LastTenSales for its data, you'd type `=LastTenSales` in the Data Range box.

A sparkline fits inside a single cell, so once you've inserted one, you can change its size by resizing the cell it's in. For example, you can resize the cell's row or column or use Recipe 1.6 to make the cell span more than one row and/or column.

You can customize the sparkline by selecting it and using the available options in the Sparkline tab. These options are as follows:

Edit Data

Select the Edit Single Sparkline's Data option to open the Edit Sparkline Data dialog box and change the sparkline's source data range; if you want to also move the sparkline or you're using a sparkline group (see Recipe 13.11), select the Edit Group Location & Data option instead. To control how to display empty cells or handle hidden rows and columns, use the Hidden & Empty Cells option to open the Hidden & Empty Cell Settings dialog box.

Type

Select Line, Column, or Win/Loss to change the sparkline's type.

Show

This group contains a series of check boxes that you can use to highlight specific characteristics, such as high and low points, negative points, the first and last points, and to include point markers.

Style

You can apply a different style to the sparkline by choosing one from the style gallery, use the Sparkline Color option to change the sparkline's color and weight, and use the Marker Color option to change the colors it applies to its markers. For example, you can use different colors for the high and low points of the sparkline.

Group

Use the Axis option to control the sparkline's axes. For example, you can display a horizontal axis (if there are any negative data points) by selecting the Show Axis option, or change the minimum and maximum bounds for the vertical axis. Use the Group and Ungroup options to manually create a sparkline group or ungroup an existing one (see Recipe 13.11). Finally, select the Clear option to delete a sparkline (or sparkline group).

Discussion

A sparkline is handy if you have a numeric data series and want to quickly visualize its trend, compare relative values, highlight high and low points, or scan for positive, negative, or zero values. If you have more than one data series, you can create a sparkline group (see Recipe 13.11).

13.11 Using Sparkline Groups

Problem

You have more than one data series—for example, each row in a table—and want to create a sparkline for each one.

Solution

Suppose you have a table containing stock data, and the data for each stock is in a separate row. You can quickly create a sparkline (see Recipe 13.10) for each row using a sparkline group.

To create the sparkline group, follow these steps (see Figure 13-4):

1. Select the data range for each row on which you want to base the sparkline, excluding any headings.

2. Choose Insert ⇒ Sparklines and select the type of sparklines you want to insert (for example, Line).

3. In the Create Sparklines dialog box, ensure that the data range entered in the Data Range box is correct, then use the Location Range box to specify where to insert the sparklines. For example, if the data range includes five rows and you want to insert a sparkline for each row, the location range must be one column wide and five rows high.

4. Click OK to create the sparkline group.

Stock no	Jan	Feb	Mar	Apr	May	
1	201	191	192	178	176	
2	53	56	60	57	55	
3	-5	10	20	40	80	
4	100	101	99	100	100	
5	176	178	192	191	201	

Figure 13-4. A table with a sparkline group

You can also manually group sparklines you've already created by selecting the sparklines and choosing Sparkline ⇒ Group ⇒ Group. To remove a sparkline from a group, select the sparkline and choose Sparkline ⇒ Group ⇒ Ungroup.

Sparkline groups make it easy to apply changes to every sparkline in the group because when you select one of the sparklines, Excel automatically selects them all. For example, to change every sparkline in a group to a Column sparkline, select one of the sparklines, then choose Sparkline ⇒ Type ⇒ Column.

> By default, Excel automatically scales the vertical axis for each sparkline, so each sparkline in the group may use a different scale. To use the same scale for each sparkline, select one of the sparklines, choose Sparkline ⇒ Group ⇒ Axis, and specify minimum and maximum values for the vertical axis.

If you have a sparkline group based on table data, the group doesn't automatically accommodate any new rows or columns you add. However, you can manually edit the group by selecting one of the sparklines and choosing Sparkline ⇒ Edit Data ⇒ Edit Group Location & Data to open the Edit Sparklines dialog box. You can also add new sparklines to a sparkline group by clicking and dragging a sparkline cell's fill handle (see Recipe 1.12).

Discussion

Sparkline groups are handy if you want to display a sparkline for each row in a table or change several sparklines simultaneously. This recipe gives an overview of how to do this.

13.12 Using 3D Maps

Problem

You want to display geographic data on a 3-D map and (optionally) show changes over time.

Solution

Suppose you have geographic data in a table (see Figure 13-5) and you want to plot it on an interactive 3-D map. If you're using Excel for Windows, you can use 3D Maps: a 3-D data visualization tool that lets you explore and create virtual tours of geographic data, including how it changes over time.

State	Population	Date
California	39,538,223	1/1/2020
California	37,253,956	1/1/2010
California	33,871,648	1/1/2000
California	29,760,021	1/1/1990
Nevada	3,104,614	1/1/2020
Nevada	2,700,551	1/1/2010
Nevada	1,998,257	1/1/2000
Nevada	1,201,833	1/1/1990
Arizona	7,151,502	1/1/2020
Arizona	6,392,017	1/1/2010
Arizona	5,130,632	1/1/2000
Arizona	3,665,228	1/1/1990
Utah	3,271,616	1/1/2000
Utah	2,763,885	1/1/2010
Utah	2,233,169	1/1/2000
Utah	1,722,850	1/1/1990

Figure 13-5. Geographic data to plot using 3D Maps

To create a 3-D map or tour, select a cell in the table and choose Insert ⇒ Tours ⇒ 3D Map. If this is the first time you've used 3D Maps, Excel will prompt you to enable an add-in, then open 3D Maps in a new window. If you've previously used 3D Maps, Excel will display a list of existing tours, and you'll need to click the New Tour button to create a new one.

The 3D Maps window displays a Layer pane on the right (see Figure 13-6). You use this to configure how to use the data in the map: You do so as follows:

1. In the Data section, select the type of visualization you want to display on the map—choose from Stacked Column, Clustered Column, Bubble, Heat Map, and Region.

2. Make sure the Location box includes any table columns—or *fields*—with the geographic data you want to plot and the tool has set each one to the correct type. Change any incorrect type and click Add Field to add any fields you want to include.

3. To see how confident 3D Maps is that it can correctly plot each location, click the percentage just above the Location. The percentage tells you the degree of confidence, and 3D Maps reports on any points it's uncertain about.

4. If the table includes date/time columns and you want to see how the data changes over time, add these to the Time section. Once you've added one, you can optionally use its drop-down box to specify the granularity you want to apply, such as using just its year.

5. Use the remaining boxes in the Data section to specify any extra fields you want to use for the visualization's data; these sections will depend on the option you choose in step 1. For example, if you select a clustered column, you use the Height box to determine column heights.

6. You can optionally adjust the Filters and Show Values sections to filter the data; specify whether to show zero, negative, or null values; and update other options.

Figure 13-6 shows a 3-D map with completed Layer pane sections. The map shows a column for each state; the column height represents the population size.
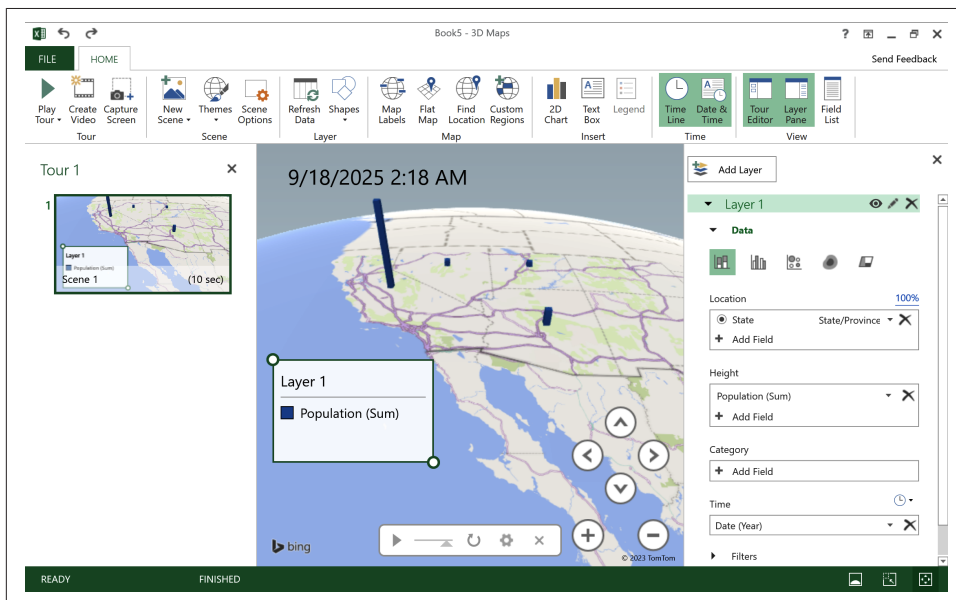
Figure 13-6. A 3-D map showing population data for four states

To add a pie chart to the map, choose the Bubble option in step 1 and add fields to the Size and Category boxes in step 5. The Category box determines the pie slice categories, and the Size box determines the size of each slice. If you don't add any fields to the Category box, the Size box determines the size of each bubble.

Once you've configured the data, the 3D Maps tool displays it on the globe in the center of its window. Use the on-screen controls or mouse to rotate the globe or zoom in and out. If you've completed the layer's Time section, you'll also see a slider you can adjust to see the data at a particular time. You can control this animation by clicking the clock icon ⏱· above the Time box and choosing one of the available options to specify whether the data stays for an instant, accumulates over time, or stays until it's replaced.

If you want to display the data on a flat map instead of a globe, choose Home ⇒ Map ⇒ Flat Map. You can also include labels such as country, state, and city by choosing Home ⇒ Map ⇒ Map Labels and apply a different theme by choosing Home ⇒ Scene ⇒ Themes.

When you hover your mouse pointer over each data point, the tool displays a data card showing the data it's using for that point. You can customize these cards by selecting one and then clicking the settings icon ⚙ in its top-right corner.

> Each layer describes a type of visualization using specific data. Generally, each 3-D map or tour includes a single layer. However, you can build more complex models by adding extra layers, each specifying a separate visualization. You can choose whether to show or hide each layer by clicking its Show/Hide icon 👁 in the Layers pane.

Discussion

This recipe gives an overview of using Excel's 3D Maps tool to explore geographic data and see how it varies over time. This tool is far more advanced than using the Filled Map chart described in Recipe 12.1.

13.13 Creating Videos with 3D Maps

Problem

You have geographic data and want to use it to create a virtual tour or video.

Solution

Suppose you've used Recipe 13.12 to plot data using a 3-D map or tour. You can create a video of the tour by configuring one or more scenes. For example, you can create a video that circles the map's visualizations, showing changes over time.

By default, 3D Maps creates a scene for the visualization, which you can find in the Tour Editor pane to the left of the map (see Figure 13-6). To configure the scene, open the Scene Options dialog box by hovering your mouse pointer over it, clicking its settings icon ⚙, or selecting the scene and choosing Home ⇒ Scene ⇒ Scene Options. Then, use the available options to specify the scene duration, apply effects, and specify a time interval if you've added data to the layer's Time section. For example, you can create a scene lasting 30 seconds that circles the map and shows columns building over time.

To play a scene, hover your mouse pointer over it and click its play icon.

To add a new scene, choose Home ⇒ Scene ⇒ New Scene, then select the scene to edit it; you can move to a different part of the map, change the zoom level, change the map type (by choosing Home ⇒ Map ⇒ Flat Map), show a different visualization by configuring a layer, and apply different effects using the options in the Scene Options dialog box.

Once you've added and configured each scene, choose Home ⇒ Tour ⇒ Play Tour to play the entire tour or Home ⇒ Tour ⇒ Create Video to create a video with an optional soundtrack.

Video creation uses a lot of processor power and memory, so you may get an *Out of Memory* error. If this occurs, try creating a lower-resolution video.

Discussion

This recipe gives an overview of how to create a video of a 3-D map or tour. Try experimenting with the available options to see what you can achieve.

What-If Analysis

What-if analysis involves trying different values to examine their effect. For example, you may want to determine how changing an interest rate affects loan repayments or find the most efficient way to schedule your workforce.

The recipes in this chapter cover four kinds of what-if analysis tools available in Excel: Data Tables, Scenarios, Goal Seek, and Solver. The areas covered include:

- Producing results tables showing the impact of one or two variables assuming different values
- Saving sets of values as scenarios and switching between them
- Finding the cell values needed to meet goals and return specific results
- Solving optimization problems, such as maximizing profit given constraints, workforce scheduling, task allocation, and route planning

14.1 Creating a One-Variable Data Table

Problem

You want to create a table showing how substituting different values in a single cell changes the results of one or more formulas.

Solution

Suppose you have a worksheet that calculates the amount you would need to pay each month if you took out a fixed-rate loan, where cell B1 contains the loan amount, B2 contains the loan term in months, B3 contains the annual rate, and cell B4 uses the

PMT function (see Recipe 10.1) to calculate the monthly payment. You want to see how changing the loan term affects the monthly payment and total amount to pay.

You can solve this problem by creating a *one-variable data table*: a range of cells that show how substituting different values in a single cell changes the results of one or more formulas. So, in this example, you can create a one-variable data table that compares the monthly payments and the total amount to pay for different loan terms.

To create the data table:

1. Identify the *input cell*: the cell you want to substitute different values for and see their effect. In this example, the input cell is B2 (the loan term in months).

2. Type the values you want to use for the input cell in a single column, leaving a few empty rows and columns on either side. In this example, you'd type the numbers **12**, **24**, **36**, and **48** in the range E3:E6.

3. Type the formula you want to see the results for in the cell one row above and one column to the right of the values you entered in step 2. So, to see the results for the monthly payment formula, you'd type **=B4** in cell F2.

4. If you want to see the results of other formulas, type them in the cells to the right of the first formula. So, to additionally see the total amount paid for each loan term, you'd type **=B4*B2** in cell G2.

5. Select the range of cells containing the input values and formulas—in this example, E2:G6. Then, choose Data ⇒ Forecast ⇒ What-If Analysis ⇒ Data Table to open the Data Table dialog box.

6. Since the values for the input cell are in a column, put a cell reference to the input cell in the Column Input Cell box. So, in this example, you'd type **$B$2** in the Column Input Cell box.

7. Click OK to insert the data table.

8. Optionally, add headings and format the data table to clarify its purpose.

Figure 14-1 shows the completed data table.

	A	B	C	D	E	F	G
1	Loan amount:	200000			**Loan Term**	**Monthly Payment**	**Total**
2	Loan term (months):	48			**(Current)**	-$4,515.81	-$216,758.92
3	Annual rate:	4%			**12**	-$17,029.98	-$204,359.77
4	Monthly payment:	-$4,515.81			**24**	-$8,684.98	-$208,439.63
5					**36**	-$5,904.80	-$212,572.69
6					**48**	-$4,515.81	-$216,758.92

Figure 14-1. A column-oriented one-variable data table

The input cell and data table must be in the same worksheet, so you can't add a data table to a different worksheet than the input cell.

Discussion

This recipe shows how to create a one-variable data table, displaying how formula results change when substituting different values in a single cell.

While you can achieve similar results using regular cell formulas, using a data table includes the following benefits:

- Excel generates the results for you, so you don't have to copy formulas to other cells or worry about using relative or absolute references (see Recipe 2.1).

- It's harder to accidentally modify a data table's results because, behind the scenes, Excel creates data tables using the TABLE array function: a function you can't manually edit and that prevents you from modifying part of the data table. You can, however, edit data tables using Recipe 14.4.

- You have more control over whether a data table recalculates automatically (see Recipe 3.17).

See Also

This recipe creates a column-oriented data table with the input cell values arranged in a column. To create a row-oriented data table, see Recipe 14.2.

To create a data table with two input cells, see Recipe 14.3.

To use more than two input cells, consider using scenarios; see Recipe 14.5.

14.2 Creating a Row-Oriented One-Variable Data Table

Problem

You want to create a one-variable data table by arranging the input cell values in a row.

Solution

Suppose you want to create a similar one-variable data table to the one created in Recipe 14.1 but with its rows and columns transposed. You can do so by creating a row-oriented one-variable data table: a one-variable data table whose input cell values are listed in a row instead of a column.

To create the data table:

1. Identify the input cell, which, in this example, is B2 (the loan term in months).

2. Type the values you want to use for the input cell in a single row, leaving a few empty rows and columns on either side. In this example, you'd type the numbers **12**, **24**, **36**, and **48** in the range F2:I2.

3. Type the formula you want to see the results for in the cell one row below and one column to the left of the values you entered in step 2. So, to see the results for the monthly payment formula, you'd type **=B4** in cell E3.

4. If you want to see the results of any other formulas, type them in the cells below the first formula. So, to additionally see the total amount paid for each loan term, you'd type **=B4*B2** in cell E4.

5. Select the range of cells containing the input values and formulas—in this example, E2:I4. Then, choose Data ⇒ Forecast ⇒ What-If Analysis ⇒ Data Table to open the Data Table dialog box.

6. Since the values for the input cell are arranged in a row, put a cell reference to the input cell in the Row Input Cell box. So, in this example, you'd type **$B$2** in the Row Input Cell box.

7. Click OK to insert the data table.

8. Optionally, add headings and format the data table to clarify its purpose.

Figure 14-2 shows the completed data table.

	A	B	C	D	E	F	G	H	I
1	Loan amount:	200000							
2	Loan term (months):	48		**Loan Term**	**(Current)**	**12**	**24**	**36**	**48**
3	Annual rate:	4%		**Monthly Payment**	-$4,515.81	-$17,029.98	-$8,684.98	-$5,904.80	-$4,515.81
4	Monthly payment:	-$4,515.81		**Total**	-$216,758.92	-$204,359.77	-$208,439.63	-$212,572.69	-$216,758.92

Figure 14-2. A row-oriented one-variable data table

Discussion

This recipe is similar to Recipe 14.1 except that it transposes the data table's rows and columns.

14.3 Creating a Two-Variable Data Table

Problem

You want to create a table showing how substituting different values in two cells changes the results of a single formula.

Solution

Suppose you have a worksheet that calculates the amount you would need to pay each month if you took out a fixed-rate loan, where cell B1 contains the loan amount, B2 contains the loan term in months, B3 contains the annual rate, and cell B4 uses the PMT function (see Recipe 10.1) to calculate the monthly payment. You want to see how changing the loan term and annual rate affects the monthly payment.

You can solve this problem by creating a *two-variable data table*: a range of cells that show how substituting different values in two cells changes the results of a single formula. So, in this example, you can create a two-variable data table that compares the monthly payments for different loan terms and annual rates.

To create the data table:

1. Identify the two input cells: the cells you want to substitute different values for and see their effect. In this example, the input cells are B2 (the loan term in months) and B3 (the annual rate).

2. Type the values you want to use for the first input cell in a single column, leaving a few empty rows and columns on either side. In this example, you'd type the numbers **12**, **24**, **36**, and **48** in the range E4:E7.

3. Type the values you want to use for the second input cell in a single row, starting from the cell one row above and one column to the right of the values you entered in step 2. In this example, you'd type the values **2%**, **4%**, **6%**, and **8%** in the range F3:I3.

4. Type the formula you want to see the results for in the data table area's top-left corner—the cell one row above the values you entered in step 2 and one column to the left of the values you entered in step 3. So, in this example, you'd type **=B4** in cell E3.

5. Select the range of cells containing the input values and formula—in this example, E3:I7. Then, choose Data ⇒ Forecast ⇒ What-If Analysis ⇒ Data Table to open the Data Table dialog box.

6. In the Row Input Cell box, put a reference to the input cell whose values you listed in a row in step 3. In this example, you'd type **$B$3** in the Row Input Cell box.

7. In the Column Input Cell box, put a reference to the input cell whose values you listed in a column in step 2. In this example, you'd type **$B$2** in the Column Input Cell box.

8. Click OK to insert the data table.

9. Optionally, add headings and format the data table to clarify its purpose.

Figure 14-3 shows the completed data table.

	A	B	C	D	E	F	G	H	I
1	Loan amount:	200000					**Monthly Payments**		
2	Loan term (months):	48					**Annual Rate**		
3	Annual rate:	4%			-$4,515.81	**2%**	**4%**	**6%**	**8%**
4	Monthly payment:	-$4,515.81			12	-$16,847.77	-$17,029.98	-$17,213.29	-$17,397.69
5					24	-$8,508.05	-$8,684.98	-$8,864.12	-$9,045.46
6					36	-$5,728.52	-$5,904.80	-$6,084.39	-$6,267.27
7					48	-$4,339.02	-$4,515.81	-$4,697.01	-$4,882.58

Figure 14-3. A two-variable data table

Discussion

This recipe shows how to create a two-variable data table: a data table with two input cells that shows how their different values change the results of a single formula. A two-variable data table is like a one-variable data table (see Recipe 14.1), except that it has two input cells instead of one and can include only one formula.

If you want to see the impact of two input cells on multiple formulas or use more than two input cells, consider using scenarios (see Recipe 4.5).

14.4 Editing Data Tables

Problem

You have a one- or two-variable data table and want to add, remove, or edit its input values or formulas.

Solution

Suppose you have an existing data table and want it to show the results for different input values or formulas. Your approach depends on whether you want to keep the data table the same size or add or remove rows or columns.

If you want to keep the data table the same size and use the same cells for its input cells and formulas, you can update them with new values or formulas. The data table will automatically update if you've set Excel's calculation options to Automatic; if not, you can use Recipe 3.17 to manually recalculate its formulas.

If you want to add rows and/or columns to the data table so it calculates the results for new input values and/or formulas, you can do so as follows:

1. Add the new input values and/or formulas to the data table. So, if you have a column-oriented one-variable data table with input values in cells E3:E6, you'd list any new input values in cells E7, E8, and so on.

2. Select the range of cells containing the input values and formulas (including the ones you added in step 1) and then choose Data ⇒ Forecast ⇒ What-If Analysis ⇒ Data Table to open the Data Table dialog box.

3. Complete the Row Input Cell and/or Column Input Cell boxes, depending on the type of data table; then click OK to update the data table.

If you want to remove rows and/or columns from the data table so it calculates the results for fewer input values and/or formulas, you can do so by following these steps:

1. Select the data table's result cells—the ones using the `TABLE` function—and press the Delete key to delete their values.

2. Delete any input values and/or formulas you no longer want to include.

3. Select the new range of cells containing the input values and formulas and then choose Data ⇒ Forecast ⇒ What-If Analysis ⇒ Data Table to open the Data Table dialog box.

4. Complete the Row Input Cell and/or Column Input Cell boxes, depending on the type of data table; then click OK to update the data table.

> If you omit step 1, Excel displays an error message when you try to open the Data Table dialog box in step 3. Ensure that you first delete the data table's calculated values to avoid this error.

Discussion

This recipe gives an overview of how to edit a data table's input values and/or formulas to display different results. Unless you want it to stay the same size and use the same cells for its input cells and formulas, you generally need to re-create the data table since you can't edit part of a data table's results.

14.5 Using Scenario Manager

Problem

You want to explore different scenarios by saving groups of values and substituting them in a worksheet.

Solution

Suppose you have a worksheet that calculates the profit of completing a piece of work, where cell B1 contains the hourly cost, B2 contains the number of hours, B3

contains the amount charged, and B4 calculates the profit using the formula =B3-B1*B2. You want to see how using different sets of values for the hourly cost, number of hours, and amount charged affects the profit.

You can solve this problem by saving each set of values as a *scenario*: a group of values you can substitute into up to 32 worksheet cells to see their impact. So, in this example, you could save one set of values as a best-case scenario, another as a worst-case scenario, and so on.

You define scenarios using Excel's Scenario Manager as follows (see Figure 14-4):

1. Choose Data ⇒ Forecast ⇒ What-If Analysis ⇒ Scenario Manager to open the Scenario Manager dialog box. Then click Add to add a new scenario.

2. In the Scenario Name box, type a name for the scenario (for example, **Best Case**).

3. Select the cells you want to save values for in the Changing Cells box. So, in this example, you select the cells B1:B3 to save values for the hourly cost, number of hours, and amount charged. Then click OK.

4. Enter a value for each cell you selected in step 3. In this example, type **30** in the box for B1, **25** in the box for B2, and **2000** in the box for B3. Then click OK to save the scenario; you can also click Add to add another scenario or Cancel to cancel your changes.

5. When you click OK, Excel displays the saved scenario in Scenario Manager.
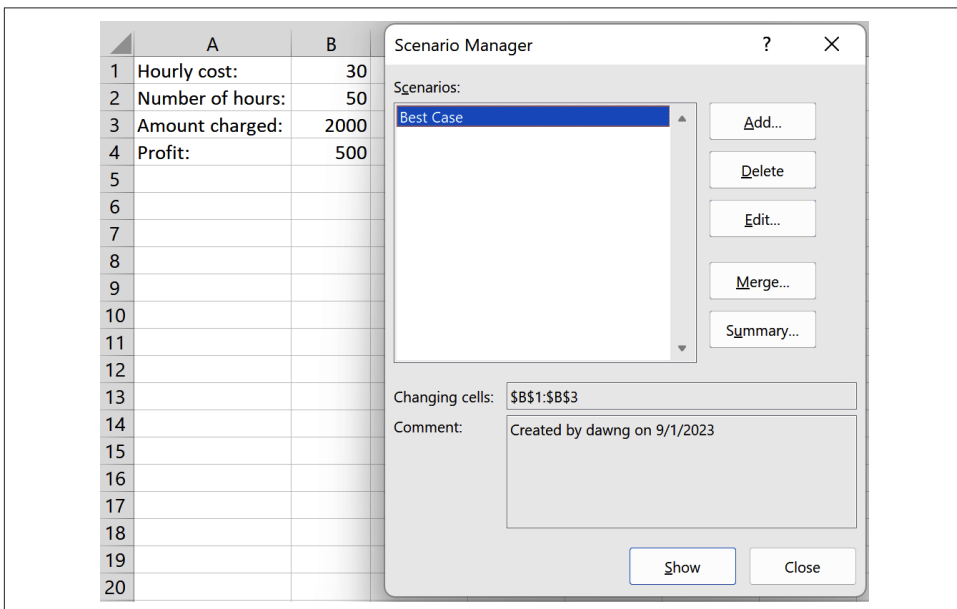
Figure 14-4. The Scenario Manager dialog box

If you've protected the worksheet (see Recipe 1.8), you can use the Prevent Changes check box to stop the scenario from being edited and the Hidden check box to stop it from being displayed. If the worksheet is unprotected, these options aren't applied.

Once you've added a scenario, you can substitute its values in the worksheet by choosing Data ⇒ Forecast ⇒ What-If Analysis ⇒ Scenario Manager to open the Scenario Manager dialog box, selecting the scenario, then clicking Show. You can also click Edit to edit the scenario's values or Delete to remove them.

Showing a scenario substitutes its values in the worksheet and replaces any values already displayed. If you want to return to the worksheet's original values, save them in a separate scenario.

Excel saves each scenario against a specific worksheet, so you can see only those scenarios when you open Scenario Manager. However, you can add scenarios from other open workbooks using Recipe 14.6.

Discussion

This recipe shows how to add scenarios to Scenario Manager and display them in the worksheet. You'll typically want to create multiple scenarios, and there's no limit to how many you can save.

Scenarios are helpful when you want to see the impact of changing cell values in a worksheet and switch between different sets of values. Similarly to data tables (see Recipes 14.1 and 14.3), you specify which cells can change and the values you want to substitute for them. However, a scenario can save values for up to 32 changing cells (instead of 1 or 2) and you can see their effect on any formula.

See Also

To copy one worksheet's scenarios to another, see Recipe 14.6. You can also use Recipe 14.7 to generate a summary report or PivotTable of the worksheet's scenarios.

14.6 Merging Scenarios

Problem

You want to copy scenarios from one worksheet to another.

Solution

Suppose you have a worksheet that uses scenarios (see Recipe 14.5) and you want to copy the scenarios to another worksheet. You can do so by merging the scenarios as follows:

1. Open the workbook you want to copy scenarios from.

2. Go to the worksheet to which you want to copy the scenarios and choose Data ⇒ Forecast ⇒ What-If Analysis ⇒ Scenario Manager to open the Scenario Manager dialog box.

3. Click Merge, select the workbook and worksheet from which you want to copy scenarios, and then click OK. Excel copies all the scenarios to the current worksheet, including each one's name, changing cell values, and who created it.

> When you merge scenarios, ensure that both worksheets have the same cell structure and use any changing cells for the same purpose. For example, if one worksheet uses B1 for the hourly cost, B2 for the number of hours, and B3 for the amount charged, the other worksheet should as well.

Discussion

This recipe shows how to merge scenarios from other worksheets and workbooks quickly. This is handy, for example, if you want to collect and merge scenarios from other team members or departments.

See Also

If you want to see who created each scenario, generate a PivotTable summary; see Recipe 14.7.

14.7 Generating Scenario Summaries

Problem

You want to generate a report of a worksheet's scenarios, such as each one's changing cell values, results, and who created it.

Solution

Scenario Manager (see Recipe 14.5) can generate two types of reports: a scenario summary that generates an outline (see Recipe 2.17) showing each scenario's

changing cell values and results and a PivotTable (see Chapter 11) that lets you filter by who created each scenario. You can generate either report as follows:

1. Open Scenario Manager by choosing Data ⇒ Forecast ⇒ What-If Analysis ⇒ Scenario Manager.

2. Click Summary to open the Scenario Summary dialog box.

3. Select a report type. Choose the "Scenario summary" option to compare changing cell values and results and "Scenario PivotTable report" to create a PivotTable showing each scenario's results and filter by who created it.

4. In the Result Cells box, select any formula cells whose results you want to see in the summary. Then click OK.

Figure 14-5 shows the report Scenario Manager generates when you choose the "Scenario summary" option.

Scenario Summary				
	Current Values:	Best Case	Worst Case	Expected
Changing Cells:				
$B$1	30	30	45	35
$B$2	25	25	50	30
$B$3	2000	2000	2000	2000
Result Cells:				
$B$4	1250	1250	-250	950

Notes: Current Values column represents values of changing cells at time Scenario Summary Report was created. Changing cells for each scenario are highlighted in gray.

Figure 14-5. Scenario summary report generated by Scenario Manager

You can improve the reports Scenario Manager generates by naming any cells you want to display (see Recipe 2.6). When you do so, Scenario Manager uses the cell's name in the report instead of its cell reference.

Discussion

This recipe shows how to use Scenario Manager to generate scenario reports. Generally, the scenario summary report outputs results similar to using a data table, while the PivotTable report is handy if you want to filter by who created each scenario.

Because the scenario summary report is an outline, you can use the - and + buttons on the left to show or hide its details. For example, you can hide the changing cell values.

14.8 Using Goal Seek

Problem

You want to determine the value a single input cell should be to return a specific result in a dependent formula cell.

Solution

Suppose you have a worksheet that calculates the profit for completing a piece of work, where cell B1 is the hourly cost, B2 is the number of hours it takes, B3 is the amount charged, and B4 calculates the profit using the formula =B3-B1*B2. If the hourly cost is $30 and you charge $3,000, you want to know the hours needed to break even.

You can solve this problem using *Goal Seek*: a tool that calculates the value you need to enter in an input cell to produce a specific result in a dependent formula cell. So, in this example, you can use it to find the value B2 (the input cell) needs to be for B4 (the dependent formula cell) to be 0.

To use Goal Seek, follow these steps (see Figure 14-6):

1. Choose Data ⇒ Forecast ⇒ What-If Analysis ⇒ Goal Seek to open the Goal Seek dialog box.

2. Put a reference to the formula cell—in this example, B4—in the "Set cell" box.

3. Type the value you want the formula cell to be—in this example, 0—in the "To value" box.

4. Put a reference to the input cell—in this example, B2—in the "By changing cell" box. Then click OK.
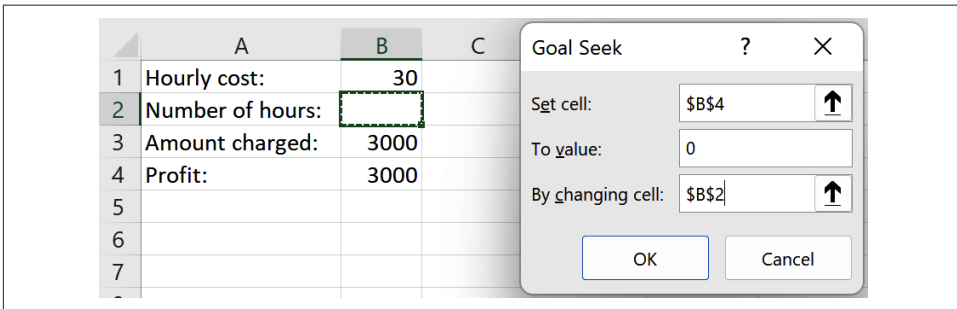
Figure 14-6. Using Goal Seek to find the number of hours where you break even

When you click OK, Goal Seek uses iteration to search for a solution, starting from the current value of the input cell, and opens the Goal Seek Status dialog box to

inform you of its progress. When it's reached the end of its search, the dialog box tells you whether it has found a solution.

When Goal Seek finds a solution, it updates the value of the input cell specified in step 4 and displays the target and current value of the formula cell in the Goal Seek Status dialog box. So, in this example, Goal Seek puts 100 in cell B2 for the number of hours because this value means cell B4 (the profit) is 0 (see Figure 14-7).
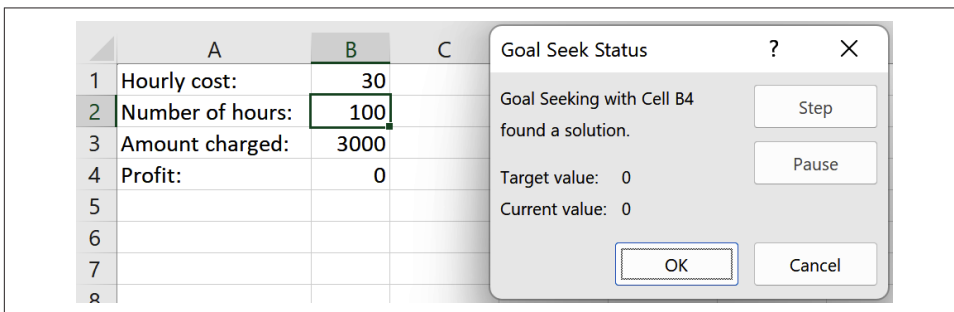
Figure 14-7. The solution found by Goal Seek

Click OK to accept the input cell's new value or Cancel to revert to its original value.

Depending on the formula, Goal Seek may return an approximate solution that you want to be more precise. You can address this by changing Goal Seek's precision. If you're using Excel for Windows, choose File ⇒ Options ⇒ Formulas and change the number in the Maximum Change box; the smaller the number, the more precise the result. If you're using Excel for Mac, choose Excel ⇒ Preferences ⇒ Calculation.

If Goal Seek can't find a solution, it displays the closest it can find in the Goal Seek Status dialog box. If you're sure a solution exists, try one of the following:

- Change the value in the input cell. Since Goal Seek starts iterating from the input cell value, changing the cell's value changes where it starts iterating from. This can help if the formula in the formula cell is discontinuous or displays an error value such as #DIV/0! for a specific input cell value (see Recipe 14.10).

- Increase the maximum number of iterations. If you're using Excel for Windows, you can make Goal Seek search for longer by choosing File ⇒ Options ⇒ Formulas and increasing the number in the Maximum Iterations box; if you're using Excel for Mac, you can find this option by choosing Excel ⇒ Preferences ⇒ Calculation.

- Resolve any circular references (see Recipe 3.19). Goal Seek may not function properly if any circular references involve the formula cell (or the cells on which it depends).

Discussion

This recipe shows how to use Goal Seek to help meet your goals and reach a desired outcome.

In some respects, Goal Seek is the polar opposite of using a one-input data table; instead of displaying the results for a set of input cell values, it figures out the value an input cell should be to get a specific result.

See Also

Goal Seek can return only a single solution at a time. If you have a formula that may have multiple solutions, see Recipe 14.9.

Goal Seek can find only exact matches for a single input cell. If you want to minimize or maximize a formula cell or have multiple input cells, try using Solver instead; see Recipe 14.12.

14.9 Finding Multiple Solutions with Goal Seek

Problem

You want to use Goal Seek to find multiple solutions to a problem.

Solution

Suppose cell B1 contains the formula =A1^2 and you want to determine the values A1 should be for B1 to return 49.

This problem has two solutions: 7 and –7. You can use Goal Seek to find each one by putting different initial values in cell A1, then following the steps in Recipe 14.8. Goal Seek returns the solution closest to the initial value, so putting 10 in cell A1 returns a solution that's approximately 7, and putting –10 in A1 returns one that's approximately –7 (see Figure 14-8).
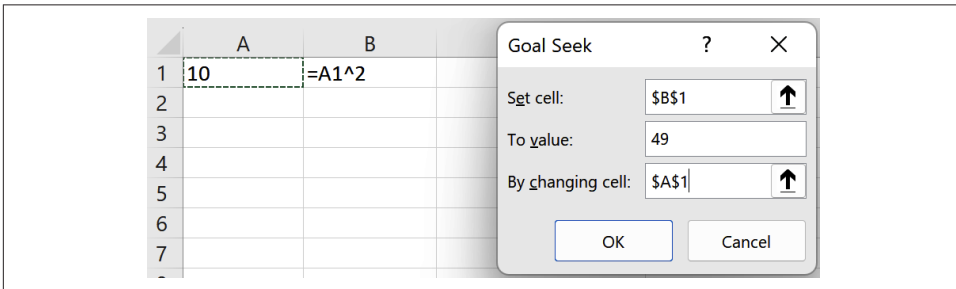
Figure 14-8. Trying an initial value of 10 in the input cell to find the first solution

Generally, you can find multiple solutions to a problem by trying different initial values with Goal Seek.

Discussion

Goal Seek can return only one solution at a time. However, changing the input cell's initial value can drive out multiple solutions. The values you try depend on the specific formula you want to solve.

14.10 Handling Discontinuous Formulas with Goal Seek

Problem

You want to use Goal Seek to find a solution for a discontinuous formula.

Solution

Suppose cell B1 contains the formula =1/A1, and you want to determine the value of A1 that returns 0.25 and the one that returns –0.25. This formula is discontinuous when A1 is 0 since Excel can't evaluate =1/0 and returns a #DIV/0! error value (see Figure 14-9).
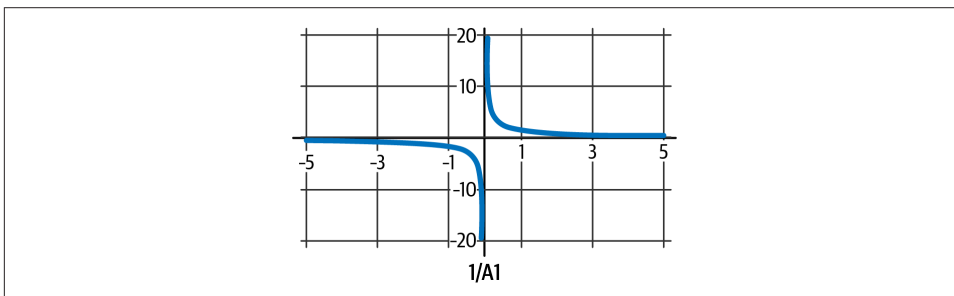
Figure 14-9. Chart showing 1/A1, which is discontinuous when A1 is 0

When you have a formula with one or more discontinuities, finding solutions with Goal Seek can be tricky because it can't generally cross discontinuities. However, you can get around this by trying different initial values on either side of each discontinuity, similarly to Recipe 14.9. For example, if you try an initial value of 10 in cell A1, Goal Seek can find the value of A1 for which B1 is 0.25 (the solution is 4) but won't find the value for when B1 is –0.25. Similarly, if you try an initial value of –10 in cell A1, Goal Seek will be able to find the value of A1 for which B1 is –0.25 (the solution is –4) but won't find the value for when B1 is 0.25 (see Figure 14-10).
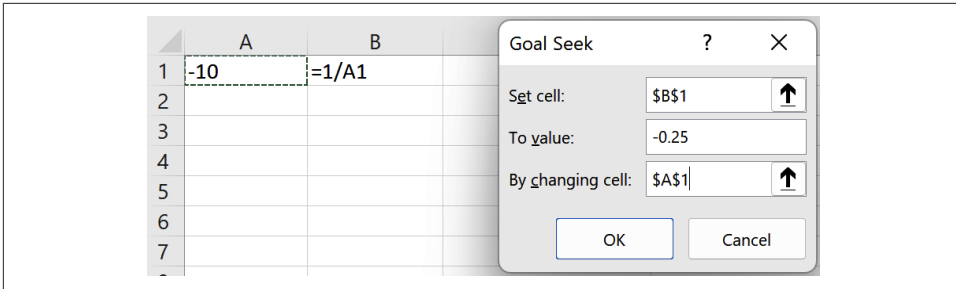
Figure 14-10. Trying an initial value of –10 to solve when B1 is –0.25

> If you try using an initial value for the input cell—in this example, 0—which results in an error value, Goal Seek returns the error message "Formula in cell must result in a number" instead of searching for a solution. You can correct this by changing the value of the input cell so the formula cell returns a number.

Discussion

Finding solutions to formulas with discontinuities can be tricky because Goal Seek can't evaluate or cross these discontinuities. This recipe shows how to handle this problem by trying different initial values on either side of each discontinuity.

14.11 Enabling Solver

Problem

You have a problem you want to solve that's more complex than Goal Seek can handle, and you want to install a tool to help you solve it.

Solution

The Solver add-in lets you solve complex problems. Once it's enabled, you can access Solver from the Data menu by going to the Analyze group and choosing Solver.

To enable Solver in Excel for Windows:

1. Choose File ⇒ Options.

2. Select Add-ins.

3. In the Manage box at the bottom of the screen, choose the Excel Add-ins option and click Go.

4. In the Add-ins dialog box, check the Solver Add-In check box and click OK.

To load Solver in Excel for Mac, choose Tools ⇒ Excel add-ins to open the Add-ins dialog box, place a check in the Solver Add-In check box, and click OK.

Discussion

Solver is a powerful optimization tool that can solve more complex problems than Goal Seek can handle, such as ones with multiple input cells or where you want to minimize or maximize the formula cell subject to specified constraints. See Recipe 14.12 and onward for more details on how to use Solver in specific situations.

14.12 Solving an Optimization Problem with Solver

Problem

You want to know how to solve an optimization problem with Solver.

Solution

Suppose you manufacture three products, each taking a specified number of labor hours to produce and return a specific profit. You want to determine how many units to manufacture of each product to maximize your profit when you have limited labor, need to meet the demand for each product, and can produce only a specific total number of units.

To solve this problem with Solver, you first need to model the problem in a worksheet, as shown in Figure 14-11. In this worksheet, A2:A4 shows the name of each product, B2:B4 the labor required per unit, C2:C4 the profit per unit, and D2:D4 the minimum number you need to produce for each product. E2:E4 shows the number of units you need to manufacture for each product (which Solver will evaluate), F2:F4 calculates the total labor required, G2:G4 calculates the total profit, and Row 5 calculates the sum of the E, F, and G columns. Finally, E8 shows the maximum number of units you can produce, and F8 shows the maximum labor.

	A	B	C	D	E	F	G
1	Product	Labor per Unit	Profit per Unit	Min. Units	No. Units	Total Labor	Total Profit
2	A	6	20	6		=B2*E2	=C2*E2
3	B	7	25	4		=B3*E3	=C3*E3
4	C	4	15	10		=B4*E4	=C4*E4
5					Total: =SUM(E2:E4)	=SUM(F2:F4)	=SUM(G2:G4)
6							
7					Max Units	Max Labor	
8					40	200	

Figure 14-11. The model used to determine how many units to manufacture

The model includes three components, which Solver will use to solve the problem:

An objective or target cell you want to minimize, maximize, or set to a value
In this example, you want to maximize the total profit in cell G5.

One or more input or changing cells
In this example, these are E2:E4—the number of units.

Constraints
In this example, the number of units (E2:E4) must be greater than or equal to the minimum number of units to manufacture (D2:D4), the total units (E5) must be less than or equal to the maximum number of units (E8), and the total labor (F5) must be less than or equal to the maximum labor (F8).

Once you've modeled the problem, you can solve it with Solver by following these steps:

1. Choose Data ⇒ Analyze ⇒ Solver to open the Solver Parameters dialog box; if there's no Solver option, see Recipe 14.11 for how to load it.

2. In the Set Objective box, select the objective cell to minimize, maximize, or set to a value. So, in this example, you must choose cell G5 (the total profit).

3. Choose a To option—Max, Min, or Value Of—from the available options, depending on your objective in step 2. In this example, you must choose the Max option to maximize the total profit (cell G5).

4. In the By Changing Variable Cells box, select the input or changing cells for which you want Solver to find values. In this example, you need to choose E2:E4.

5. Next, you must add constraints to the Subject to the Constraints box. To add the first constraint, click Add to open the Add Constraint dialog box, choose E2:E4 in the Cell Reference box, select >= from the drop-down list, then choose D2:D4 in the Constraint box. Then click Add to add the next constraint.

6. To add the second constraint, choose E5 in the Cell Reference box, select <= from the drop-down list, and choose E8 in the Constraint box. Then click Add to add the next constraint.

7. To add the final constraint, choose F5 in the Cell Reference box, select <= from the drop-down list, and choose F8 in the Constraint box. Then click OK to close the Add Constraint dialog box and return to the Solver Parameters dialog box.

8. Place a check in the Make Unconstrained Variables Non-Negative check box; this ensures that all the changing cells—the range you entered in step 4—will be greater than or equal to 0.

9. Select Simplex LP from the Select a Solving Method drop-down list, which you use to solve linear problems (see "Discussion" on page 377).

Figure 14-12 shows the completed options in the Solver Parameters dialog box.
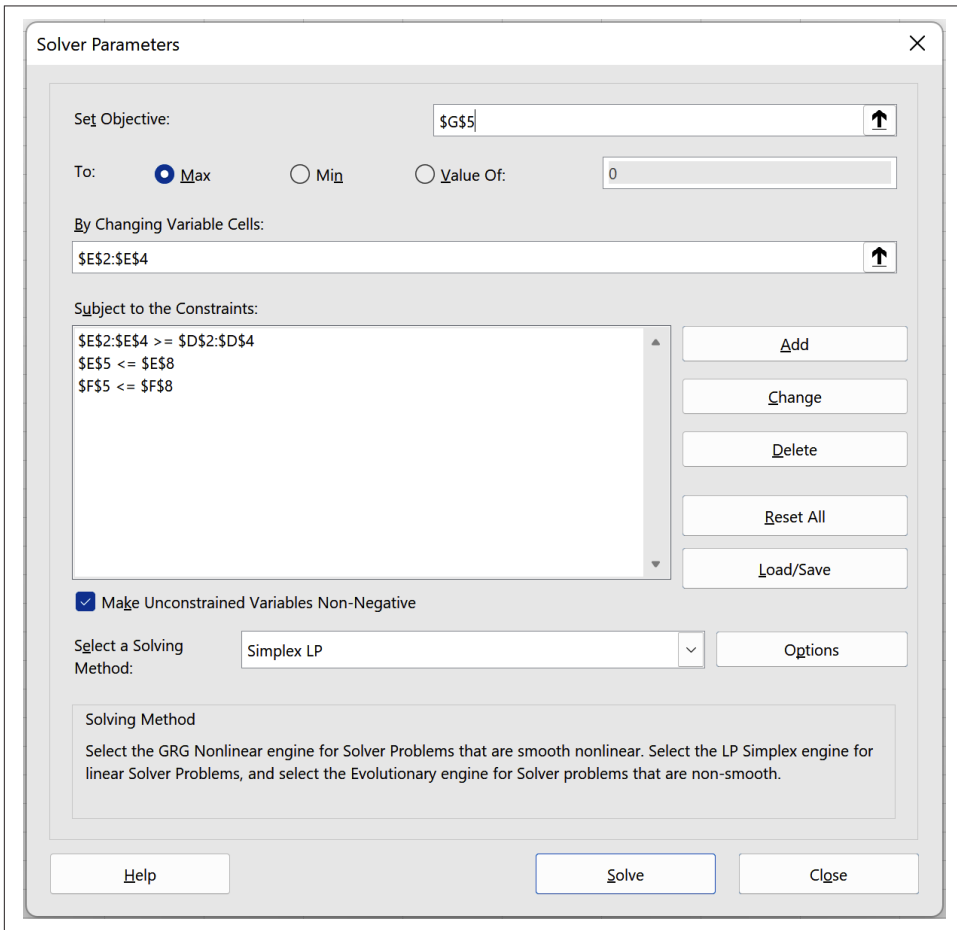
Figure 14-12. The options needed in the Solver Parameters dialog box to solve the optimization problem

Once you've selected the appropriate options in the Solver Parameters dialog box, click Solve to solve the problem. Solver searches for a solution, and if it finds one, updates the changing cells in the worksheet (see Figure 14-13).

