Microsoft Office

Microsoft

Create Dynamic Charts in Microsoft Office Excel 2007 and Beyond

CD Includes:
- 150+ sample charts
- Complete eBook

Reinhold Scheck

How to access your CD files

The print edition of this book includes a CD. To access the CD files, go to http://aka.ms/625440/files, and look for the Downloads tab.

Note: Use a desktop web browser, as files may not be accessible from all ereader devices.

Questions? Please contact: mspinput@microsoft.com

Microsoft Press

PUBLISHED BY
Microsoft Press
A Division of Microsoft Corporation
One Microsoft Way
Redmond, Washington 98052-6399

Library of Congress Control Number: 2008938214

Printed and bound in the United States of America.

1 2 3 4 5 6 7 8 9   QWT   3 2 1 0 9 8

Distributed in Canada by H.B. Fenn and Company Ltd.

A CIP catalogue record for this book is available from the British Library.

Microsoft Press books are available through booksellers and distributors worldwide. For further information about international editions, contact your local Microsoft Corporation office or contact Microsoft Press International directly at fax (425) 936-7329. Visit our Web site at www.microsoft.com/mspress. Send comments to mspinput@microsoft.com.

The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious. No association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred.

This book expresses the author’s views and opinions. The information contained in this book is provided without any express, statutory, or implied warranties. Neither the authors, Microsoft Corporation, nor its resellers, or distributors will be held liable for any damages caused or alleged to be caused either directly or indirectly by this book.

Acquisitions Editor: Juliana Aldous Atkinson
Developmental Editor: Sandra Haynes
Project Editors: Valerie Woolley and Thomas Pohlmann
Editorial Production: Media Service Gerhard Alfes
Translation: Lemoine International Inc., Salt Lake City, UT
Cover: Turnstyle Design

Body Part No. X14-95102

Contents at a Glance

Table of Contents

How to Use This Book

Why Would Anyone Need This Book in the First Place?

A few hundred pages solely dedicated to Excel charts? Using a program in which it has always been so easy to present numbers in a spreadsheet? Is it not simply a case of selecting a range and starting the Chart Wizard, which will then guide you through a few key steps to ensure that you make the right decisions in the right places? Furthermore, the new features available in Microsoft Excel 2007 enable you to create better, more aesthetically pleasing charts in less time. Or do they?