> In this example, Solver returns decimal numbers in the changing cells. If needed, you can restrict their values to integers by adding an extra constraint; see Recipe 14.13 for an example.

	A	B	C	D	E	F	G
		Labor	**Profit**	**Min.**		**Total**	**Total**
1	**Product**	**per Unit**	**per Unit**	**Units**	**No. Units**	**Labor**	**Profit**
2	A	6	20	6	6	36	120
3	B	7	25	4	9.3333333	65.333	233.3
4	C	4	15	10	24.666667	98.667	370
5				**Total:**	40	200	723.3
6							
7					**Max Units**	**Max Labor**	
8					40	200	

Figure 14-13. In the optimization solution, Solver has updated cells E2:E4, so the maximum total profit meeting the constraints is 723.3

Solver also displays a Solver Results dialog box, indicating whether it found a feasible solution. To accept the solution and keep the updated values, choose the Keep Solver Solution and click OK. To discard the solution, click Cancel or choose the Restore Original Values option and click OK (see Figure 14-14).
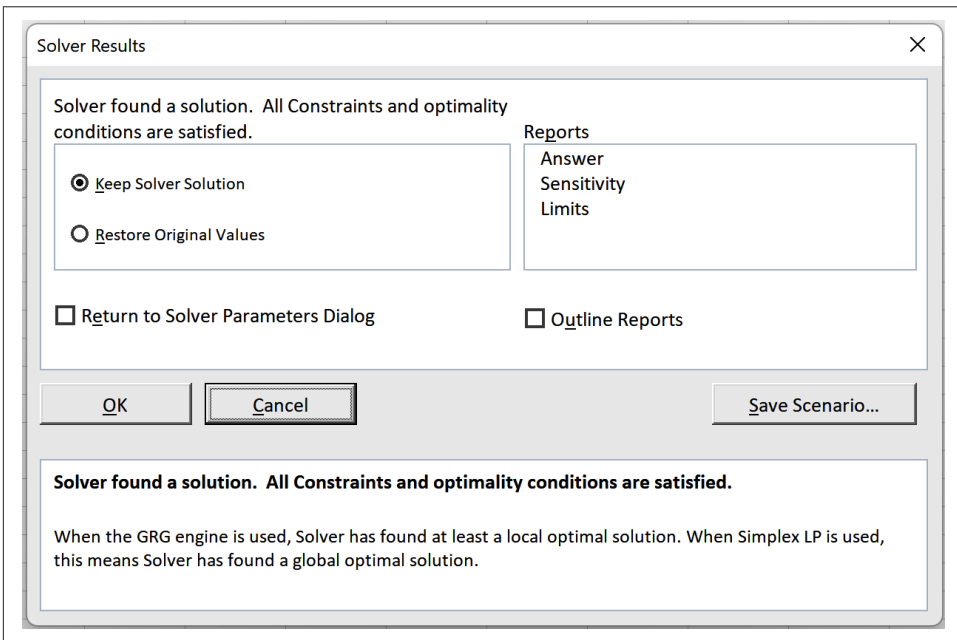
Figure 14-14. The Solver Results dialog box

Discussion

This recipe shows how to use Solver to solve a linear optimization problem where you must maximize a cell value subject to specified constraints. First, you formulate a worksheet model to describe the problem you want to solve, including an objective cell, one or more changing cells, and any constraints you need to apply. You then use Solver to find the optimal values of the changing cells needed to meet your objective.

This example uses the *Simplex LP* solving method, which you use to solve linear optimization problems. In these problems, the objective cell and constraints are calculated by multiplying a changing cell by a constant and summing the results. So, in this example, the objective cell is the total profit, which multiplies each changing cell by the profit per unit and sums the result. Similarly, the total labor constraint refers to a cell that multiplies each changing cell by the labor per unit and sums the result. See Recipes 14.13 and 14.14 for other examples of using this method to solve a problem.

If you want to solve a problem where the objective cell or a constraint is not linear, use the *GRG Nonlinear* solving method. Choose this option to solve problems involving everyday mathematical operations, such as multiplying changing cells together or raising them to a power.

If you have a problem where the objective cell or a constraint can abruptly change—for example, because they use the IF, SUMIF, ABS, or MAX functions—you can use the *Evolutionary* solving method. This is the most flexible solving method because it can solve a wide variety of problems. However, it's much slower than using the Simplex LP or GRG Nonlinear solving methods. See Recipes 14.15 and 14.16 for examples of using this solving method.

Don't worry if you need help deciding which solving method to use. If you choose an inappropriate method for a specific problem, Solver displays a message informing you of this in its Solver Results dialog box. You can then try solving the problem again with a different solving method.

See Also

You can adjust Solver's default options to change the precision, limit the running time, and more; see Recipe 14.19.

14.13 Using Integer-Only Constraints with Solver

Problem

You want to solve a problem with Solver where the changing cells must be integers.

Solution

Suppose your company operates a warehouse that opens seven days a week. All employees work five consecutive days full-time, and you want to determine the smallest number of employees required to maintain the minimum cover each day and their working patterns. You can solve this problem with Solver by using an *integer* constraint: a constraint that forces changing cells to assume integer values.

You first need to formulate a worksheet model that includes the following:

An objective cell
> In this example, you want to determine the minimum number of employees, so you must include a cell that calculates their total.

Changing cells
> You want to find the number of employees with each working pattern—the number who work Monday to Friday, Tuesday to Saturday, Wednesday to Sunday, and so on—so you must include cells to record this.

Constraints
> For each day, the number of employees in the warehouse must be greater than or equal to the minimum cover for that day. Furthermore, the number of employees with each working pattern must be integers because all employees work full-time.

Figure 14-15 shows a model for this problem, where B3:B9 lists the start day of each possible working pattern, C3:I9 specifies the days on which an employee with each pattern works (1 means they work that day, 0 means they don't), and C10:I10 specifies the minimum cover required for each day. L3:L9 specifies how many employees have each working pattern (the changing cells, which must all have integer values), L10 calculates the total number of employees (the objective cell to be minimized), and C11:I11 uses the SUMPRODUCT function to calculate the total cover for each day (to be used as a constraint).

	A	B	C	D	E	F	G	H	I	J	K	L
1			Working Pattern									
2			Mon	Tues	Wed	Thurs	Fri	Sat	Sun			Employees
3		Mon	1	1	1	1	1	0	0			0
4		Tues	0	1	1	1	1	1	0			0
5	Start Day	Wed	0	0	1	1	1	1	1			0
6		Thurs	1	0	0	1	1	1	1			0
7		Fri	1	1	0	0	1	1	1			0
8		Sat	1	1	1	0	0	1	1			0
9		Sun	1	1	1	1	0	0	1			0
10		Min Cover:	20	21	18	20	25	15	10		Total:	=SUM(L3:L9)
11		Total Cover:	=SUMPRODUCT($L$3:$L$9,C3:C9)	=SUN	=SUN	=SUMP	=SU	=SUI	=SUN			

Figure 14-15. The model used to schedule the workforce

Once you've formulated the model, you can solve the problem by following these steps in Solver:

1. Choose Data ⇒ Analyze ⇒ Solver to open the Solver Parameters dialog box.

2. In the Set Objective box, select the objective cell L10 (the total number of employees).

3. Choose a To option of Min.

4. In the By Changing Variable Cells box, choose L3:L9.

5. Click Add to open the Add Constraint dialog box, choose C11:I11 in the Cell Reference box, select >= from the drop-down list, and choose C10:I10 in the Constraint box. Then click Add to add the next constraint.

6. For the second constraint, choose L3:L9 in the Cell Reference box, and select int from the drop-down list; this specifies that L3:L9 must be integers and automatically puts "integer" in the Constraint box. Then click OK to close the Add Constraint dialog box and return to the Solver Parameters dialog box.

7. Place a check in the Make Unconstrained Variables Non-Negative check box to ensure that all the changing cells will be greater than or equal to 0.

8. Select Simplex LP from the Select a Solving Method drop-down list.

9. Click the Options button, uncheck the Ignore Integer Constraints check box in the All Methods tab (see Recipe 14.19), then click OK.

Figure 14-16 shows the completed options in the Solver Parameters dialog box.
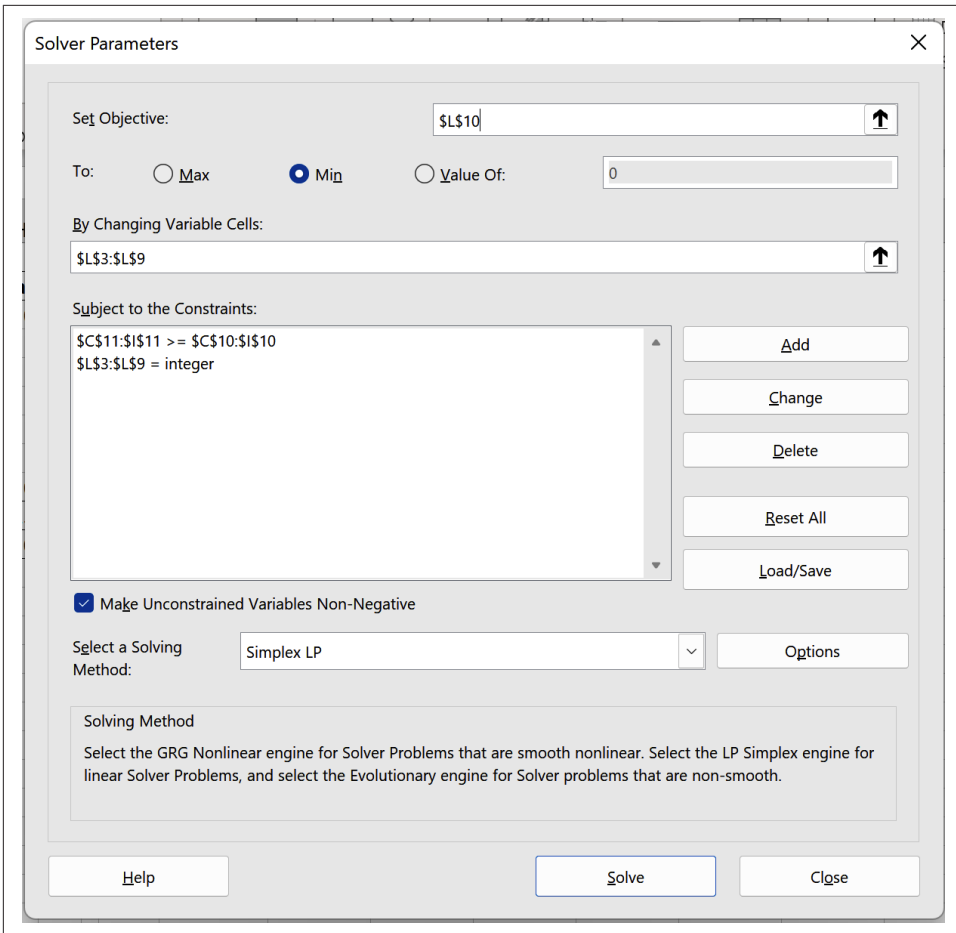
Figure 14-16. The options needed in the Solver Parameters dialog box to solve the work-force scheduling problem

Once you've selected the appropriate options in the Solver Parameters dialog box, click Solve to solve the problem. Solver searches for a solution, then updates the changing cells in the worksheet, as shown in Figure 14-17.

Figure 14-17. In the workforce scheduling solution, Solver has updated cells L3:L9, so the minimum number of employees is 26

Discussion

This recipe shows how to restrict Solver's changing cells to integer values by solving a workforce scheduling problem. The crucial part of this recipe is step 6, which adds a constraint limiting the values of the changing cells.

See Also

See Recipe 14.14 if you want to limit cell values to 0 or 1, and see Recipe 14.15 if you want the cell values to be all different integers.

14.14 Using Binary-Only Constraints with Solver

Problem

You want to solve a problem with Solver where the changing cells must be 0 or 1.

Solution

Suppose you have three tasks and three employees, and you want to allocate a single task to each employee, and each task requires a single person. Employees perform each task at different rates, and you want to determine which employee to allocate to each task while minimizing the total cost. You can solve this problem using a *binary* constraint: a constraint that forces changing cells to assume binary—0 or 1—values.

You first need to formulate a worksheet model that includes the following:

An objective cell
> The total cost that you want to minimize.

Changing cells
> To record which task to allocate to each employee.

Constraints
> Each employee must have one task, and each task must be allocated to one employee (so a task can be allocated to each employee only one or zero times).

Figure 14-18 shows a model for this problem, where C3:E5 lists the cost of each employee working on a task, I3:K3 specifies which employees are allocated to each task (the changing cells), L3:L5 specifies the number of tasks each employee must have, M3:M5 calculates each employee's total number of tasks, I6:K6 specifies the number of employees needed for each task, and I7:K7 calculates the number of employees allocated to each task. Finally, I9 calculates the total cost—the objective cell to be minimized.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1				Task						Task			
2		Cost	1	2	3			Allocation	1	2	3	Equals	Total
3	Employee	A	15	40	20		Employee	A	0	0	0	1	=SUM(I3:K3)
4		B	30	20	10			B	0	0	0	1	=SUM(I4:K4)
5		C	25	30	25			C	0	0	0	1	=SUM(I5:K5)
6								Equals	1	1	1		
7								Total	=SUM(I3:I5)	=SUM(J3:J5)	=SUM(K3:K5)		
8													
9								Total Cost:	=SUMPRODUCT(C3:E5,I3:K5)				

Figure 14-18. The model used for task allocation

Once you've formulated the model, you can solve the problem by following these steps in Solver:

1. Choose Data ⇒ Analyze ⇒ Solver to open the Solver Parameters dialog box.

2. Select the objective cell I9 (the total cost) in the Set Objective box.

3. Choose a To option of Min.

4. In the By Changing Variable Cells box, choose I3:K5.

5. Click Add to open the Add Constraint dialog box, choose I7:K7 in the Cell Reference box, select = from the drop-down list, and choose I6:K6 in the Constraint box. Then click Add to add the next constraint.

6. For the second constraint, choose M3:M5 in the Cell Reference box, select = from the drop-down list, and choose L3:L5 in the Constraint box. Then click Add to add the next constraint.

7. For the final constraint, choose I3:K5 in the Cell Reference box, and select bin from the drop-down list; this specifies that I3:K5 must be binary, so their values can be only 0 or 1, and automatically puts "binary" in the Constraint box. Then click OK to close the Add Constraint dialog box and return to the Solver Parameters dialog box.

8. Place a check in the Make Unconstrained Variables Non-Negative check box to ensure that all the changing cells will be greater than or equal to 0.

9. Select Simplex LP from the Select a Solving Method drop-down list.

10. Click the Options button, uncheck the Ignore Integer Constraints check box in the All Methods tab (see Recipe 14.19), then click OK.

Figure 14-19 shows the completed options in the Solver Parameters dialog box.
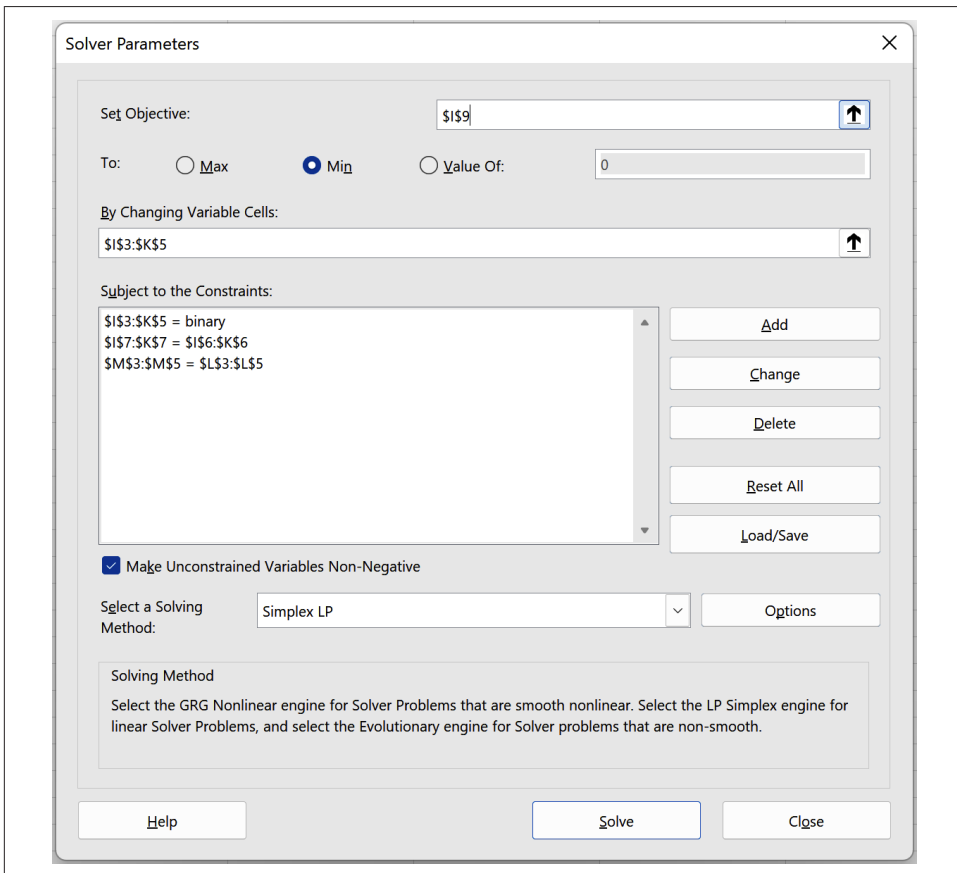
Figure 14-19. The options needed in the Solver Parameters dialog box to solve the task allocation problem

Once you've selected the appropriate options in the Solver Parameters dialog box, click Solve to solve the problem. Solver searches for a solution, then updates the changing cells in the worksheet, as shown in Figure 14-20.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1				Task						Task			
2		Cost	1	2	3			Allocation	1	2	3	Equals	Total
3	Employee	A	15	40	20		Employee	A	1	0	0	1	1
4		B	30	20	10			B	0	0	1	1	1
5		C	25	30	25			C	0	1	0	1	1
6								Equals	1	1	1		
7								Total	1	1	1		
8													
9								Total Cost:	55				

Figure 14-20. In the task allocation solution, Solver has updated cells I3:K5, so the minimum total cost is 55

Discussion

This recipe shows how to restrict Solver's changing cells to binary values by solving a task allocation problem. This works similarly to Recipe 14.14, except the changing cell values are limited to 0 and 1.

14.15 Making Changing Cells All Different with Solver

Problem

You want to solve a problem with Solver where the changing cells must have different integer values.

Solution

Suppose you need to deliver goods to five cities. You want to determine the sequence in which to travel to each city (and return to the first city) so that you minimize the total distance traveled.

This type of problem is known as a *traveling salesperson problem*, and you can solve it with Solver by using an *alldifferent* constraint: a constraint that forces the changing cells to assume unique integer values.

You first need to formulate a worksheet model that includes the following:

An objective cell
    The total distance that you want to minimize

Changing cells

To record the sequence to visit each city, where each cell needs to have a different integer value (the constraint)

Figure 14-21 shows a model for this problem, where A3:A7 numbers each city from 1 to 5, C3:G7 lists the distance between each city, I3:I7 lists the city numbers in the order in which you need to visit them (the changing cells), J3:J7 uses the INDEX function to look up the city name, K3:K7 uses INDEX to look up the distance between the current and previous city, and K8 calculates the total distance—the objective cell you want to minimize.

	A	B	C	D	E	F	G	H	I	J	K
1				Distances Between Cities						City Delivery Order	
2			Albuquerque	St Louis	Minneapolis	Bozeman	Omaha		No	Name	Miles
3	1	Albuquerque	0	940	980	769	716			=INDEX(B3:B7,I3)	=INDEX(C3:G7,I3,I7)
4	2	St Louis	940	0	461	1168	353			=INDEX(B3:B7,I4)	=INDEX(C3:G7,I4,I3)
5	3	Minneapolis	980	461	0	862	293			=INDEX(B3:B7,I5)	=INDEX(C3:G7,I5,I4)
6	4	Bozeman	769	1168	862	0	811			=INDEX(B3:B7,I6)	=INDEX(C3:G7,I6,I5)
7	5	Omaha	716	353	293	811	0			=INDEX(B3:B7,I7)	=INDEX(C3:G7,I7,I6)
8										Total Distance:	=SUM(K3:K7)

Figure 14-21. The model used to determine the city order

Once you've formulated the model, you can solve the problem by following these steps in Solver:

1. Choose Data ⇒ Analyze ⇒ Solver to open the Solver Parameters dialog box.

2. In the Set Objective box, select the objective cell K8 (the total distance).

3. Choose a To option of Min.

4. In the By Changing Variable Cells box, choose I3:I7.

5. Click Add to open the Add Constraint dialog box, choose I3:I7 in the Cell Reference box, and select dif from the drop-down list; this specifies I3:I7 must be different integers and automatically puts "AllDifferent" in the Constraint box. Then click OK to close the Add Constraint dialog box and return to the Solver Parameters dialog box.

6. Place a check in the Make Unconstrained Variables Non-Negative check box to ensure that all the changing cells will be greater than or equal to 0.

7. Select Evolutionary from the Select a Solving Method drop-down list because the objective cell depends on values evaluated using the INDEX function.

8. Click the Options button, uncheck the Ignore Integer Constraints check box in the All Methods tab (see Recipe 14.19), then click OK.

Figure 14-22 shows the completed options in the Solver Parameters dialog box.
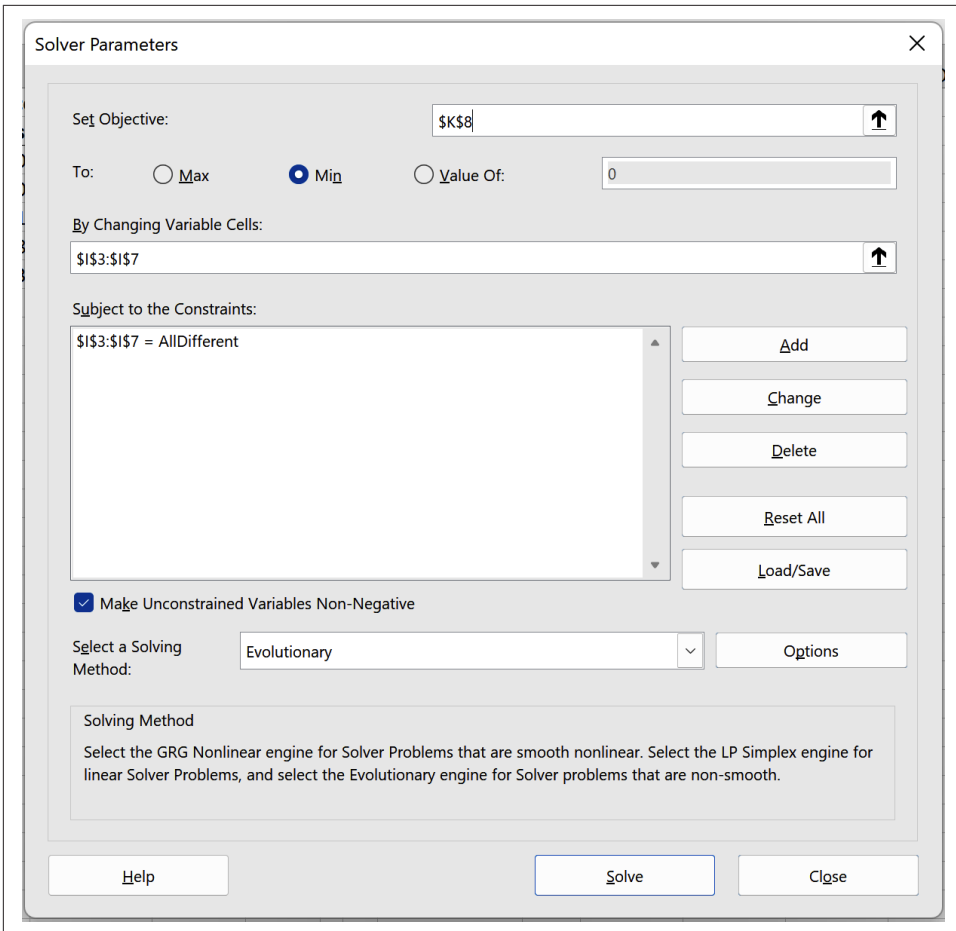
Figure 14-22. The options needed in the Solver Parameters dialog box to solve the sequencing problem

Once you've selected the appropriate options in the Solver Parameters dialog box, click Solve to solve the problem. Solver searches for a solution and then updates the changing cells in the worksheet (see Figure 14-23).

	A	B	C	D	E	F	G	H	I	J	K
1			**Distances Between Cities**							**City Delivery Order**	
2			Albuquerque	St Louis	Minneapolis	Bozeman	Omaha		No	Name	Miles
3	1	**Albuquerque**	0	940	980	769	716		1	Albuquerque	769
4	2	**St Louis**	940	0	461	1168	353		5	Omaha	716
5	3	**Minneapolis**	980	461	0	862	293		2	St Louis	353
6	4	**Bozeman**	769	1168	862	0	811		3	Minneapolis	461
7	5	**Omaha**	716	353	293	811	0		4	Bozeman	862
8										**Total Distance:**	**3161**

Figure 14-23. In the optimal sequence, Solver has updated cells I3:I7, so the minimum total distance is 3161

Discussion

This recipe shows how to restrict Solver's changing cells so they are all different values. This option is handy if you want to solve an ordering or sequencing problem, such as the example used in this recipe.

14.16 Handling Discontinuities with Solver

Problem

You have a formula with one or more discontinuities and want to know how to solve it using Solver.

Solution

Suppose cell B1 contains the formula =1/A1, which, as described in Recipe 14.10, is discontinuous when A1 is 0. You want to use Solver to find the value of A1, for which B1 is 0.25.

You can solve this problem using Solver's Evolutionary solving method as follows:

1. Choose Data ⇒ Analyze ⇒ Solver to open the Solver Parameters dialog box.

2. Select the objective cell B1 (the formula) in the Set Objective box.

3. Choose the To option Value Of and type **0.25** in the value box.

4. In the By Changing Variable Cells box, choose A1.

5. Click Add to open the Add Constraint dialog box, choose A1 in the Cell Reference box, select <= from the drop-down list, and type **10** in the Constraint box; this restricts the values the Evolutionary solving method will try so it runs quicker. Then click OK to close the Add Constraint dialog box and return to the Solver Parameters dialog box.

6. Place a check in the Make Unconstrained Variables Non-Negative check box to further restrict the values the solving method will try.

7. Select Evolutionary from the Select a Solving Method drop-down list.

Figure 14-24 shows the completed options in the Solver Parameters dialog box.
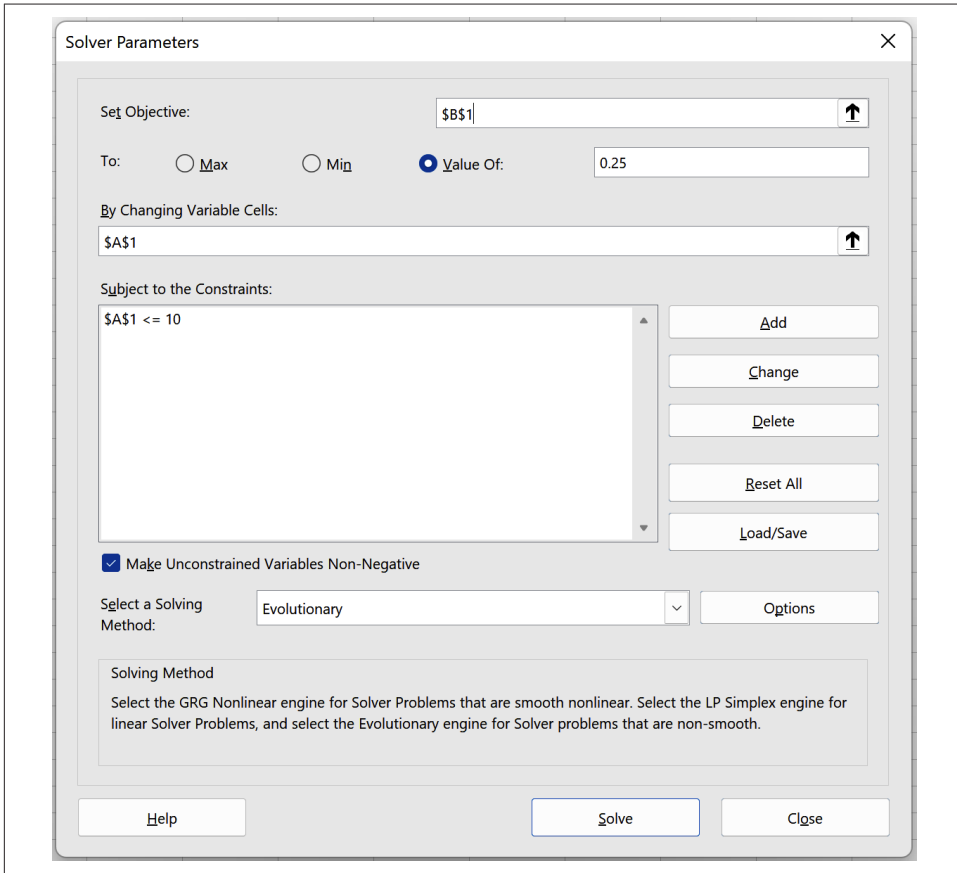
Figure 14-24. The options needed in the Solver Parameters dialog box to solve the discontinuous formula

Once you've completed the Solver Parameters dialog box, click Solve to solve the problem. Solver searches for a solution and then updates the changing cell in the worksheet—in this example, with a value that's approximately 4.

Discussion

This recipe shows how to solve a problem with discontinuities using Solver's Evolutionary solving method. Unlike Recipe 14.10, it doesn't matter if the formula is

discontinuous at the changing cell's initial value since the Evolutionary method automatically tries different starting points. For this reason, it can also cross the discontinuity in search of solutions.

In the example used for this recipe, you check the Make Unconstrained Variables Non-Negative check box because you don't need Solver to test negative values of A1. If you want Solver to test negative values (for example, when finding the value of A1 where B1 is –0.25), you should uncheck the check box and add an extra constraint specifying a lower bound—for example, A1 >= –10.

Generally, you should try to limit the range in which Solver searches for a solution when using the Evolutionary solving method because this helps it run quicker; you may need to adjust these constraints if Solver can't find a feasible solution.

> If you don't want to include upper or lower constraints, click Options in the Solver Parameters dialog box, navigate to the Evolutionary tab, and uncheck the Require Bounds on Variables check box (see Recipe 14.19). However, doing so means Solver will take longer to find a solution.

14.17 Finding Multiple Solutions with Solver

Problem

You want to use Solver to find multiple solutions to a problem.

Solution

Suppose cell B1 contains the formula =A1^2, and you want to determine the values A1 should be for B1 to return 49. Recipe 14.9 shows how to use Goal Seek to solve this problem by trying different initial values in the changing cell, and you can adopt a similar approach to solve it with Solver; depending on the solving method you choose, Solver will return the solution closest to each initial value.

To find the first solution:

1. Type **10** in cell A1.
2. Choose Data ⇒ Analyze ⇒ Solver to open the Solver Parameters dialog box.
3. Select the objective cell B1 (the formula) in the Set Objective box.
4. Choose a To option of Value Of and type **49** in the box next to it.
5. In the By Changing Variable Cells box, choose A1.
6. Uncheck the Make Unconstrained Variables Non-Negative check box since you don't need to restrict A1 to 0 or positive values.

7. Select GRG Nonlinear from the Select a Solving Method drop-down list since the formula in B1 is smooth but not linear.

8. Click Options to open the Options dialog box, select the GRG Nonlinear tab, and ensure that the Use Multistart check box is unchecked. Then, close the Options dialog box.

9. Click Solve to solve the problem. When you do so, Solver finds a solution that's approximately 7.

Figure 14-25 shows the completed options in the Solver Parameters dialog box.
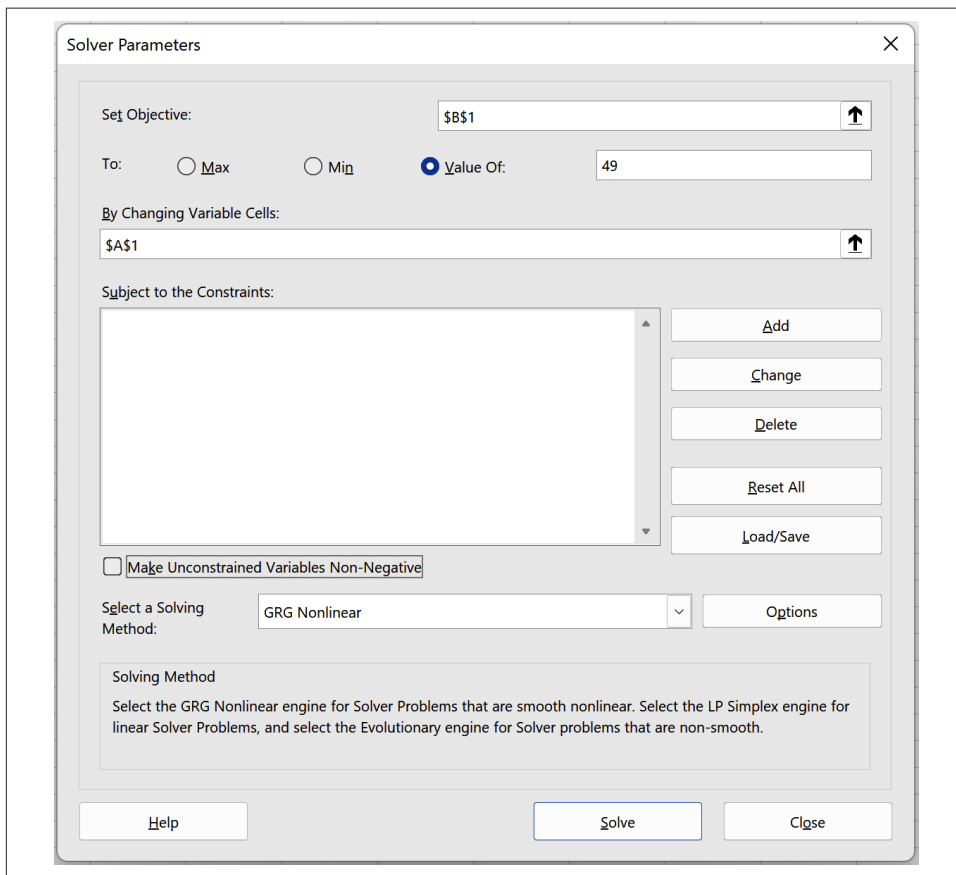
Figure 14-25. The options needed in the Solver Parameters dialog box to find the first solution

To find the second solution:

1. Type **-10** in cell A1.

2. Choose Data ⇒ Analyze ⇒ Solver to open the Solver Parameters dialog box; then click Solve to find a solution that's approximately –7. You don't need to change any parameters because Solver automatically saves the last set.

Discussion

This recipe shows how to solve problems with multiple solutions using Solver instead of Goal Seek. Generally, Solver returns the result closest to the changing cell's initial value, so you can try different values to drive out separate solutions.

14.18 Finding a Formula's Global Minimum or Maximum with Solver

Problem

You have a nonlinear formula and want to find the changing cell values that minimize or maximize it.

Solution

Suppose cell B1 contains the formula =6*A1^3+4*A1^2-7*A1+10 and you want to find the value of A1 between 2 and –2 that maximizes B1.

Global optimization problems like this can be tricky since they may have local minimum and maximum values. Solver may return one of these instead of the global minimum or maximum. For example, if A1 has an initial value of 0, maximizing cell B1 using the GRG Nonlinear solving method's default options returns a value that's approximately –0.88 for A1 (a local maximum) instead of 2 (the global maximum within the specified range); see Figure 14-26.
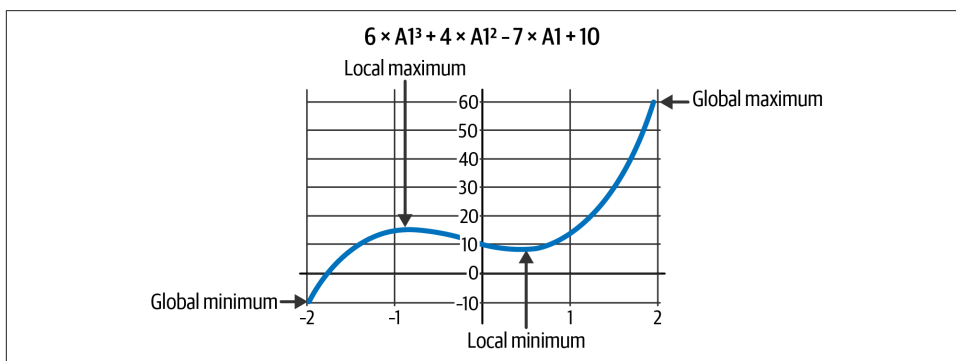
Figure 14-26. Chart showing the formula's local and global minima and maxima

A more reliable way of solving this problem is to use the multistart version of the GRG Nonlinear solving method. This method tries different initial values for any changing cells—in this example, A1—before converging on a value, so it's more likely to find the globally optimal solution.

These steps show how to solve the problem using this solving method:

1. Choose Data ⇒ Analyze ⇒ Solver to open the Solver Parameters dialog box.

2. Select the objective cell B1 (the formula) in the Set Objective box.

3. Choose a To option of Max.

4. In the By Changing Variable Cells box, choose A1.

5. Click Add to open the Add Constraint dialog box, choose A1 in the Cell Reference box, select <= from the drop-down list, and type **2** in the Constraint box. Then click Add to add the next constraint.

6. For the second constraint, choose A1 in the Cell Reference box, select >= from the drop-down list, and type -**2** in the Constraint box. Then click OK to close the Add Constraint dialog box and return to the Solver Parameters dialog box.

7. Uncheck the Make Unconstrained Variables Non-Negative check box because you don't need to restrict A1 to 0 or positive values.

8. Select GRG Nonlinear from the Select a Solving Method drop-down list since the formula in B1 is smooth but not linear.

9. Click Options to open the Options dialog box, select the GRG Nonlinear tab, and place a check in the Use Multistart check box. Then, close the Options dialog box.

10. Click Solve to solve the problem. When you do so, Solver finds a solution that's approximately 2.

Discussion

This recipe shows how to solve a global optimization problem using the multistart version of the GRG Nonlinear solving method. This method will likely find a globally optimal solution because it generates multiple starting points.

14.19 Adjusting Solver's Options

Problem

You want to adjust Solver's default options, such as setting time limits or changing the precision.

Solution

You can adjust Solver's Options by choosing Data ⇒ Analyze ⇒ Solver to open the Solver Parameters dialog box; then click Options to open the Options dialog box.

The All Methods tab includes the following options:

Constraint Precision
> This is the required precision for any constraints; the smaller the number, the higher the precision.

Use Automatic Scaling
> Checking this option reduces the impact of extremely large or small values on the solution.

Show Iteration Results
> Check this to pause after each iteration and display its results.

Ignore Integer Constraints
> Check this to make Solver ignore any integer, binary, or alldifferent constraints (see Recipes 14.13, 14.14, and 14.15).

Integer Optimality (%)
> This is the tolerance for integer solutions. Its default value is 1, which finds a near-optimal solution relatively quickly. To spend more time looking for the best solution, set this option to 0.

Max Time and Iterations
> The maximum time (in seconds) to spend searching for a solution and the maximum number of trial solutions.

Max Subproblems and Max Feasible Solutions
> The maximum number of subproblems and feasible solutions to explore when using the Evolutionary solving method or integer constraints.

The GRG Nonlinear tab includes options that Solver uses with the GRG Nonlinear solving method; the options are as follows:

Convergence
> This is the relative change you want to allow in the last five iterations before Solver decides it has converged on a solution. Typing a smaller number means Solver may find a better solution.

Derivatives
> You can choose to estimate derivatives through forward or central differencing. Central differencing may be more accurate but involves more calculations and takes longer.

Use Multistart, Population Size, and Random Seed

Checking the Use Multistart option runs the GRG Nonlinear solving method repeatedly with different starting values (see Recipe 14.18). If Use Multistart is checked, the Population Size option specifies how many starting points to use, and the Random Seed box specifies a fixed seed to generate them randomly; if Random Seed is blank, Solver will use a different seed each time it runs.

Require Bounds on Variables

If the Use Multistart option is checked, this option controls whether there must be upper and lower constraints for the changing cells.

The Evolutionary tab includes options that Solver uses with the Evolutionary solving method; the options are as follows:

Convergence

This is the maximum percentage difference in the objective values for the top 99% of the population. Typing a smaller number means Solver may find a better solution, but it will take longer.

Mutation Rate

This is a number between 0 and 1, which specifies the relative frequency at which a population member can mutate. Increasing the rate may find a better solution, but it will take longer.

Population Size

This is the number of points you want Solver to maintain when searching for a solution.

Random Seed

This specifies a fixed seed Solver should use to make random choices; if it's blank, Solver will use a different seed each time it runs.

Maximum Time without Improvement

This is the number of seconds you want Solver to run without any significant improvement.

Require Bounds on Variables

This option controls whether the changing cells must have upper and lower constraints.

> To restore the default options, click Reset All in the Solver Parameters dialog box. Doing so also clears any constraints and changing cells, and changes the objective cell to the worksheet's active cell.

Discussion

Tweaking Solver's options can be handy if you need to diverge from its default settings—for example, to speed up the Evolutionary method, use the GRG Nonlinear Multistart option, or find a more accurate solution. This recipe gives an overview of the options and the effect each one has.

14.20 Saving and Loading Solver Parameters

Problem

You want to save a set of Solver parameters or load a previously saved set.

Solution

Suppose you're using Solver to solve an optimization problem and want to try different constraints to find solutions for different situations. You can save the parameters for each situation in a worksheet range as follows:

1. Open the Solver Parameters dialog box by choosing Data ⇒ Analyze ⇒ Solver.
2. Complete the parameters needed to solve the problem—the objective cell, changing cells, constraints, and so on.
3. Click Load/Save to open the Load/Save Model dialog box.
4. In the range box, select a range of empty cells to which you want to save the parameters. When you click Save, Solver saves the parameters.

To load a previously saved set of parameters:

1. Open the Solver Parameters dialog box, then click Load/Save.
2. In the range box, select the worksheet range containing the saved parameters.
3. Click Load to load the parameters to the Solver Parameters dialog box.

Discussion

Solver automatically saves the last set of parameters you enter in the Solver Parameters dialog box, so you don't need to start from scratch each time you reopen it. This recipe shows how to save other parameters so you can reload them when needed.

14.21 Saving Solver-Generated Scenarios

Problem

You've solved a problem using Solver and want to save its changing cell values as a scenario.

Solution

Once you've solved a problem with Solver, you can save its changing cell values as a scenario (see Recipe 14.5).

To save the scenario, click Solve in the Solver Parameters dialog box; then in the Solver Results dialog box, click Save Scenario. When prompted, type the scenario's name, then click OK to save the scenario.

Discussion

This recipe offers a convenient way of adding a Solver solution to Scenario Manager so you can quickly refer back to its results. Doing so saves the solution's changing cell values as a new scenario, using a name you specify.

14.22 Displaying Solver Reports

Problem

You want to generate a report when Solver finds a solution or encounters a problem.

Solution

Solver uses the Solver Results dialog box to inform you when it has solved or encountered a problem, and it includes a set of reports—listed in the Reports box—that you can use to find out more about its results.

To generate a report, select it in the Reports box, optionally check the Outline Reports check box to generate an outline for it (see Recipe 2.17), then click OK; Solver adds the report to the workbook as a new worksheet.

Discussion

Solver can display a variety of reports, depending on the solving method, whether there are integer constraints, and the result. For example, if Solver can't find a feasible solution, you can generate a report showing any constraints Solver can't satisfy so you can double-check them.

Power Query

Power Query is a powerful data transformation tool that lets you import external data, perform a set of data manipulations, and then load the results into Excel. If the underlying data changes, you can update it in Excel by refreshing it. Power Query has four main phases:

1. In the *connect* phase, you specify which data you want to import and how to access it.

2. Once you've connected to the data, you can *transform* it using the tools in the Power Query Editor. For example, you can apply filters, split or merge columns, and invoke custom functions.

3. You can optionally *combine* datasets by appending or merging their data.

4. Finally, you *load* the data into Excel.

This chapter includes recipes for each phase.

> Power Query, or Get & Transform, is available in Excel 2016 or later for Windows. Many features are available in Excel 365 for Mac, Excel for Web supports some features, and other Microsoft products—such as Power BI—include it too. This chapter focuses on those features available in Excel 365 for Windows.

15.1 Getting and Loading Data

Problem

You want to get data from a file, the web, a database, or another source and load it into Excel.

Solution

Power Query lets you get data from various sources, including files, databases, and websites (depending on your version of Excel). Once you've specified which data to import, you can shape and transform it using the Power Query Editor and then load the results into Excel.

To get the data, choose Data ⇒ Get & Transform Data ⇒ Get Data, and select the appropriate option. Depending on your version of Excel, the options are as follows:

From File
> You can import data from Excel, Text/CSV, XML, JSON, PDF files, or files in a folder with the same type and structure (see Recipe 15.2).

From Database
> Supported databases include SQL Server, Microsoft Access, and SQL Server Analysis Services.

From Azure
> This option includes Azure Data Lake Storage Gen2 and Azure Data Explorer.

From Power Platform
> This includes Dataflows and Dataverse options.

From Other Sources
> You can load data from a table or range, a web page, a Microsoft Query, an OData feed, an ODBC or OLEDB connection, or a picture. You can also create a blank query, which you can use, for example, to create a custom function (see Recipe 15.25).

Once you've selected an option, use the wizard to specify which data to import—for example, a specific Excel file and table. Then click Load to load the data to a default destination in Excel, Load To (from the Load button's drop-down menu) to choose the destination (see Recipe 15.3), or Transform to shape the data using the Power Query Editor before loading it.

When you load the data into Excel, Power Query creates a query containing the steps needed to get and transform it. You can see a list of the workbook's queries in the Queries & Connections pane's Queries tab, which Excel generally opens by default. If this pane is closed, you can open it by choosing Data ⇒ Queries & Connections ⇒ Queries & Connections.

Discussion

This recipe focuses on connecting to data and loading it into Excel. See Recipe 15.2 for a specific example.

15.2 Getting and Loading Data from Files in a Folder

Problem

You have a folder containing files with identically structured data that you want to combine and load into Excel.

Solution

Suppose you have identically structured data for multiple teams or departments in separate Excel files. You want to load the data into a new Excel table that includes all the rows in each file as a one-off or repeat task. You can solve this problem by creating a query that gets and loads data from a folder. The steps to do so are as follows:

1. Put the files whose data you want to combine in a dedicated folder.

2. Create or open the Excel workbook to which you want to load the data, ensuring it's in a different folder from that in step 1 and not one of its subfolders.

3. Choose Data ⇒ Get & Transform Data ⇒ Get Data ⇒ From File ⇒ From Folder.

4. Browse to the folder containing the files you want to load from, then click Open.

5. Power Query displays a list of the folder's files. Click the Combine button's drop-down arrow and select one of the options. For example, to combine the data and load it straight into Excel, choose Combine & Load from the Combine menu.

6. Select a sample file in the Combine Files dialog box, which Power Query uses as the basis for its queries. Then select the data to load, place a check in the "Skip files with errors" check box, and click OK to create the query.

Figure 15-1 shows the Combine Files dialog box.

Once you've created the query, you can include any new files added to the folder or accommodate any data updates by refreshing the query (see Recipe 15.5). Doing so reruns the query's steps and reloads the data.
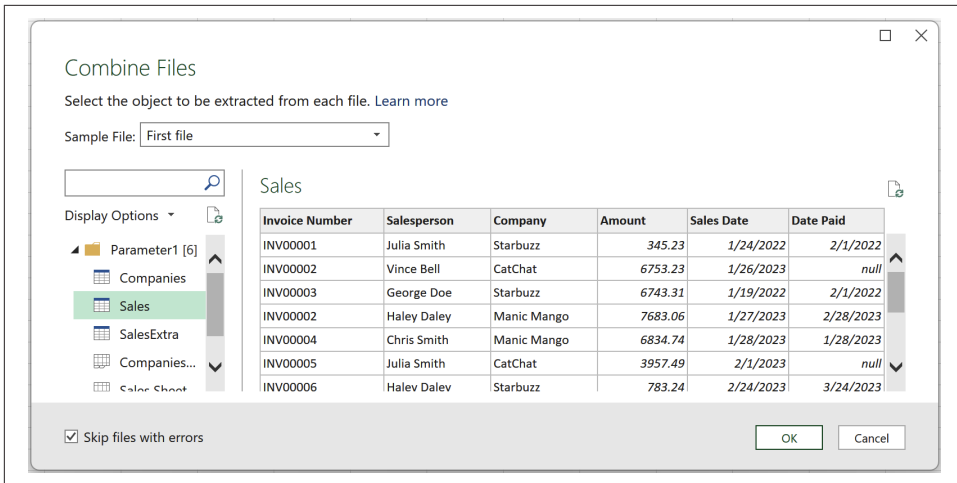
Figure 15-1. The Combine Files dialog box

You can also use this feature to load data from multiple folders—for example, if different departments have files in subfolders. To import data in this situation, choose the parent folder in step 3, then follow the rest of the steps to include all files. If necessary, you can then use Recipe 15.12 to filter the files.

Discussion

Manually combining the data from multiple files can be time-consuming and error-prone. This recipe helps simplify this process because you can use Power Query's Get Data From Folder wizard to append the data for each file and load it into Excel.

This recipe is especially helpful if you need to include any changes to the original data—for example, any extra files that get added to the folder—to create monthly reports because you can refresh the query to reload the data (see Recipe 15.5).

15.3 Specifying Where to Load Data To

Problem

You have a query and want to specify where to load its data.

Solution

By default, Power Query loads the data from a single query to a table in a new worksheet and the data from multiple queries at once to the data model (see Chapter 16). However, depending on your version of Excel, you can choose to load the data to a different location.

For new queries, you can choose where to load the data by choosing the Load To option in the Get Data wizard (see Recipe 15.1) or Home ⇒ Close ⇒ Close & Load ⇒ Close & Load To in the Power Query Editor (see Recipe 15.7). Doing so opens the Import Data dialog box, which includes the following options:

Table
    This default option loads the data to an Excel table.

PivotTable Report and PivotChart
    These options load the data to a PivotTable cache (see Recipe 11.28) so you can create a PivotTable or PivotChart without first creating a table.

Only Create Connection
    This option gives other queries access to the query's results without loading its data into Excel. This can be handy if, for example, you want to use the output of one query to filter the results of another, as in Recipe 15.31.

Where do you want to put the data?
    If you've chosen the Table, PivotTable Report, or PivotChart option, you can choose whether to use a new or existing worksheet.

Add this data to the Data Model
    Check this check box if you want Power Query to load the data to the workbook's data model (see Chapter 16).

Figure 15-2 shows the Import Data dialog box.
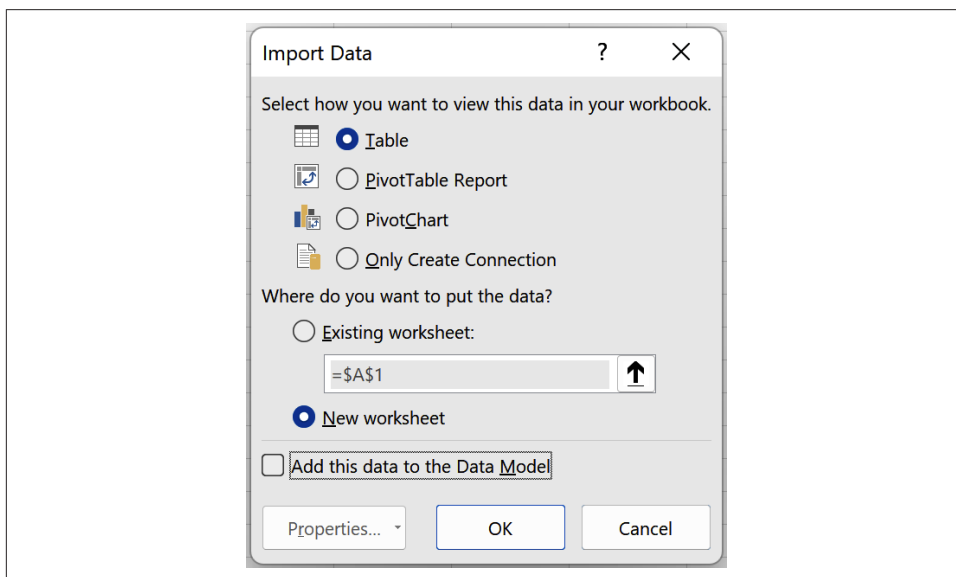
Figure 15-2. The Import Data dialog box

You can open the Import Data dialog box for existing queries by right-clicking the query in Excel's Queries & Connections pane and selecting Load To.

> To change where Power Query loads data by default, choose Data ⇒ Get & Transform Data ⇒ Get Data ⇒ Query Options in Excel (or File ⇒ Options and Settings ⇒ Query Options in the Power Query Editor), select Data Load in the Global section, and then use the options under Default Query Load Settings.

Discussion

This recipe shows you how to specify where to load a query's data, depending on your version of Excel; at the time of writing, Excel for Mac can load data only to an Excel table, so the Import Data dialog box isn't available.

15.4 Editing Data Source Settings and Security

Problem

You want to change the location or credentials of a data source.

Solution

Suppose you have one or more queries that get data from a file you've subsequently moved. To update the file's location, choose Data ⇒ Get & Transform Data ⇒ Get Data ⇒ Data Source Settings in Excel (or Home ⇒ Data Sources ⇒ Data Source Settings in the Power Query Editor) to open the Data Source Settings dialog box (see Figure 15-3). Then, choose the "Data sources in current workbook" option, select the data source, click Change Source, and navigate to the file's new location.

> If you want to change the data source of a single query—for example, to refer to a different file or table—you can do so by editing the query's Source and/or Navigation steps. This is useful if you want to copy a query and apply its transformations to a different data source (see Recipe 15.27).

You can also use the Data Source Settings dialog box to change the permissions of a data source used in the current workbook or update global permissions; choose the "Data sources in current workbook" or "Global permissions" option, select the data source, then click Edit Permissions or Clear Permissions, depending on what you want to do.
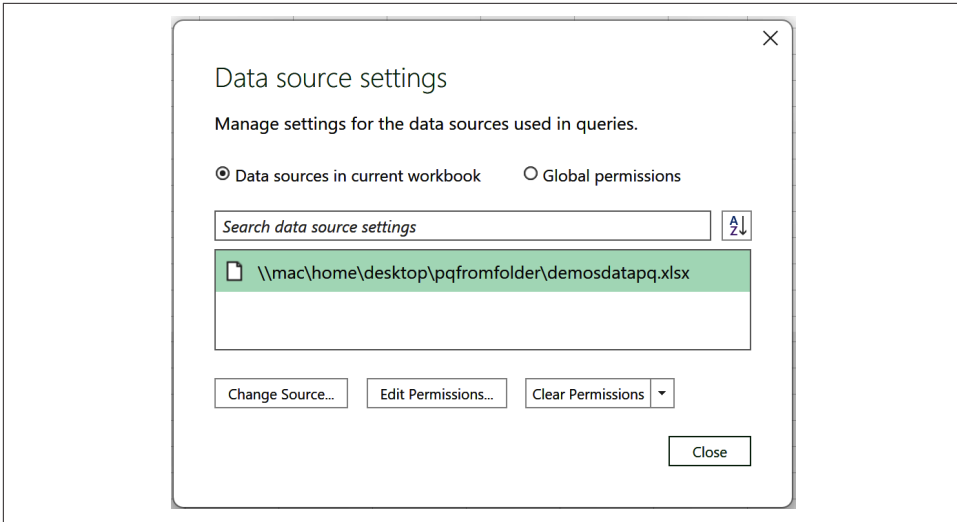
Figure 15-3. The Data Source Settings dialog box

To control how strict security should generally be for different types of data sources, choose Data ⇒ Queries & Connections ⇒ Get Data ⇒ Query Options in Excel (or File ⇒ Options and Settings ⇒ Query Options in the Power Query Editor), select Security in the Global section, then use the available options, shown in Figure 15-4.
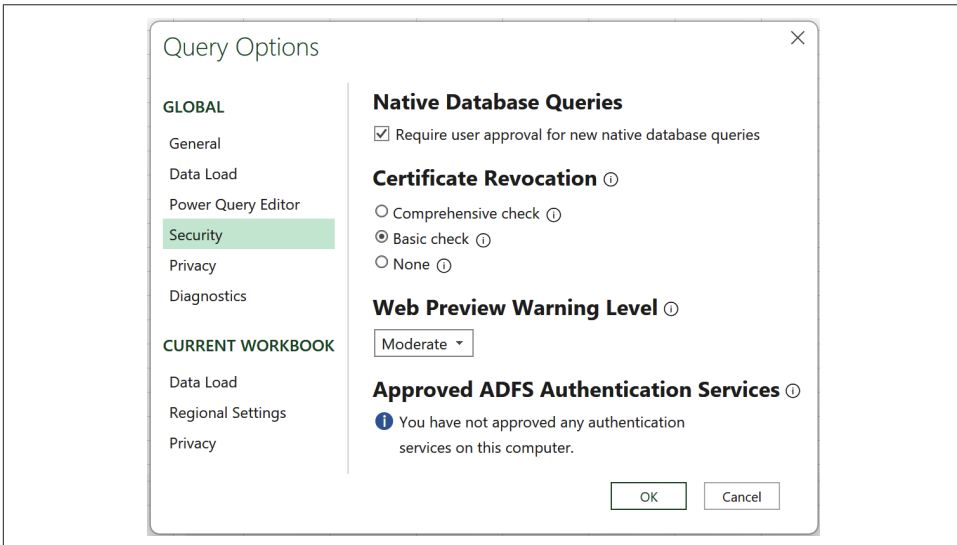
Figure 15-4. The Query Options dialog box

Discussion

This recipe shows how to manage a data source's settings, such as location or permissions. You can use this approach, for example, if you have queries that load data from a file you've subsequently moved.

15.5 Refreshing a Query's Data

Problem

You have a query whose underlying data has changed and want to refresh the data loaded in Excel.

Solution

Similarly to a PivotTable (see Recipe 11.5), a query provides a snapshot of the underlying data. If the data changes, you need to refresh the query to load these changes to Excel. Doing so reruns the query's steps, applying any transformations to the updated data.

You can refresh a query using the following methods:

- Hover your mouse over the query in Excel's Queries & Connections pane and then click its Refresh icon .
- Right-click the query in the Queries & Connections pane and choose Refresh.
- Choose Data ⇒ Queries & Connections ⇒ Refresh ⇒ Refresh or Refresh All.
- Use Recipe 15.7 to edit the query; then choose Home ⇒ Close ⇒ Close & Load in the Power Query Editor.

Generally, choose the Refresh option to refresh the selected query and Refresh All to refresh all the data in the workbook, including queries, PivotTables, and Power Pivot connections.

You can also make Excel automatically refresh the data in specific queries every hour or when you open the workbook. If you're using Excel for Windows, right-click the query in the Queries & Connections pane, choose the Properties option, select the Usage tab, and then use the settings in the Refresh Control section to specify when you want the data to refresh. If you're using Excel for Mac, you can find these settings by selecting a cell in the table you've loaded to and then choosing Data ⇒ Properties.

Discussion

When you refresh a query, Power Query reruns each step, applying any transformations to the updated data. For example, if you have a query that loads data from files in a folder (see Recipe 15.2), refreshing the data will automatically include any new files you've added, saving you from manual, time-consuming edits.

Being able to refresh a query's data also means you can effectively automate many data import or transformation tasks without having to use macros or VBA. For example, you may generate monthly reports by creating queries you periodically refresh.

15.6 Managing Queries

Problem

You want to rename a query, delete it, copy it to a different workbook, or add it to a group.

Solution

You can generally manage a query by right-clicking it in Excel's Queries & Connections pane and choosing one of the available options:

Copy and Paste
These options let you copy and paste a query, including to a different workbook.

Edit, Delete, and Rename
Use these to edit the query (see Recipe 15.7), delete it, or change its name.

Refresh and Load To
Use these to refresh the query's data (see Recipe 15.5) or change where it's loaded (see Recipe 15.3).

Duplicate, Reference, Merge, and Append
Use these options to duplicate the query, create a query based on it, or merge or append its data with the data in another query; see Recipes 15.27, 15.28, 15.29, and 15.30.

Export Connection File
Use this option to share an external data source. Selecting it creates an Office Data Connection (ODC) file containing the data source connection information and the Power Query steps and formulas. To use the shared data source, double-click the ODC file, which opens the file in a new Excel workbook. Then, use the Import Data dialog box to specify how to load the data.

Move To Group, Move Up, and Move Down

These options let you group queries or change a query's position in a group.

Show the Peek

This displays a pop-up window showing a preview of the query's data, its columns, when it was last refreshed, and other information.

Properties

Use this to open the Query Properties dialog box. You can use this, for example, to automatically refresh a query's data (see Recipe 15.5).

Figure 15-5 shows the Queries & Connections pane and a query's right-click menu.
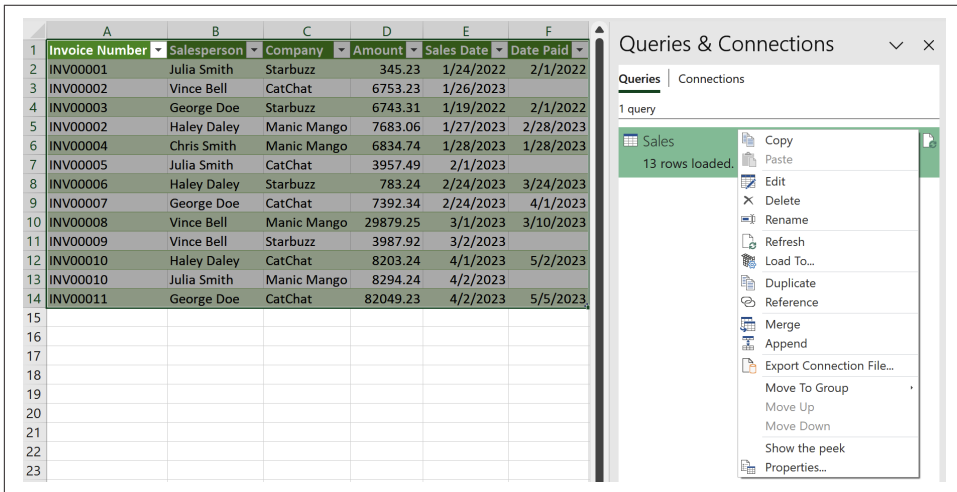
Figure 15-5. The Queries & Connections pane (on the right), including a query's right-click menu

> You can see other options by choosing Data ⇒ Get & Transform Data ⇒ Get Data ⇒ Query Options in Excel or File ⇒ Options and Settings ⇒ Query Options in the Power Query Editor. For example, you can choose to not show the query peek when hovering your mouse over a query using the "Query peek" option in the Global ⇒ General section.

Discussion

This recipe gives an overview of managing queries and performing basic tasks such as renaming, deleting, or organizing queries in groups.

You can find many of these options in the Power Query Editor by right-clicking the query in the query list. You can also rename a query by typing a new name in the Query Settings pane's Name box (see Recipe 15.8).

15.7 Editing a Query

Problem

You want to edit a query to shape and transform the data it returns.

Solution

Suppose you've used Power Query to load the data into Excel, and you want to transform the data. You can do so by editing the query.

You edit queries in the Power Query Editor; use one of the following methods to open it:

- Choose Data ⇒ Get & Transform Data ⇒ Get Data ⇒ Launch Power Query Editor.
- Double-click the query in the Queries & Connections pane.
- Right-click the query and choose Edit.

The Power Query Editor opens in a new window, shown in Figure 15-6.
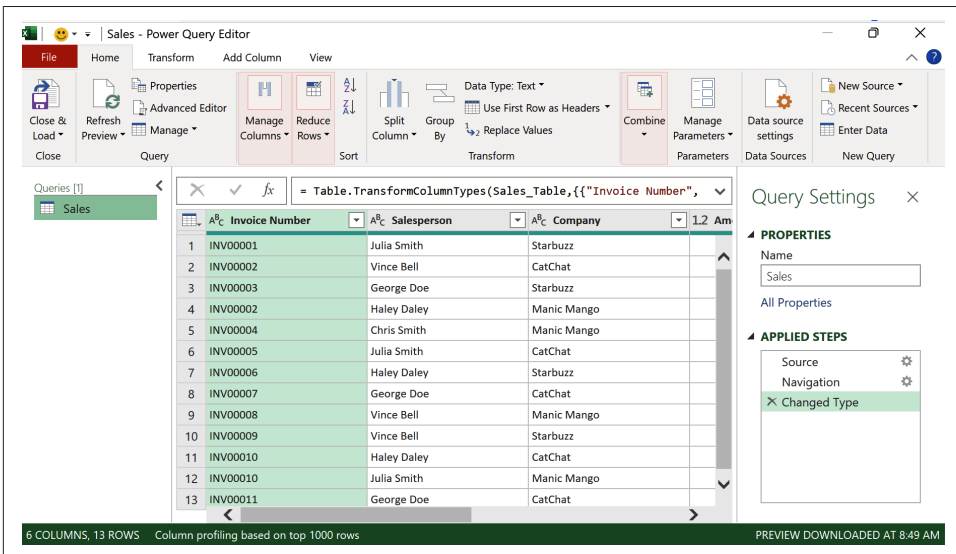
Figure 15-6. The Power Query Editor window

The Power Query Editor displays a list of queries on the left, which you use to select the query you want to edit. When you select a query, a preview of the first 1,000 rows of its data appears in the window's central area, which shows the effect of any transformations you apply. Selecting a query also displays the Query Settings pane on the right, which displays a list of the query's steps (see Recipe 15.8).

The Power Query Editor's data preview uses a saved cache of the data, and Power Query may prompt you to refresh the preview if it's old. You can do so by clicking the Refresh button in the prompt, choosing Home ⇒ Query ⇒ Refresh Preview, or choosing Home ⇒ Query ⇒ Refresh Query ⇒ Refresh All to refresh the cache for all queries. To clear the cache, choose File ⇒ Options and Settings ⇒ Query Options in the Power Query Editor (or Data ⇒ Get & Transform Data ⇒ Get Data ⇒ Query Options in Excel) to open the Query Options dialog box; then choose Global ⇒ Data Load and click Clear Cache in the Data Cache Management Options section.

To transform the data, use the options in the ribbon or column headings described in later recipes. Then, when you've finished editing the query, you choose Home ⇒ Close ⇒ Close & Load to close the Power Query Editor and load the data into Excel.

Discussion

This recipe shows how to open the Power Query Editor window, which you use to edit a query. See other recipes in this chapter for specific details on the available options.

15.8 Managing a Query's Steps

Problem

You have a query and want to delete or edit one of its steps.

Solution

A query's steps specify its data source and any transformations used to shape the data. Power Query adds a new step for each transformation you apply.

The Power Query Editor displays a list of the query's steps in the Query Settings pane, which appears on the right (see Figure 15-6); if this pane is closed, you can open it by choosing View ⇒ Layout ⇒ Query Settings. The Query Settings pane displays the name of the current query in a Name box, and the Applied Steps section contains a list of the steps in the order in which it applies them.

To see how a particular step transforms the data, click the step; this changes the data preview to show the step's effect. You can also edit a step (where possible) by hovering your mouse over it and clicking the settings icon ⚙ that appears to the right and

delete a step by clicking the delete icon ✕ that appears to the left. You can find both of these options (and more) by right-clicking the step.

Discussion

Unlike Excel, Power Query has no Undo command. However, you can edit or delete steps (as described in this recipe) to undo changes you've made. You can also discard any changes you've made since launching the Power Query Editor by closing the Power Query window and clicking the Discard option when it asks if you want to keep your changes.

15.9 Managing Columns

Problem

You have a query and want to rename, move, or delete a column.

Solution

Suppose you have a query and you want to transform its data by renaming, moving, or deleting a column.

To rename a column, double-click its name in the Power Query Editor's data preview (see Figure 15-6), then type its new name. You can move a column by selecting it (by clicking the column header) and dragging it to the new location, and delete a column by selecting it and pressing the Delete key. You can also perform these actions by right-clicking the column and choosing the Rename, Move, or Remove option.

> In some situations, such as importing data from the web, you may want to use the first row of data for the column names or the column names as the first row. You can do so by choosing Home ⇒ Transform ⇒ Use First Row as Headers and selecting the appropriate option (see Recipe 15.14).

Discussion

This recipe shows you how to transform data by renaming, moving, or deleting a query's column.

If your query has many columns, you may find it easier to select or go to them by choosing Home ⇒ Manage Columns ⇒ Choose Columns and using the Choose Columns and Go To Column options. You can also delete the selected—or unselected—columns by choosing Home ⇒ Manage Columns ⇒ Remove Columns, or move the

selected columns to the left, right, beginning, or end by choosing Transform ⇒ Any Column ⇒ Move.

15.10 Using Data Types

Problem

You have a column and want to view or edit the type of data it can contain.

Solution

Power Query assigns a data type to each column, which controls the type of data it can include and the operations you can apply to it. For example, a column with a Whole Number data type represents integer values, and you can apply numeric filters and operations.

To view or change a column's data type, select the column in the Power Query Editor and choose Home ⇒ Transform ⇒ Data Type. The available options are as follows:

Decimal Number, Currency, Whole Number, and Percentage
These are number data types that let you apply numeric filters and operations. Decimal Number represents a floating-point number and handles numbers with fractional values; Currency has a fixed location for the decimal separator; Whole Number represents integer values; and Percentage is like the Decimal Number data type except it formats values as percentages.

Date/Time, Date, Time, Date/Time/Timezone
These are date and/or time data types, where Date/Time has date and time components, Date is just a date, Time is just a time, and Date/Time/Timezone is a Coordinated Universal Time (UTC) Date/Time with a time-zone offset.

Duration
This represents a length of time, and you can use Duration operations to extract the number of days, hours, minutes, and so on. You can also use it to perform calculations with date and/or time values.

Text
This data type represents a text string.

True/False
This is for Boolean True and False values.

Binary
This represents data in a binary format.

The Power Query Editor displays an icon to the left of each column name indicating the column's data type. You can click this icon to open a list of data types, then either select one to change the data type or press the Esc key to close the list without making changes.

Discussion

This recipe provides an overview of Power Query's available data types and how to use them.

By default, Power Query automatically detects each column's data type by examining its values. You can control this behavior by choosing File ⇒ Options and Settings ⇒ Query Options in the Power Query Editor (or Data ⇒ Queries & Connections ⇒ Get Data ⇒ Query Options in Excel) and configuring the Type Detection options in the Global ⇒ Data Load and Current Workbook ⇒ Data Load sections. You can also choose Transform ⇒ Any Column ⇒ Detect Data Type to automatically detect the data type of a selected column.

15.11 Sorting and Filtering Data

Problem

You have a query and want to sort and/or filter the data in one or more columns.

Solution

You can sort and/or filter data in the Power Query Editor similarly to an Excel table.

To sort the data, select the column you want to sort by, choose Home ⇒ Sort, and use the options to sort the column in ascending or descending order. You can also reverse the order of the rows by choosing Transform ⇒ Table ⇒ Reverse Rows.

To filter the data, click the drop-down arrow to the right of the column name to open its drop-down menu; then use the available options. You can remove rows with empty values, specify which values to include or exclude, or use more advanced filters based on the column's data type (see Recipe 15.10). For example, if the column's data type is Date, you can apply a filter that returns only dates in the current year by choosing Date Filters ⇒ Year ⇒ This Year.

You can also filter data by right-clicking a cell containing the value you want to filter by and choosing the Filters option. For example, if you have a Salesperson column containing a list of names and want to see rows only for a specific name, you could right-click a cell containing that name and choose Text Filters ⇒ Equals.

Once you've applied a filter, you can modify it (for example, to add extra conditions) by editing its step, which opens the Filter Rows dialog box. Use the Basic option to apply one or two conditions to the same column, or choose the Advanced option to add extra clauses, which you can apply to different columns, as shown in Figure 15-7.
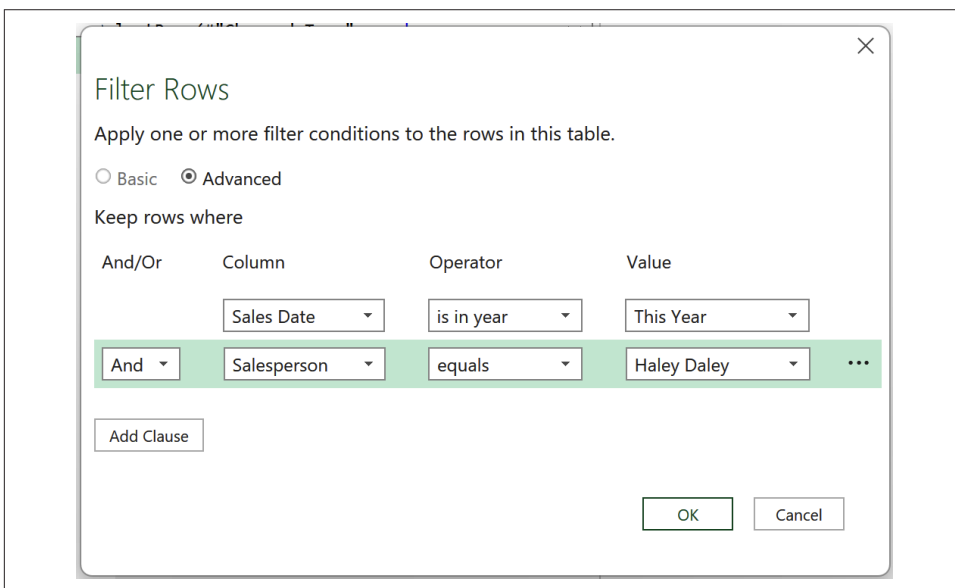
Figure 15-7. The Filter Rows dialog box

Discussion

Filtering data in the Power Query Editor reduces the number of rows the query loads into Excel, which is more efficient than loading the entire dataset. This is especially helpful when working with large datasets, where the number of rows may exceed Excel's row limit.

See Also

You can also apply a filter using a parameter value (see Recipe 15.24) or the results of another query (see Recipe 15.31).

15.12 Filtering Files When Loading Data from a Folder

Problem

You have a folder containing files with identically structured data and want to filter the files, combine their data, and load it into Excel.

Solution

Suppose you have identically structured data in separate Excel files in a folder. You want to combine and load the data from some files in the folder and specify which ones to include or exclude.

You can solve this problem as follows:

1. Follow the steps in Recipe 15.2, making sure you choose the Combine & Transform Data option from the Combine menu. Doing so creates a query and opens it in the Power Query Editor.

2. Click the Source step in the Power Query Editor's Query Settings pane to edit this step.

3. The Power Query Editor's data preview displays a list of the folder's files with columns including File Name, Extension, and Folder Path. Apply a filter to exclude any files whose data you don't want to load.

4. Choose Home ⇒ Close ⇒ Close & Load to close the Power Query Editor and load the data to Excel.

Discussion

This recipe builds on Recipe 15.2 and shows how to exclude particular files, giving you more control over your data. For example, you can exclude files in specific folders or include only those whose name begins with *Team*.

15.13 Removing Duplicates, Blank Rows, and Errors

Problem

You have a query that returns duplicate rows, blank rows, or error values and want to remove them.

Solution

Suppose you have a query that has duplicate or blank rows or that contains errors. You can remove these rows by choosing Home ⇒ Reduce Rows ⇒ Remove Rows and selecting from the available options: Remove Top Rows, Remove Bottom Rows, Remove Alternate Rows, Remove Duplicates, Remove Blank Rows, and Remove Errors.

You can also keep particular rows by choosing Home ⇒ Reduce Rows ⇒ Keep Rows and selecting one of these options: Keep Top Rows, Keep Bottom Rows, Keep Range of Rows, Keep Duplicates, or Keep Errors.

Discussion

This recipe provides a convenient way of removing or keeping particular row types. This is handy if you need to remove blank or duplicate rows from the data or keep rows with error values to help you track down the source of these errors.

15.14 Transforming Data in Columns

Problem

You have a query and want to transform the date held in one or more columns.

Solution

The Power Query Editor's Transform menu includes many options to shape the data held in columns.

The Transform menu's Table group contains options that apply to the query's entire dataset; these are as follows:

Group By
　This groups the data and calculates aggregations for each row. Use the Basic option to group by one column and calculate a single aggregation, and use the Advanced option to group by multiple columns or include more than one calculation.

Use First Row as Headers
　Use this to use the values in the first row as column names or use the column names as the data's first row.

Transpose
　This transposes the rows and columns, translating each row into a separate column.

Reverse Rows
　This reverses the order of the rows.

Count Rows
　This replaces the columns with a count of how many there are.

The Any Column group contains options for any column. Select the column—or columns—you want to transform and then choose one of the following options:

Data Type
　This displays the column's data type and lets you change it (see Recipe 15.10).

Detect Data Type

This automatically detects the data type of a selected column based on its values.

Rename and Move

Use these to rename or move the column to the left, right, beginning, or end.

Replace Values

Use the options in this group—Replace Values and Replace Errors—to find and replace values or errors in the selected column. If you choose the Replace Values option, you can use the advanced options to specify whether to match the cell's entire contents and use special characters.

You can also right-click a cell and choose Replace Values. Doing so opens the Replace Values dialog box, prefilling the "Value to find" box with the cell's value.

Fill

This fills empty cells with the value in neighboring cells in the same column. Choose Down to copy values down and fill empty cells with the value immediately above them and choose Up to copy values up and fill empty cells with the value immediately below.

Pivot Column

This produces output resembling a basic Excel PivotTable; see Recipe 15.16 for more details.

Unpivot Columns

This translates columns into attribute-value pairs, and it's handy for structuring data to use with PivotTables (see Recipe 15.17 for more details).

Convert to List

This converts the column to a list (see Recipe 15.19).

The Text Column group contains options that work with text values; select the column you want to transform, then choose one of the following options:

Split Column and Merge Columns

These options let you split or merge columns (see Recipe 15.15).

Format

This lets you format the column's values in different ways. Use the Lowercase, Uppercase, and Capitalize Each Word options to change the text case; Trim to remove leading and trailing whitespaces; Clean to remove nonprintable characters; Add Prefix to insert text before each value; and Add Suffix to add text after.

Extract

This lets you extract characters from the values. Use Length to return the length of the text string; First Characters, Last Characters, and Range to return a specified number of characters from the start, end, or middle of the text string; and Text Before Delimiter, Text After Delimiter, and Text Between Delimiters to return text using delimiters.

Parse

Use this to extract rows and columns from text formatted as XML or JSON.

The Number Column group contains options you can use with number columns; select the number column you want to transform, then select an option from the following:

Statistics

This replaces the query's columns with a single statistic derived from the selected column (see Recipe 15.19). Choose from Sum, Minimum, Maximum, Median, Average, Standard Deviation, Count Values, and Count Distinct Values.

Standard

This transforms the selected column using standard mathematical operations: Add, Multiply, Subtract, Divide, Integer-Divide, Modulo, Percentage, and Percent Of. For example, you can use this option to add a number to each value in the column or multiply each one by a specific rate.

Scientific

This includes the following operations: Absolute Value, Power, Square Root, Exponent, Logarithm, and Factorial.

Trigonometry

This includes Sine, Cosine, Tangent, Arcsine, Arccosine, and Arctangent trigonometry operations.

Rounding

Use this to round values up to the next integer or down to the previous one, or to specify how many decimal places to round to.

Information

Use this to return each value's sign or indicate whether it's positive or negative.

The Date & Time Column group contains options for date and/or time columns and durations. Select the column, then choose one of the following options, depending on its data type:

Date

This includes options you can use with columns that include a date component. For example, you can return the age, year, quarter, month, week, day, and the earliest or latest date.

The Age option returns a duration that's the difference between each date and today in days. To transform the age to the number of complete years, choose Transform ⇒ Date & Time Column ⇒ Duration ⇒ Total Years to convert the duration to years; then choose Transform ⇒ Number Column ⇒ Rounding ⇒ Round Down to round the result down to the previous integer.

Time

This includes options you can use with columns that include a time component. For example, you can return the hours, minutes, and seconds, and the earliest or latest time.

Duration

This includes options you can use with Duration data types. For example, you can return the day, hour, minute, or second component or the total number of years, days, hours, minutes, or seconds. You can also multiply or divide a duration by a specified value or return statistics such as the sum or average.

The Structured Column group contains options you can use with structured data—for example, a column that returns a table for each value. See Recipe 15.18 for details on how to work with this type of data.

Discussion

This recipe overviews the many Power Query Editor's Transform menu options. Some options transform the data as a whole, some transform any column, while others apply only to specific data types.

Generally, the options in the Transform menu replace the values in the columns they transform. If you want to keep the original column values and add a new column containing the transformation, use the options in the Add Column menu instead (see Recipe 15.20).

15.15 Splitting and Merging Columns

Problem

You want to split a column by delimiter, number of characters, positions, case, or type of character or merge columns.

Solution

To split a column, select the column, choose Home ⇒ Transform ⇒ Split Column, and select one of the options:

By Delimiter
> This splits values using a delimiter you specify, such as a space. For example, you can use this option to split the value *John Smith* into two columns, the first containing *John* and the second containing *Smith*. You can split columns using the delimiter's leftmost, rightmost, or all occurrences. The advanced options let you specify whether to split values into columns (the default) or rows and, optionally, how many columns.

By Number of Characters
> This splits values into fragments of a specified length. You can split values once—as far left or far right as possible—or repeatedly. The advanced options let you specify whether to split values into columns (the default) or rows and, optionally, how many columns.

By Positions
> This splits values into fragments at the positions you specify. For example, typing **0, 2** for the positions puts the first two characters in one column and the remaining characters in another. The advanced options let you specify whether to split values into columns (the default) or rows.

By Lowercase to Uppercase
> This splits values based on when the value switches from lowercase to uppercase. For example, it splits the value *JohnJosephSmith* into three columns, the first containing *John*, the second *Joseph*, and the third *Smith*.

By Uppercase to Lowercase
> This splits values based on when the value switches from uppercase to lowercase. For example, it splits the value *JohnJosephSmith* into four columns, the first containing *J*, the second *ohnJ*, the third *osephS*, and the fourth *mith*.

By Digit to Non-Digit
> This splits values based on when the value switches from numeric to nonnumeric characters. For example, it splits the value *NE123SW456* into two columns, the first containing *NE123* and the second *SW456*.

By Non-Digit to Digit

This splits values based on when the value switches from nonnumeric to numeric characters. For example, it splits the value *NE123SW456* into three columns, the first containing *NE*, the second *123SW*, and the third *456*.

You can also merge columns as follows:

1. Select the columns you want to merge in the order you want to merge them.

2. To create a new merged column that replaces the columns you selected in step 1, choose Transform ⇒ Text Column ⇒ Merge Columns. To create a new column and keep the columns you selected, choose Add Column ⇒ From Text ⇒ Merge Columns.

3. When prompted, choose a character to separate values by (the default is None), and specify the new column's name. Then click OK to create the column.

Discussion

Splitting columns using Excel functions can be tricky, depending on your version of Excel and how you want to split the data. Using Power Query offers a flexible alternative.

When merging columns, you can choose whether to replace or keep the columns you want to merge; use the option in the Transform menu to replace the columns and the option in the Add Column menu (see Recipe 15.20) to keep them.

15.16 Pivoting Columns

Problem

You have a query and want to use a column's unique values as headings for new columns and include aggregations of another column for their values.

Solution

Suppose you have a query with columns named Salesperson, Company, and Amount. You want to replace the Company and Amount columns with a new column for each Company value and calculate the sum of the Amount values for each one.

You can achieve this using the Pivot Column option as follows:

1. Select the Company column.

2. Choose Transform ⇒ Any Column ⇒ Pivot Column to open the Pivot Column dialog box.

3. Select Amount from the Values Column drop-down list.

4. Optionally, open the Advanced section to choose a specific aggregation—the default is Sum.

5. Click OK to transform the data.

Figure 15-8 shows an example output from pivoting the data.

ABC Salesperson	$ Starbuzz	$ CatChat	$ Manic Mango
1 Chris Smith	null	null	6,834.74
2 George Doe	6,743.31	89,441.57	null
3 Haley Daley	783.24	8,203.24	7,683.06
4 Julia Smith	345.23	3,957.49	8,294.24
5 Vince Bell	3,987.92	6,753.23	29,879.25

Figure 15-8. Pivoting the Company column, calculating the sum of the Amount for each

Discussion

This recipe shows you how to transform data so it's organized similarly to a basic Excel PivotTable. This can save you from loading the entire dataset into Excel if you want to see only summary values.

An alternative way of calculating aggregations is to use the Transform menu's Group By option (see Recipe 15.14).

15.17 Unpivoting Columns

Problem

You have a query and want to translate column values into attribute-value pairs.

Solution

Suppose you have a query with columns named Starbuzz, CatChat, and Manic Mango, which contain numeric values. You want to replace these with Company and Amount columns so you can more effectively use the dataset with a PivotTable (see Recipe 11.1).

You can solve this problem using the Unpivot Columns option as follows:

1. Select the Starbuzz, CatChat, and Manic Mango columns.

2. Choose Transform ⇒ Any Column ⇒ Unpivot Columns ⇒ Unpivot Only Selected Columns.

3. Power Query replaces the selected columns with one named Attribute (containing the companies) and another named Value (containing the amounts). Name the Attribute column **Company** and the Value column **Amount**.

Figure 15-9 shows the effect of unpivoting the data before renaming the Attribute and Value columns.

	ABC Salesperson	ABC Attribute	$ Value
1	Chris Smith	Manic Mango	6,834.74
2	George Doe	Starbuzz	6,743.31
3	George Doe	CatChat	89,441.57
4	Haley Daley	Starbuzz	783.24
5	Haley Daley	CatChat	8,203.24
6	Haley Daley	Manic Mango	7,683.06
7	Julia Smith	Starbuzz	345.23
8	Julia Smith	CatChat	3,957.49
9	Julia Smith	Manic Mango	8,294.24
10	Vince Bell	Starbuzz	3,987.92
11	Vince Bell	CatChat	6,753.23
12	Vince Bell	Manic Mango	29,879.25

Figure 15-9. Unpivoting the Starbuzz, CatChat, and Manic Mango columns

Discussion

Unpivoting a dataset's columns is sometimes needed to make it easier to use with PivotTables. However, transforming the data with Excel formulas can take time and effort. This recipe shows how to unpivot the data with Power Query, which is generally more straightforward.

15.18 Transforming Structured Columns

Problem

You have a column containing a record, table, or list of data you want to expand.

Solution

A *structured column* contains structured data instead of a specific data type. There are three main types of structured data, as follows:

List

> This stores a list of values.
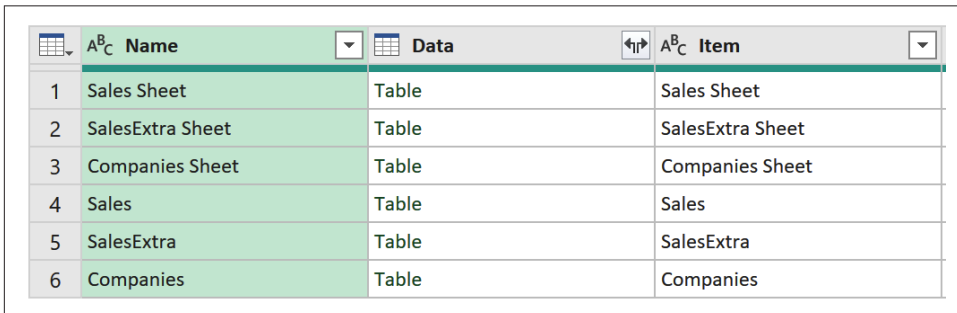
Record

> This stores a set of named fields and has a one-to-one relationship with the query's primary data.

Table

> This stores a table with rows and columns and may have a one-to-many relationship with the query's primary data.

Figure 15-10 shows an example data preview with structured table data in the Data column.

	A^{B}C Name			Data		A^{B}C Item	
1	Sales Sheet			Table		Sales Sheet	
2	SalesExtra Sheet			Table		SalesExtra Sheet	
3	Companies Sheet			Table		Companies Sheet	
4	Sales			Table		Sales	
5	SalesExtra			Table		SalesExtra	
6	Companies			Table		Companies	

Figure 15-10. The Data column holds structured data (a table)

When you get data from a data source, Power Query retrieves it as structured data and automatically expands it into columns. However, in some circumstances—such as when merging queries (see Recipe 15.30)—you need to do this yourself. Do the following to expand structured data:

1. Select the structured column.

2. Choose Transform ⇒ Structured Column ⇒ Expand, or click the Expand button ⇥ to the right of the column heading and select the Expand option.

3. Check any columns you want to add to the query, uncheck any you don't, and (optionally) type a prefix for the column names.

4. Click OK to add the columns.

You can also include aggregations of table data as follows:

1. Select the structured column.

2. Choose Transform ⇒ Structured Column ⇒ Aggregate, or click the Expand button ⇥ to the right of the column heading and select the Aggregate option.

3. Check any columns you want to include aggregations for.

4. Specify the aggregations for each column by hovering your cursor over each one, clicking the drop-down arrow on the right, and then checking the options you want to display.

5. Click OK to add the aggregations.

Discussion

This recipe provides an overview of how to transform structured columns. You can use these techniques if, for example, you've merged two queries and want to expand the second one's data into columns.

15.19 Returning a Value or List

Problem

You have a query and want to use it to return a list of values or a single value.

Solution

Suppose you have a query with a Salesperson column and you want to transform it into a *list*: a sequence of values that you can load to Excel or that other queries can refer to—for example, to filter another query (see Recipe 15.31). You can do so by selecting the column and choosing Transform ⇒ Any Column ⇒ Convert to List.

Once you've created the list, you can shape it using the options in the List Tools ⇒ Transform menu. For example, you can remove duplicate values, sort them, or use them to perform statistical operations.

A query can also return a single value, such as a statistical summary, the number of rows in the query, or the value in a specific cell. To return a statistic, select the column you want to calculate it from, choose Transform ⇒ Number Column ⇒ Statistics, and select the statistic you wish to return. To return the number of rows, choose Transform ⇒ Table ⇒ Count Rows to return the number of rows. Finally, you can return the value in a specific cell by right-clicking it and choosing Drill Down.

Discussion

This recipe shows how to transform a query so it returns a list of values, along with techniques for returning a single value. Returning a value or list from a query is helpful if, for example, you want to derive one or more values that other queries can refer to and use (see Recipe 15.23).

An example of when this might be useful is for filtering a query using the value in an Excel cell. You can use the Drill Down technique in this recipe to get the cell value, then use Recipe 15.31 to filter by that value.

15.20 Adding New Columns

Problem

You have a query and want to add new columns to it.

Solution

The Power Query Editor's Add Column menu includes many options to add new columns while keeping the original ones intact.

The From Text, From Number, and From Date & Time groups contain similar options to the Transform menu's Text Column, Number Column, and Date & Time Column groups (see Recipe 15.14). The main differences are as follows:

- Using the options in the Add Column menu keeps the original columns, while those in the Transform menu replace them.
- You can use the Statistics options in the From Number group only if you select multiple columns—for example, to calculate their average for each row.
- You can apply mathematical operations to multiple columns using the Standard options in the From Number group. For example, if you select two number columns and choose Add Column ⇒ From Number ⇒ Standard ⇒ Add, Power Query adds a new column containing the sum for each row.
- The Date, Time, and Duration groups in the From Date & Time section include extra Subtract options, which you can use to calculate the difference between two date/time or duration columns. For example, you can use it to calculate the difference in days between two dates.

The General group includes options for adding columns as follows:

Duplicate Column
    This copies the currently selected column, including its values.

Index Column
    This adds a new column containing index values, which you can use to support other transformations based on a row's position. Choose From 0 or From 1 to make the index values start from 0 or 1, and choose Custom to specify the starting value and increment.

Column From Examples

This option adds a column similar to Flash Fill in Excel (see Recipe 15.21).

Conditional Column

This creates a column using one or more criteria (see Recipe 15.22).

Custom Column

This lets you add a new column by writing Power Query M code (see Recipe 15.23).

Invoke Custom Function

This adds a new column by invoking a custom function you've created (see Recipe 15.26).

Discussion

This recipe provides an overview of the Power Query Editor's Add Column menu options. Generally, these options add new columns and keep the original ones intact; if you want to replace them, you can either manually delete them after adding the new column or use the options described in Recipe 15.14.

15.21 Adding a Column Based on Examples

Problem

You have a query and want to add a new column by typing examples that Power Query can try to pattern match.

Solution

Suppose you have a Salesperson column containing first names and last names, and you want to create a new column that displays the names in the format *last name, first name*. For example, *John Smith* becomes *Smith, John*; *Jane Doe* becomes *Doe, Jane*; and so on.

You can achieve this using Column From Examples. This feature works similarly to Flash Fill (see Recipe 1.14) in Excel; you provide examples of the results you want for the first few rows, and if Power Query spots a pattern, it completes the rest of the rows.

You use Column From Examples as follows:

1. To pattern match using a specific column or columns—in this example, the Salesperson column—select the columns and choose Add Column ⇒ General ⇒ Column From Examples ⇒ From Selection. To pattern match using all columns

instead, choose Add Column ⇒ General ⇒ Column From Examples ⇒ From All Columns. Whichever option you choose, Power Query inserts a new column.

2. Type a few example values in the new column. For example, if you selected the Salesperson column in step 1 and the values in the first two rows are *John Smith* and *Jane Doe*, you'd type **Smith, John** in the first row and **Doe, Jane** in the second.

3. When Power Query spots a pattern, it suggests values for the remaining rows. If these suggestions are incorrect, add more example values until it spots the correct pattern. You can also edit an individual value by double-clicking its cell.

4. When Power Query correctly identifies the pattern, press Ctrl+Enter or click OK to accept the values.

Figure 15-11 shows an example of using Column From Examples with example values in the Custom column's first three rows and suggestions for the remaining rows.
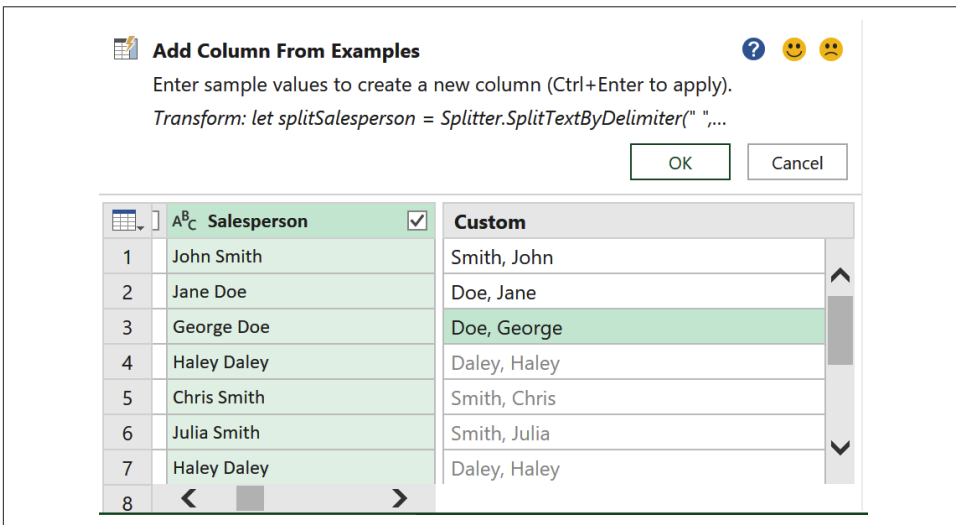
Figure 15-11. Using Column From Examples

Discussion

Adding a column using example values can be convenient if transforming the data will take multiple steps or if you need help deciding which option to use. This recipe provides an overview of using this feature.

15.22 Adding a Conditional Column

Problem

You have a query and want to add a new column whose values depend on meeting one or more conditions.

Solution

Suppose you have a query with a column named Company, and you want to add a new column whose value is 1 for rows where the company is *Starbuzz*, 2 where it's *CatChat*, and 3 for all other values. You can achieve this by adding a *conditional column*, which uses criteria to determine values.

To add the column:

1. Choose Add Column ⇒ General ⇒ Conditional Column to open the Add Conditional Column dialog box.

2. Type the name of the new column in the "New column name" box (for example, **Company Code**).

3. Use the If row in the dialog box to specify the first condition and the corresponding value you want to add to the new column. So, in this example, choose Company from the Column Name drop-down box, Equals from the Operator drop-down box, then type **Starbuzz** in the Value box and **1** in the Output box.

4. To add the second condition, click Add Clause to add a new Else If row. Choose Company from the Column Name drop-down box, Equals from the Operator drop-down box, then type **CatChat** in the Value box and **2** in the Output box.

5. Repeat step 4 to add any more conditions.

6. Type the value you want to return if no conditions are met in the Else box. In this example, you'd type **3**. Then click OK to add the new column.

Figure 15-12 shows the Add Conditional Column dialog box.

> If you want to delete or move a clause, hover your cursor over it in the Add Conditional Column dialog box, click the ellipsis (…) that appears to the right of the row, and then select the appropriate option.
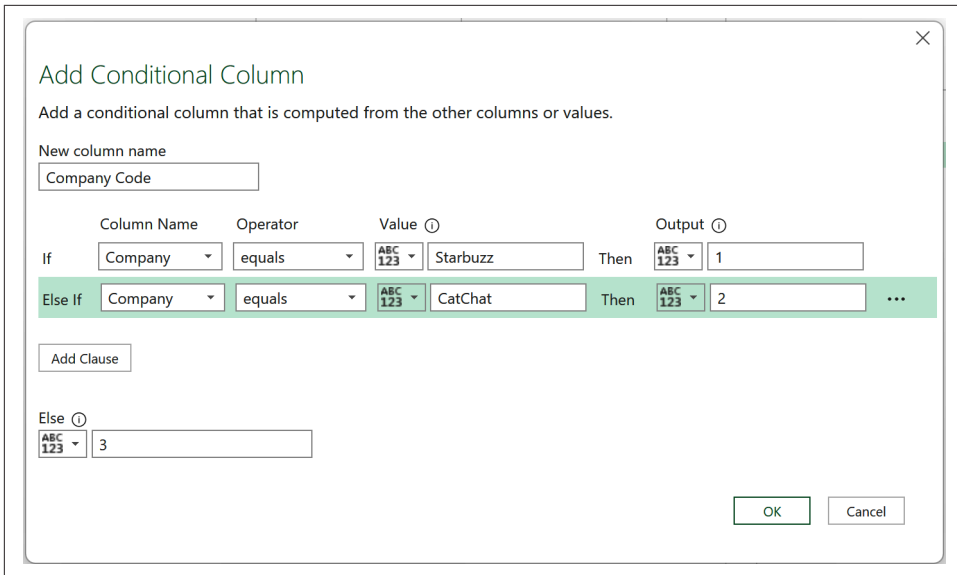
Figure 15-12. The Add Conditional Column dialog box

You can also use a conditional column to compare the values in two columns or return a value from a column when a condition is met. For example, you can compare the dates in two date columns and return the earliest date. To compare two columns, choose the name of the first column from the Column Name drop-down box, choose "Select a column" from the Value drop-down box, and then select the name of the second column. To return a column value, choose "Select a column" from the Output drop-down box, then select the column's name.

Discussion

A conditional column offers a flexible way of inserting a column whose values depend on criteria. In some ways, it works similarly to using the IF or IFS Excel functions to build criteria (see Recipe 7.5).

The examples in this recipe describe constructing criteria using constant or column values. A third option is to use parameter values (see Recipe 15.24) to learn more about these.

15.23 Adding a Custom Column

Problem

You have a query and want to add a column that uses a formula written in the Power Query M formula language.

Solution

Suppose you want to add a new column to a query, but the transformation you want to apply isn't available from the Add Column menu. In this situation, you can use a *custom column*: a column that uses a formula to derive a value for each row.

You generally add a custom column as follows:

1. Choose Add Column ⇒ General ⇒ Custom Column to open the Custom Column dialog box.

2. Type a name for the new column in the "New column name" box.

3. Type the formula in the "Custom column formula" box.

4. Click OK to insert the column.

5. Make sure the new column's data type is correct, and change it if needed (see Recipe 15.10).

Figure 15-13 shows the Custom Column dialog box with some example code.
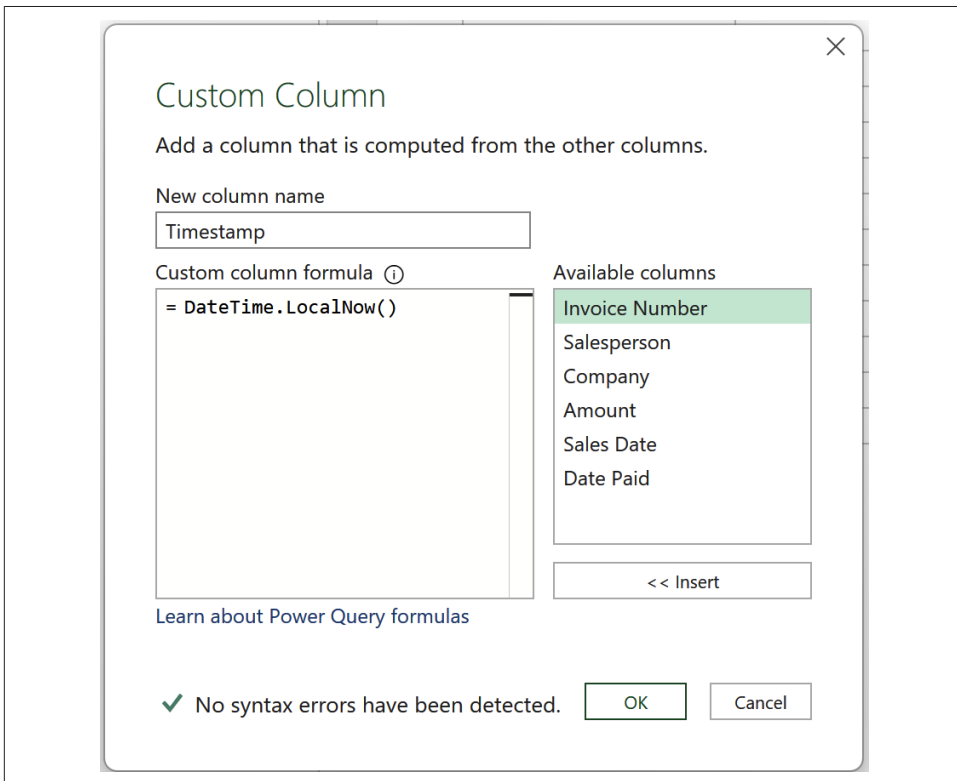
Figure 15-13. The Custom Column dialog box

You write the custom column's formula using the Power Query M formula language. M is a case-sensitive language that you can use to refer to queries, columns, and parameters and use operators and M functions to return the desired results. Some examples are as follows:

`DateTime.LocalNow()`

This puts the current date and time in the column. It's the M equivalent of using `NOW()` in Excel.

`[Amount]*Rate`

This multiplies the values in the Amount column by a parameter named Rate (see Recipe 15.24).

`[Salesperson]=SalespersonName`

This checks whether each value in the Salesperson column equals a query named SalespersonName, which returns a single value (see Recipe 15.19).

`List.Contains(SalespersonNames, [Salesperson])`

This checks whether SalespersonNames—a query that returns a list (see Recipe 15.19)—contains the value in the Salesperson column.

You can add a column to the formula using the "Available columns" list in the Custom Column dialog box; double-click the column name you wish to add, or select it and click Insert.

> Unlike Excel formulas, the Power Query M formula language is case-sensitive. Ensure that you type any formulas correctly, or the code will display a syntax error.

Discussion

This recipe provides an overview of adding a column to a query that uses the Power Query M formula language. This can be helpful for transformations that Power Query doesn't provide out of the box.

To learn more about using the Power Query M formula language, click the "Learn about Power Query formulas" link in the Custom Column dialog box.

See Also

If you want to write a more complex formula or one you might want to reuse in multiple queries, consider writing a custom function; see Recipes 15.25 and 15.26.

15.24 Using Parameters

Problem

You want to assign a name to a constant value so you can refer to it by name in a conditional column or code or use it to filter rows.

Solution

Suppose you have a value—for example, a rate or text string—that you want to use in several columns or queries. You want to be able to refer to the value by a name instead of hardcoding it, so if the value changes, you don't have to update each query.

You can achieve this by adding the value as a *parameter*: a constant value, which you can refer to by name in conditional columns, filters, custom columns, and code. You create one as follows:

1. Choose Home ⇒ Parameters ⇒ Manage Parameters ⇒ New Parameter to open the Manage Parameters dialog box.
2. Type the name of the parameter in the Name box—for example, **Rate**.
3. Optionally, type a description for the parameter in the Description box.
4. Place a check in the Required check box; you should uncheck this check box only if the parameter's value can be left blank.
5. Use the Type drop-down box to choose the type of data (see Recipe 15.10) the parameter should return—for example, Decimal Number.
6. The Suggested Values option specifies whether you want Power Query to suggest values for the parameter if you decide to edit it. Choose Any Value if you don't want Power Query to suggest values (as in this example), choose List of Values to suggest values from a hardcoded list (which you specify, type the values, then specify the default value). If you want to suggest values from a query, set Suggested Values to Query, then select the query; this must be a query that returns a list.
7. Type the parameter's value in the Current Value box (for example, **0.25**)—you must complete this box if you checked the Required check box in step 4. Then click OK to create the parameter.

Figure 15-14 shows the Manage Parameters dialog box with an example parameter.
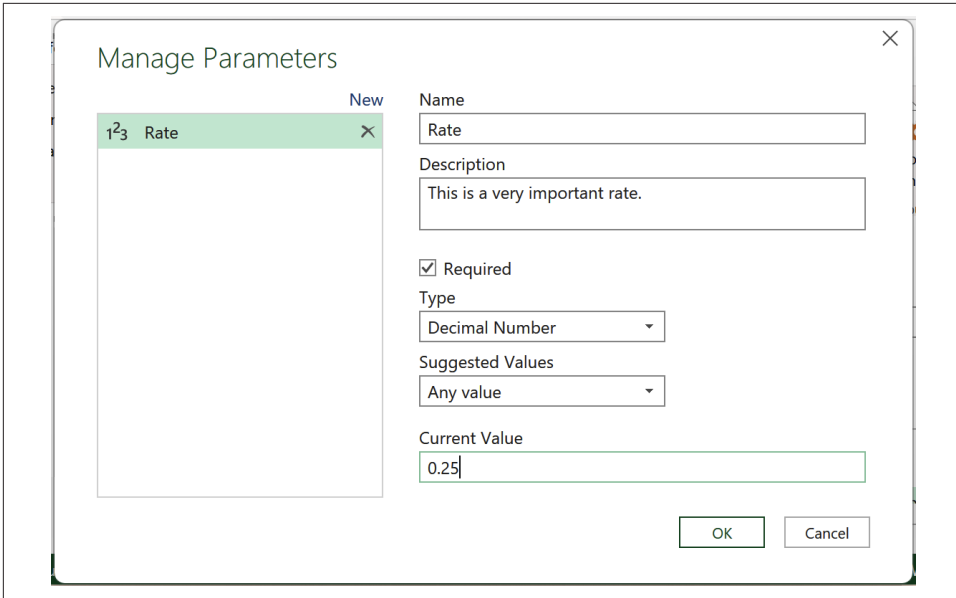
Figure 15-14. The Manage Parameters dialog box

Once you've created the parameter, you can refer to it in conditional columns, filters, custom columns, and code. To use the parameter in a conditional column, choose Parameter from the Value or Output drop-down box, then select the parameter's name. Similarly, you can filter using a parameter value by choosing Parameter in the Filter Rows dialog box. To refer to a parameter in a custom column or code, type or select the parameter name.

When you've added one or more parameters, you can edit their values by choosing Home ⇒ Parameters ⇒ Manage Parameters ⇒ Edit Parameters. You can either type the new value for each parameter you want to change or select the value from the drop-down list if its Suggested Values option is List of Values or Query. You can also edit or delete parameters by choosing Home ⇒ Parameters ⇒ Manage Parameters.

Discussion

Using parameters is a convenient way of referring to constant values by name in conditional columns, filters, custom columns, functions, and code. They save you from hardcoding values, making queries that use them more maintainable.

A limitation of using parameters is that you can't use one to store the result of a calculation or a summary value returned by a query. However, you can refer to the query name using custom columns, functions, or code (see Recipes 15.23, 15.25, and 15.30).

15.25 Creating a Custom Function

Problem

You want to write a function using the Power Query M formula language.

Solution

Suppose you have a formula that calculates commissions by multiplying amounts greater than 1,000 by 0.1. You want to use this formula in multiple queries or apply it to different columns, and you can do so by writing a custom function with the Power Query M formula language (see Recipe 15.23).

You create a custom function as follows:

1. Create a blank query by choosing Data ⇒ Get & Transform Data ⇒ Get Data ⇒ From Other Sources ⇒ Blank Query in Excel, or Home ⇒ New Query ⇒ New Source ⇒ Other Sources ⇒ Blank Query in the Power Query Editor.

2. Name the function by changing the query's name to that of the function. So, in this example, you'd rename the query *Commission* to define a function named `Commission`.

3. Choose View ⇒ Advanced ⇒ Advanced Editor to open the Advanced Editor window (see Figure 15-15).
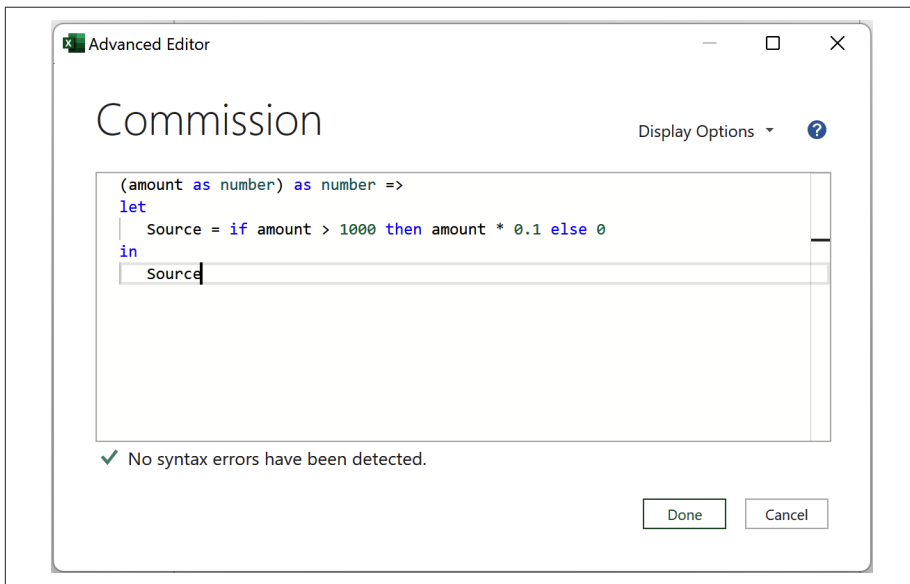
Figure 15-15. A custom function

4. Type the function's code in the window's code box. For example, to calculate the commission by multiplying amounts greater than 1,000 by 0.1, you'd type the code listed in Example 15-1.

5. When you've finished typing the code, click Done.
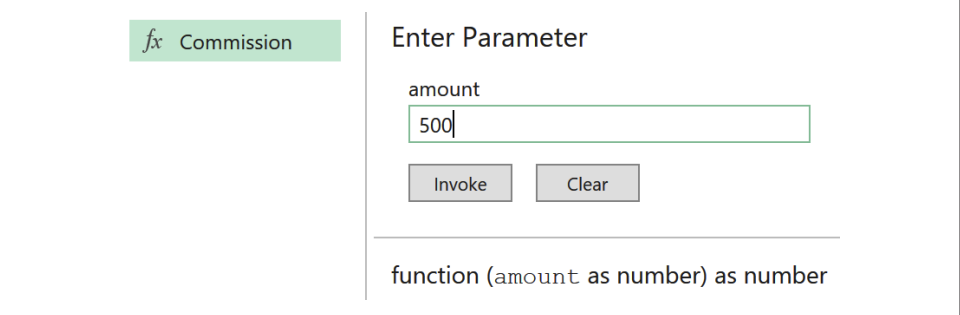
Example 15-1. Code for the `Commission` function

```
(amount as number) as number =>
let
    Source = if amount > 1000 then amount * 0.1 else 0
in
    Source
```

This code works as follows:

- The first line, `(amount as number) as number =>`, defines a number argument named `amount` and specifies that the function returns a number.

- The `let` expression in line 2 specifies any calculations the formula needs to perform—in this example, the calculation in line 3.

- Line 3 calculates the commission by multiplying the `amount` argument by 0.1 if it's greater than 1,000. The code then assigns the result to a variable named `Source`.

- The `in` expression in line 4 specifies the function's return value—in this example, the value of the `Source` variable in line 5.

Once you've created the function, you can test it with different values to ensure it returns the correct result. To do so, select the query in the Power Query Editor's query list, enter the value of any arguments (in this example, the amount), and click Invoke to see the result. For example, you could test the `Commission` function by invoking it with values of 500 and 5,000 and make sure it returns values of 0 and 500, respectively (see Figure 15-16).

Figure 15-16. Invoking a custom function

Each time you click the Invoke button, Power Query creates a new query that calls the function with the specified arguments and returns the result. You can delete these queries once you're satisfied the function works correctly. If you don't, Power Query will load their results into Excel the next time you close the Power Query Editor and load the data.

Discussion

This recipe introduces you to creating custom functions using the Power Query M formula language. Once you've defined a function, you can call it by following Recipe 15.26 or using M code (see Recipes 15.23 and 15.30).

15.26 Adding a Column by Invoking a Custom Function

Problem

You have a custom function and want to use it to add a new column to a query.

Solution

Suppose you've used Recipe 15.25 to define a custom function named `Commission` and you want to use it to calculate the commission for each row in the Amount column and put the results in a new column. You can do so as follows:

1. Select the query you want to add the column to and choose Add Column ⇒ General ⇒ Invoke Custom Function to open the Invoke Custom Function dialog box.

2. Type a name for the new column in the "New column name" box—for example, **Commission Earned**.

3. Select the name of the custom function from the "Function query" drop-down list. So, in this example, you'd select Commission.

4. Specify the values or column names to use for any arguments. In this example, you'd select Column Name from the amount argument's drop-down list and then select the Amount column.

5. Click OK to insert the column.

Figure 15-17 shows an example of using the Invoke Custom Function dialog box to create a column that invokes the `Commission` function, passing it the Amount column.
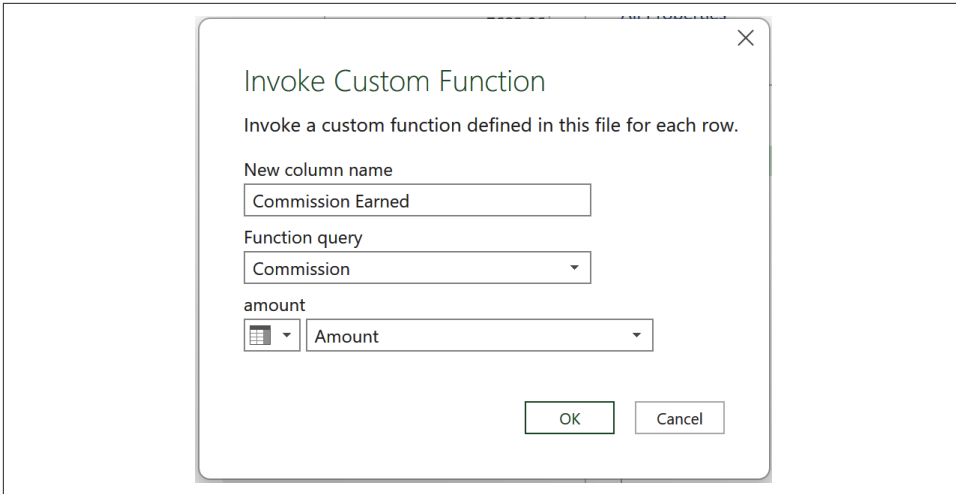
Figure 15-17. Using the Invoke Custom Function dialog box to add a new column

Discussion

This recipe shows how to add a new column by invoking a custom function you've previously defined. It offers a convenient alternative to inserting a custom column (see Recipe 15.23) and manually invoking the function in the column's code.

15.27 Duplicating a Query

Problem

You want to make an identical copy of a query and (optionally) apply its steps to another data source.

Solution

Suppose you have a query that gets data from an Excel workbook and transforms its data. You want to copy the query to apply the same transformations to the data in another workbook with the same schema.

You can solve this problem by duplicating the query and adjusting its steps.

To duplicate the query, select it in the Power Query Editor's query list and choose Home ⇒ Query ⇒ Manage ⇒ Duplicate, or right-click it and choose Duplicate. Then, rename the new query.

Once you've duplicated the query, you can change the new query's data source by editing its Source and Navigation steps. Editing the Source step lets you select a different filepath (by browsing to the file or using a parameter value) and optionally

change the file type. Editing the Navigation step lets you choose the resource in the file to get data from.

Discussion

Duplicating a query can be helpful if you want to make a copy before making changes or apply a query's transformations to another data source. Instead of re-creating the query from scratch, you can copy it and edit its data source.

15.28 Referencing a Query

Problem

You want to create a new query based on the steps in another query.

Solution

Suppose you have a query that transforms data. You want to create a second query that applies all the steps in the first and then includes extra transformations.

In this situation, you can create a new query that references the original so it uses the original query as its source. Doing so avoids having copies of the same steps in different places, making the queries more maintainable.

To create the new query, select the query you want to reference and choose Home ⇒ Query ⇒ Manage ⇒ Reference, or right-click it and choose Reference. Then, rename the new query.

If you want to create a new query that references the original query's first few steps, you can extract them into a new query, which you can reference. You do so as follows:

1. Select the query whose steps you want to extract.
2. In the Query Settings pane, right-click the step immediately after the ones you want to refer to and then select Extract Previous; this opens the Extract Steps dialog box.
3. Type a name for the new query to which you're extracting the steps, then click OK; this creates a new query containing the steps, which the original query—the one you selected in step 1—now references.
4. Select the new query you created in step 3, then choose Home ⇒ Query ⇒ Manage ⇒ Reference to create a separate query that references it. You can now use this query to apply more transformations.

To see the dependencies between queries, choose View ⇒ Depen-
dencies ⇒ Query Dependencies. This option displays a diagram
showing every query, their data sources, and any dependencies.

Discussion

This recipe is handy if you want to create multiple queries based on the same data
source and apply a standard set of transformations to each one. Doing so helps pre-
vent maintaining copies of the same steps in multiple queries.

15.29 Appending Data from Multiple Queries

Problem

You want to create a new query that includes all rows from two or more other quer-
ies.

Solution

Suppose you have two queries with matching column names, and you want to create
a new query that returns all the rows from both queries. You can do this by append-
ing the data in a new query as follows:

1. Choose Home ⇒ Combine ⇒ Append Queries ⇒ Append Queries as New to
   open the Append dialog box.
2. Select the names of the two queries in the "First table" and "Second table" boxes.
3. If you want to append the data from more than two queries, select the "Three or
   more tables option" and add the rest of the queries.
4. Click OK to create the new query, which you can now rename.

Figure 15-18 shows an example of using the Append dialog box to create a new query
that appends the data from two queries named Sales and SalesExtra.
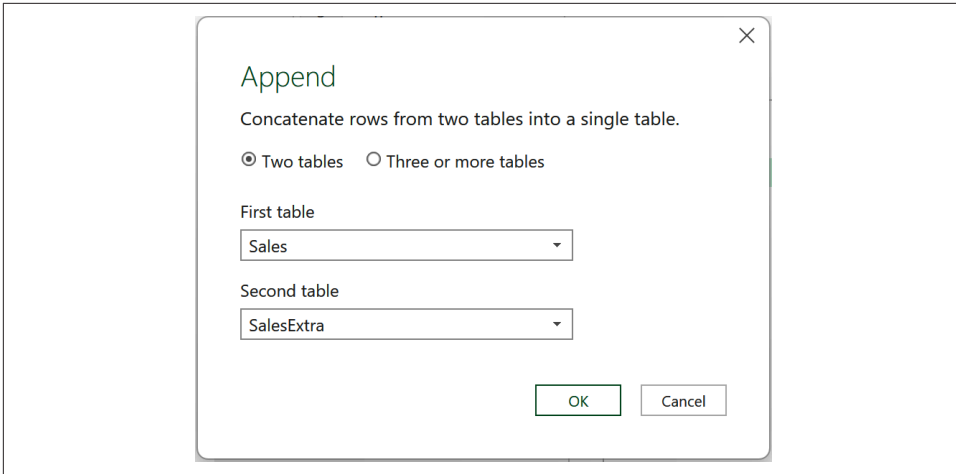
Figure 15-18. Using the Append dialog box

The new query contains every row and column from the queries you select in the order in which you select them. It matches columns with the same name and puts null values in any columns it can't match.

Discussion

This recipe shows how to append the data from multiple queries, creating a new query. For example, you can use this approach to append rows from multiple data sources or different file types.

You can also append data to an existing query. To do so, select the query to which you want to add rows and choose Home ⇒ Combine ⇒ Append Queries.

See Also

If you want to create a query that appends the data from files in a folder, see Recipe 15.2.

To join related data in two queries, see Recipe 15.30.

15.30 Merging Data from Multiple Queries

Problem

You want to create a new query that joins two related queries.

Solution

Suppose you have two queries named Sales and Companies, both of which include a Company column. You want to create a new query containing all the columns and rows from the Sales query and add company information from the Companies query where the rows match.

You can achieve this by merging the queries as follows:

1. Choose Home ⇒ Combine ⇒ Merge Queries ⇒ Merge Queries as New to open the Merge dialog box.

2. Select the first query you want to merge from the first drop-down list—in this example, the Sales query. This displays a preview of the query's rows and columns.

3. Select the second query you want to merge from the second drop-down list—in this example, the Companies query. This displays a preview of the second query.

4. Select the columns you want to use to match or join the data. In this example, you select the Company column from both queries because you want to match the data in both queries using this column.

5. Select an option from the Join Kind drop-down box that describes how you want to match the data. For example, the "Left Outer (all from the first, matching from the second)" option includes every row from the first query (Sales) and any matching rows from the second (Companies).

6. Optionally, specify whether you want to merge the data by finding an exact match (the default) or using fuzzy matching.

7. Click OK to create the query.

Figure 15-19 shows the Merge dialog box.

The new query includes all the columns from the first query and an extra one for the second query's data. This final column contains structured data, which you can transform using Recipe 15.18.
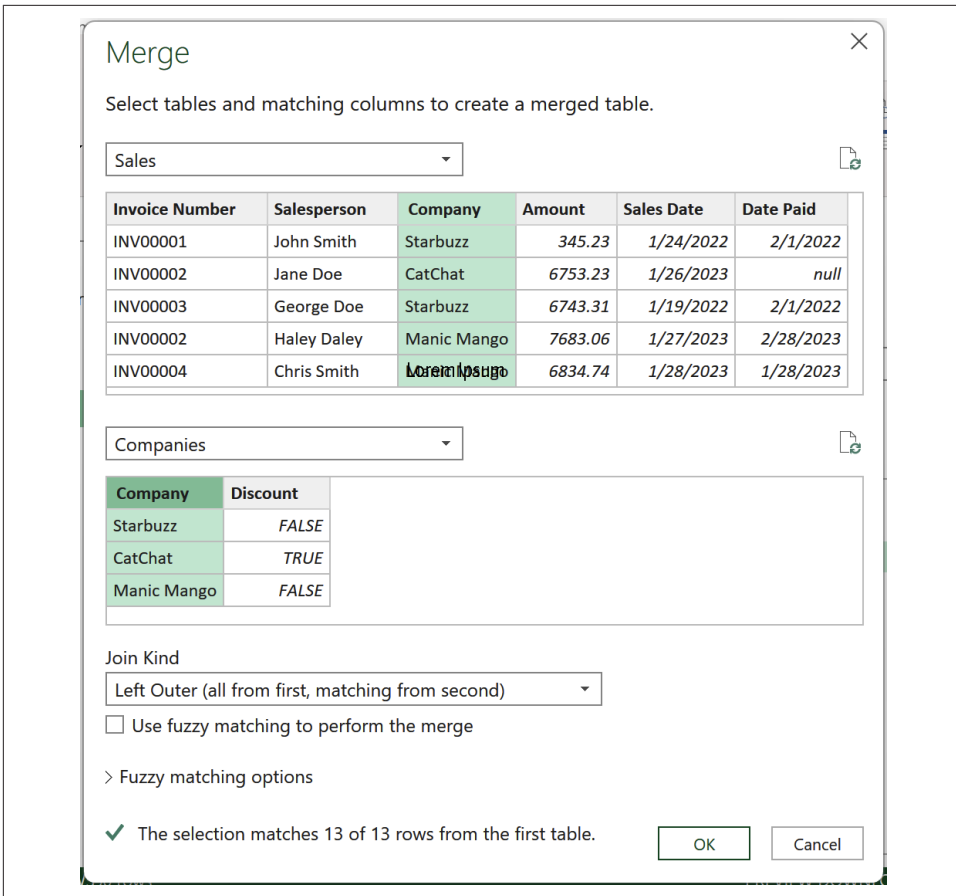
Figure 15-19. Using the Merge dialog box

Discussion

This recipe shows how to combine queries by merging them. This approach is helpful if you have data in two queries you want to join based on matching values.

You can also merge data to an existing query. To do so, select the query you want to use as the primary query and choose Home ⇒ Combine ⇒ Merge Queries.

See Also

To append the data from two or more queries, see Recipe 15.29.

15.31 Editing a Query's M Code

Problem

You have a query and want to view or edit the underlying code for the entire query.

Solution

Behind the scenes, Power Query generates M code for every query you create and its steps. You can see the code for a single step by selecting the step and expanding the formula bar, and view the query's entire code by choosing View ⇒ Advanced ⇒ Advanced Editor. For example, if you have a step that filters a Salesperson column to return only rows for *John Doe*, the code in the formula bar might be as follows:

```
= Table.SelectRows(#"Changed Type", each ([Salesperson] = "John Doe"))
```

In some situations, you may need—or prefer—to edit the query's M code instead of using the Power Query Editor's commands. For example, suppose you want to change the Salesperson filter to use the value returned by a query named Salesperson-Name instead of hardcoded text. In this case, you'd replace "John Doe" with the name of the query as follows:

```
= Table.SelectRows(#"Changed Type", each ([Salesperson] = SalespersonName))
```

Similarly, to change the filter so it refers to a list of values returned by a query named SalespersonNames, you'd update the code as follows:

```
= Table.SelectRows(#"Changed Type",
                   each List.Contains(SalespersonNames, [Salesperson]))
```

When you've finished updating the code, click Done if you've edited it in the Advanced Editor window, or press Enter/Return if you've used the formula bar.

Discussion

This recipe provides an overview of how to edit a query's code instead of using the Power Query Editor's command. Behind the scenes, each query is written in the Power Query M formula language.

See Also

See Recipes 15.23 and 15.25 for other examples that use the Power Query M formula language.

Power Pivot and the Data Model

By default, you can base a PivotTable on only a single table, which is problematic if your data is in multiple related tables. Furthermore, Excel's row limits can make handling large datasets tricky.

If you're using Excel for Windows, a solution to these limitations is to use the *data model*: a collection of related tables saved with your workbook. Using the data model has the following advantages:

- You can create a PivotTable that uses data from multiple related tables.
- There's no row limit; you can import millions of rows from multiple data sources.
- Using the data model is faster for large datasets because of its efficient compression algorithms.

The recipes in this chapter show how to work with Excel's data model using Power Pivot: an add-in that interfaces with the data model. Areas covered include adding data to the data model; defining relationships between tables; creating calculated columns, measures, and key performance indicators (KPIs); and using PivotTables and Cube formulas to retrieve and analyze data-model data.

16.1 Installing Power Pivot

Problem

You're using Excel for Windows and want to install the Power Pivot add-in so you can interact with Excel's data model.

Solution

If you're using Excel for Windows, you can use the Power Pivot add-in to interact with the workbook's data model. You enable Power Pivot as follows:

1. Choose File ⇒ Options.

2. Select Add-ins.

3. In the Manage box at the bottom of the screen, choose the COM Add-ins option and click Go.

4. In the COM Add-ins dialog box, place a check in the Microsoft Power Pivot for Excel check box and click OK.

Discussion

Power Pivot brings extra features to Excel that let you interact with the workbook's data model and perform more advanced data analyses. See the recipes in the rest of this chapter for more details on how to use this powerful tool.

16.2 Adding Data to the Data Model

Problem

You have data you need to analyze and want to add it to the workbook's data model.

Solution

In most situations, the best way to add data to the data model is using Power Query, which lets you clean and transform the data before importing it. Generally, use Recipe 15.1 to get the data, choose the Load To option to open the Import Data dialog box (see Recipe 15.3), then place a check in the "Add this data to the Data Model" check box and click OK.

If you want to load a table in the current workbook to the data model without transforming its data, choose Power Pivot ⇒ Tables ⇒ Add to Data Model. Doing so adds the table to the data model and opens the Power Pivot window, as shown in Figure 16-1.
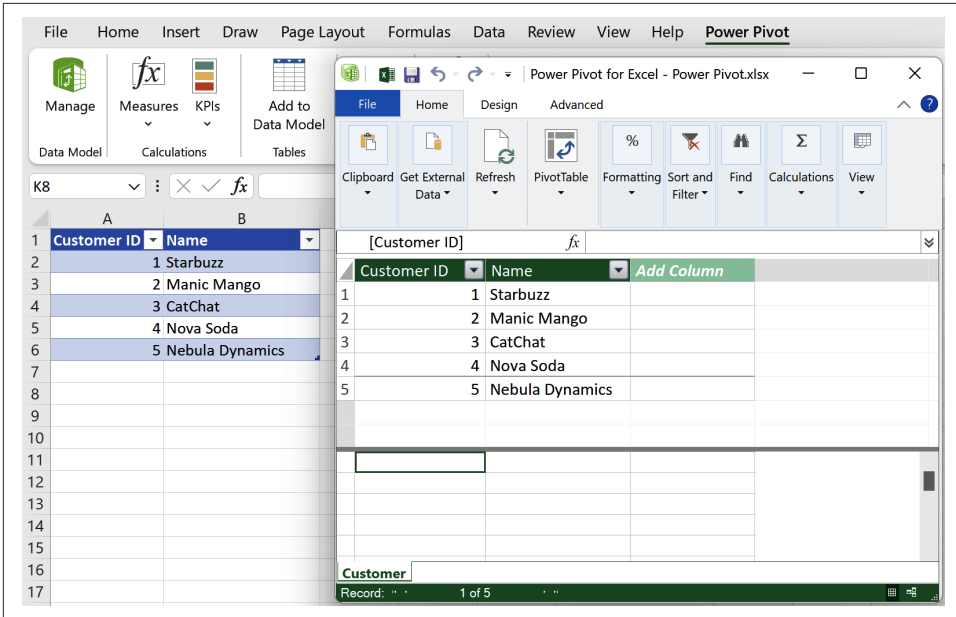
Figure 16-1. Adding an Excel table to the data model

You can also use Power Pivot's import tools to load data directly to the data model without adding it to a worksheet. Choose Power Pivot ⇒ Data Model ⇒ Manage to open the Power Pivot window, choose Home ⇒ Get External Data, then select one of these available options:

From Database
> This includes SQL Server, Access, Analysis Services, and Power Pivot options.

From Data Service
> This lets you import data from an OData (Open Data Protocol) data feed.

From Other Sources
> This includes many options, including Oracle, Teradata, Sybase, Informix, and IBM DB2.

Existing Connections
> This lets you select an existing connection.

Once you've selected an option, follow the wizard's instructions to import the data.

Discussion

This recipe explains how to add data to the workbook's data model. It's best to use Power Query for this task because it lets you clean and transform the data before importing it. Once it's imported, you can then model the data using Power Pivot.

16.3 Managing Power Pivot Data Connections

Problem

You've used Power Pivot to add data to the data model and want to edit its connection to the source data.

Solution

When you add data to the data model using the Power Pivot import options (see Recipe 16.2), Power Pivot creates a connection to the data source. You can view these connections by choosing Power Pivot ⇒ Data Model ⇒ Manage to open the Power Pivot window, then choosing Home ⇒ Get External Data ⇒ Existing Connections to open the Existing Connections dialog box (see Figure 16-2).
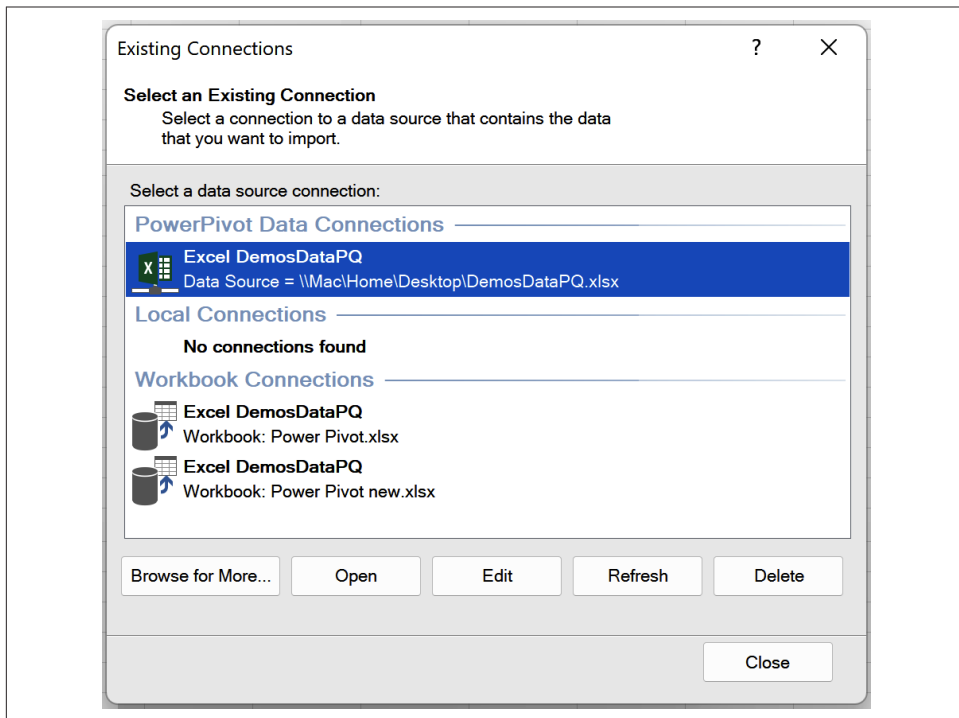
Figure 16-2. The Existing Connections dialog box

The Existing Connections dialog box lets you open a connection to reenter the table import wizard, edit the connection to change its name and other settings, refresh the data, or delete the connection. To do so, select the connection and choose the appropriate option.

You can also see a list of connections in Excel's Queries & Connections pane, including one for the data model. To see these connections, choose Data ⇒ Queries & Connections ⇒ Queries & Connections, then select the Connections option (see Figure 16-3). You can then view or edit a connection's properties by right-clicking it and choosing the Properties option.
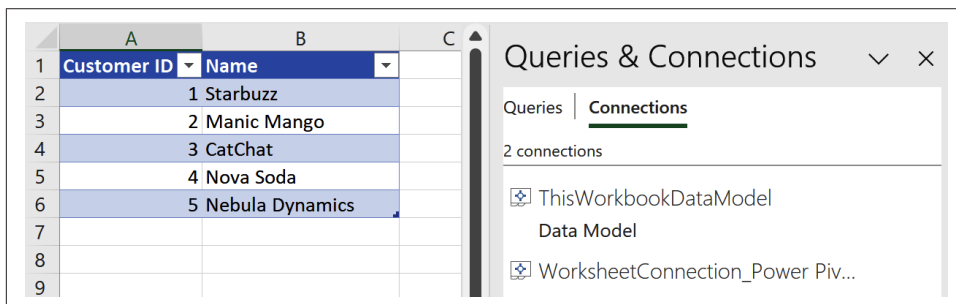
Figure 16-3. The Queries & Connections pane

Discussion

This recipe shows how to manage connections to data sources created using Power Pivot's import tools. For data added using Power Query, use Recipe 15.4 to change the query settings instead.

16.4 Viewing and Managing the Data Model's Tables

Problem

You've added data to the data model and want to view, rename, or delete its tables.

Solution

Importing data to the data model adds one or more tables, which you can view and manage using Power Pivot's data or diagram view.

To use the data view, choose Power Pivot ⇒ Data Model ⇒ Manage to open the Power Pivot window, and then choose Home ⇒ View ⇒ Data View (if it's not already selected). The Power Pivot window displays each table's columns and data, using a separate tab for each table (see Figure 16-4).
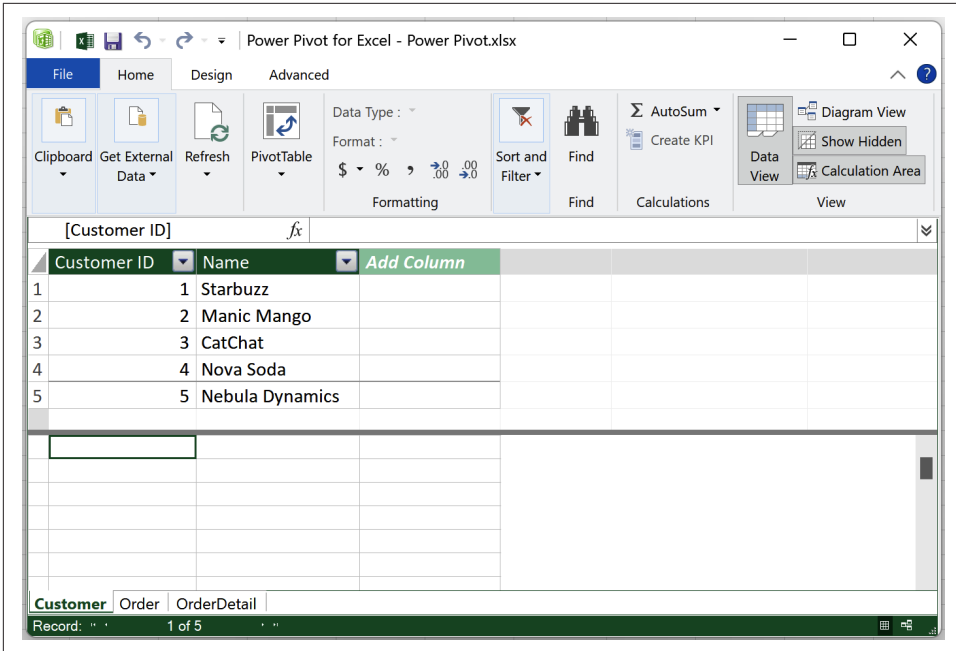
Figure 16-4. Power Pivot's data view

Right-clicking a table's tab gives you a set of options as follows:

Delete
> This lets you delete a table from the data model. If you've added the data using Power Query, you must remove it by editing the query instead (see Recipe 15.3).

Rename
> Use this option to rename the table.

Move
> This lets you rearrange the tabs.

Description
> Use this to add a description of the table.

Hide/Unhide from Client Tools
> Use these options to control whether the table is hidden from tools such as Pivot-Tables. You can control whether to show hidden tables in Power Pivot by choosing Home ⇒ View ⇒ Show Hidden.

Show Calculation Area
> This controls whether the table's calculation area is visible. You can also control this by choosing Home ⇒ View ⇒ Calculation Area.

To use the diagram view, choose Power Pivot ⇒ Data Model ⇒ Manage to open the Power Pivot window, and then choose Home ⇒ View ⇒ Diagram View, as shown in Figure 16-5. This view displays each table (with its columns) as a separate block, and you can rearrange these by clicking and dragging. You can rename a table by double-clicking its label, delete a table by selecting it and pressing the Delete key, and right-click a table to display a menu of options similar to the ones available in the data view.
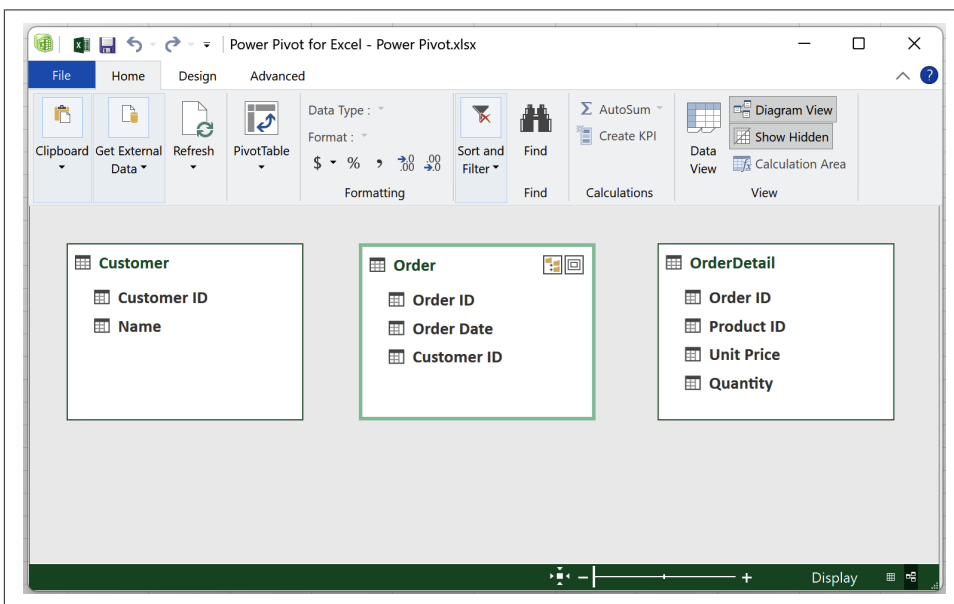
Figure 16-5. Power Pivot's diagram view

Discussion

Once you've added data to the data model, you can view its tables, rename them, or remove them from the data model when you no longer need them. This recipe shows how to do this using Power Pivot's data and diagram views.

16.5 Refreshing the Data Model's Data

Problem

You've added data to the data model and want to refresh it to include changes to the underlying data.

Solution

The data model uses a snapshot of the underlying data, so if the underlying data changes, you need to refresh the data in the data model.

You can refresh the entire data model by choosing Data ⇒ Queries & Connections ⇒ Refresh ⇒ Refresh All or opening the Power Pivot window and choosing Home ⇒ Refresh ⇒ Refresh All. You can also use Recipe 15.5 to refresh any data added using Power Query or refresh the data added using a Power Pivot connection by hovering your cursor over the connection in Excel's Queries & Connections pane and then clicking its Refresh icon.

Similarly to Recipe 15.6, you can make Excel refresh specific connections every hour or when you open the workbook. To do so, right-click the connection in the Queries & Connections pane (see Recipe 16.3), choose the Properties option, select the Usage tab, and then use the settings in the Refresh Control section to specify when you want the data to refresh.

Discussion

This recipe outlines how to refresh the data in the data model. The most straightforward approach is to use Refresh All to refresh all the data, including PivotTables and queries. However, if needed, you can refresh the queries and connections individually.

16.6 Working with Table Columns

Problem

You have a table in the data model and want to rename a column, hide or delete it, change its data type, or apply a filter.

Solution

Suppose you've added a table to the data model and want to change its columns. You can do so using Power Pivot's data view (see Recipe 16.4).

The data model controls the type of data a column can contain using a data type. To view or change a column's data type, select the column by clicking it and choose Home ⇒ Formatting ⇒ Data Type; the available options (depending on the data in the column) include Text, Date, Decimal Number, Whole Number, Currency, and TRUE/FALSE.

Format the column's data using the options in the Home ⇒ Formatting group. Depending on the column's data type, you can apply a percentage or currency format, specify the number of decimal places, choose a specific date format, and so on.

To sort or filter a column, click the drop-down arrow in the column header and choose from the available options. For example, you can sort the data in ascending or descending order, include or exclude specific values, or apply a filter based on the column's data type. You can also choose Home ⇒ Sort and Filter to sort the data, clear any sorts or filters you've applied, or use the Sort by Column option to sort one column by the value in another. For example, if you have a date table (see Recipe 16.13) containing columns for the month name and month number, you can choose to sort the month name by the month number (see Figure 16-6) so January (month 1) appears before February (month 2).
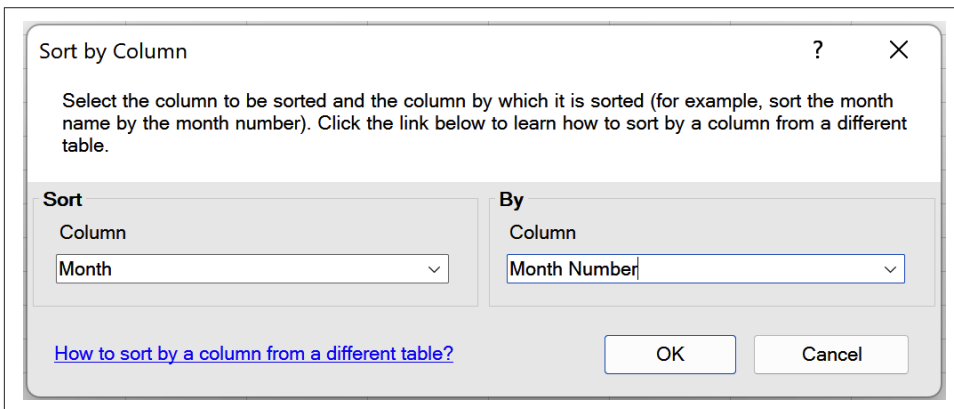
Figure 16-6. Using the Sort by Column dialog box to sort one column by another

To rename a column, either double-click the column heading and type a new name or right-click the column and choose Rename Column. You can also use this right-click menu to perform other operations, such as deleting the column, hiding it from client tools (for example, PivotTables), or navigating to a related table (see Recipe 16.7).

Finally, you can move a column by clicking its column header and dragging it to the new location.

Discussion

This recipe presents the general changes you can make to the data model's table columns, focusing on those available in Power Pivot's data view. Many of these options are also available in the diagram view, but it's generally more challenging to see their effect using this view.

See Also

You can also add calculated columns to a data model table; see Recipe 16.8 for more details.

16.7 Creating and Editing Relationships

Problem

You have a data model containing tables and want to specify how they relate.

Solution

Suppose you have two tables in the data model named Customer and Order containing details of customers and the orders they've made, respectively. Each table includes a number column named Customer ID, which matches the data in the two tables and specifies which customer placed which orders.

When you have related tables you need to define a relationship specifying how the tables are related. So, in this example, you need to create a relationship specifying that the Customer ID column in the Customer table matches the Customer ID column in the Order table.

To define the relationship, follow these steps:

1. Open the Power Pivot window by choosing Power Pivot ⇒ Data Model ⇒ Manage.

2. Choose Design ⇒ Relationships ⇒ Create Relationship to open the Create Relationship dialog box.

3. In the Create Relationship dialog box, use the two drop-down boxes to specify which two tables you want to define the relationship for. So, in this example, you select Customer in one drop-down box and Order in the other.

4. Select which columns contain matching values in the preview of each table shown in the dialog box. In this example, you select the Customer ID column for each table.

5. Click OK to create the relationship.

Figure 16-7 shows the Create Relationship dialog box.

You can also create the relationship using Power Pivot's diagram view (see Recipe 16.4). Either right-click a table and choose Create Relationship to open the Create Relationship window or draw a line between the columns containing related data by clicking and dragging from one column to the other. So, in this example, you could create the required relationship by clicking and dragging the Customer ID column in the Customer table to the Customer ID column in the Order table, as shown in Figure 16-8.
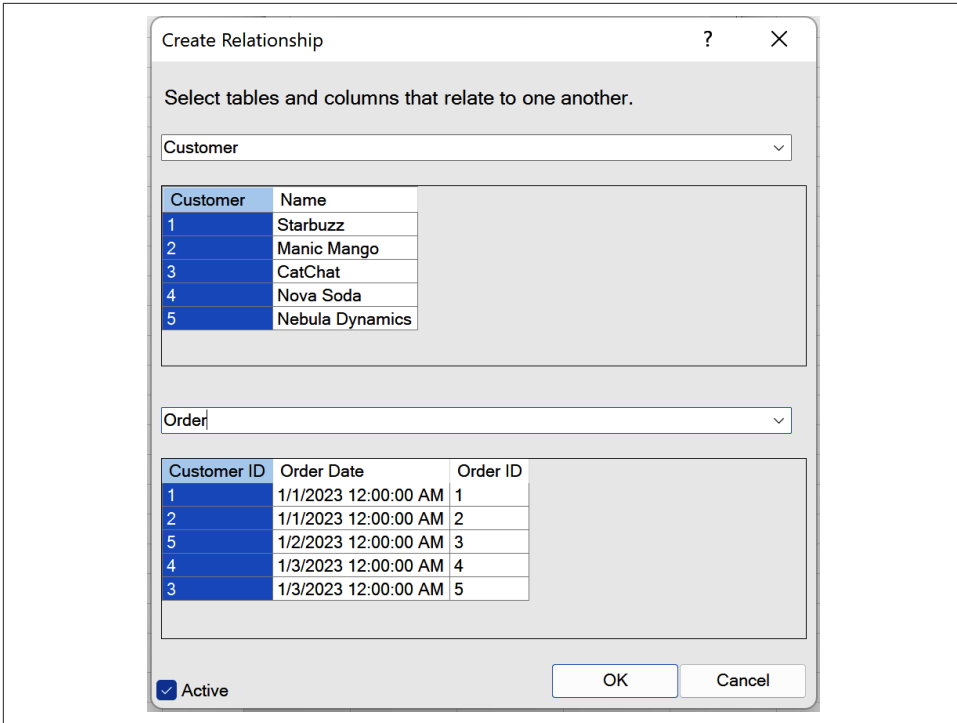
Figure 16-7. Using the Create Relationship dialog box to define a relationship
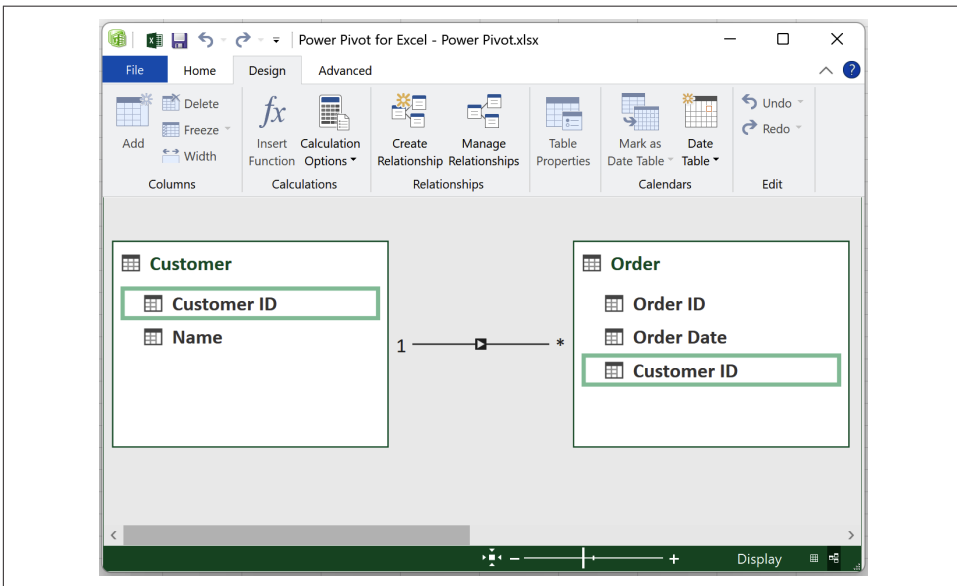
Figure 16-8. A relationship between two tables in the diagram view

When you define a relationship between two columns, ensure that their data types match, or you may get errors. To view or change a column's data type, see Recipe 16.6.

To edit or delete an existing relationship, choose Design ⇒ Relationships ⇒ Manage Relationship to open the Manage Relationships dialog box. Then select the relationship and click Edit or Delete, depending on the action you want to perform. You can also double-click a relationship in the diagram view to edit it, select it, and press the Delete key to delete it or right-click it to open a menu showing these options.

An alternative to deleting a relationship is to make it inactive; this has the same effect as deleting the relationship but makes it easier to reinstate later on. To make a relationship inactive, edit it and then uncheck the Active box in the Edit Relationship dialog box or right-click it in the diagram view and choose Mark as Inactive.

Discussion

When you have two or more tables in the data model, you specify how they relate by defining relationships. These relationships are important because they affect calculations (for example, those shown in PivotTables) and your data model may return incorrect results if the relationships are missing.

This recipe outlines how to create and manage relationships using Power Pivot. If you haven't enabled Power Pivot, you can create and manage relationships by choosing Data ⇒ Data Tools ⇒ Relationships.

16.8 Adding a Calculated Column

Problem

You have a table in the data model and want to add a new column that uses a formula.

Solution

Suppose you have a table named OrderDetail in the data model that includes columns named Unit Price and Quantity. You want to insert a new column that calculates the product of these two columns for each row.

You can solve this problem by adding a calculated column to the table, similar to adding one to an Excel worksheet table. You add the column to the data model table as follows (see Figure 16-9):

1. Open the Power Pivot window by choosing Power Pivot ⇒ Data Model ⇒ Manage.

2. Switch to the data view by choosing Home ⇒ View ⇒ Data View and navigate to the table to which you want to add the column—in this example, OrderDetail.

3. Double-click the Add Column heading to the right of the table's columns, type a name for the new column—in this example, **Total**—then press Enter/Return to create the new column.

4. Select the new column, type the formula you want to use for its values in the formula bar, and then press Enter/Return. For example, to multiply the values in the Unit Price and Quantity columns, you'd type `=[Unit Price]*[Quantity]`.

When you add a calculated column to a data model table, Power Pivot automatically derives its data type. You can view or change the data type or change the column's format using the options described in Recipe 16.6.
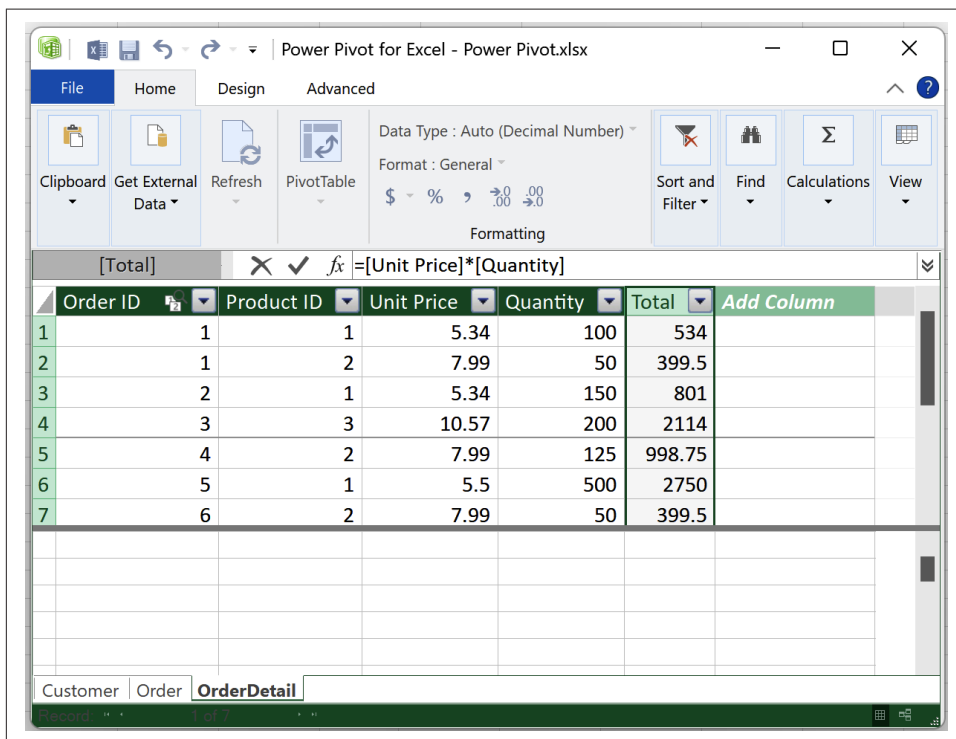
Figure 16-9. Creating the Total calculated column

Calculated columns in data model tables can also refer to columns in related tables. For example, suppose there's a relationship between the OrderDetail and Order

tables, with matching values in a column named Order ID. You want to use this relationship to add a new column to the Order table that calculates the sum of the Order-Detail Total column for each related row. You can add the new column as follows (see Figure 16-10):

1. Navigate to the Order table in the Power Pivot window's data view.

2. Double-click the Add Column heading to the right of the table's columns, type a name for the new column—in this example, **Amount**—then press Enter/Return to create the column.

3. Select the new column and then type the formula **=CALCULATE(SUM(Order Detail[Total]))** in the formula bar and press Enter/Return.
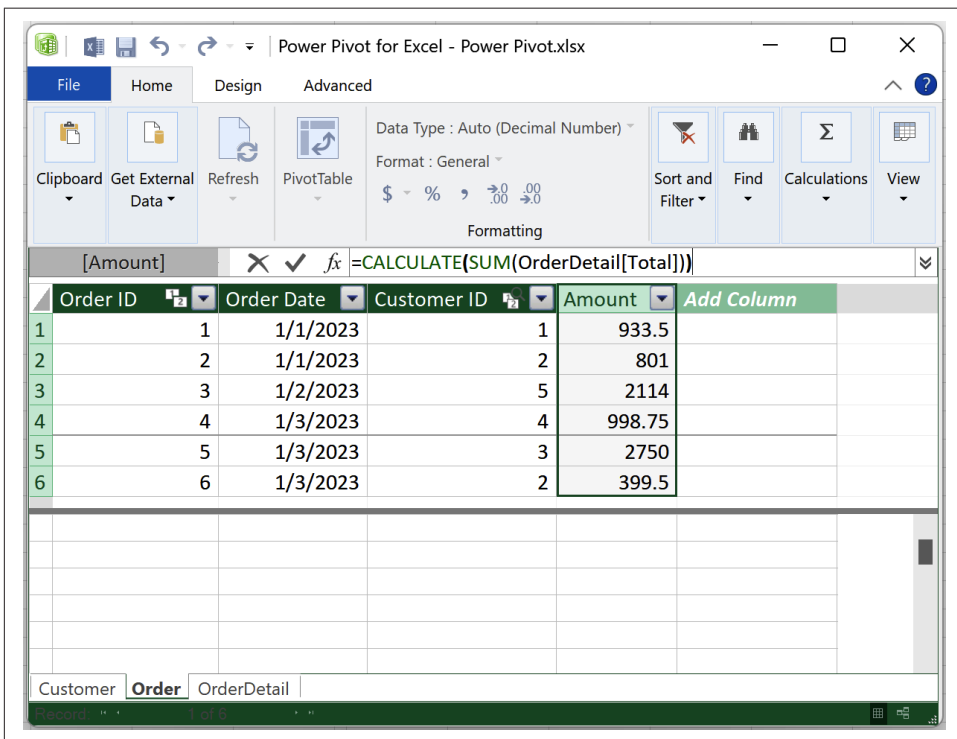
Figure 16-10. Creating the Amount calculated column

The formula =CALCULATE(SUM(OrderDetail[Total])) uses OrderDetail[Total] to refer to the Total column in the OrderDetail column, and the SUM function adds together its values. The CALCULATE function uses the relationship between the Order and OrderDetail tables to filter the calculation so it sums only the values with matching Order ID values.

Another helpful function when working with calculated columns is RELATED, which returns a single related value from another table. For example, suppose the Order table includes a column named Order Date. In that case, you can add the date of each order to the Order Details table by creating a calculated column with the formula =RELATED(Order[Order Date]).

Discussion

Adding a calculated column to a data model works similarly to adding one to a worksheet table. The main difference is that you write data model formulas using the Data Analysis Expressions (DAX) formula language instead of Excel formulas.

DAX formulas have a similar syntax to Excel formulas and include many functions with the same names (for example, SUM). However, extra functions, such as CALCULATE and RELATED, allow you to refer to column values in related tables and filter results based on their relationships.

See Also

You can also add calculations to the data model using measures; see Recipe 16.10.

16.9 Basing a PivotTable or PivotChart on Data Model Tables

Problem

You want to insert a PivotTable or PivotChart that uses one or more tables from the data model.

Solution

Suppose you have two related tables in the data model named Customer and Order containing details of customers and the orders they have made, respectively. The Customer table includes a Name column specifying each customer name, the Order table includes an Amount column specifying the amount of each order, and you want to create a PivotTable that displays each customer name and their total amount.

To solve this problem, you need to create a PivotTable that's based on the data model and uses data from both tables; you do this as follows (see Figure 16-11):

1. In the Power Pivot window, choose Home ⇒ PivotTable ⇒ PivotTable to open the Create PivotTable dialog box, choose whether to put it in a new or existing worksheet, and then click OK to insert the PivotTable.

2. Select the All tab in the PivotTable Fields pane (if it's not already selected).

3. The All tab displays a list of available tables (excluding any hidden from client tools; see Recipe 16.4. Click or expand each table to see its fields (including any calculations); then drag any you want to use in the PivotTable to the Filters, Columns, Rows, or Values sections (see Recipe 11.3). So, in this example, you'd drag the Customer table's Name field to the Rows section and the Order table's Amount field to the Values section.
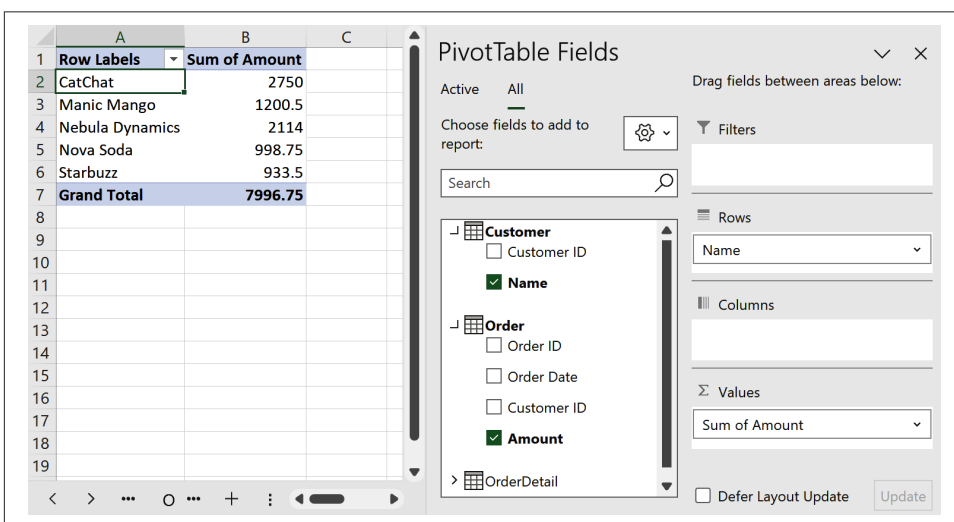
Figure 16-11. Creating the PivotTable[1]

Instead of inserting a single PivotTable in step 1, you can create a single PivotChart or different combinations of the two. Choose Home ⇒ PivotTable in the Power Pivot window to see the available options.

Discussion

You can base a PivotTable on only a single worksheet table by default. However, you can get around this limitation by adding these tables to the data model and then using Power Pivot to insert the PivotTable.

Once you've created the PivotTable, you can adjust it using most recipes in Chapter 11. However, there are several differences, including the following:

---

1 The PivotTable Fields pane has been adjusted to show the fields list and Filters, Columns, Rows, and Values sections side by side.

- PivotTables based on data model tables have different aggregation options in the Value Field Settings dialog box (see Recipe 11.9).
- You add extra calculations to the data model as calculated columns (see Recipe 16.8) and measures (see Recipe 16.10) instead of using calculated items and fields.
- You can't use Recipes 11.17 and 11.18 to group by number or text values. You can use Recipe 11.16 to group by date values or insert a date table (see Recipe 16.13) for more flexibility.
- You can hide columns in the data model (see Recipe 16.6), so they're not listed in the PivotTable fields list.

Discussion

This recipe outlines how to use Power Pivot to insert a PivotTable based on data in the model data. If you haven't enabled Power Pivot, you can add a worksheet table to the data model and create a PivotTable that uses data-model data by following Recipe 11.2 and selecting the "Add this data to the Data Model" check box in step 2.

> If you're using Excel for Mac, you can't base a PivotTable on multiple tables because this version of Excel doesn't include the data model. However, you can use functions such as XLOOKUP (see Recipe 7.9) to add extra columns to a worksheet as a workaround for this limitation.

16.10 Inserting Measures

Problem

You have a table in the data model and want to add a calculation that uses data aggregations to measure results.

Solution

Suppose you have a table in the data model named Order with a column named Amount. You want to calculate the sum of the Amount column so you can view it in the data model, add it to a PivotTable's Values section, and use it as the basis of a KPI (see Recipe 16.11).

You can achieve this by creating a *measure*: a formula you add to a data model's table that returns a result for that table. So you'd add a measure named Total Amount to the Order table, which calculates the sum of the Amount column.

You create a measure as follows (see Figure 16-12):

1. Choose Power Pivot ⇒ Data Model ⇒ Manage to open the Power Pivot window.

2. Switch to the data view by choosing Home ⇒ View ⇒ Data View and navigate to the table to which you want to add the measure (in this example, Order).

3. Select an empty cell in the calculation area below the table's rows. If you can't see this area, try choosing Home ⇒ View ⇒ Calculation Area, which toggles whether to show or hide it.

4. Once you've selected the empty cell, type the measure's formula in the formula bar; this takes the general form *measure_name:=formula*, where *measure_name* is the name of the measure, and *formula* is its formula. To create a measure named Amount Total, which sums the Amount column, you'd type `Amount Total: =SUM([Amount])`.

5. When you've finished typing the formula, press Enter/Return to create it.
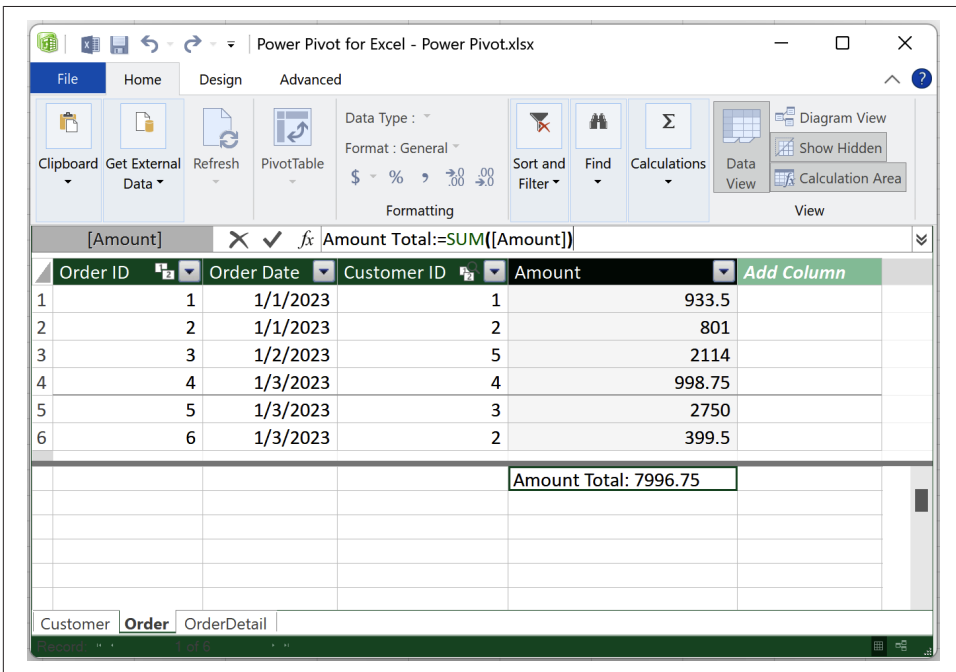
Figure 16-12. Adding a measure using the data view

If you omit the measure name and type only its formula—for example, `=SUM([Amount])`—Power Pivot gives the measure a default name. You can change this name by updating it in the formula bar.

Once you've created the measure, you can see its value in the calculation area—for example, *Amount Total: 7996.75*; you may need to widen the column to see this value, which you can do using the column header. You can also edit the measure in the formula bar, format it using the options in the Home ⇒ Formatting group, and delete it by pressing the Delete key. Alternatively, right-click the measure's cell and choose the appropriate option.

If you want to create a simple aggregation measure, you can do so using Power Pivot's AutoSum option instead of manually typing the formula. Select the column you want to base the measure on, choose Home ⇒ Calculations ⇒ AutoSum, and select one of the available aggregations. These aggregations are Sum, Average, Count, Distinct Count, Max, and Min. So, in this example, you could create a measure to calculate the sum of the Amount column by selecting the column and choosing the AutoSum option's Sum aggregation.

You can also create, edit, and delete measures within the main Excel window. To create a measure in this way, choose Power Pivot ⇒ Calculations ⇒ Measures ⇒ New Measure to open the Measure dialog box, and then use the available options to create the measure (see Figure 16-13).
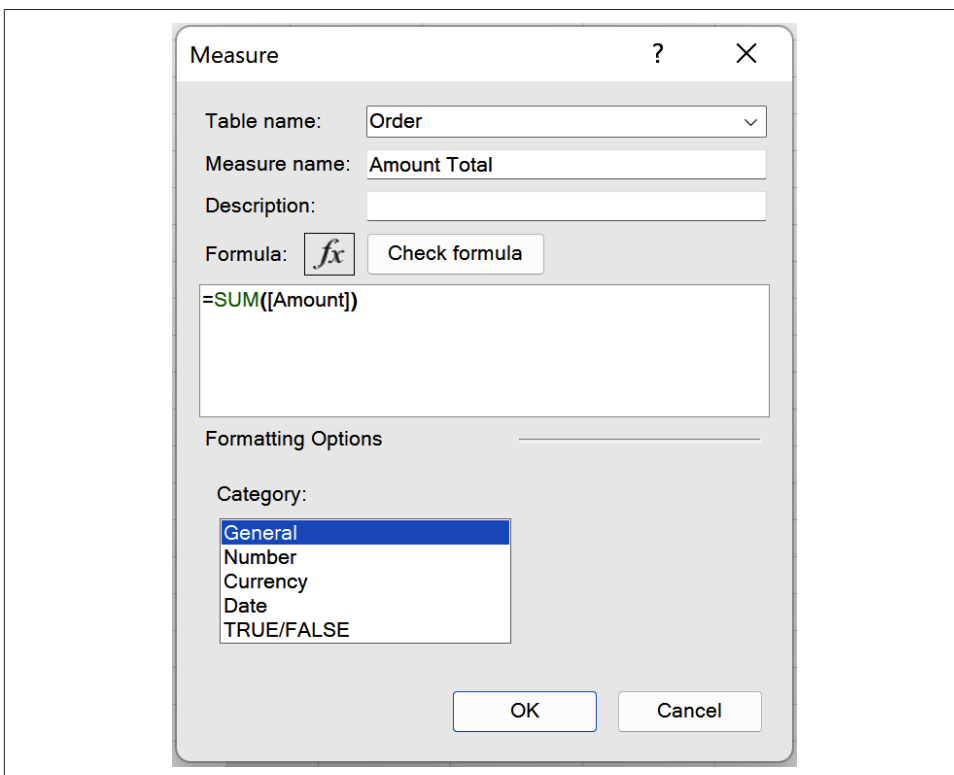
Figure 16-13. Adding a measure using the Measure dialog box

To edit or delete a measure, choose Power Pivot ⇒ Calculations ⇒ Measures ⇒ Manage Measures to open the Manage Measures dialog box, select the measure, then click the Edit or Delete button.

To add a measure to a PivotTable, find it in the PivotTable Fields pane's field list, then drag it to the Values section (see Recipe 16.9).

Discussion

This recipe explains how to add a measure to a table in the data model. Similarly to calculated columns (see Recipe 16.8), you define measures by writing DAX code so you can refer to columns in other tables and refer to other measures. Unlike with calculated columns, you can use measures to create KPIs (see Recipe 16.11).

The measures you add to the data model, as described in this recipe, are sometimes called *explicit* measures because you explicitly define them. Data models can also include *implicit* measures, which Power Pivot automatically creates when you add a field to a PivotTable's Values section. For example, if you add the Order table's Amount column to the Values section to calculate the sum, Power Pivot automatically creates an equivalent measure in the Order table. Implicit measures are read-only (so you can't update their formulas), get deleted when you remove them from the Pivot-Table, and can't be used to create KPIs.

You can show or hide a table's implicit measures in the Power Pivot data view by choosing Advanced ⇒ Show Implicit Measures.

16.11 Using KPIs

Problem

You have a measure in the data model and want to visually gauge its performance against a specified target.

Solution

Suppose you have two related tables in the data model named Customer and Order. Customer contains customer details and includes a Name column, and Order contains details of their orders and includes an Amount column. Your target total order amount per customer is $5,000, and you want to quickly gauge whether you're on track to meet this target for each customer.

You can solve this problem using KPIs: visual measures of performance that help you gauge the status of a metric against a defined target value.

To create a KPI, you must first define a measure that provides its base or current value. So, in this example, you'd add a measure named Amount Total to the Order table, which calculates the sum of the Amount column (see Recipe 16.10).

Once you've defined the measure, create the KPI as follows:

1. Open the Key Performance Indicator (KPI) dialog box using one of these methods: select the measure in the Power Pivot data view and choose Home ⇒ Calculations ⇒ Create KPI; right-click the measure and select Create KPI; or choose Power Pivot ⇒ Calculations ⇒ KPIs ⇒ New KPI in the main Excel window.

2. Ensure that the "KPI base field (value)" drop-down list shows the measure you want to base the KPI on—in this example, Amount Total. If you opened the dialog box in the main Excel window, you may need to select this measure from the drop-down list.

3. In the "Define target value" section, choose whether to use another measure (which you must select) as the KPI target value or use an absolute value. In this example, the target total order amount per customer is $5,000, so select Absolute Value and type **5000** in its value box.

4. In the "Define status thresholds" section, choose one of the styles to indicate which colors—red, yellow, or green—you want to use for low, high, and target values, where red means the value is very off target, yellow means it's on track but more work is needed, and green means it's on target. In this example, you want to use red for low values, yellow for values slightly below the target, and green for values above the target, so you can keep the default red-yellow-green style.

5. Once you've chosen a color style, type values in the threshold boxes to indicate when you want the status color to change. So, to change the status from red to yellow when the value reaches 2,000, you'd type **2000** in the box between these two colors. Similarly, to change the status from yellow to green when the value reaches 4,000, you'd type **4000** in the box between these two colors.

6. In the "Select icon style" section, choose the icon set to use for the KPI.

7. If you want to add descriptions to the KPI, click Descriptions to expand this section and then type descriptions for KPI, base value, status, and target in the appropriate boxes.

8. Click OK to create the KPI.

Figure 16-14 shows the Key Performance Indicator (KPI) dialog box, including the options used to create the KPI.
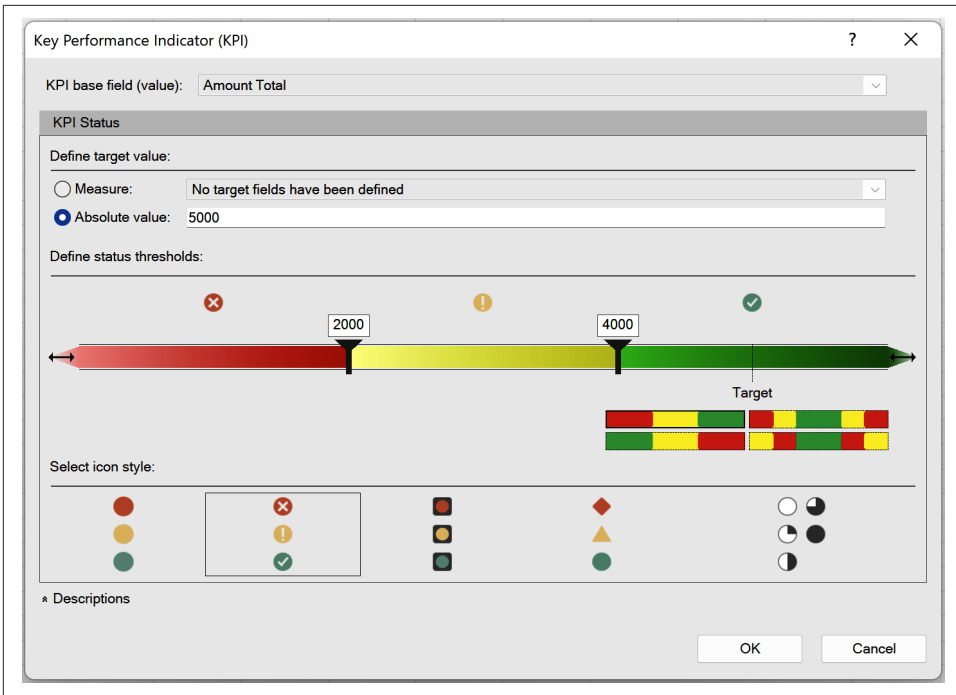
Figure 16-14. Adding a KPI using the Key Performance Indicator (KPI) dialog box

Once you've created the KPI, you display its status by adding it to a PivotTable. So, to show the Amount Total KPI in this example, you'd follow these steps:

1. Create a PivotTable that uses the data model (see Recipe 16.9).

2. In the PivotTable Fields pane, drag the Customer table's Name field to the Rows section to display each customer's name.

3. Expand the Order table and the Amount Total KPI in the PivotTable Fields pane to show its Value, Goal, and Status fields. Then, drag the Status field to the Values section to display the status of the KPI for each customer.

4. If you want to show each customer's current value and goal, drag the KPI's Value and Goal fields to the Values section as well.

Figure 16-15 shows the KPI's status displayed in a PivotTable.
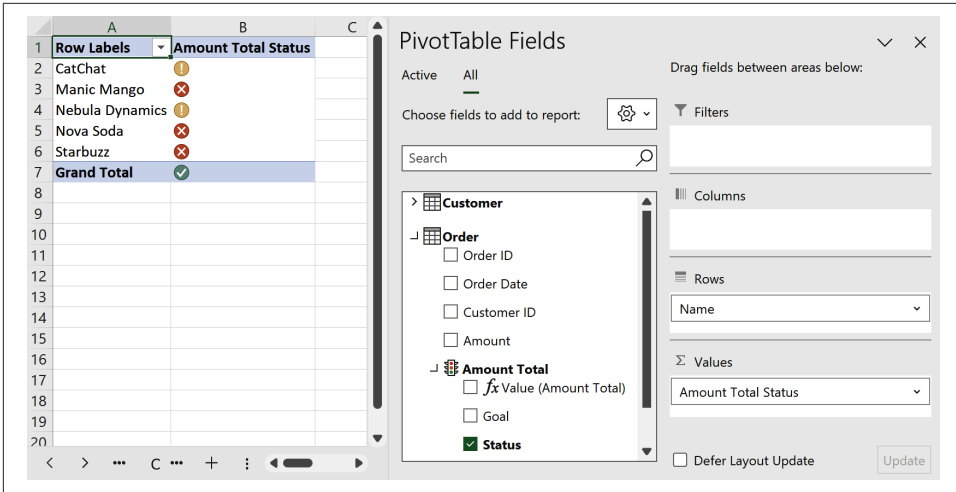
Figure 16-15. Displaying a KPI's status in the PivotTable[2]

If you need to edit or delete a KPI, right-click the measure in Power Pivot's data view and choose Edit KPI Settings or Delete KPI. Alternatively, in the main Excel window, choose Power Pivot ⇒ Calculations ⇒ KPIs ⇒ Manage KPIs.

If you change the data model, any KPI statuses used in PivotTables may stop displaying correctly. To rectify this, remove the status from the PivotTable and then put it back again.

Discussion

KPIs offer a convenient way of gauging a measure's performance against a specific target. This recipe gives an overview of creating one and then displaying its status using a PivotTable.

16.12 Creating Hierarchies

Problem

You want to organize selected fields in a PivotTable's field list as a hierarchy, which you can use to group and drill down into data.

---

2 The PivotTable Fields pane has been adjusted to show the fields list and Filters, Columns, Rows, and Values sections side by side.

Solution

Suppose you have a Country table in the data model containing country data, including columns named Country, Region, State, and City. You want to group data by these fields in any PivotTables you create so you can drill down from Country to Region and so on.

You can achieve this by creating a *hierarchy*: a group of fields that appears as a single item in the PivotTable's field list. In this example, you'd create a hierarchy with Country as the parent, followed by Region, State, and City.

You create the hierarchy as follows:

1. Open the Power Pivot window by choosing Power Pivot ⇒ Data Model ⇒ Manage.

2. Switch to the diagram view (if it's not already open) by choosing Home ⇒ View ⇒ Diagram View.

3. Find the table you want to create the hierarchy for—in this case, Country—right-click the column you want to use as its parent (the Country column), and select Create Hierarchy; this adds the hierarchy to the bottom of the table's column list and copies the parent column to it.

4. Type a name for the hierarchy—for example, **Country Hierarchy**.

5. Add more columns to the hierarchy by clicking and dragging them or right-clicking them and choosing Add to Hierarchy. In this example, you'd add the Region, State, and City columns to the hierarchy.

Once you've created the hierarchy, you can rearrange its columns by clicking and dragging them into position, and remove a column by selecting it in the hierarchy and pressing the Delete key. You can also right-click the column in the hierarchy and choose from the available options.

Figure 16-16 shows a hierarchy in the Country table.

To use the hierarchy, create a PivotTable based on the data model (see Recipe 16.9), and then, in the PivotTable Fields pane's field list, expand the table containing the hierarchy. You can then treat the hierarchy as a single item and drag it to one of the pane's sections to add all its fields at once (see Figure 16-17).
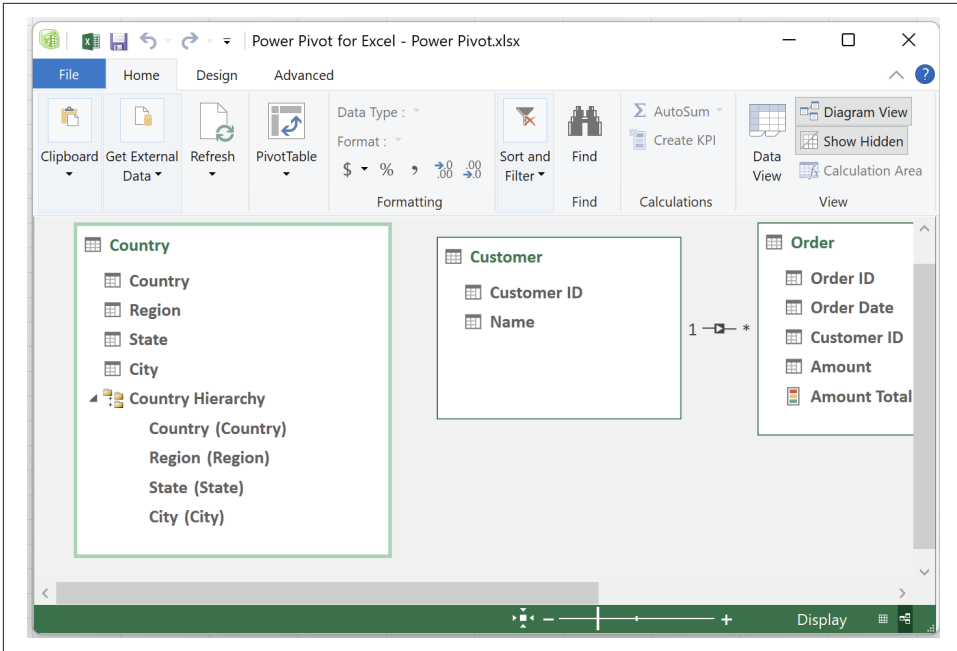
Figure 16-16. Adding a hierarchy to the Country table
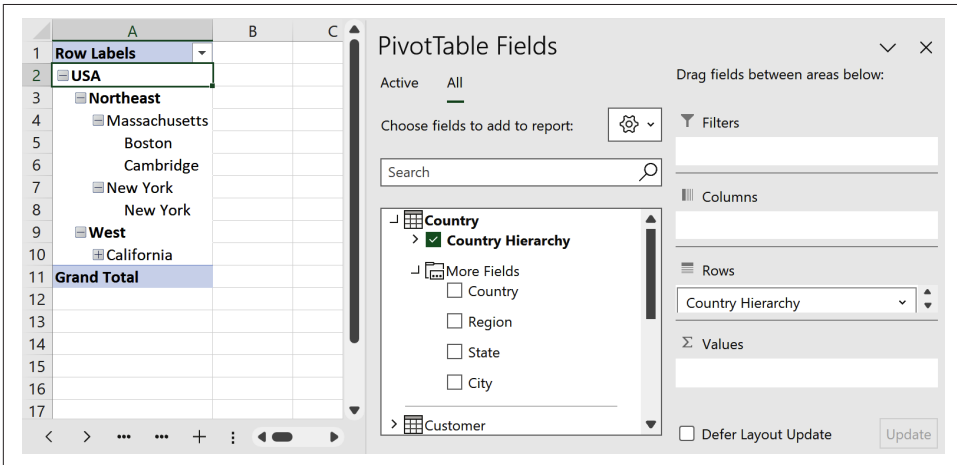
Figure 16-17. Using a hierarchy for a PivotTable's rows

Discussion

Hierarchies can be handy if you often need to group and drill down using a specific group of fields. This recipe shows how to add a hierarchy to the data model so you can use it in PivotTables.

16.13 Creating a Date Table

Problem

You have a PivotTable based on the data model and want to be able to customize how you group any date columns, including by financial year and quarter.

Solution

Suppose you have a table in the data model named Order containing a date column named Order Date. You want to use this column to group the data in a PivotTable.

To group the data by year, quarter, and/or month, you can use Recipe 11.16 to specify the groups. However, if you need to group the data in another way—for example, by fiscal year and fiscal quarter—you can create and use a date table instead.

A *date table* is a table you add to the data model that contains a contiguous set of dates with columns that represent the date in different ways—for example, month, year, fiscal quarter, and so on. You can then use the date table in PivotTables to group data by one or more of these columns.

You create a date table as follows (see Figure 16-18):

1. Choose Power Pivot ⇒ Data Model ⇒ Manage to open the Power Pivot window.

2. Switch to the data view (if it's not already open) by choosing Home ⇒ View ⇒ Data View.

3. Choose Design ⇒ Calendars ⇒ Date Table ⇒ New to create the date table. Power Pivot names the table Calendar, populates it with dates, and includes calculated columns representing the date in different ways—for example, Year, Month, and Day Of Week.

4. The date table must include all the dates you'll use in the data model. To change the range of dates included in the table, choose Design ⇒ Calendars ⇒ Date Table ⇒ Update Range, and then select the new range.

5. Create a relationship between the date table and the date column in the other table. For example, suppose the data model includes a table named Order with a date column named Order Date. In that case, you'd create a relationship between the Order table's Order Date column and the Calendar table's Date column.
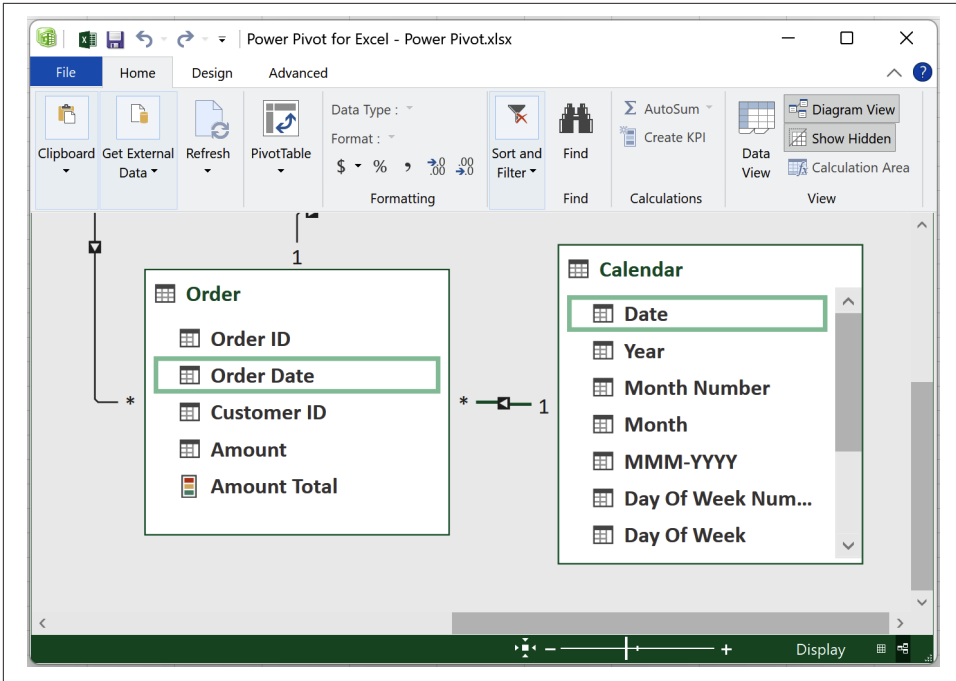
Figure 16-18. Adding a date table named Calendar

Once you've created the date table, you can add new calculated columns, depending on how you want to group dates. For example, say you want to be able to group dates by fiscal quarter and year and your financial year starts on October 1. You could add a calculated column named Fiscal Quarter with the formula `=SWITCH(MONTH ([Date]),10,1,11,1,12,1,1,2,2,2,3,2,4,3,5,3,6,3,4)`—where months 10, 11, and 12 return 1; months 1, 2, and 3 return 2; months 4, 5, and 6 return 3; and the remaining months return 4. Then you could add a column named Fiscal Year with the formula `=SWITCH([Fiscal Quarter],1, YEAR([Date])+1, YEAR([Date]))`. These formulas use the DAX `SWITCH` function, which works similarly to Excel's (see Recipe 7.8).

You use the date table in a PivotTable as follows:

1. Create a PivotTable that uses the data model (see Recipe 16.9).
2. Drag fields to the Rows, Columns, and Values section as usual, but instead of using your original date field—in this example, the Order table's Order Date—use columns from the date table. So, to group rows by fiscal year and fiscal quarter, you'd drag the Calendar table's Fiscal Year and Fiscal Quarter fields to the Rows section.

If the data model contains multiple date columns you want to be able to group by, you may need to create more than one date table—one for each date column. If you've already created a date table containing extra custom columns, you can use it as the basis for any new date tables you want to create by selecting the table, choosing Design ⇒ Calendars ⇒ Date Table ⇒ Save Configuration, and then creating the new date tables. If you subsequently want to create a date table using Power Pivot's default configuration, choose Design ⇒ Calendars ⇒ Date Table ⇒ Set Default before creating the table.

Discussion

When using dates in PivotTables, you may often want to group by date columns in custom ways that Excel doesn't provide by default. Including one or more date tables provides a workaround for this.

This recipe gives instructions for creating a date table from scratch because this is generally the most efficient option. You can also mark an existing table as a date table by selecting the table, choosing Design ⇒ Calendars ⇒ Mark as Date Table, then choosing Design ⇒ Calendars ⇒ Mark as Date Table ⇒ Date Table Settings to specify which column contains the dates.

16.14 Using Named Sets

Problem

You have a PivotTable based on the data model and want to hide aggregations for different rows or columns.

Solution

Suppose you have a Result table in the data model containing the forecast and actual exam results for students in multiple schools and academic years. These details are recorded in columns named School, Student, Academic Year, Forecast, and Actual. You want to create a PivotTable that calculates the average score per school, using the Actual column for some academic years and the Forecast column for others—for example, because forecast scores are available for only some academic years. You can solve this problem using a *named set*, which lets you pick and choose which rows and columns to include in the PivotTable.

Before using a named set, you need to create the PivotTable. So, in this example, you'd create a PivotTable based on the data model, dragging the Result table's School field to the Rows section, the Academic Year field to the Columns section, and the Actual and Forecast fields to the Values section, changing their aggregation to Average. This

results in a PivotTable with rows for each school, and columns showing the forecast and actual average scores for each academic year, as shown in Figure 16-19.
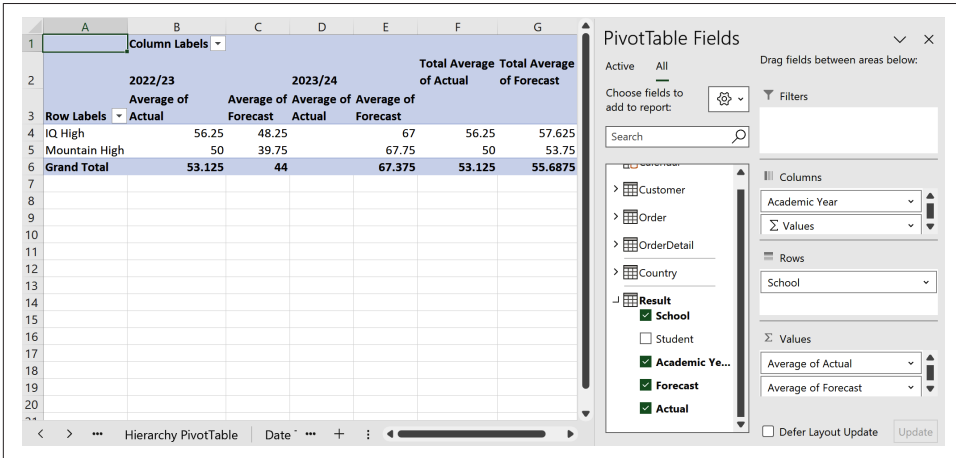
Figure 16-19. The PivotTable before defining a named set

Once you've created the PivotTable, you define a named set using the New Set dialog box. To open this dialog box, select the PivotTable and choose PivotTable Analyze ⇒ Calculations ⇒ Fields, Items, & Sets ⇒ Create Set Based on Row Items or Create Set Based on Column Items, depending on whether you want to hide PivotTable calculations using row or column values. In this example, you must select the Create Set Based on Column Items option because you want to restrict the values shown for specific academic years (the PivotTable's columns).

Once you've opened the New Set dialog box, type a name for the set in the "Set name" box, so in this example, type the name **Academic Year Scores Set**.

The New Set dialog box displays a list of the PivotTable's rows or columns, depending on which option you selected. In this example, you selected the Create Set Based on Column Items option, so the dialog box displays a list of the columns, including each Academic Year value and the average of the Forecast and Actual scores. If the Academic Year column includes the values *2022/23* and *2023/24*, the list would include rows for 2022/23 Average of Forecast, 2022/23 Average of Actual, 2023/24 Average of Forecast, and 2023/24 Average of Actual, along with All Average of Forecast and All Average of Actual rows for the grand totals (see Figure 16-20).

To remove a particular item from the PivotTable, select it by clicking to the left and then click the Delete Row button. To display only the average of the Actual column for 2022/23 and the Forecast column for 2023/24, you'd select and delete each item except for 2022/23 Average of Actual and 2023/24 Average of Forecast (see Figure 16-21).
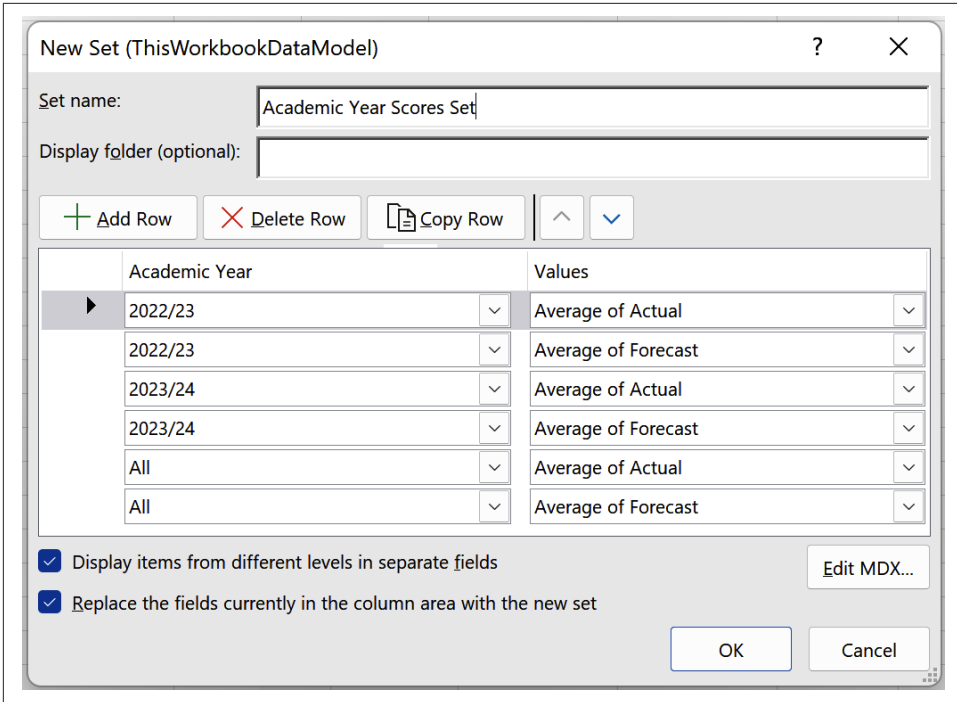
Figure 16-20. The New Set dialog box listing every column
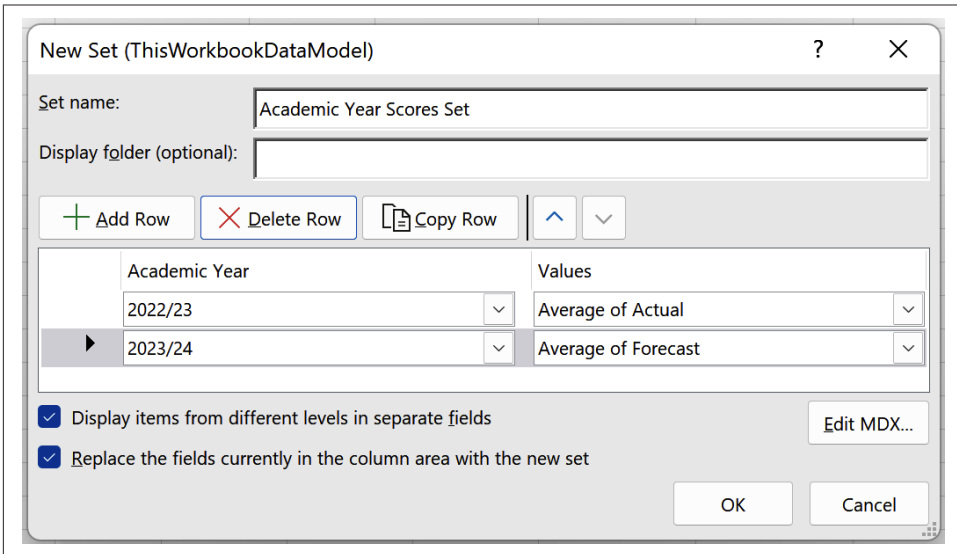
Figure 16-21. The New Set dialog box with some columns removed

Finally, make sure the "Display items from different levels in separate fields" and "Replace the fields currently in the column area with the new set" check boxes are checked. Then click OK to create the set, add it to the PivotTable Fields pane's field list, and view the changes its made to the PivotTable (see Figure 16-22).
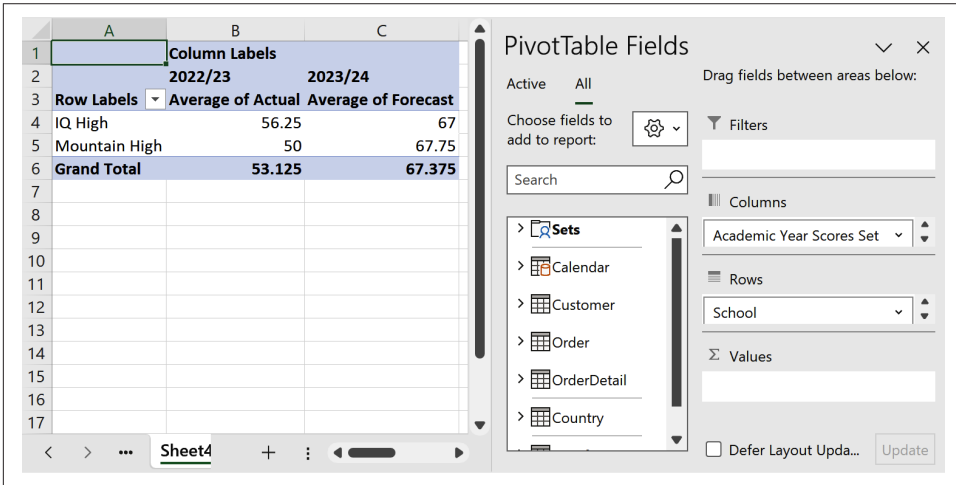
Figure 16-22. The PivotTable after defining a named set

If you want to edit or delete a named set you've previously created, choose PivotTable Analyze ⇒ Calculations ⇒ Fields, Items, & Sets ⇒ Manage Sets, and use the available options.

Discussion

Named sets are helpful if you want to create an asymmetric PivotTable, which excludes calculations for specific rows and/or columns.

This recipe shows how to create a named set using the New Set dialog box options. You can also specify the rows or columns using *Multidimensional Expressions* (MDX), a query language Excel uses to communicate with the data model. You can view or edit the set's MDX code by clicking the Edit MDX button in the New Set dialog box.

16.15 Converting a PivotTable to Formulas

Problem

You have a PivotTable based on the data model and want to break it into formulas so you can move or delete its cells.

Solution

Suppose you have a PivotTable based on the data model and you want to rearrange its cells in ways that aren't possible using Recipes 11.7 and 16.14. For example, you might want to use only selected cells, manually insert rows or columns, or combine or intersperse the rows from two separate PivotTables so you can compare their values.

You can solve this problem by converting the PivotTable's cells to formulas that get values directly from the data model. Doing so means you can move or delete cells and still get up-to-date values from the data model by refreshing the data.

> When you convert a PivotTable to formulas, you can no longer use interactive PivotTable features such as sorting data and expanding and collapsing levels, and you can't convert the formulas back to a PivotTable. Make sure you save a copy of your work in case you need to return to it.

Before converting the PivotTable, you first need to make sure it resembles as closely as possible the result you want. For example, the PivotTable should include any values you want to generate values for and include any necessary filters. Once satisfied, you convert the PivotTable by choosing PivotTable Analyze ⇒ Calculations ⇒ OLAP Tools ⇒ Convert to Formulas.

Excel generally converts the PivotTable as soon as you select the Convert to Formulas option. However, if the PivotTable Fields pane's Filters section contains fields, Excel first opens the Convert to Formulas dialog box. This dialog box lets you specify whether to convert the entire PivotTable or keep the filters; if you choose to keep the filters, you can continue to use them to filter the data.

> You can also use slicers and timelines to filter the data returned by the formulas. See Recipe 16.17 for more information about how to do this.

Once the PivotTable has been converted to formulas, you can treat its cells like any other ones containing formulas, so you can move or delete them as necessary. Figure 16-23 shows example results from converting a PivotTable to formulas.

	A	B	C	D	E	F
1	Row Labels	Sum of Amount				
2	=CUBEMEMBER("ThisWorkbookDataModel","[Customer].[Name].&[CatChat]")					
3	Manic Mango	1200.5				
4	Nebula Dynamics	2114				
5	Nova Soda	998.75				
6	Starbuzz	933.5				
7	Grand Total	7996.75				

Figure 16-23. Converting a PivotTable to formulas

Discussion

PivotTables offer a convenient way of analyzing data-model data, but making them appear the way you want can be tricky because of their layout restrictions. This recipe shows how to overcome these limitations by breaking the PivotTable into formulas.

Behind the scenes, Excel converts the PivotTable's cells to Cube formulas, which get data directly from the data model. This approach is more flexible than using the GET PIVOTDATA function (see Recipe 11.32), which relies on the PivotTable being there. You can learn more about Cube formulas in Recipe 16.16.

16.16 Using Cube Formulas

Problem

You have tables and measures in the data model and want to be able to access their data using formulas instead of PivotTables.

Solution

Suppose you have one or more tables in the data model, and you want to access their values without creating a PivotTable. You can do so with Excel's *Cube* functions: functions you can use to get data directly from the data model.

The most important Cube function is CUBEVALUE, which returns a value from the data model, such as an aggregation from one of the data model's measures. You generally use the formula =CUBEVALUE(*connection, member_expression_1, member_expression_2, …*), where *connection* is a text string giving the name of the connection to the data model, and *member_expression_1, member_expression_2*, and so on are optional text strings specifying the value you want to retrieve and any filters you want to apply. To connect to the data model and get the value of a measure named Amount Total, you'd use =CUBEVALUE("ThisWorkbookDataModel", "[Measures].[Amount Total]"). Similarly, to get the value of the Amount Total

measure where the value of the Customer ID column in the Customer table is 1, you would use =CUBEVALUE("ThisWorkbookDataModel", "[Measures].[Sum of Total]", "[Customer].[Customer ID].[1]").

When you start typing Cube formulas, Excel's IntelliSense provides help with each function's arguments, such as the name of the data model's connection string and any *members*: the measures, tables, fields, items, and so on, in the data model. You specify members using the MDX query language, where [Customer] refers to the data model's Customer table, [Customer].[Customer ID] to its Customer ID column, and [Customer].[Customer ID].[1] to the customer whose customer ID is 1.

The CUBEMEMBER function gets a reference to a member of the data model, such as an item, and you can use it to verify whether a member exists. You generally use the formula =CUBEMEMBER(*connection, member_expression, caption*), where *connection* is the connection to the data model, *member_expression* references the member you want to return, and *caption* (optional) is alternative text you want to display instead of the member's label. To get the customer from the Customer table where the Customer ID is 1, you'd use =CUBEMEMBER("ThisWorkbookDataModel", "[Customer].[Customer ID].[1]"); doing so returns 1—the Customer ID value—if this customer exists or #N/A if it doesn't.

You can use the CUBEMEMBER function to simplify CUBEVALUE formulas. If cell A1 contains the formula =CUBEMEMBER("ThisWorkbookDataModel","[Measures].[Sum of Total]") and cell A2 contains =CUBEMEMBER("ThisWorkbookDataModel", "[Customer].[Customer ID].[1]"), you can use the formula =CUBEVALUE("This WorkbookDataModel", A1, A2) instead of =CUBEVALUE("ThisWorkbookDataModel", "[Measures].[Sum of Total]", "[Customer].[Customer ID].[1]").

The CUBESET function gets a reference to a set of members from the data model, such as the items in a table's column. The function doesn't display these members when you use it as a cell's formula, but you can use it to simplify other cube formulas—for example, CUBEVALUE, CUBESETCOUNT, and CUBERANKEDMEMBER. You generally use the formula =CUBESET(*connection, set_expression, caption, sort_order, sort_by*), where the arguments are as follows:

connection
This is a text string giving the connection's name to the data model.

set_expression
This is a text string specifying the members you want to return. For example, the expression "[Customer].[Name].Children" specifies each unique item in the Customer table's Name column, and "[Customer].[Name].Members" includes an extra All item.

caption (optional)

> This lets you display text in a cell when you use the formula in one. If you omit this argument, the formula doesn't display any text.

sort_order and *sort_by* (optional)

> These arguments let you sort the members in the set, where *sort_order* is an integer specifying how to sort, and *sort_by* specifies a value to sort by. Setting *sort_order* to 0 leaves the set in the existing order, 1 sorts it in ascending order of *sort_by*, 2 sorts it in descending order of *sort_by*, 3 sorts it in alpha ascending order, 4 sorts it in alpha descending order, 5 sorts it in natural ascending order, and 6 sorts it in natural descending order.

So, to return a set of the items in the Customer table's Name column, you'd use `=CUBESET("ThisWorkbookDataModel", "[Customer].[Name].Children")`. Similarly, to order these names by the Amount Total measure so the ones with the highest amounts appear first, you'd use `=CUBESET("ThisWorkbookDataModel","[Customer].[Name].Children", "Customers", 2, "[Measures].[Amount Total]")`.

The `CUBESETCOUNT` function returns the number of items in a set. You generally use the formula `=CUBESETCOUNT(set)`, where *set* is the set whose members you want to count, usually one returned by the `CUBESET` function. So if cell B3 contains the formula `=CUBESET("ThisWorkbookDataModel", "[Customer].[Name].Children")`, the formula `=CUBESETCOUNT(B3)` counts how many items it returns.

The `CUBERANKEDMEMBER` function returns the item at a specific position in a set. You generally use the formula `=CUBERANKEDMEMBER(connection, set_expression, rank, caption)`, where *connection* is the connection name, *set_expression* refers to the set (usually one returned by the `CUBESET` function), *rank* is the rank of the item you want to return (1 for the first value, 2 for the second, and so on), and *caption* (optional) is alternative text you want to display instead of the member's label. So if cell A3 contains the formula `=CUBESET("ThisWorkbookDataModel", "[Customer].[Name].Children")`, the formula `=CUBERANKEDMEMBER("ThisWorkbookDataModel", A3, 1)` returns its first item.

The `CUBEKPIMEMBER` function gets a reference to a KPI's value, goal, or status, which you can use in a `CUBEVALUE` formula. You generally use the formula `=CUBEKPIMEMBER(connection, kpi_name, kpi_property, caption)`, where *connection* is the connection name, *kpi_name* is the name of the KPI, *kpi_property* indicates which property to return (returns the current or actual value, 2 the target goal, and 3 the current status), and *caption* (optional) is alternative text you want to display instead of the KPI property's label. So the formula `=CUBEKPIMEMBER("ThisWorkbookDataModel", "Amount Total", 3)` returns the status of the Amount Total KPI, and if you put this formula in cell A4, the formula `=CUBEVALUE("ThisWorkbookDataModel",`

`A4, "[Customer].[Customer ID].[1]")` returns the status of the KPI for the customer from the Customer table where the Customer ID is 1.

Discussion

This recipe shows how to use Excel's Cube functions to get data directly from the data model. The easiest way to learn these functions is by using Recipe 16.15 and examining the results because Excel replaces the PivotTable cells with Cube formulas.

16.17 Filtering Cube Formulas with Slicers and Timelines

Problem

You have a `CUBEVALUE` formula and want to filter its results using a slicer or timeline.

Solution

Suppose you have a cell containing a `CUBEVALUE` formula that gets data from the data model, and you want to be able to filter its results dynamically. You can do so by inserting a slicer or timeline (see Recipe 11.14) that's connected to the data model and then updating the `CUBEVALUE` formula to refer to it.

You add the slicer or timeline as follows:

1. Select an empty cell and choose Insert ⇒ Filters ⇒ Slicer or Insert ⇒ Filters ⇒ Timeline; this opens the Existing Connections dialog box.
2. Click the Data Model tab in the Existing Connections dialog box and then click Open.
3. Select the table column for which you want to insert the slicer or timeline and click OK to create it.

Once you've inserted the slicer or timeline, you can add it to a `CUBEVALUE` formula as an extra argument. So if you have a cell that uses the formula `=CUBEVALUE("ThisWork bookDataModel", "[Measures].[Amount Total]")` to return the value of the Amount Total measure and you want to filter it using an Order Date timeline, you'd update the formula to `=CUBEVALUE("ThisWorkbookDataModel", "[Measures]. [Amount Total]", Timeline_Order_Date)`, where *Timeline_Order_Date* is the name of the timeline.

When you add a slicer or timeline to your workbook, Excel automatically assigns a name to it, which you can use in formulas. These names begin with *Slicer* or *Timeline*, so you can use Excel's IntelliSense to choose the correct one. You can view a slicer's name by selecting it and choosing Slicer ⇒ Slicer ⇒ Slicer Settings; however, there is no equivalent option for timelines.

Discussion

This recipe shows how to use slicers and timelines to filter Cube formulas; this can be helpful if you've used Recipe 16.15 to convert a PivotTable's cells to Cube formulas and want to filter their results.

LET, LAMBDA, and LAMBDA Helper Functions

Custom functions make it easier to use complex formulas because they let you call each formula by a user-friendly name. However, until recently, you could write these functions only using macros or VBA.

The LAMBDA function in Excel 365 is one of Excel's most significant additions because it lets you write custom functions using formulas. So if you frequently use an Excel formula—such as calculating a date's fiscal quarter—you can use the LAMBDA function to save that formula as a custom function and call it using a name of your choice.

The recipes in this chapter cover the following areas:

- How to make Excel formulas more efficient and readable using the LET function

- How to create custom functions using the LAMBDA function

- How to use Excel's *LAMBDA helper* functions—functions that accept a LAMBDA argument—to perform complex operations using a single formula

17.1 Improving Formula Efficiency

Problem

You have a formula that repeats references or calculations, and you want to make the formula more efficient and readable.

Solution

Suppose you have a cell containing the formula `=IFS(SUM(A:A)>50, "Large", SUM(A:A)>30, "Medium", SUM(A:A)<=30, "Small")`, which returns *Large*, *Medium*, or *Small*, depending on the sum of the A column. This formula refers to—and has to calculate—`SUM(A:A)` three times, making it inefficient.

If you're using Excel 2021 or Excel 365, you can use the `LET` function to help make formulas with repeat calculations more efficient. This function lets you assign names to values or calculations so that you can refer to them within a formula, so it computes repeat calculations only once instead of multiple times.

You generally use the formula `=LET(name1, value1, name2, value2, …, formula)`, where *name1* is the name of *value1*, *name2* is the name of *value2*, and so on, and the final *formula* argument is a formula that uses the names instead of the values. For example, typing **=LET(x, 4, x+1)** sets x to 4 and then calculates x + 1, returning 5. Similarly, typing **=LET(x, SUM(A:A), IFS(x>50, "Large", x>30, "Medium", x<=30, "Small"))** sets x to SUM(A:A) and then uses it to decide which text to return; because the formula refers to SUM(A:A) only once, it doesn't need to recalculate it several times, making the formula more efficient.

Discussion

This recipe is a handy way of making formulas with repeat calculations more efficient because it reduces the number of times the formula computes each one. It can also make complex functions more readable by using friendly names instead of references or calculations.

17.2 Writing and Testing a LAMBDA Formula

Problem

You want to write and test a formula that uses Excel's `LAMBDA` function so you can use it with other recipes, such as Recipe 17.4.

Solution

Suppose you have a formula that you frequently use in a workbook, such as calculating a date's calendar or fiscal quarter (see Recipe 6.4). If you're using Excel 365, you can make this formula easier to use by turning it into a custom function using Excel's `LAMBDA` function.

The `LAMBDA` function lets you create bespoke functions using Excel formulas, so you don't have to use macros or VBA. You first write and test the `LAMBDA` formula in a

worksheet cell, and once you're satisfied it works the way you want, you can use Recipe 17.4 to convert it to a named function or pass it to a LAMBDA helper function (see Recipe 17.7 onward).

The function takes the form LAMBDA(*name1, name2, …, formula*), where *name1*, *name2*, and so on, are the (optional) names of variables, and the final *formula* argument is a formula that uses them. Some example LAMBDA formulas are as follows:

LAMBDA(x, x+1)
> This calculates x + 1 for a value of x. If x is 5, the formula returns 6.

LAMBDA(x, y, x-y)
> This calculates x – y for values of x and y. If x is 10 and y is 2, the formula returns 8.

LAMBDA(date, ROUNDUP(MONTH(date)/3, 0))
> This returns a date's calendar quarter. If date is today's date, the formula returns today's calendar quarter.

LAMBDA(RANDBETWEEN(1, 10))
> This formula returns a random integer from 1 to 10. Notice that this example includes only a +*formula*+ argument because no +*name*+ variables are needed.

Unlike other formulas, you execute a LAMBDA formula by typing it into a cell, followed by an additional set of parentheses containing any values you want to pass to it. These values correspond to the formula's name arguments, so to execute the formula =LAMBDA(x, x+1), where x is 5, you'd type **=LAMBDA(x, x+1)(5)**. Some more examples of executing LAMBDA formulas include the following:

=LAMBDA(x, x+1)(6)
> This assigns 6 to x, calculates 6 + 1, and returns 7.

=LAMBDA(x, y, x-y)(6, 4)
> This assigns 6 to x and 4 to y, calculates their difference, and returns 2.

=LAMBDA(date, ROUNDUP(MONTH(date)/3, 0))(TODAY())
> This assigns today's date to date, returning today's calendar quarter.

=LAMBDA(RANDBETWEEN(1, 10))()
> The additional parentheses are empty because the LAMBDA formula has no +*name*+ arguments. The formula returns a random integer from 1 to 10.

See Figure 17-1 for examples of executing the LAMBDA formulas.

Formula	Result
=LAMBDA(x, x+1)(5)	6
=LAMBDA(x,y, x-y)(6, 4)	2
=LAMBDA(date, ROUNDUP(MONTH(date)/3, 0))(TODAY())	4
=LAMBDA(RANDBETWEEN(1, 10))()	3

Figure 17-1. Executing LAMBDA formulas

> If you try typing a LAMBDA formula in a cell and omit the extra parentheses, Excel displays a #CALC! error value. If you encounter this error, make sure you've included the extra parentheses.

Once you've tested the LAMBDA formula by executing it with various parameters and are satisfied it produces the correct results, you can use it with other recipes—for example, to define a custom function. See Recipe 17.4 for more details.

Discussion

The LAMBDA function works similarly to LET (see Recipe 17.1) because you use it to specify a formula that (optionally) uses variable names instead of hardcoded values. However, while LET includes arguments assigning values to these names, the LAMBDA function doesn't. Instead, you pass values for any variable names from outside the LAMBDA formula.

While this approach can initially seem awkward, it means you can use LAMBDA formulas for different purposes, such as defining custom functions. See the rest of the recipes in this chapter for more details and examples.

See Also

If you omit a LAMBDA formula's argument when calling it, Excel returns a #VALUE! error value. See Recipe 17.3 for how to avoid these errors.

17.3 Making LAMBDA Arguments Optional

Problem

You have a LAMBDA formula and want one or more of its arguments to be optional.

Solution

If you want to specify that an argument in a LAMBDA formula is optional, you can do so by putting its name in square brackets ([]). So if you have the formula =LAMBDA(x, y, x+y) and want to make y optional, you'd rewrite the formula as =LAMBDA(x, [y], x+y).

When you execute a LAMBDA formula, Excel treats any optional parameters you've omitted as empty. So typing **=LAMBDA(x, [y], x+y)(6)** assigns 6 to x, leaves y empty, and returns 6. Similarly, typing **=LAMBDA(text1, [text2], text1&text2)("Hello")** assigns "Hello" to text1, leaves text2 empty, and returns the text *Hello*.

You can check whether an argument has been omitted using ISOMITTED(*argument*), which returns TRUE if the argument has been omitted or FALSE if it hasn't. For example, if you have the formula =LAMBDA(x, [y], x*y) and want to set the y argument to 1 if it's been omitted, you could rewrite the formula as =LAMBDA(x, [y], x*IF(IS OMITTED(y), 1, y)); in this example, the IF(ISOMITTED(y), 1, y) part of the formula checks whether y has been omitted, and returns 1 if it has or the value of y if it hasn't. So typing **=LAMBDA(x, [y], x*IF(ISOMITTED(y), 1, y))(6)** assigns 6 to x, 1 to y, and returns 6.

Discussion

If you call a LAMBDA formula and omit a required argument, Excel displays a #VALUE! error value. This recipe shows how to avoid this issue by making the argument optional and checking whether it's been omitted.

17.4 Defining a Custom LAMBDA Function

Problem

You want to define a custom function you can use throughout your workbook without using macros or VBA.

Solution

Suppose you have a formula that you frequently use in a workbook, such as calculating a date's calendar or fiscal quarter (see Recipe 6.4). If you're using Excel 365, you can turn this formula into a custom function by adding a LAMBDA formula to Excel's Name Manager (see Recipe 2.6).

You create the function as follows (see Figure 17-2):

1. Use Recipe 17.2 to write and test a `LAMBDA` formula for the function in a worksheet cell. So, in this example, you'd use the `LAMBDA` formula `=LAMBDA(date, ROUNDUP(MONTH(date)/3, 0))` to compute a date's calendar quarter.

2. Choose Formulas ⇒ Defined Names ⇒ Define Name to open the New Name dialog box.

3. Type the function's name in the Name box—for example, **QUARTER**.

4. Choose Workbook from the scope list to use the function throughout the workbook.

5. Type a description of the formula in the Comment box—for example, **Returns a date's calendar quarter, an integer from 1 to 4**.

6. Type the `LAMBDA` formula you wrote in step 1 in the "Refers to" box; in this example, `=LAMBDA(date, ROUNDUP(MONTH(date)/3, 0))`. Then click OK.

New Name	— □ ✕
Name:	QUARTER
Scope:	Workbook ⌄
C**o**mment:	Returns a date's calendar quarter, an integer from 1 to 4.⬆⬇
Refers to:	=LAMBDA(date, ROUNDUP(MONTH(date)/3, 0)) ⬆
	OK Cancel
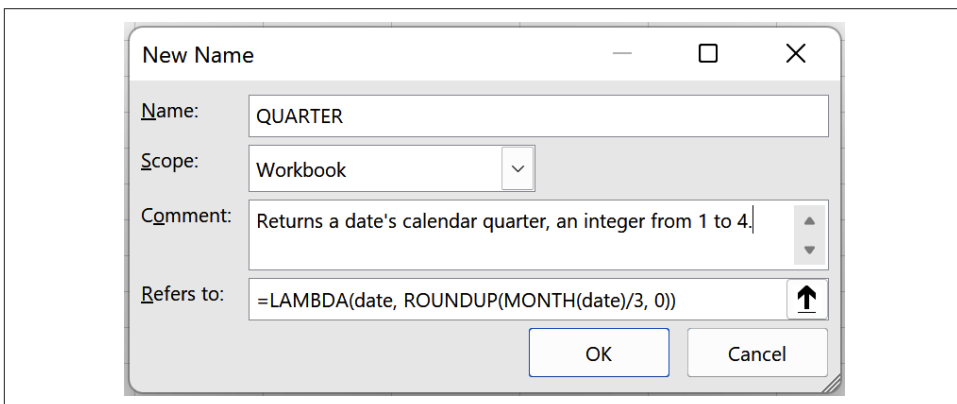
Figure 17-2. Defining the QUARTER custom function

When you click OK, Excel saves the `LAMBDA` formula as a custom function in the workbook so that you can call it using its name. So, in this example, typing `=QUARTER(TODAY())` in a cell returns today's calendar quarter.

Discussion

This powerful recipe demonstrates how to use a `LAMBDA` formula to define a custom function you can use throughout your workbook without using macros or VBA. Creating custom functions can help make your formulas more readable and save you from copying complex formulas into different cells; if a formula needs updating, you need to change it in only a single place.

See Also

To create custom functions using VBA, see Recipe 18.9.

17.5 Writing Recursive LAMBDA Formulas

Problem

You want to create a custom LAMBDA function that calls itself.

Solution

Suppose you want to create a custom LAMBDA function (see Recipe 17.4) that returns the *n*th number in the Fibonacci sequence. This sequence is 1, 1, 2, 3, 5, 8, 13, and so on, where the first and second numbers are 1, and each number after that is the sum of the previous two. So the third number in the sequence is 2 because it's the second (1) plus the first (1), the fourth number is 3 because it's the third (2) plus the second (1), and the fifth number is 5 because it's the fourth (3) plus the third (2); see Figure 17-3.

n:	1	2	3	4	5	6	7	8	9	10	11	12
Fibonacci number:	1	1	2	3	5	8	13	21	34	55	89	144

Figure 17-3. The first 12 numbers in the Fibonacci sequence

Because each Fibonacci number greater than 3 depends on the value of the previous two, you can generate each number by defining a *recursive* custom LAMBDA function: a custom LAMBDA that calls itself. So, in this example, you could create a custom function named FIBONACCI, where FIBONACCI(n) is FIBONACCI(n-1) + FIBONACCI(n-2), where n is an integer greater than 2; if n is 1 or 2, FIBONACCI(n) returns 1.

You create the function as follows (see Figure 17-4):

1. Choose Formulas ⇒ Defined Names ⇒ Define Name to open the New Name dialog box.

2. Type the function's name in the Name box—for example, **FIBONACCI**.

3. Choose Workbook from the scope list to use the function throughout the workbook.

4. Type a description of the formula in the Comment box; for example, **Returns the nth number in the Fibonacci sequence where n is a positive integer**.

5. Type the following formula in the "Refers to" box: **=LAMBDA(n, IF(n>2, FIBONACCI(n-1)+FIBONACCI(n-2), 1))**. Then click OK.
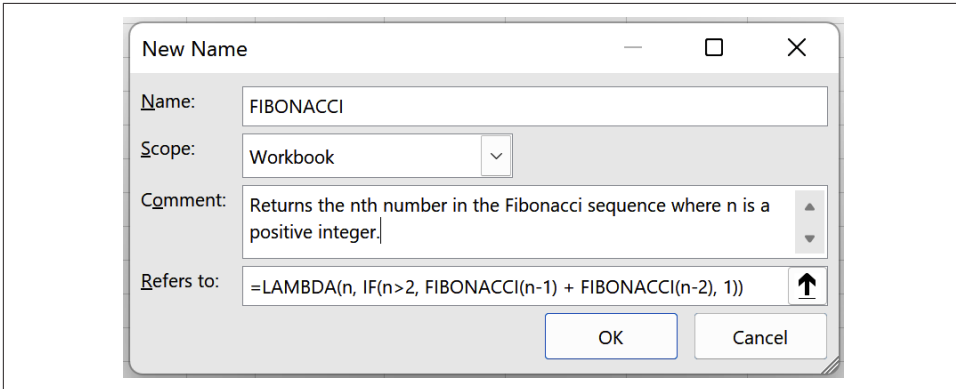
Figure 17-4. Defining the FIBONACCI custom function

Once you've created the function, you can use it to generate numbers in the Fibonacci sequence. For example, typing **=FIBONACCI(12)** returns 144, the 12th number in the sequence.

Generally, recursive custom LAMBDA functions take the form =LAMBDA(*arguments,* IF(*condition, formula, value*)), where *arguments* refers to the function's arguments, *condition* is a condition specifying when you want the function to continue iterating, *formula* is the code you want the LAMBDA function to evaluate when *condition* is TRUE, and *value* is the value you want to return when *condition* is FALSE.

Recursive custom LAMBDA functions must always include a condition controlling when the function should continue to iterate and when it should stop. If you don't include a condition, the function will keep calling itself without exiting.

Discussion

This recipe shows how to create a recursive custom LAMBDA function that calls itself. This technique means that you can use Excel formulas to create functions that loop, so you can use it to perform iterative calculations or create functions that loop through every character in a text string.

17.6 Copying a Custom LAMBDA Function to Another Workbook

Problem

You have a workbook containing custom LAMBDA functions and want to copy one or more of them to another workbook.

Solution

Let's say you used Recipe 17.4 to create a custom LAMBDA function named QUARTER, and you want to copy it to a different workbook. You can copy a single custom function to another workbook as follows:

1. Open the workbook that contains the custom function.
2. Select a cell that uses the custom function and copy it by pressing Ctrl+C or choosing Home ⇒ Clipboard ⇒ Copy.
3. Go to the workbook you want to copy the function to.
4. Select an empty cell and press Ctrl+V or choose Home ⇒ Clipboard ⇒ Paste to paste the formula. When you do so, Excel adds the custom function to the workbook's Name Manager.

If you want to copy every custom function to another workbook, including any other defined names, you can do so as follows:

1. Open the workbook containing the custom functions and the one you want to copy them to.
2. In the workbook from which you want to copy custom functions, right-click an empty worksheet and choose Move or Copy.
3. In the Move or Copy dialog box, select the workbook to which you want to copy the custom functions, place a check in the Create a Copy check box, and click OK. When you do so, Excel copies the worksheet to the destination workbook, along with any defined names, including custom functions.

Discussion

This recipe provides two ways of copying any custom LAMBDA functions you've defined to another workbook, saving you from manually re-creating them.

Because you create a custom LAMBDA function using a defined name, you can use a workbook reference to call a function in another workbook instead of copying it. See Recipe 2.6 for more details.

17.7 Applying a LAMBDA Formula to Each Column

Problem

You have an array or range and want to apply a LAMBDA formula to each of its columns and return a dynamic array of the results.

Solution

Suppose the range B1:G4 contains numbers, and you want to calculate the sum of each column multiplied by 0.5. If you're using Excel 365, one way of doing this is to use the BYCOL function.

The BYCOL function takes an array or range, applies a LAMBDA formula to each column, and returns a dynamic array containing the result for each column. You generally use the formula =BYCOL(*array, lambda*) with arguments as follows:

array
> This is the array or range you want to use in the formula—the one whose columns you want to apply the LAMBDA formula to.

lambda
> This is the LAMBDA formula, which takes the form LAMBDA(*column, formula*), where *column* is the variable name you want to use for a column in the array and *formula* is the formula you want to apply to it.

To create a dynamic array that puts the sum of each column in B1:G4 multiplied by 0.5 in cells B5:G5, you'd type **=BYCOL(B1:G4, LAMBDA(c, SUM(c)*0.5))** in cell B5, as shown in Figure 17-5.

B5			f_x	=BYCOL(B1:G4,LAMBDA(c, SUM(c)*0.5))		
	A	B	C	D	E	F	G
1		1	2	3	4	5	6
2		7	8	9	10	11	12
3		13	14	15	16	17	18
4		19	20	21	22	23	24
5	BYCOL:	20	22	24	26	28	30

Figure 17-5. Using the BYCOL function

Behind the scenes, the `BYCOL` formula works as follows:

1. The formula takes the first column in the array and passes it to the `LAMBDA` formula.

2. The `LAMBDA` formula sums the column's values and multiplies the result by 0.5. The result for this column is the first element in the dynamic array returned by the formula.

3. The formula repeats steps 1 and 2 for every column in the array, putting the result for each column in the dynamic array.

> Because the `BYCOL` function returns a dynamic array, you can use it to make functions like `SUM` spill that wouldn't otherwise do so; this can save you from copying formulas to adjacent cells.

Discussion

`BYCOL` is a `LAMBDA` helper function that applies a `LAMBDA` formula to each column in an array, returning a dynamic array of the results.

The example in this recipe uses an array of numbers, but you can use other types of values with this function too. For example, suppose B1:G4 contains text values, and you want to concatenate the values in each column using a space delimiter. In that case, you'd type the formula `=BYCOL(B1:G4, LAMBDA(c, TEXTJOIN(" ", TRUE, c)))` in cell B5; in this example, the `LAMBDA` formula uses `TEXTJOIN` to concatenate the text (see Figure 17-6).

	A	B	C	D	E	F	G
1		Q	W	E	R	T	Y
2		U	I	O	P	A	S
3		D	F	G	H	J	K
4		L	Z	X	C	V	B
5	BYCOL:	QUDL	WIFZ	EOGX	RPHC	TAJV	YSKB

Figure 17-6. Using the BYCOL function with text values

17.8 Applying a LAMBDA Formula to Each Row

Problem

You have an array or range and want to apply a LAMBDA formula to each of its rows and return a dynamic array of the results.

Solution

Suppose the range A2:F5 contains numbers, and you want to calculate the sum of each column multiplied by 0.5. If you're using Excel 365, you can do this using the BYROW function.

The BYROW function works similarly to BYCOL (see Recipe 17.7), except BYROW applies a LAMBDA formula to each row in an array or range instead of its columns and returns a dynamic array containing the result for each row. You generally use the formula =BYROW(*array*, *lambda*) with arguments as follows:

array
>   This is the array or range you want to use in the formula—the one whose rows you want to apply the LAMBDA formula to.

lambda
>   This is the LAMBDA formula, which takes the form LAMBDA(*row*, *formula*), where *row* is the variable name you want to use for a row in the array and *formula* is the formula you want to apply to it.

To create a dynamic array that puts the sum of each row in A2:F5 multiplied by 0.5 in cells G2:G5, you'd type **=BYROW(A2:F5, LAMBDA(r, SUM(r)*0.5))** in cell G2 (see Figure 17-7).

G2			fx	=BYROW(A2:F5, LAMBDA(r, SUM(r)*0.5))		
	A	B	C	D	E	F	G
1							BYROW:
2	1	2	3	4	5	6	10.5
3	7	8	9	10	11	12	28.5
4	13	14	15	16	17	18	46.5
5	19	20	21	22	23	24	64.5

Figure 17-7. Using the BYROW function

Behind the scenes, the BYROW formula works as follows:

1. The formula takes the first row in the array and passes it to the LAMBDA formula.

2. The LAMBDA formula sums the row's values and multiplies the result by 0.5. The result for this row is the first element in the dynamic array returned by the formula.

3. The formula repeats steps 1 and 2 for every row in the array, putting the result for each row in the dynamic array.

Discussion

BYROW is a LAMBDA helper function that applies a LAMBDA formula to each column in an array, returning a dynamic array of the results.

As with the BYCOL function, you can use BYROW with different value types. For example, if A2:F5 contains text values and you want to concatenate the values in each column using a space delimiter, you'd type the formula **=BYROW(A2:F5, LAMBDA(r, TEXT JOIN(" ", TRUE, r)))** in cell G2 (see Figure 17-8).

G2			⌄ ⋮ ✕ ✓ *fx*	=BYROW(A2:F5,LAMBDA(r, TEXTJOIN(" ", TRUE, r)))		
	A	B	C	D	E	F	G
1							BYROW:
2	Q	W	E	R	T	Y	Q W E R T Y
3	U	I	O	P	A	S	U I O P A S
4	D	F	G	H	J	K	D F G H J K
5	L	Z	X	C	V	B	L Z X C V B

Figure 17-8. Using the BYROW function with text values

17.9 Creating an Array of Calculated Values

Problem

You want to create an array that uses a LAMBDA formula to calculate each of its values.

Solution

Suppose you want to insert a 7 × 7 multiplication table in the range A1:G7. If you're using Excel 365, you can do this using the MAKEARRAY function.

MAKEARRAY creates a dynamic array with a specified number of rows and columns, and each value is calculated using a LAMBDA formula. You generally use the formula =MAKEARRAY(*rows, columns, lambda*) with arguments as follows:
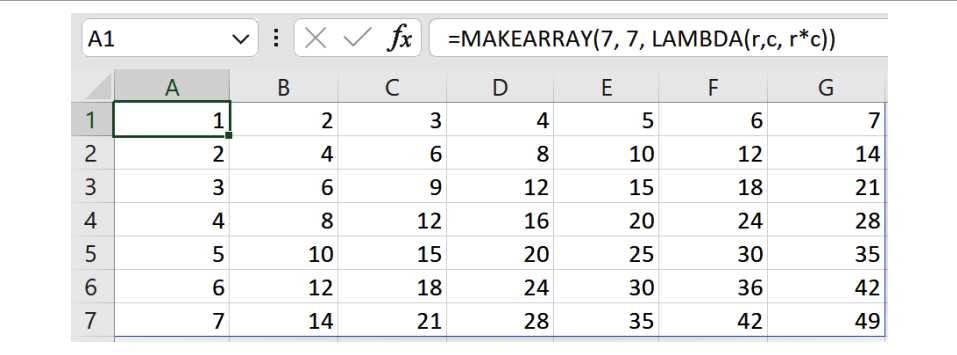
rows and columns

These specify the number of rows and columns you want in the dynamic array.

lambda

This is the `LAMBDA` formula, which takes the form `LAMBDA(row, column, for` `mula)`, where `row` and `column` are the row and column index of each value and `formula` is the formula you want to use to calculate each value.

To create a dynamic array that puts a 7 × 7 multiplication table in A1:G7, you'd type `=MAKEARRAY(7, 7, LAMBDA(r, c, r*c))` in cell A1 (see Figure 17-9).

A1			✓ fx	=MAKEARRAY(7, 7, LAMBDA(r,c, r*c))			
	A	B	C	D	E	F	G
1	1	2	3	4	5	6	7
2	2	4	6	8	10	12	14
3	3	6	9	12	15	18	21
4	4	8	12	16	20	24	28
5	5	10	15	20	25	30	35
6	6	12	18	24	30	36	42
7	7	14	21	28	35	42	49

Figure 17-9. Using the MAKEARRAY function

Behind the scenes, the `MAKEARRAY` formula works as follows:

1. The formula creates a dynamic array with seven rows and seven columns.

2. The formula moves to the first position in the dynamic array and passes its row and column index values (1 and 1) to the `LAMBDA` formula.

3. The `LAMBDA` formula multiplies the row and column index values together. The `MAKEARRAY` function then uses the result (1) for the dynamic array's first value.

4. The formula repeats steps 2 and 3 for every position in the dynamic array.

Discussion

`MAKEARRAY` is a `LAMBDA` helper function that uses a `LAMBDA` formula to create a dynamic array of a specified size.

This recipe shows how to calculate each value using its row and column index—in this example, by multiplying them. However, while you must always pass these indices to `LAMBDA`, you don't have to use them in its formula. For example, the formula `=MAKEARRAY(5, 5, LAMBDA(r, c, RAND()))` creates a 5 × 5 array of random numbers, producing the same effect as using `=RANDARRAY(5, 5)`.

17.10 Transforming the Values in Arrays

Problem

You have one or more arrays or ranges and want to use a LAMBDA formula to transform their values and return a dynamic array of the results.

Solution

Suppose A2:A6 contains one set of numbers, B2:B6 contains another, and you want to determine the highest value for each row. If you're using Excel 365, you can do so using the MAP function.

MAP takes one or more arrays, uses a LAMBDA formula to transform their values, and then returns a dynamic array of the results. You generally use the formula =MAP(*array1,array2,…,lambda*) with arguments as follows:

array1, array2, …
> These are the arrays whose values you want the LAMBDA formula to transform—in this example, the two columns.

lambda
> This is the LAMBDA formula, which takes the form LAMBDA(*array1_value, array2_value, … , formula*), where *array1_value* is a value *array1*, *array2_value* is the value in the same position in *array2*, and *formula* is the formula you want to use to transform them.

To create a dynamic array that compares A2:A6 and B2:B6 and returns the highest value in each row, you'd type **=MAP(A2:A6, B2:B6, LAMBDA(a, b, MAX(a, b)))** in cell C2 (see Figure 17-10).

Figure 17-10. Using the MAP function

Behind the scenes, the MAP formula works as follows:

1. The formula takes the values in the first position of each array and passes them to the LAMBDA formula.

2. The LAMBDA calculates the highest of these two values.

3. The MAP formula puts the result from step 2 in the first position of the dynamic array returned by the formula.

4. The formula repeats steps 1 to 3 for every position, populating the dynamic array with each result.

You can also use MAP with the AND and OR functions to perform logical tests. For example, suppose A2:A7 contains a list of company names and B2:B7 contains a list of corresponding amounts. In that case, you can create a dynamic array showing which rows have the company name *Starbuzz* and an amount greater than $5,000 by typing `=MAP(A2:A7, B2:B7, LAMBDA(company, amount, AND(company="Starbuzz", amount>5000)))` in cell C2 (see Figure 17-11).

	A	B	C	D	E	F	G	H	I	J
1	Company:	Amount:	Starbuzz > 5000?							
2	CatChat	$1,768.00	=MAP(A2:A7, B2:B7, LAMBDA(company,amount, AND(company = "Starbuzz", amount > 5000)))							
3	Manic Mango	$7,398.00	FALSE							
4	CatChat	$2,875.00	FALSE							
5	Starbuzz	$5,187.00	TRUE							
6	CatChat	$6,209.00	FALSE							
7	Starbuzz	$4,820.00	FALSE							

Figure 17-11. Using the MAP function with AND

> You can also use the MAP function to transform the values in a single array. So if A2:C5 contains numbers, you can create a dynamic array that shows each value multiplied by 0.5 by typing `=MAP(A2:C5, LAMBDA(x, x*0.5))`. However, most of the time, you can obtain the same results using a more straightforward dynamic array formula—in this example, by typing `=A2:C5*0.5` (see Recipe 3.4).

Discussion

This recipe shows how to use MAP—a LAMBDA helper function that transforms the values in one or more arrays and returns a dynamic array of the results. This is handy when you want to return a dynamic array using formulas that otherwise wouldn't do so—for example, when aggregating data or using logical functions.

17.11 Calculating Cumulative Values

Problem

You have an array and want to use a LAMBDA formula to calculate cumulative values, such as running totals or successive text concatenations.

Solution

Suppose A2:A6 contains a list of amounts and you want to generate a running total of the amounts. If you're using Excel 365, you can do so using the SCAN function.

SCAN takes an initial value and an array and uses a LAMBDA formula to calculate successive values, such as a running total. The function then returns a dynamic array of the results. You generally use the formula =SCAN(*initial_value, array, lambda*) with arguments as follows:

initial_value
> This is the calculation's initial value. You usually set this to 0 if you want to add numbers together, 1 to multiply them, and "" to concatenate text values.

array
> This is the array whose values you want to use in the LAMBDA formula.

lambda
> This is the LAMBDA formula, which takes the form LAMBDA(*accumulator, value, formula*): *accumulator* is the cumulative value (for example, the running total), *value* is the next value in the *array*, and *formula* is the formula you want to use to calculate the next running total.

To create a dynamic array that returns the running total of the amounts in A2:A6, you'd type **=SCAN(0, A2:A6, LAMBDA(total, x, total+x))** in cell B2 (see Figure 17-12).
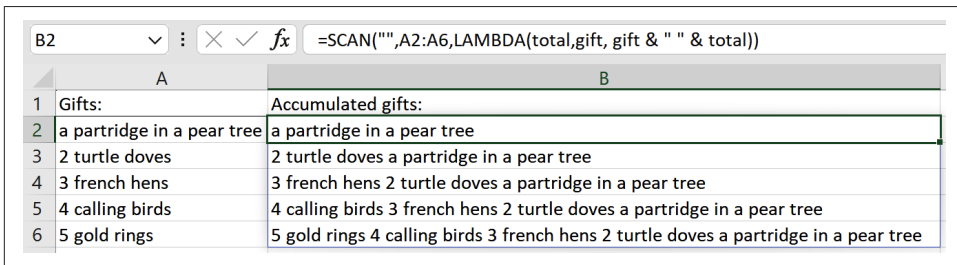
	A	B	C	D	E	F
	B2		fx	=SCAN(0, A2:A6, LAMBDA(total,x, total + x))		
1	Amount:	Running total:				
2	$1,243.00	$1,243.00				
3	$5,342.00	$6,585.00				
4	-$3,756.00	$2,829.00				
5	$123.00	$2,952.00				
6	-$5,674.00	-$2,722.00				

Figure 17-12. Using the SCAN function to return a running total

Behind the scenes, the formula works as follows, assuming the first two amounts are 1243 and 5342:

1. The SCAN formula sets the running total's initial value to 0.

2. The formula passes the first value in the array—1243—to the LAMBDA formula, which adds it to the running total. The running total is now 1243, which is the first value in the dynamic array returned by the formula.

3. The formula moves to the second value in the array—5342—and passes it to the LAMBDA formula. The LAMBDA formula adds this value to the running total, making it 6585, which is the second value in the dynamic array.

4. The formula continues this way for every value in the array, outputting the new running total for each value.

You can also use the SCAN function with text values. For example, if A2:A6 contains a list of gifts, you can create a dynamic array showing an accumulated list of gifts at each stage by typing **=SCAN("", A2:A6, LAMBDA(total, gift, gift&" "&total))** in cell B2 (see Figure 17-13).

B2	⌄ : ✕ ✓ *fx*	=SCAN("",A2:A6,LAMBDA(total,gift, gift & " " & total))
	A	B
1	Gifts:	Accumulated gifts:
2	a partridge in a pear tree	a partridge in a pear tree
3	2 turtle doves	2 turtle doves a partridge in a pear tree
4	3 french hens	3 french hens 2 turtle doves a partridge in a pear tree
5	4 calling birds	4 calling birds 3 french hens 2 turtle doves a partridge in a pear tree
6	5 gold rings	5 gold rings 4 calling birds 3 french hens 2 turtle doves a partridge in a pear tree

Figure 17-13. Using the SCAN function with text values

Discussion

This recipe shows how to use SCAN—a LAMBDA helper function that returns a dynamic array of cumulative values, such as running totals.

The SCAN function is handy if you want to see each intermediate calculation. If you want to see only the final result, you can use the REDUCE function instead (see Recipe 17.12).

17.12 Returning the Final Value of a Cumulative Calculation

Problem

You have an array and want to use a LAMBDA formula to return the end result of a cumulative calculation.

Solution

Suppose A1:E5 contains numbers, and you want to count how many are odd. If you're using Excel 365, you can do so using the REDUCE function.

REDUCE works similarly to the SCAN function (see Recipe 17.11); it takes an initial value and an array and uses a LAMBDA formula to calculate successive values. However, the REDUCE function returns only the final result.

To use REDUCE, you generally use the formula =REDUCE(*initial_value, array, lambda*) with arguments as follows:

initial_value
> This is the calculation's initial value. You usually set this to 0 if you want to add numbers together, 1 to multiply them, and "" to concatenate text values.

array
> This is the array whose values you want to use in the LAMBDA formula.

lambda
> This is the LAMBDA formula, which takes the form LAMBDA(*accumulator, value, formula*): *accumulator* is the cumulative value (for example, the running count), *value* is the next value in the *array*, and *formula* is the formula you want to use to calculate the next running count.

To count the odd values in A1:E5, you'd type **=REDUCE(0, A1:E5, LAMBDA(count, x, IF(ISODD(x), count+1, count)))** in cell H1 (see Figure 17-14).

	A	B	C	D	E	F	G	H	I
	fx	=REDUCE(0, A1:E5, LAMBDA(count,x, IF(ISODD(x), count + 1, count)))							
1	3	4	3	9	3		Count odd:	16	
2	5	7	6	9	10				
3	11	2	6	7	1				
4	6	3	7	1	5				
5	4	7	9	2	4				

Figure 17-14. Using the REDUCE function to return the final total

Behind the scenes, the formula works as follows, assuming the first two values are 3 and 4:

1. The REDUCE formula sets the running count's initial value to 0.

2. It passes the first value in the array (3) to the LAMBDA formula. Because this number is odd, the formula adds 1 to the running count, which is now 1.

3. The formula moves to the second value in the array (4) and passes it to the LAMBDA formula. Because this number is even, the formula doesn't update the running count, which is still 1.

4. The formula continues this way for every value in the array, adding 1 to the running count each time it encounters an odd value.

5. When no more values exist, the formula outputs the final running count value.

Discussion

This recipe shows how to use REDUCE—a LAMBDA helper function that reduces an array to a single result by returning the final result of a cumulative calculation. REDUCE is structured identically to the SCAN function, except SCAN returns a dynamic array showing each intermediate calculation.

Developer Tools: Macros, VBA, Controls, and XML

Excel's Developer tab contains advanced features you can use to customize and automate Excel. For example, you can automate routine or repetitive tasks, create custom functions, and use controls to develop Excel-based applications.

The recipes in this chapter demonstrate how to use these features and include the following areas:

- Recording and running macros, including using absolute and relative references
- Writing VBA code—Excel's built-in programming language—to define macros and custom functions and work with events
- Using the personal macro workbook and Excel add-ins to share custom functions and procedures
- Using Form controls, ActiveX controls, and UserForms to create dynamic user interfaces* Importing and exporting XML by mapping XML elements to Excel cells and table columns

18.1 Showing the Developer Tab

Problem

You want to use developer options such as form controls and macros but don't know where to find them.

Solution

The Excel Developer tab includes options for working with controls, macros, and VBA. Excel hides this tab by default.

To show the Developer tab, use Recipe 1.19 to customize the ribbon, placing a check against the Developer option in the Customize the Ribbon's Main Tabs section.

Discussion

This recipe shows how to enable the ribbon's Developer tab. You'll need this tab for every feature described in this chapter, including using macros, VBA, Form and ActiveX controls, and UserForms.

18.2 Recording a Macro

Problem

You have a set of actions you frequently need to repeat and want to automate them.

Solution

Suppose you frequently need to perform the same actions in Excel—for example, merging cells and rotating and centering its text. You can automate this by recording a *macro*: a set of actions saved as VBA code, which you can rerun.

You can record a macro to merge and format cells as follows:

1. Select a range of cells to merge and format—for example, A1:A5.

2. Choose Developer ⇒ Code ⇒ Record Macro to open the Record Macro dialog box.

3. Type a name for the macro in the "Macro name" box—**MergeAndRotate**, for example. The macro name must start with a letter or underscore (_), contain no spaces, and not be an existing name in the workbook.

4. In the "Shortcut key" box, type an uppercase letter to use as a keyboard shortcut for running the macro (optional). For example, typing **M** means you can run the macro via Ctrl+Shift+M. Generally, the keyboard shortcut you use to run the macro is Ctrl+Shift+*letter* where *letter* is uppercase.

5. Use the "Store macro in" drop-down box to specify where to store the macro. The options are: This Workbook, which stores the macro in the current workbook); New Workbook, which stores it in a new one; and Personal Macro Workbook, which makes the macro available to all open workbooks (see Recipe 18.3). In this example, select This Workbook to store the macro in the current workbook.

6. Type a description of the macro in the Description box (optional). So, in this example, type **Merges cells and rotates and centers the text**.

7. Click OK to start recording the macro.

8. Perform any actions you want to record. In this example, choose Home ⇒ Alignment ⇒ Merge & Center, then Home ⇒ Alignment ⇒ Orientation ⇒ Rotate Text Up, and then Home ⇒ Alignment ⇒ Middle Align.

9. Choose Developer ⇒ Code ⇒ Stop Recording to stop recording the macro.

Figure 18-1 shows the Record Macro dialog box with the completed options for the MergeAndRotate macro.
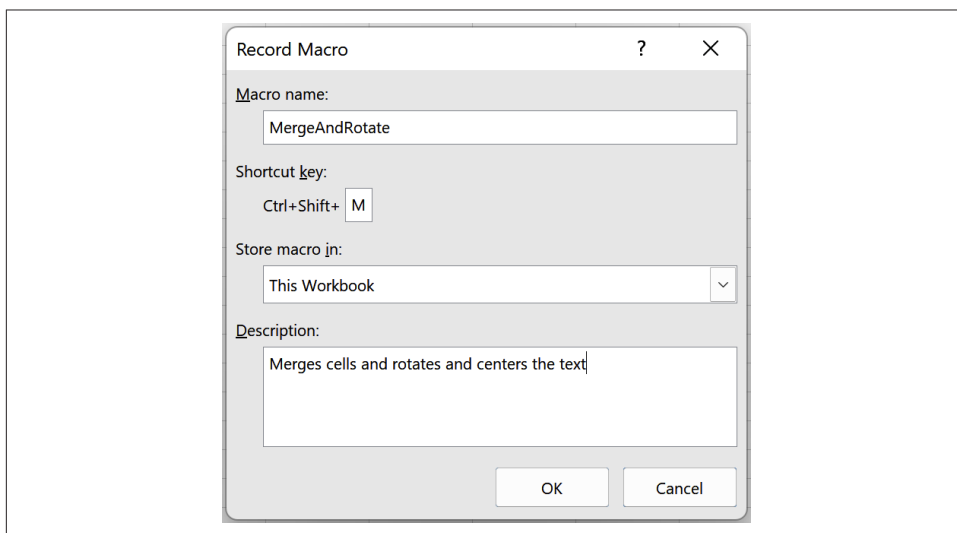
Figure 18-1. The Record Macro dialog box

You can also use a lowercase letter as a keyboard shortcut so you can run the macro by pressing Ctrl+*letter*. However, doing so may override an equivalent built-in Excel keyboard shortcut. For example, typing a lowercase letter **z** in the "Shortcut key" box overrides Excel's Undo keyboard shortcut Ctrl+Z.

Discussion

This recipe shows how to create a basic macro by recording a series of keystrokes and actions. Once you've created the macro, you can run it as often as you like using Recipe 18.5.

Behind the scenes, Excel records the macro as VBA code. You can view and edit this code using Recipe 18.6.

When you add macros or VBA code to an Excel workbook, you must save it as an Excel macro-enabled workbook with an *.xlsm* extension. If you save the file as a normal workbook, Excel deletes the macros and VBA code.

18.3 Using a Personal Macro Workbook

Problem

You want to record a macro you can access from every Excel file you open on your computer.

Solution

Suppose you want to record a macro you can use in every workbook you open. You can do so by storing it in a personal macro workbook.

A *personal macro workbook* is a file named *Personal.xlsb* stored in a hidden folder named *XLSTART*. When you add macros or custom functions (see Recipe 18.9) to this file, they're automatically available to any Excel files you open without you having to copy code between workbooks, making the personal macro workbook a handy place to store generally applicable macros and VBA code.

You add a macro to the personal macro workbook by following the steps in Recipe 18.2 but choosing the Personal Macro Workbook option in step 5. This option creates the file *Personal.xlsb* (if it doesn't already exist) and inserts the macro.

Once you've added a macro to the personal macro workbook, save it by closing Excel and, when prompted, choose the option to save the changes made. You can also save changes to the personal macro workbook by selecting its project in the Visual Basic Editor (see Recipe 18.6) and choosing File ⇒ Save.

Discussion

The personal macro workbook is a hidden file that Excel automatically creates for you when you add a macro to it. Storing macros in the personal macro workbook gives every workbook you open on your computer access to them, saving you from copying them to different workbooks. Behind the scenes, Excel looks for this file when it launches and opens the file if it exists.

Excel lists any macros added to the personal macro workbook as `PERSONAL.XLSB!` *macro_name*—for example, when using Recipe 18.4. Likewise, you call any functions in the personal macro workbook using `PERSONAL.XLSB!`*function_name* (see Recipe 18.9). If you don't want to use the `PERSONAL.XLSB!` prefix, consider using Recipe 18.20 instead.

18.4 Editing a Macro's Options

Problem

You have a macro and want to edit its keyboard shortcut or description.

Solution

When you record a macro, Excel prompts you to enter a keyboard shortcut and type a description (see Recipe 18.2). If you want to edit these options, or you want to add a keyboard shortcut or description for a macro you've created using VBA code (see Recipe 18.8), you can do so as follows:

1. Choose Developer ⇒ Code ⇒ Macros to open the Macro dialog box.

2. Select the macro you want to edit and click Options to open the Macro Options dialog box.

3. Edit the shortcut key and/or description; then click OK, followed by Cancel, to close the two dialog boxes.

Figure 18-2 shows the Macro dialog box.
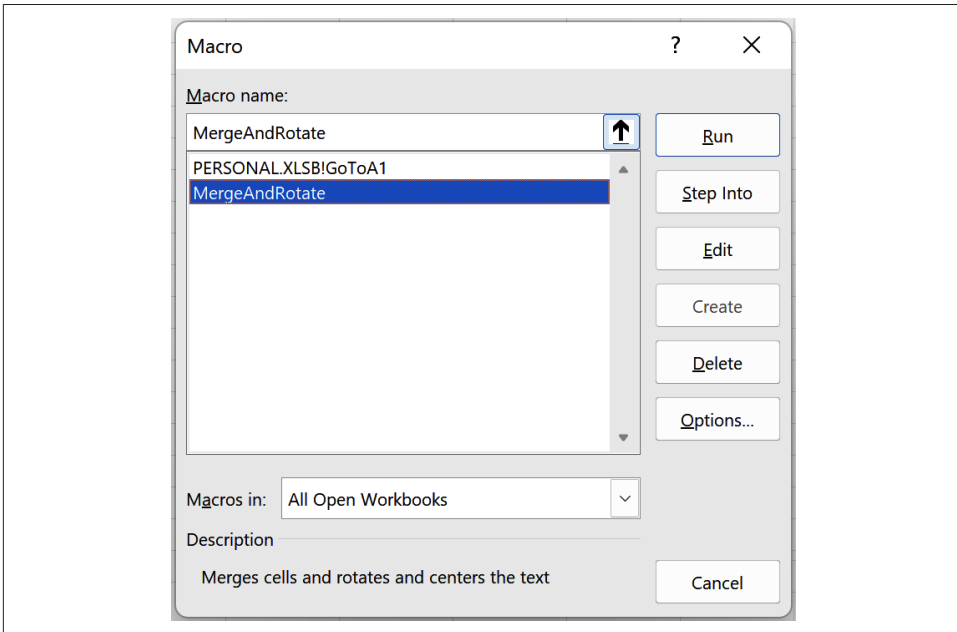
Figure 18-2. The Macro dialog box showing macros in the current and personal macro workbook

Discussion

This recipe shows how to update a macro's keyboard shortcut and/or description. As discussed in Recipe 18.2, you should generally use uppercase letters as shortcut keys to avoid overriding any of Excel's built-in keyboard shortcuts.

18.5 Running a Macro

Problem

You've created a macro and want to know how to run it.

Solution

Suppose you've recorded a macro and want to run it. You can do so using one of the following methods:

- Use the macro's keyboard shortcut, which you assigned to it using Recipe 18.2 or Recipe 18.4. For example, if you assigned the shortcut key M to the macro, you'd run the macro by pressing Ctrl+Shift+M.
- Choose Developer ⇒ Code ⇒ Macros to open the Macro dialog box, select the macro from the available list, and click Run.
- Add the macro to the Quick Access Toolbar so it runs when clicked. To add the macro, use Recipe 1.20 to modify the toolbar, select Macros from the "Choose commands from" drop-down list, select the macro, and click Add. You can then optionally click Modify to change the macro's default icon.
- Add a custom tab containing macros to the ribbon. To do so, use Recipe 1.19 to customize the ribbon, click New Tab to create a new tab and group, then add macros to the group similarly to adding them to the Quick Access Toolbar.
- Assign the macro to a button Form control so the macro runs when clicked. To do so, add the button or shape to the worksheet (see Recipes 18.17 and 13.3), right-click it, and choose Assign Macro to open the Assign Macro dialog box. Then select the macro you want the button or shape to run and click OK.
- Run the macro in an ActiveX control's Click event (see Recipe 18.18).
- Run the macro from the Visual Basic Editor (see Recipe 18.6).

Discussion

This recipe outlines various methods for running a macro, and you can choose whichever one best suits your situation. For example, if you want to run a macro in a specific worksheet, consider assigning the macro to a button Form control on that worksheet. However, if the macro is more general and you want to run it from any

workbook or worksheet, consider adding it to the Quick Access Toolbar or ribbon instead.

18.6 Viewing or Editing a Macro's VBA Code

Problem

You have a macro and want to view or edit its VBA code.

Solution

Suppose you've recorded a workbook macro named `HideSheet` that hides a worksheet. To view or edit the macro's VBA code, choose Developer ⇒ Code ⇒ Macros to open the Macro dialog box, select the macro, and click Edit. When you do so, Excel opens the *Visual Basic Editor* in a new window: an editor that lets you view, edit, and interact with VBA code (see Figure 18-3).
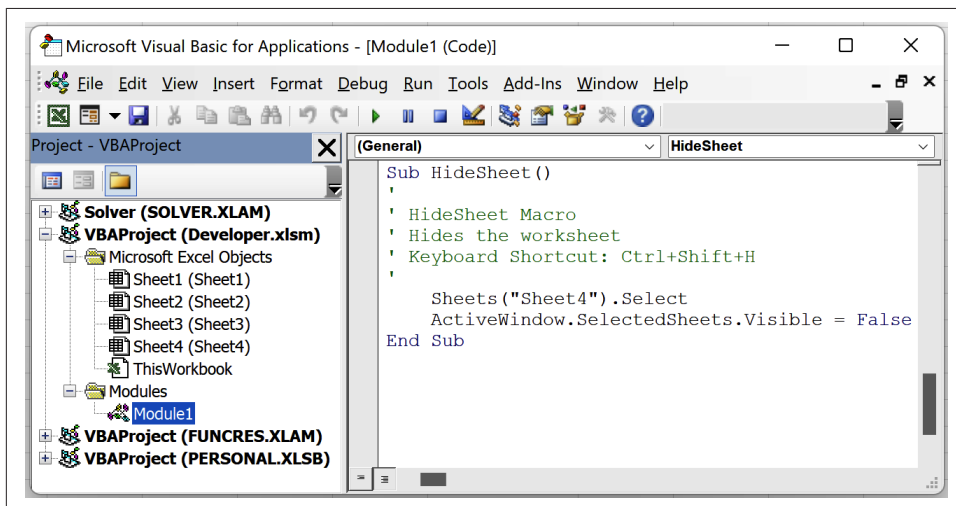
Figure 18-3. The Visual Basic Editor showing the code for the HideSheet macro

> You can also open the Visual Basic Editor by choosing Developer ⇒ Code ⇒ Visual Basic. Use this option to view or edit the code for any macros stored in the personal macro workbook because clicking Edit in the Macro dialog box may display an error message.

The Visual Basic Editor window displays a project explorer on the left containing a list of VBA projects—one for each open workbook, the personal macro workbook (listed as `PERSONAL.XLSB`), and any add-ins (see Recipe 18.20). When you expand

these items by double-clicking them, the explorer usually displays two folders—
Microsoft Excel Objects and *Modules*—for each.

The *Modules* folder contains one or more *modules*: organizational structures that
store VBA code. When you record a macro, Excel usually adds it to a module named
Module*n*, where *n* is a number. To view the VBA code created for any macros, expand
the *Modules* folder, and double-click each module to open its code window.

The module's code window displays all the VBA code stored there. For example, if
you've recorded a macro named HideSheet that hides a specific worksheet (in this
case, a worksheet named Sheet4), the code in the module's code window will look
similar to the following:

```
Sub HideSheet()
'
' HideSheet Macro
' Hides the worksheet
' Keyboard Shortcut: Ctrl+Shift+H
'
    Sheets("Sheet4").Select
    ActiveWindow.SelectedSheets.Visible = False
End Sub
```

In this example, the lines

```
Sub HideSheet()

End Sub
```

specify the name of the macro and its start and end. To be more precise, the Sub and
End Sub lines define the start and end of a *subprocedure*—a set of actions or instruc-
tions written in VBA—named HideSheet().

The lines:

```
'
' HideSheet Macro
' Hides the worksheet
' Keyboard Shortcut: Ctrl+Shift+H
'
```

are comments, denoted by the ' character at the start of each row. Excel automatically
adds these lines when you record the macro.

> If you edit a macro's options (see Recipe 18.4), Excel doesn't auto-
> matically update the comments in the VBA code. However, you can
> manually update these comments by editing them in the VBA code
> window.

Finally, the lines

```
Sheets("Sheet4").Select
ActiveWindow.SelectedSheets.Visible = False
```

describe the macro's actions. In this example, the macro selects the worksheet named Sheet4 and sets its visibility to `False`.

You can change the VBA code by editing it in the code window. For example, you could change the macro to hide the currently active worksheet (instead of Sheet4) by updating the code as follows:

```
Sub HideSheet()
'
' HideSheet Macro
' Hides the worksheet
' Keyboard Shortcut: Ctrl+Shift+H
'
    ActiveSheet.Visible = xlSheetHidden
End Sub
```

Once you've finished editing the code, you can run it in the Visual Basic Editor by clicking the Run Sub/UserForm command in the Editor's toolbar or choosing Run ⇒ Run Sub/UserForm.

Discussion

When you record a macro, Excel saves its actions as VBA code. This recipe provides an overview of how to view and edit this code using the Visual Basic Editor, which can be handy if you want to tweak or generalize the code generated by Excel.

Generally, VBA code lets you automate actions, interact with Excel worksheets, workbooks, or other Office applications, and customize how you use Excel. The following is a quick guide to some useful VBA syntax:

- To refer to the currently active worksheet, use `ActiveSheet`; to refer to a specific worksheet, use `Sheets("sheet_name")` .

- Use a `.` to access properties whose values you can get and set—for example, `ActiveSheet.Visible`. You can also use a `.` to access any *methods*: actions you can take on an object. For example, use `Sheets("Sheet4").Select` to select Sheet4.

- Use `Range("cell_or_range")` to refer to a cell or a range—for example, `Range("C2")`; see Recipe 18.7.

- Use `Dim` to declare a variable; see Recipe 18.8 for an example.

- You can loop through each item in a collection using a `For-Each` construct; see Recipe 18.8 for an example of this syntax.

- You can use an `If-Then` construct to perform actions if one or more conditions are `True` or `False`; see Recipe 18.15 for an example.

- You can use the `Select-Case` construct instead of `If-Then` to test a single condition's value; see Recipe 18.16 for an example.

- Use a Space followed by an underscore (_) to break a single statement into multiple lines. See Recipe 18.10 for an example of this.

18.7 Using Absolute and Relative References

Problem

You want to record a macro and choose whether it should refer to another cell by its absolute cell reference or its position relative to another cell.

Solution

Suppose you want to record a macro that navigates to another cell and adds some text. When you do so, you can choose whether to generate VBA code that uses the cell's absolute reference—for example, cell C2—or that refers to its position relative to another cell—for example, one row down and two columns to the right.

By default, recording a macro generates code that uses *absolute references*: references that explicitly refer to a cell's position. For example, if you select cell A1 and then record a macro that enters the text *Hello* in cell C2, the generated VBA code looks like this:

```
Sub AbsoluteDemo()
'
' AbsoluteDemo Macro
' Demonstrates absolute references
' Keyboard Shortcut: Ctrl+Shift+A
'
    Range("C2").Select
    ActiveCell.FormulaR1C1 = "Hello"
End Sub
```

In this example, the code `Range("C2").Select` explicitly navigates to cell C2, regardless of which cell is active when you run the macro.

An alternative approach is to use *relative references*: references relative to another cell, similar to Recipe 7.14. To generate the equivalent code using relative references, choose Developer ⇒ Code ⇒ Use Relative References to enable this option, select cell A1, and then record a macro entering the text *Hello* in cell C2. The generated VBA code looks like this:

```
Sub RelativeDemo()
'
' RelativeDemo Macro
' Demonstrates relative references
' Keyboard Shortcut: Ctrl+Shift+R
'
    ActiveCell.Offset(1, 2).Range("A1").Select
    ActiveCell.FormulaR1C1 = "Hello"
End Sub
```

In this example, the code `ActiveCell.Offset(1, 2).Range("A1").Select` navigates to the cell one row down and two columns to the right of the active one. Calling the macro from cell A1 populates cell C2 (one row down and two columns to the right of cell A1), and calling it from cell B2 populates cell D3 (one row down and two columns to the right of cell B2).

To revert to using absolute references, choose Developer ⇒ Code ⇒ Use Relative References again to turn off the Use Relative References option.

Discussion

This recipe shows how to record macros using absolute and relative references and their impact on the generated VBA code. In general, use absolute references when you want to perform an action on a specific cell or range, and relative references when you want to perform an action relative to the current selection.

18.8 Creating a Macro by Writing VBA

Problem

You want to create a macro by writing a VBA procedure instead of recording keystrokes and actions.

Solution

Suppose you want to add a macro named `UnhideWorksheets` to the current workbook that unhides any hidden worksheets. You can do so by typing the macro's VBA code directly in the Visual Basic Editor as follows:

1. Choose Developer ⇒ Code ⇒ Visual Basic to open the Visual Basic Editor.
2. In the project explorer, navigate to the current workbook's VBA project and double-click the module you want to add the macro code to. If the workbook doesn't include any modules, right-click the project and choose Insert ⇒ Module to create one.

3. In the module's code window, enter the code for the new macro (in this case, the one named `UnhideWorksheets` shown in Example 18-1), making sure you don't edit the code of any existing macros. One way of doing this is to scroll to the end of the file and enter the new code on a new line.

4. Optionally, use Recipe 18.4 to add a keyboard shortcut and description for the new macro.

The code for the `UnhideWorksheets` macro is shown in Example 18-1. The code first declares a `Worksheet` variable named `ws` and then uses it to loop through every worksheet, making each one visible.

Example 18-1. VBA code for the `UnhideWorksheets` macro

```
Sub UnhideWorksheets()
    Dim ws As Worksheet
    'Loop through each worksheet and make each one visible
    For Each ws In Worksheets
        ws.Visible = xlSheetVisible
    Next
End Sub
```

Discussion

This recipe shows how to create a macro using VBA code instead of recording keystrokes and actions. This approach is handy if you want to develop more complex macros and are already familiar with writing VBA code, or you have some prewritten code you want to be able to run.

18.9 Creating a Custom VBA Function

Problem

You have an Excel formula you commonly use and want to rewrite it as a VBA function.

Solution

Suppose you have a formula you frequently use in a workbook, such as generating a random integer between 1 and 10 or calculating a date's calendar or fiscal quarter (see Recipe 6.4). You can make this formula easier to use by rewriting it as a VBA custom function.

You create a VBA function by following steps similar to those in Recipe 18.8. The main differences are that functions use `Function` and `End Function` instead of `Sub` and `End Sub` to denote the function's start and end, and you usually use a function to

return a value. For example, to add a `RandTen` function to the current workbook that returns a random integer between 1 and 10, you'd follow these steps:

1. Choose Developer ⇒ Code ⇒ Visual Basic to open the Visual Basic Editor.

2. In the project explorer, navigate to the current workbook's VBA project and double-click the module you want to add the function to. If the workbook doesn't include any modules, right-click the project and choose Insert ⇒ Module to create one.

3. In the module's code window, enter the code for the new `RandTen` function as follows, making sure you don't edit any existing code:

```
Function RandTen() As Integer
    'Return a random integer from 1 to 10
    RandTen = WorksheetFunction.RandBetween(1, 10)
End Function
```

In this example, the first line of code declares a function named `RandTen` that returns an integer. The function then generates an integer between 1 and 10 using Excel's RANDBETWEEN function; it calls this function using the code `WorksheetFunction.Rand Between(1, 10)`. The code assigns this integer to `RandTen`—the name of the function—which tells VBA to return this value. Finally, the `End Function` line signifies the end of the function.

> `WorksheetFunction` lets you call an Excel worksheet function, and it's handy for functions with no VBA equivalent. For example, using `WorksheetFunction.RandBetween` in VBA calls Excel's RANDBETWEEN function.

To specify that a VBA function accepts one or more arguments, you put them in parentheses as a comma-separated list. For example, to insert a function named `Quarter` that returns a month's calendar quarter, you could use the following code:

```
Function Quarter(d As Date) As Integer
    'Return the calendar quarter
    Quarter = (Month(d) + 2) \ 3
End Function
```

In this example, the function uses the code `Quarter(d As Date)` to define a function named `Quarter` with a single date argument named d.

Similarly, you could use the following code to add a function named `FiscalQuarter` that returns a date's fiscal quarter:

```
Function FiscalQuarter(d As Date) As Integer
    Dim m, qtr As Integer
```

```
    'Get the month
    m = Month(d)
    'Calculate and return the fiscal quarter
    qtr = Switch(m < 4, 4, m < 7, 1, m < 10, 2, m < 13, 3)
    FiscalQuarter = qtr
End Function
```

Once you've created a custom function, you can call it from a worksheet cell like any other function. So if you've added the RandTen function to the current workbook, you'd call it by typing **=RandTen()** in a worksheet cell. Similarly, if you've added the Quarter function to the current workbook and want to use it to return today's calendar quarter, type **=QUARTER(TODAY())** in a cell.

Similarly to macros, any functions you add to the personal macro workbook are automatically available to any open workbook. Generally, you call a function from the personal macro workbook using the syntax =PERSONAL.XLSB!*function_name*, where *function_name* is the name of the VBA function. So if you've added the RandTen function to the personal macro workbook, you'd call it by typing **=PERSONAL.XLSB! RandTen()**.

> If you want to use custom functions in any open workbook but want to avoid using the PERSONAL.XLSB! prefix, you can create a custom Excel add-in instead. See Recipe 18.20 for more details.

To remove a function you've created, delete its VBA code using the steps in Recipe 18.13.

Discussion

This recipe shows how to define a custom function using VBA code. Creating custom functions can help make your formulas more readable and save you from copying complex formulas into different cells; if the formula needs updating, you need to change it in only a single place.

See Also

If you're using Excel 365, you can also create custom functions using Recipe 17.4.

18.10 Using Worksheet and Workbook Events

Problem

You want to run some VBA code when you perform an action in the current worksheet or workbook.

Solution

Suppose you want to run some VBA code when you perform an action in Excel, such as double-clicking a cell in a worksheet. You can do so using an *event*: an action that can cause VBA code to run. If you want to run code when you double-click a cell, you add the code to the worksheet's double-click event.

Worksheet events occur for actions in a specific worksheet, such as double-clicking or changing a cell value. You add code to a worksheet event as follows:

1. Choose Developer ⇒ Code ⇒ Visual Basic to open the Visual Basic Editor.

2. In the project explorer, double-click the worksheet whose event you want to add code to, to open its code window. For example, if you want to add code to an event in Sheet1 of the current workbook, navigate to the current workbook's VBA project, open its Microsoft Excel Object folder, and double-click Sheet1.

3. At the top of the worksheet's code window, select Worksheet from the Object drop-down list on the left.

4. Use the Procedure drop-down list (to the right of the Object drop-down list) to select the event to which you want to add code. For example, to write code for the worksheet's double-click event, you'd select BeforeDoubleClick. When you select an event, the Visual Basic Editor adds a corresponding empty procedure to the worksheet's code. So if you select the BeforeDoubleClick event, the Editor inserts a procedure named `Worksheet_BeforeDoubleClick`.

5. You edit the procedure added in the previous step to specify what should happen when the event occurs. For example, to change the background color of a worksheet cell when you double-click it, you'd update the code to the following:

```
Private Sub Worksheet_BeforeDoubleClick(ByVal Target As Range, _
                                        Cancel As Boolean)
    'Change the target cell's color
    Target.Interior.Color = 255
End Sub
```

Figure 18-4 shows the worksheet's BeforeDoubleClick event code in the Visual Basic Editor.
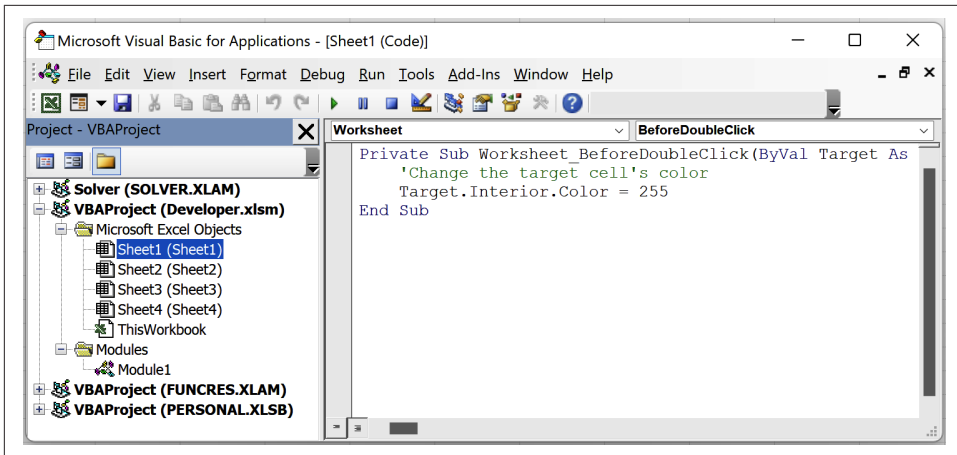
Figure 18-4. The Visual Basic Editor

Workbook events occur for actions in a specific workbook, such as opening the workbook or inserting a new worksheet. Adding code to a workbook event is similar to adding it to a worksheet event, except you double-click the ThisWorkbook object in step 2 and select Workbook from the Object drop-down list in step 3. For example, to display a message when you open the workbook, you'd use the following steps:

1. Choose Developer ⇒ Code ⇒ Visual Basic to open the Visual Basic Editor.

2. In the project explorer, navigate to the current workbook's VBA project, open its Microsoft Excel Object folder, and double-click ThisWorkbook.

3. At the top of the worksheet's code window, select Workbook from the Object drop-down list on the left, and then select the event you want to use from the Procedure drop-down list on the right—in this example, Open. When you do so, the Visual Basic Editor creates a corresponding procedure.

4. Edit the procedure added in the previous step. So to display a message when you open the workbook (see Recipe 18.16), you'd update the code as follows:

```
Private Sub Workbook_Open()
    'Display a message
    MsgBox("Hello!")
End Sub
```

Figure 18-5 shows the workbook's Open event code in the Visual Basic Editor.
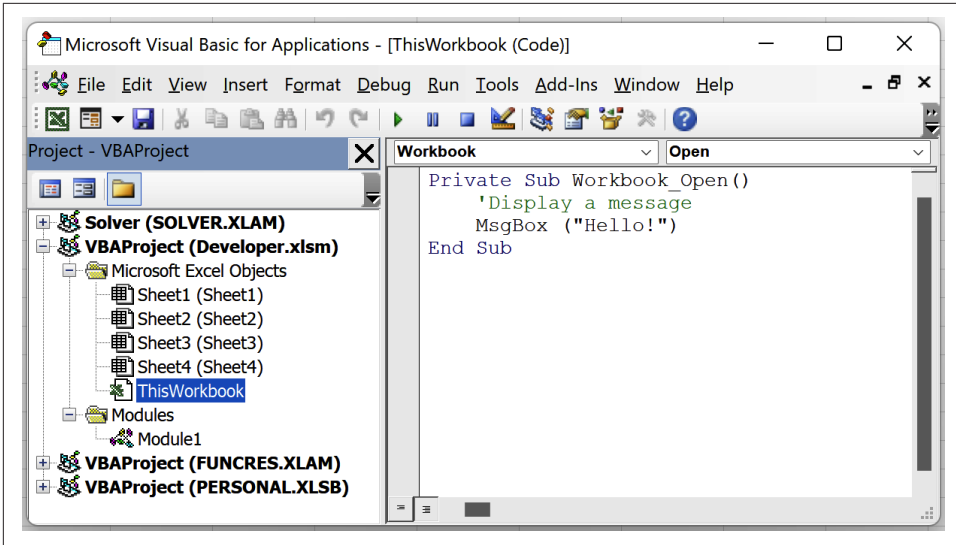
Figure 18-5. The Visual Basic Editor

Discussion

This recipe shows how to use worksheet and workbook events to run VBA code when specific actions occur. You can browse a list of possible events by examining the options in the Procedure drop-down at the top of the worksheet or workbook code window.

Excel also includes application events, which run when specific actions occur in the application. However, these are generally less useful than worksheet and workbook events because they apply to the entire Excel application.

18.11 Overriding Keystrokes with OnKey

Problem

You have a macro or procedure and want to run it when you press a specific key combination.

Solution

When you create a macro, you can assign a letter shortcut key to it so the macro runs when you press Ctrl+*letter* for lowercase letters or Ctrl+Shift+*letter* for uppercase (see Recipe 18.2).

If you want to run a macro using another key combination, do so by adding the code `Application.OnKey` *key, macro* to a VBA function or procedure, where *key* is the

key or key combination you want to use as a keyboard shortcut and *macro* is the macro or procedure you want to run when you press it. For example, the following code defines a macro named `SetupShortcutKeys`, which, when run, overrides the Page Up key so it calls the `PageUpShortcut` macro when pressed and selects cell A1:

```
Sub SetupShortcutKeys()
    'Call PageUpShortcut() when Page Up is pressed
    Application.OnKey "{PgUp}", "PageUpShortcut"
End Sub

Sub PageUpShortcut()
    On Error Resume Next
    'Go to cell A1
    Range("A1").Select
End Sub
```

In this example, the *key* argument in the `SetupShortcutKeys` macro is "{PgUp}", which corresponds to the Page Up key; you can find a complete list of key characters by searching the VBA Help for *OnKey*. The line

```
On Error Resume Next
```

specifies that if an error occurs (for example, because you've tried to press the Page Up key in a chart sheet with no cells), the code should continue execution from the next line.

Figure 18-6 shows the code for `SetupShortcutKeys` and `PageUpShortcut` in a workbook's code module.
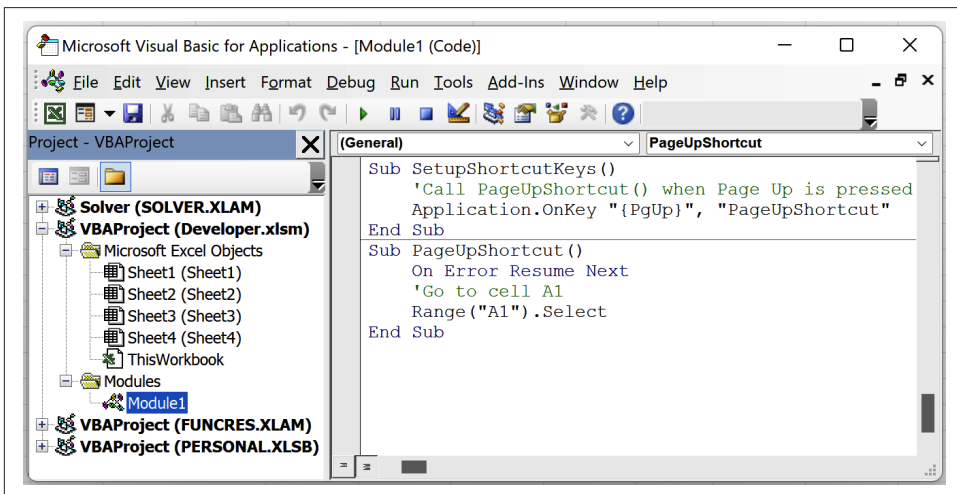
Figure 18-6. The Visual Basic Editor showing the code to set up shortcut keys

To stop overriding a key combination so it reverts to its default behavior, use the following code:

```
Application.OnKey "key"
```

For example, the following code defines a macro named `CancelShortcutKeys`, which, when run, makes the Page Up key revert to its default behavior:

```
Sub CancelShortcutKeys()
    Application.OnKey "{PgUp}"
End Sub
```

You can also turn off a shortcut key using this code:

```
Application.OnKey "key", ""
```

Discussion

This recipe shows how to use the `Application.OnKey` method to define shortcut keys. Behind the scenes, this technique uses the application's `OnKey` event, which listens for and responds to keystrokes.

18.12 Scheduling Code with OnTime

Problem

You have a macro or procedure and want to run it at a specific time or after a certain interval.

Solution

Suppose you want to run a macro or procedure at a specific time—for example, to display a message dialog box at 2 p.m. You can do so by adding `Application.OnTime` *time, macro* to a VBA function or procedure, where *macro* is the macro you want to run and *time* is when you want to run it. For example, the following code defines a macro named `SetReminders`, which, when run, executes the `TeaBreakReminder` macro at 2 p.m.:

```
Sub SetReminders()
    'Call TeaBreakReminder() at 2 p.m.
    Application.OnTime TimeSerial(14, 0, 0), "TeaBreakReminder"
End Sub

Sub TeaBreakReminder()
    MsgBox("Time for a break!")
End Sub
```

This example uses the `TimeSerial` function to specify the time in hours, minutes, and seconds, so `TimeSerial(14, 0, 0)` corresponds to 2 p.m.

Figure 18-7 shows the code for `SetReminders` and `TeaBreakReminder` in a workbook's code module.
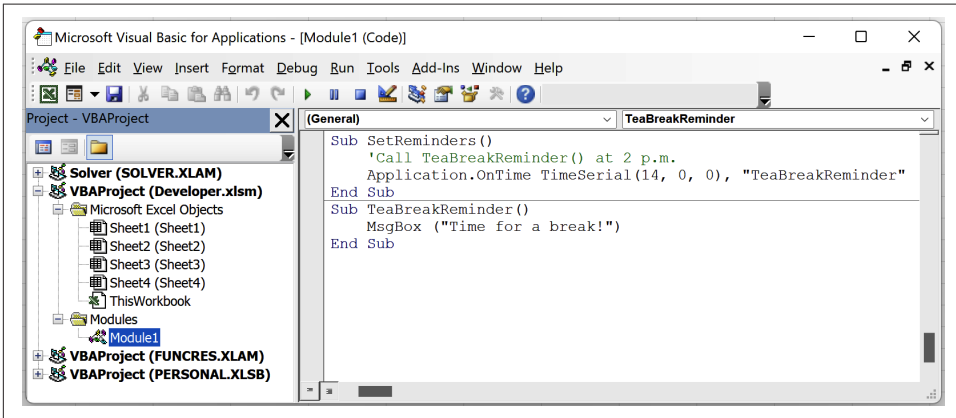
Figure 18-7. The Visual Basic Editor showing the code to schedule running code

You can also use the `Application.OnTime` method to run a macro or procedure after a specific interval. For example, to run the `TeaBreakReminder` macro 15 minutes from now, you'd use the line

```
Application.OnTime Now + TimeSerial(0, 15, 0), "TeaBreakReminder"
```

where `Now` is a VBA function returning the current time.

> For the code to run at the specified time, you must execute the macro containing the call to `Application.OnTime` and keep your computer switched on with Excel running. If you don't execute the macro or you close Excel, the code won't run.

Discussion

This recipe shows how to use the `Application.OnTime` method to run a macro or procedure at a specified time. Behind the scenes, this method uses the application's `OnTime` event.

18.13 Deleting a Macro or Function

Problem

You have a macro or function in a VBA project and want to delete it.

Solution

If you want to remove a macro from a workbook, the simplest way is as follows:

1. Open the workbook containing the macro.
2. Choose Developer ⇒ Code ⇒ Macros to open the Macro dialog box.
3. Select the macro you want to remove and then click Delete.

If you want to delete a function or remove a macro from your system's personal macro workbook folder, you can delete its VBA code with the following steps:

1. Choose Developer ⇒ Code ⇒ Visual Basic to open the Visual Basic Editor.
2. Expand the VBA project containing the code by double-clicking it.
3. Double-click the module in its *Modules* folder to open its code window.
4. In the code window, navigate to the VBA code by selecting its name from the Procedure drop-down list in the window's top-right.
5. Select the code you want to remove—from its opening `Sub` or `Function` line to its closing `End Sub` or `End Function`—and then press Delete.

> An alternative to deleting a macro or function is to comment out its lines by prefixing each one with a `'`. Doing so means you can no longer run the code, and you can remove the comments to get the code back.

To delete an entire module including its code, right-click it and choose Remove.

Discussion

This recipe outlines deleting a macro or function from a VBA project. Removing macros from workbooks is generally easier because you can use the Macro dialog box instead of deleting the underlying VBA code.

18.14 Copying Code to Another VBA Project

Problem

You have code in one VBA project and want to copy it to another.

Solution

Suppose you have code in a VBA project that you want to copy to another. You can use several techniques, depending on your situation.

If you want to copy an individual macro or function from one project to another, you can do so by following these steps:

1. Open both projects and then open the Visual Basic Editor by choosing Developer ⇒ Code ⇒ Visual Basic.

2. In the project explorer, double-click the module containing the code you want to copy, select the code in the code window, and press Ctrl+C to copy it.

3. Open the code window for the module you want to copy the code to—if needed, you can insert a new module by right-clicking the project and choosing Insert ⇒ Module. Then, paste the code in the code window by pressing Ctrl+V.

If you want to copy an entire module from one VBA project to another, you can follow these steps instead:

1. Open both projects and then open the Visual Basic Editor.

2. In the project explorer, click the module you want to copy and drag it to the destination project.

If you want to copy a module to a workbook on another computer or share it with someone else, you can export the module as a *.bas* file and then import it into the destination workbook. This approach is handy if, for example, you want to copy the code in your personal macro workbook to another computer. The steps to export and import a module are as follows:

1. Open the workbook containing the module you want to export and then open the Visual Basic Editor.

2. In the project explorer, right-click the module and choose Export File. When prompted, browse to where you want to save the file, name it, and click Save.

3. To import the *.bas* file, open the workbook to which you want to copy the code, and then open the Visual Basic Editor.

4. In the project explorer, right-click the workbook and choose Import File. When prompted, navigate to the *.bas* file and click Open.

Discussion

This recipe covers several techniques for copying code from one VBA project to another. Generally, choose whichever approach is most straightforward and appropriate for your situation.

18.15 Debugging VBA Code

Problem

You have some VBA code and want to know how to debug it.

Solution

Suppose you've added the following procedure to a module to test whether the value in the active cell is numeric:

```
Sub TestCellValue()
    Dim val
    val = ActiveCell.Value
    If IsNumeric(val) Then
        MsgBox ("Numeric value")
    Else
        MsgBox ("Non-numeric value")
    End If
End Sub
```

You want to debug the code to ensure it's working as intended.

The Visual Basic Editor includes several tools and features to debug code. For example, you can pause execution at specific points, step through the running code line by line, and examine the values of any variables.

To pause execution at a specific line, you can add a breakpoint to it. You add the breakpoint by clicking the code window margin to the left of the line or clicking somewhere in the line and choosing Debug ⇒ Toggle Breakpoint; doing so highlights the line and puts a circle icon in the margin next to it, as shown in Figure 18-8.
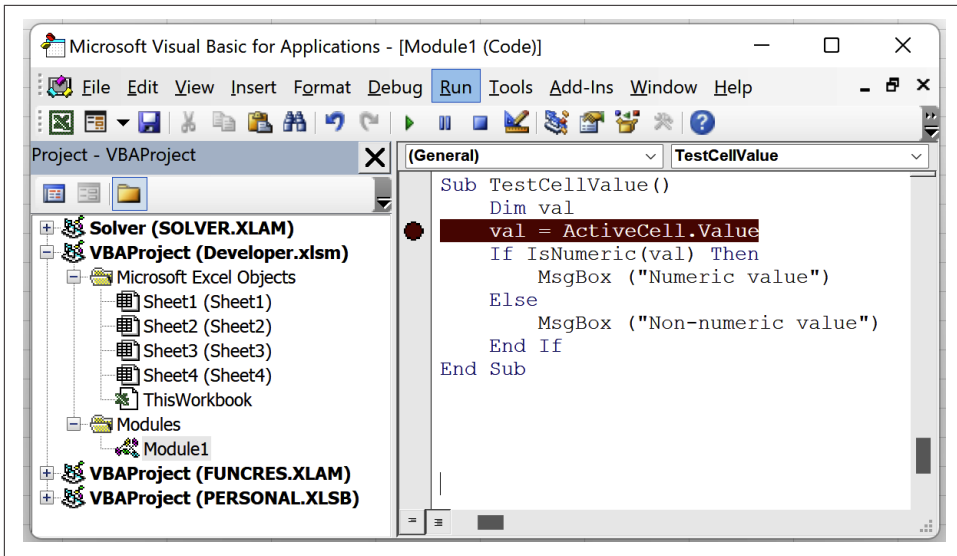
Figure 18-8. The Visual Basic Editor showing a breakpoint

Once you've added a breakpoint, running the code pauses execution when it reaches the breakpoint. You can then choose one of these options from the Debug menu to step through the code:

Step Into (F8)

> This traces each line of code and steps into any procedures they call.

Step Over (Shift+F8)

> This behaves similarly to Step Into, except it steps over any procedures the code calls and doesn't go into them.

Step Out (Ctrl+Shift+F8)

> This executes the remaining code in a procedure, and you can use it to exit a procedure you've stepped into.

Run To Cursor (Ctrl+F8)

> This lets you execute the code up to where you've inserted the cursor, letting you step over sections of code.

You can also choose Run ⇒ Continue to continue running the code until it encounters another breakpoint or Run ⇒ Reset to stop execution so you can start again.

When debugging code, you often need to experiment with setting multiple breakpoints. If you no longer need a specific breakpoint, you can remove it by clicking its icon in the margin or clicking its line and choosing Debug ⇒ Toggle Breakpoint. To remove all breakpoints, choose Debug ⇒ Clear All Breakpoints.

When you've paused execution, you can monitor the value of any variables using the Locals window. You open this window by choosing View ⇒ Locals window, and it lists each variable's name, current value, and data type (see Figure 18-9). The window also includes a call stack list toward the top, showing a queue of the executing procedures.
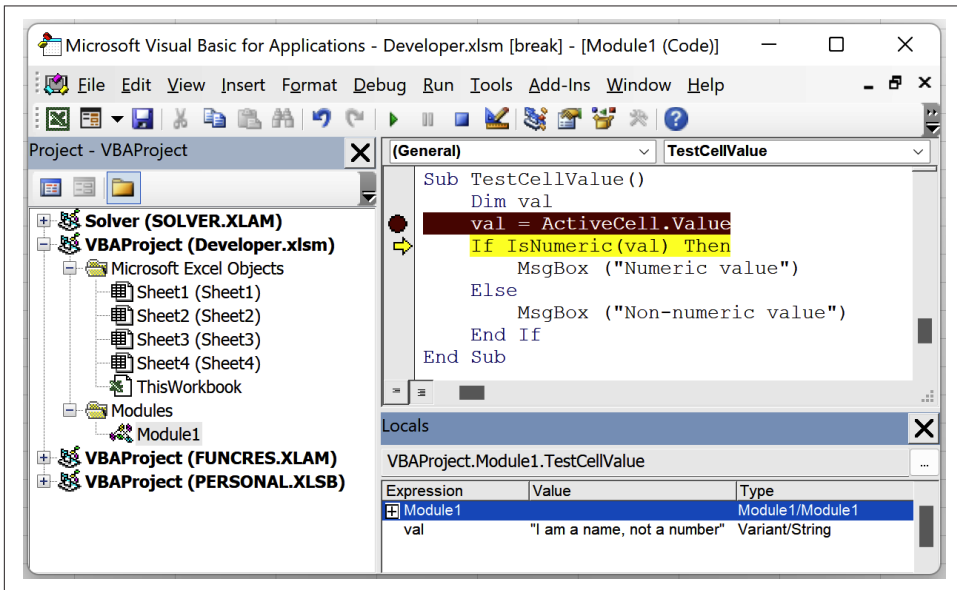
Figure 18-9. The Visual Basic Editor showing the Locals window

The Locals window displays every variable. If you want to focus on specific ones or particular expressions, you can use the Watch window instead, which you open by choosing View ⇒ Watch window (see Figure 18-10). The Watch window is similar to the Locals window, except it includes an extra Context column—the variable or expression's module or procedure—and doesn't include a call stack list.
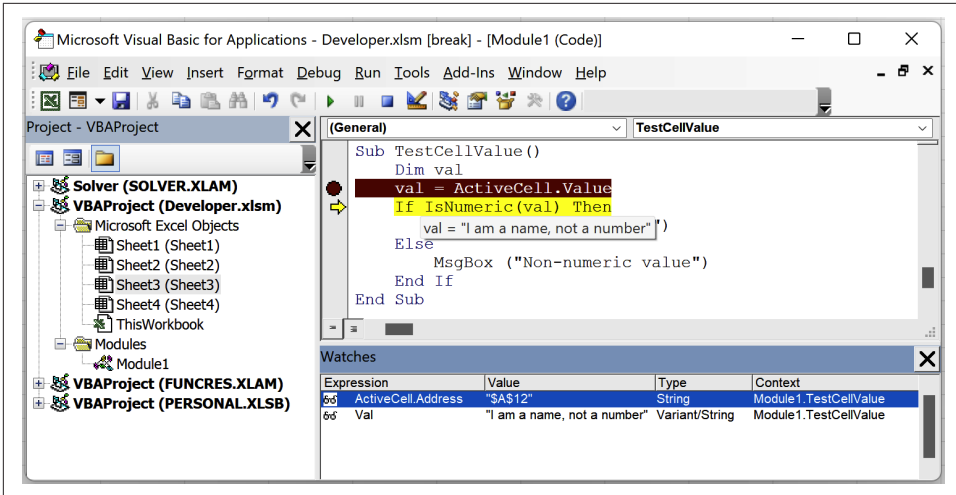
Figure 18-10. The Visual Basic Editor showing the Watch window

You add a variable or expression to the Watch window as follows:

1. Choose Debug ⇒ Add Watch to open the Add Watch dialog box.

2. Type the variable name or expression in the Expression box—for example, **ActiveCell.Address**.

3. Use the module and procedure drop-down lists in the Context section to specify when you want the expression evaluated, usually the current procedure and module.

4. Select a Watch Type; choose Watch Expression to add the expression to the Watch window, or choose Break When Value Is True or Break When Value Changes to additionally pause execution when the variable or expression is `True` or changes value. In this example, choose the Watch Expression option and click OK to add the watch.

Figure 18-11 shows the Add Watch dialog box.

To edit a watch you've added, right-click the watch and choose Edit Watch to open the Edit Watch dialog box. To delete a watch, select the watch and press the Delete key or right-click it and choose Delete Watch.

You can also debug VBA code by outputting ad hoc debug messages. To do so, add the statement `Debug.Print` *text* to the code, where *text* is the message you want to output. For example, the line

```
Debug.Print "Active cell is " & ActiveCell.Address
```

outputs the active cell's address.
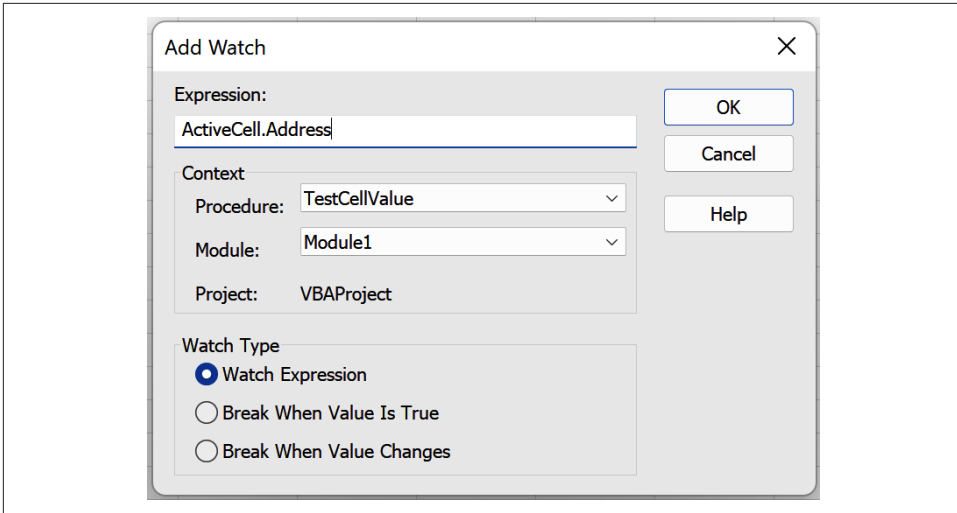
Figure 18-11. The Visual Basic Editor showing the Add Watch dialog box

Any debug messages you add get displayed in the Immediate window, which you open by choosing View ⇒ Immediate window. You can also use the Immediate window to get or set variable values while the code is paused. For example, typing **? val** in the Immediate window gets the current value of the variable named val, and typing **val = 7** sets it to 7 (see Figure 18-12).
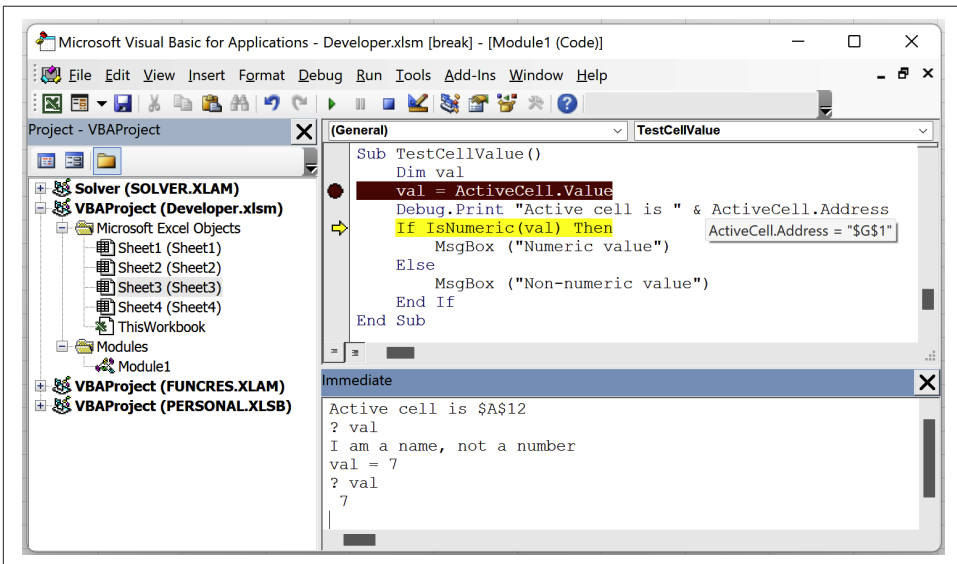
Figure 18-12. The Visual Basic Editor showing the Immediate window

Discussion

This recipe presents the debug features and tools in the Visual Basic Editor. These tools are handy if you have a macro or function that displays an error or behaves unexpectedly.

18.16 Using Built-in Dialog Boxes

Problem

You want to use one of Excel's dialog boxes to display information or get a response.

Solution

Excel includes several VBA functions that you can use to display a message or get information from the user.

The simplest function is `MsgBox`, which displays a simple message. For example, the following line uses the function to display the text *Hello* in a dialog box with a single OK button:

```
MsgBox ("Hello")
```

Generally, the function takes the form `MsgBox(prompt, buttons, title)`, where *prompt* is the text you want to display, *buttons* (optional) specifies the buttons to include and any icons, and *title* (optional) specifies the title in the dialog box heading. For example, the code

```
MsgBox ("Do you take sugar?", vbYesNo, "Sugar")
```

displays a dialog box titled *Sugar* with Yes and No buttons. You can then respond to the button the user clicks using code like this:

```
response = MsgBox("Do you take sugar?", vbYesNo, "Sugar")
Select Case response
Case vbYes
    'Code to run if Yes button
Case vbNo
    'Code to run if No button
End Select
```

If you want to get text input when you run a macro, you can use `InputBox`. This function takes the form `InputBox(prompt, title, default)`, where *prompt* is the text to display, *title* (optional) specifies the dialog box title, and *default* (optional) specifies the default value. For example, the following code asks the user to type their name:

```
InputBox ("What is your name?")
```

You can also use the `Application.GetOpenFilename` function to display Excel's Open File dialog box and the `Application.GetSaveAsFilename` function to show the Save As File dialog box.

Discussion

This recipe shows how to use built-in dialog box functions to display simple messages and/or get a response. To create more complex dialog boxes, see Recipe 18.19.

18.17 Using Form Controls

Problem

You need to work with data in worksheet cells and want to use simple controls to make this easier.

Solution

Suppose you want to make entering or editing data in worksheet cells easier. Instead of typing the values directly in the cells, you can use *Form controls*: simple controls you add to a worksheet that let you interact with cells. For example, you could use a check box control to enter TRUE or FALSE in a cell instead of typing it directly.

Excel includes the following Form controls:

Button
> Use a button to run a macro when clicked.

Combo box
> This is a drop-down list from which you can select one value.

Check box
> This control lets you set TRUE and FALSE values.

Spin button
> Use this to increase or decrease a value—for example, a number or date.

List box
> This works similarly to a combo box, except you can also select multiple adjacent or nonadjacent items from a list.

Option button
> Option buttons let you choose a single option from a mutually exclusive set of options. You usually group multiple option buttons in a group box, which lets you select a single option in the group.

Group box

> This is a frame with a label to which you can add other Form controls. You usually use it with option buttons because this lets you control which ones form part of a group.

Label

> This is a piece of text used to label another control or display descriptive text.

Scroll bar

> This lets you increase or decrease a value by scrolling through a range of values.

Figure 18-13 shows each Form control.
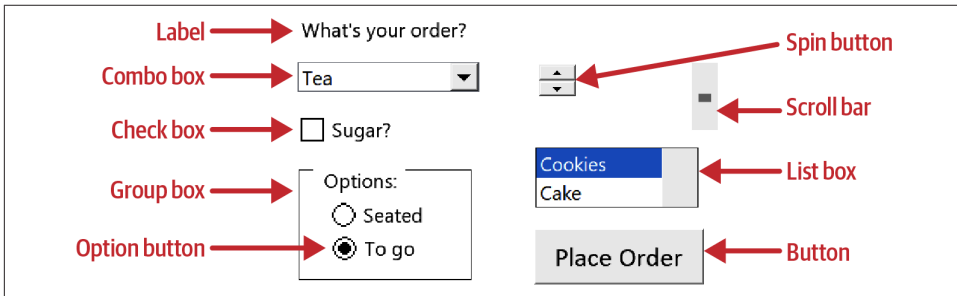
Figure 18-13. Excel's Form controls

> Excel for Windows includes two types of controls: Form controls and ActiveX controls. This recipe uses Form controls, which are simpler and embedded in Excel. ActiveX controls have a more extensive set of properties, and you can further define their behavior by writing VBA code. See Recipe 18.18 for more details about these controls.

Generally, you add a Form control to a worksheet by choosing Developer ⇒ Controls ⇒ Insert, selecting the control from the Form Controls group, then clicking and dragging where you want to place it on the worksheet. So to add a check box to a worksheet, select the check box control from the Form Controls group, and then click and drag where you want to place it on the worksheet.

Once you've added the control, you can tweak its appearance and control its behavior. Generally, you update controls as follows:

- To move or resize the control, select it by right-clicking it, then click and drag the control to move it, or click and drag its edges to resize it. You can also change its size by selecting the control and choosing Shape Format ⇒ Size, or right-clicking it, choosing the Format Control option, and then selecting the Size tab in the Format Control dialog box.

- To change the text displayed in a control (for example, a label), select the control and double-click its text.
- To specify which cell to link the control to, right-click the control, choose the Format Control option, select the Control tab in the Format Control dialog box, then put the cell reference in the Cell Link box (see Figure 18-14).
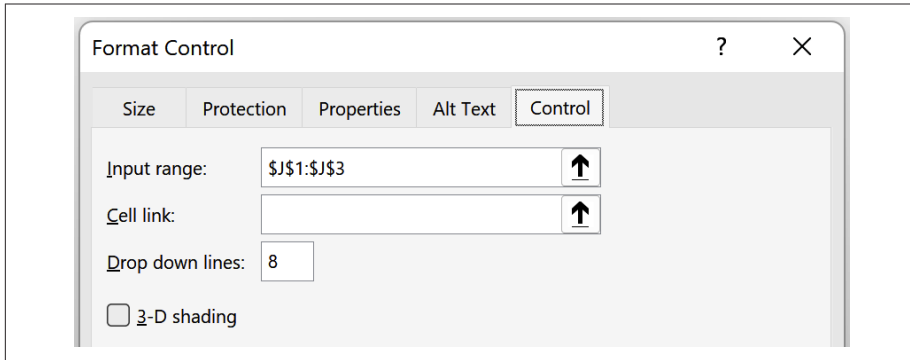
Figure 18-14. Editing a combo box's Form control's options in the Format Control dialog box

- To specify which values to use in a combo box or list box, add their descriptions to a range of cells, right-click the control, choose the Format Control option, select the Control tab in the Format Control dialog box, then put the range reference in the Input Range box, as shown in Figure 18-14. When you select an item from the list, Excel puts its index in the linked cell; you can use the INDEX function (see Recipe 7.11) to retrieve its description from the input range.
- To group option buttons so you can select only one option from the group, add a group box to the worksheet, and then move the option buttons inside it (as in Figure 18-13).
- To assign a macro to a Form control—usually a button—that runs when clicked, right-click the control and choose Assign a Macro to open the Assign Macro dialog box. You can select an existing macro to run, click Record to record a new one, or click New to write VBA code.
- To control other behavior, use the options in the Format Control dialog box, which you open by right-clicking the control and choosing Format Control. The available options depend on the type of control.

Discussion

This recipe covers how to use Excel's Form controls. These controls are embedded in Excel and provide a relatively simple way of interacting with a cell's value without

writing VBA. Unlike the more complex ActiveX controls, Form controls are available in Excel for Mac as well as Excel for Windows.

Use Form controls instead of ActiveX controls where possible because they're simpler and provide cross-platform compatibility. Try experimenting with Form controls to see if you can apply them to your situation.

18.18 Using ActiveX Controls

Problem

You want to use controls that you can customize more than Form controls, program with VBA, and add to a worksheet or UserForm.

Solution

Suppose you want to add controls to a worksheet but find Form controls too limiting. If you're using Excel for Windows, you can use ActiveX controls.

ActiveX controls look similar to Form controls but offer more flexibility. They have a more extensive set of properties, and you can control their behavior using VBA and add them to worksheets and UserForms. However, they're available in only Excel for Windows, so they're not cross-platform compatible.

Excel includes the following ActiveX controls:

Command button
    Use a command button to run a macro when clicked.

Combo box
    This lets you choose a single value from a list.

Check box
    This control turns a value on or off.

List box
    This works similarly to a combo box, except you can also select multiple adjacent or nonadjacent items from the list.

Text box
    This lets you type or edit text in a rectangular box or display static text.

Scroll bar
    This lets you scroll through a range of values.

Spin button

Use this to increase or decrease a value—for example, a number or date.

Option button

This lets you choose a single option from a mutually exclusive set of options.

Label

This is a piece of text used to label another control or display descriptive text.

Image

This lets you embed a picture.

Toggle button

Use this to indicate a binary state, such as Yes/No or On/Off.

More Controls

This lists other ActiveX controls on your computer, such as a Frame or Calendar control.

Figure 18-15 shows a selection of ActiveX controls.
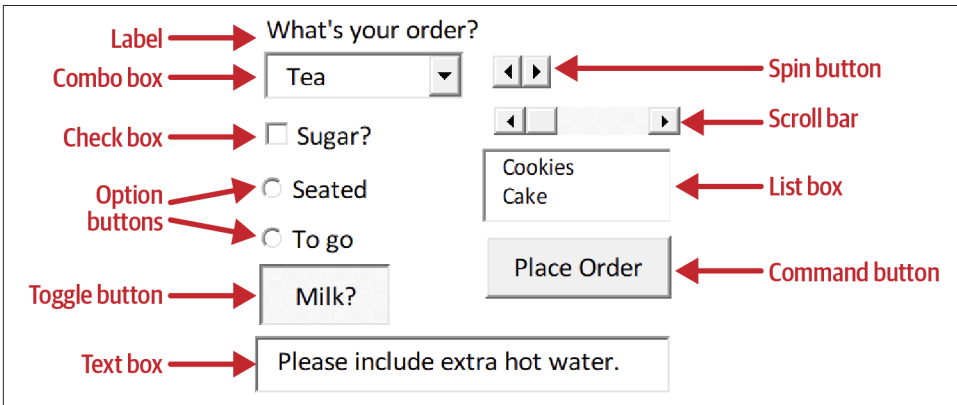
Figure 18-15. Excel's ActiveX controls

Generally, you add an ActiveX control to a worksheet by choosing Developer ⇒ Controls ⇒ Insert, selecting the control from the ActiveX Controls group, then clicking and dragging it to place it on the worksheet. So to add an ActiveX check box to a worksheet, select the check box control from the ActiveX Controls group and then click and drag it where you want to place it on the worksheet.

Once you've added the control, you need to make sure Excel has switched on Design mode before you can change the control's appearance and behavior. You can switch Design mode on or off by choosing Developer ⇒ Controls ⇒ Design Mode. Generally, Excel automatically enables Design mode when you add an ActiveX control.

To move a control, click and drag it while in Design mode. You can resize it by clicking and dragging its edges. Alternatively, right-click the control and choose Format Control.

When you're in Design mode, you specify the control's appearance and behavior by right-clicking the control and choosing Properties or selecting the control and choosing Developer ⇒ Controls ⇒ Properties—to open the Properties window. This window shows all the control's available properties in alphabetic order or by category (which is generally more useful), as shown in Figure 18-16.
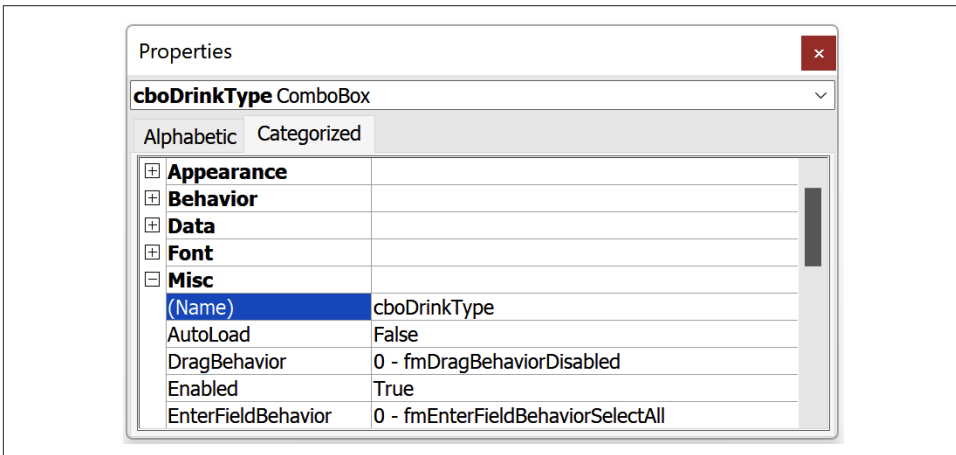
Figure 18-16. The Properties window for a combo box ActiveX control

ActiveX controls have extensive properties, which you can use to customize the control. These are generally categorized as follows:

Appearance
> This includes options specifying the control's caption, alignment, color, and any special effects.

Behavior
> The options in this group let you control behavior such as text alignment, how to match entries in a combo box, and whether you can select multiple items from a list box.

Data
> This includes options such as whether a list box should be styled as option buttons, how many rows a combo box displays, the number of columns, and its bound column—the one used to specify the control's value.

Font

This specifies the control's font.

Misc

This contains general options, such as whether the control is enabled, its name (which you can refer to in VBA code), and its linked cell. It also includes various options specific to each control, such as the data source of a combo box.

Picture

Use the options in this category to add a picture to a control, such as a command button, toggle button, or image.

Here are some common ways of using a control's properties to specify its behavior:

- Use the Caption property in the Appearance category to change the text displayed in a control such as a button, toggle button, or check box.

- To specify which cell to link the control to, type it in the LinkedCell property in the Misc category.

- To indicate which values to use in a combo box or list box, add their descriptions to a range of cells, then type this range in the ListFillRange property in the Misc category. Use the ColumnCount and ColumnHeads properties in the Data category to specify how many columns to display and whether to show column headings. To specify which column to use as the control's value, use the BoundColumn property in the Data category.

When you link an ActiveX combo box to a cell, the control puts the selected value from the bound column in that cell. This behavior differs from that of a combo box Form control, which saves its index.

- To group option buttons so you can select only one option from the group, give each one an identical GroupName property in the Misc category. Or consider using a list box control whose ListStyle property in the Data category is "1 - fmListStyleOption" because this makes the list resemble option buttons.

- Use the Accelerator property in the Misc category to assign a shortcut key to a control. For example, using *C* as a command button's Accelerator value means pressing Alt+C has the same effect as clicking the button.

ActiveX controls let you further specify behavior by writing VBA code for each control's *events*: actions that cause VBA code to run. For example, a command button includes a Click event, which you can use to specify what should happen when you click the button. Similarly, check box and combo box controls include Change events

that run when you change their value. In general, you write code for an ActiveX control's VBA events as follows:

1. Use the (Name) property in the Misc category to change the default name Excel has assigned to any ActiveX controls you want to use or refer to in the VBA code. While this step is optional, it helps make your code more readable and easier to understand.

2. Right-click the control and choose View Code to open the worksheet's code window in the Visual Basic Editor.

3. In the code window's Object drop-down list on the left, choose the control whose event you want to write VBA code for. For example, select the button to write code that runs when you click a button.

4. In the code window's Procedure drop-down list on the right, choose the event you want to write code for. For example, select Click to write code that runs when you click the control. When you do so, the Visual Basic Editor adds an empty procedure for the event to the worksheet's code. For example, if you select a command button named `cmdShowValue` in step 3 and choose its Click event, the Visual Basic Editor creates a new procedure named `cmdShowValue_Click`.

5. Edit the procedure so it performs whichever actions you require when the event triggers. For example, to display the value of a check box named chkSugar in a worksheet named OrderSheet, you'd update the code as follows:

```
Private Sub cmdShowValue_Click()
    'Display the value of chkSugar
    MsgBox(OrderSheet.chkSugar.Value)
End Sub
```

> If you want to add code to a command button's Click event, you can double-click the button instead of following steps 2 to 4. Doing so automatically inserts an empty procedure for this event.

Once you've made all the changes you need to make to the control, you can interact with it by switching off Design mode (choose Developer ⇒ Controls ⇒ Design Mode).

> If you can't interact with a control, ensure its Enabled property in the Misc category is `True`. If this property is `False`, you won't be able to use the control, even if it's displayed.

Discussion

This recipe shows how to use Excel's ActiveX controls. These controls have more extensive properties than Form controls, and you can write code for different events to fine-tune their behavior. However, ActiveX controls are more complex than Form controls and are unavailable in Excel for Mac.

In addition to adding ActiveX controls to worksheets, you can use them to create custom dialog boxes; see Recipe 18.19 for more details.

> If you don't know whether a control is a Form or ActiveX control, enable Design mode, right-click the control, and examine its options. If the control has an Assign Macro option, it's a Form control. If the control has a Properties or View Code option, it's an ActiveX control.

18.19 Creating a UserForm

Problem

You want to create a custom dialog box containing one or more ActiveX controls.

Solution

Suppose you have a workbook containing a table named tblIdeas with Idea and Timestamp columns. You want to display a free-floating dialog box that lets you quickly add a new idea to the table and the time you entered it. You can achieve this by creating a *UserForm*: a custom dialog box containing one or more ActiveX controls. Unlike Excel's default data-entry form (see Recipe 2.12), you have complete control over a UserForm's controls because you specify their behavior using VBA events.

To solve this problem, you can create a UserForm containing a text box that lets you enter a new idea, an Add command button that adds it to the table, and a Cancel command button that closes the UserForm. The steps for this are as follows:

1. Before creating the UserForm, you first need to insert the table you want to add data to. Create a table containing Idea and Timestamp columns, name the table **tblIdeas**, and name its worksheet **Ideas**.

2. You then need to create the UserForm in the Visual Basic Editor. Choose Developer ⇒ Code ⇒ Visual Basic to open the Editor, right-click the workbook's VBA project in the project explorer, and choose Insert ⇒ UserForm. When you do so, the Visual Basic Editor adds a new *Forms* folder to the project, containing an empty UserForm.

3. Next you need to change the UserForm's name and caption by updating its properties. Choose View ⇒ Properties Window to open the Properties window (if not already open) and select the UserForm. Then change its (Name) property in the Properties window's Misc category to a name you want to refer to it by in VBA code—in this example, **frmAddIdea**. Then change its Caption property in the Appearance category to the title you want to display at the top of the dialog box (in this example, **Add Idea**).

4. To add controls to the UserForm, select the UserForm and ensure the Toolbox window is open—if this is closed, choose View ⇒ Toolbox to open it. Then click each type of control you want to add, and click and drag where you want to place it on the UserForm. In this example, you want to add a text box control and two command buttons.

5. Next, update each control's properties, starting with the text box. Select the text box on the UserForm and then change its (Name) property in the Properties window's Appearance category to **txtIdea**. Changing the property's name to something more meaningful makes the VBA code easier to read and understand.

6. To update the first command button's properties, select the command button and change its (Name) property to **cmdAddIdea** and its Caption property to **Add**.

7. To update the second command button's properties, select the button and change its (Name) property to **cmdCancel** and its Caption property to **Cancel**.

8. To make the cmdAddIdea command button add a new row to the tblIdeas table when clicked, you need to write the VBA code for its Click event. To do so, double-click the command button to create an empty procedure for this event, then type the following code:

```
Private Sub cmdAddIdea_Click()
    Dim ws As Worksheet
    Dim tbl As ListObject
    Dim row As ListRow

    'Get the worksheet
    Set ws = Worksheets("Ideas")
    'Get the table
    Set tbl = ws.ListObjects("tblIdeas")
    'Add a new row
    Set row = tbl.ListRows.Add

    'Populate the first and second columns
    row.Range(1) = txtIdea.Value
    row.Range(2) = Now

    'Clear the UserForm's TextBox control
    txtIdea = ""
    'Set the focus to the TextBox control
```

```
        txtIdea.SetFocus
    End Sub
```

9. To make the `cmdCancel` command button close the form when clicked, double-click the command button to create an empty procedure for this event, then type the following code:

```
Private Sub cmdCancel_Click()
    Unload Me
End Sub
```

Figure 18-17 shows the completed UserForm in the Visual Basic Editor with the toolbox and Properties window.
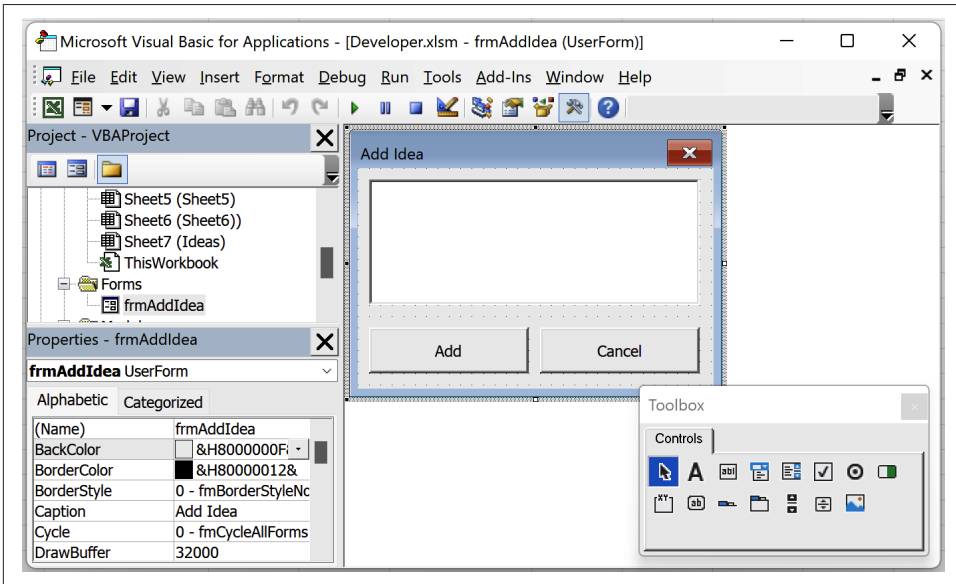
Figure 18-17. The UserForm in the Visual Basic Editor

Once you've added controls to a UserForm, you can use the Visual Basic Editor's Format menu options to align and arrange them. You can also change the tab order of the controls by right-clicking the UserForm in the Visual Basic Editor and choosing Tab Order.

To display the UserForm, you need to create a macro that, when run, displays the UserForm. To do so, double-click the workbook's module in the project explorer—if there isn't one, right-click the workbook and choose Insert ⇒ Module to insert one. Then add the code for the `ShowForm` macro as follows:

```
Sub ShowForm()
    'Show frmAddIdea
    frmAddIdea.Show
End Sub
```

When you've added the macro, run it to display the UserForm (see Figure 18-18).
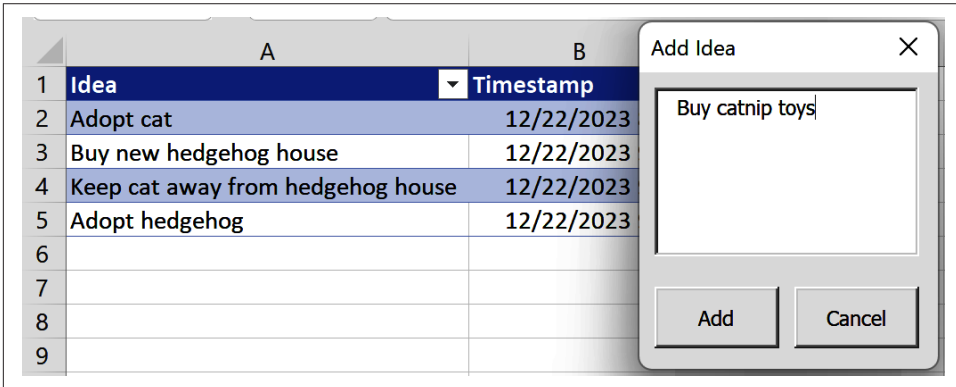
Figure 18-18. The completed UserForm and the table it adds data to

Discussion

This recipe shows how to use ActiveX controls in a custom UserForm instead of adding them to a worksheet. In general, UserForms can use the same basic ActiveX controls as worksheets, along with extra frame, tab strip, and multipage controls; the frame control displays a rectangular frame, the tab strip control displays the same controls on multiple tabs, and the multipage control lets you display multiple controls on different pages.

18.20 Creating a Custom Excel Add-in

Problem

You have custom functions and want to be able to use them in all open workbooks without using the personal macro workbook.

Solution

When you add functions to the personal macro workbook (see Recipe 18.3), they're automatically available to all open workbooks. However, you need to include a `PER SONAL.XLSB!` prefix when calling them. For example, to call a function in the personal macro workbook named `RandTen`, you type **`=PERSONAL.XLSB!RANDTEN()`**.

If you want to omit this prefix, you can save the functions in a *custom Excel add-in* instead: a file containing extra features that you can add to Excel. Creating a custom add-in gives all open workbooks access to the add-in's VBA code and makes sharing them easier.

You create a custom Excel add-in as follows:

1. Create a new Excel workbook and then open the Visual Basic Editor by choosing Developer ⇒ Code ⇒ Visual Basic.

2. In the project explorer, right-click the workbook's VBA project and select Insert ⇒ Module.

3. Double-click the module to open its code window and enter any functions you want the add-in to include.

4. In the project explorer, right-click the VBA project, select Properties to open the Project Properties dialog box, enter a project name and project description, and click OK.

5. Close the Visual Basic Editor, and in Excel, choose File ⇒ Save As. Change the file type to Excel Add-In (*.xlam*), enter a filename, choose where to save it, and save the file.

Once you've created the add-in, you enable it in Excel as follows:

1. Open an Excel workbook (or create a new one) and choose Developer ⇒ Add-ins ⇒ Excel Add-ins to open the Add-ins dialog box.

2. Click Browse, navigate to the add-in you created, and click OK.

After enabling the add-in, you can call any function without including a prefix. For example, if the add-in includes a function named `RandTen`, you can call it in a worksheet cell by typing `=RANDTEN()`.

You can also add macros and UserForms to a custom Excel add-in. When you do so, you can run the macros by adding them to the Quick Access Toolbar or ribbon (see Recipe 18.5). However, they're not listed in the Macro dialog box.

When you edit the code in a custom Excel add-in, Excel doesn't prompt you to save your changes. Instead, you must manually save your changes in the Visual Basic Editor by selecting the project and choosing File ⇒ Save.

Discussion

This recipe describes a way of sharing code between open workbooks by creating a custom Excel add-in. Once it's created, you can share the file and add it to other Excel installations.

> You can search for and manage add-ins from other solution providers by choosing Developer ⇒ Add-ins ⇒ Add-ins. For example, if you want to test experimental features from the Microsoft Excel team, try searching for Excel Labs.

18.21 Setting Security and Privacy Options

Problem

You want to control when to enable macros or ActiveX controls and specify whether to trust documents in a specific location.

Solution

When you open workbooks containing macros, ActiveX components, or connections to external data sources, Excel may prompt you to enable the workbook's content. If you're using Excel for Windows, you can control this behavior using the Trust Center: a suite of security and privacy settings for Office applications, including Excel. You can update macro and ActiveX settings, specify trusted locations and documents, and more.

To open the Trust Center, choose Developer ⇒ Code ⇒ Macro Security. Alternatively, choose File ⇒ Options to open the Excel Options dialog box, select the Trust Center option, and click Trust Center Settings.

Excel for Mac doesn't include the Trust Center. However, you can access a reduced set of security options by choosing Excel ⇒ Preferences ⇒ Security.

> Changing any Trust Center settings changes the security of your computer, data, and the computers and data on your networks. Before changing any Trust Center settings, carefully read any warnings and recommendations, consider the potential risks, and consult your system administrator.

Discussion

This recipe shows how to access the Trust Center, which you can use to change security and privacy settings. For example, you can use it to browse any trusted locations.

18.22 Importing and Exporting XML

Problem

You want to import or export data in XML format by mapping each element to a cell or table column.

Solution

Suppose you have an XML file named *ProductData.xml* containing product data. The file includes a single `<Title>` element and repeating records containing `<Name>` and `<Desc>` elements (see Example 18-2).

Example 18-2. The contents of the ProductData.xml file

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<data-set xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
    <Title>Latest Product data</Title>
        <record>
            <Name>Tea</Name>
            <Desc>Served hot</Desc>
        </record>
        <record>
            <Name>Coffee</Name>
            <Desc>As dark as a moonless night</Desc>
        </record>
</data-set>
```

You want to be able to import the data in *ProductData.xml* to specific Excel cells and table columns or export data from these cells and table columns to another file using the same XML format.

If you're using Excel for Windows, you can solve this problem using Excel's Developer XML options. These options let you map each element in an XML schema to a cell or table column and then use the map to import XML data to specific cells or export the cell values to an XML file.

> You can also import XML data using Power Query (see Recipe 15.1). However, the Developer XML options let you fine-tune your results by mapping elements to cells or table columns.

Before importing or exporting the data, you must specify where to map each element. For example, to map the `<Title>` element to cell A1 and the `<Name>` and `<Desc>` elements to a table's Product and Description columns, you'd follow these steps:

1. Use a text editor—for example, Notepad or TextEdit—to save the XML in Example 18-2 in a file named *ProductData.xml*.

2. In Excel, type **Product** in cell A3 and **Description** in cell B3. Then select A3:B3 and choose Insert ⇒ Tables ⇒ Table to convert the range to a table with headers.

3. Choose Developer ⇒ XML ⇒ Source to open the XML Source panel; then click XML Maps to open the XML Maps dialog box.

4. In the XML Maps dialog box, click Add, select the file *ProductData.xml*, and click Open to create a new XML map using the data's assumed schema. Then click OK.

5. In the XML Source panel, make sure the new XML map is selected and then drag each element to the cell or table column you want to map it to. In this example, you'd drag the `<Title>` element to cell A1, the `<Name>` element to the Product heading in cell A3, and the `<Desc>` element to the Description heading in cell B3. When you successfully map each element, Excel changes its format to bold.

Figure 18-19 shows the XML Maps dialog box containing an XML map.
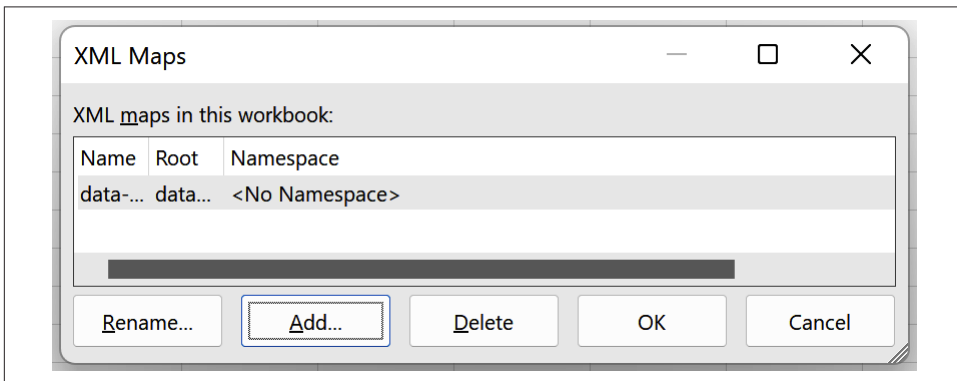
Figure 18-19. The XML Maps dialog box

Figure 18-20 shows the XML Source panel, where the XML map's elements have been mapped to cell A1 and the Product and Description table columns.

Once you've mapped each element, you can use it to import or export data.

To import data to the mapped cells and table columns, select the map in the XML Source panel (if it's not already selected), choose Developer ⇒ XML ⇒ Import, select the XML file you want to import (in this example, *ProductData.xml*), and then click Import. When you do so, Excel populates the mapped cells and table columns.
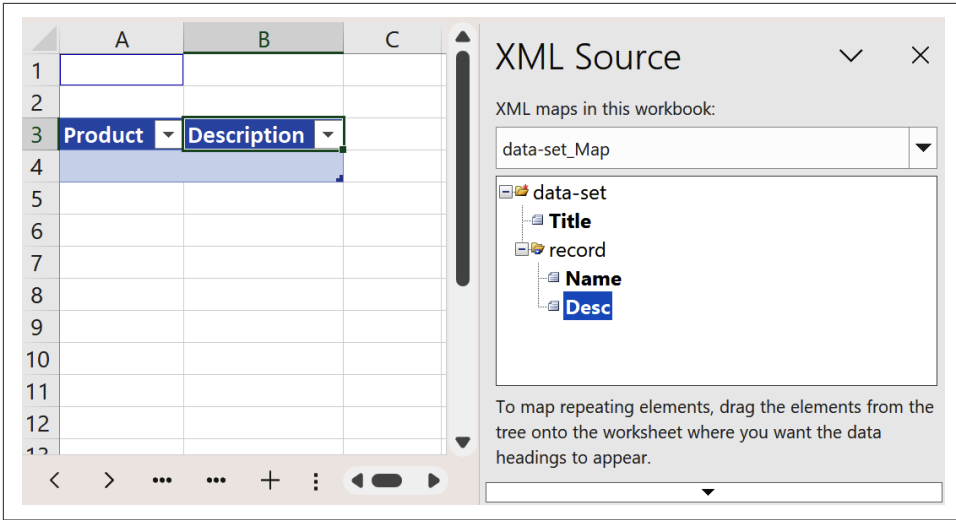
Figure 18-20. The XML Source panel

If the data in the XML file gets updated, you can refresh the Excel data by selecting a mapped cell—for example, A1—and choosing Developer ⇒ XML ⇒ Refresh Data or choosing Data ⇒ Queries & Connections ⇒ Refresh All to refresh all data.

> By default, refreshing the data or importing a different XML file overwrites any changes you've made to the data in Excel. To change this behavior, choose Developer ⇒ XML ⇒ Map Properties to open the XML Map Properties dialog box and adjust the options in the "When refreshing or importing data" section.

To export data from the mapped cells and table columns, select the map in the XML Source panel (if it's not already selected), choose Developer ⇒ XML ⇒ Export, and specify the XML file you want to export the data to (for example, *ProductDataExport.xml*), and then click Export. When you do so, Excel exports the data using the mapped cells and table columns to populate the XML elements.

Discussion

This recipe shows how to use Excel's Developer XML options to map elements to cells and table columns. Once they're mapped, you can import XML data to these cells or export their data to an appropriately structured XML file.

Next Steps

Congratulations on reaching the end of the book!

I hope you've found the recipes helpful and discovered new Excel tips, tricks, and techniques to put into practice. Try experimenting and applying them to your situation because that's the best way to make your newfound knowledge your own.

While this is the end of the book, it's not the end of your Excel journey. Microsoft is constantly adding new features and functions, and one way to keep up-to-date is by visiting my website, *TheExcelCookbook.com*. The site includes extra Excel tips and tricks and a list of my upcoming training courses. Hope to see you there!

Index

About the Author

Dawn Griffiths is an author, trainer, software developer, and the founder of *TheExcel-Cookbook.com*. She has written and cowritten several books, including *Head First Statistics*, *Head First Android Development*, *Head First Kotlin*, *Head First C*, and *React Cookbook*. She contributed to *97 Things Every Java Programmer Should Know* and created the animated video course *The Agile Sketchpad* with her husband, David.

Dawn has taught Excel to thousands of students worldwide and runs regular live online classes on the O'Reilly learning platform. You can find a list of upcoming courses at *TheExcelCookbook.com/training*.

Colophon

The animal on the cover of *Excel Cookbook* is the purple land crab (*Gecarcinus ruricola*). Also known as the black land crab, red land crab, or zombie crab, these terrestrial crabs are found in the Caribbean islands from western Cuba to Barbados.

A typical female of the species can carry around 85,000 fertilized eggs, which she releases in the sea. The eggs hatch in the sea, where they pass through several larval phases before returning to land. Back on land, they live in groups in burrows. They reach maturity after 5 years, and live for about 10 years.

The purple land crab is considered the most terrestrial land crab in the Caribbean and it is often found at high altitudes and great distances from the sea. They have adapted to life on land in part by conserving water. They are nocturnal, which helps prevent them from drying out in the sun. The crabs also have a nephritic pad, onto which they secrete waste material that is cleaned by microorganisms and then reabsorbed.

Gecarcinus ruricola has not been evaluated by the IUCN. Many of the animals on O'Reilly covers are endangered; all of them are important to the world.

The cover illustration is by Karen Montgomery, based on an antique line engraving from *Pictorial Museum of Animated Nature*. The series design is by Edie Freedman, Ellie Volckhausen, and Karen Montgomery. The cover fonts are Gilroy Semibold and Guardian Sans. The text font is Adobe Minion Pro; the heading font is Adobe Myriad Condensed; and the code font is Dalton Maag's Ubuntu Mono.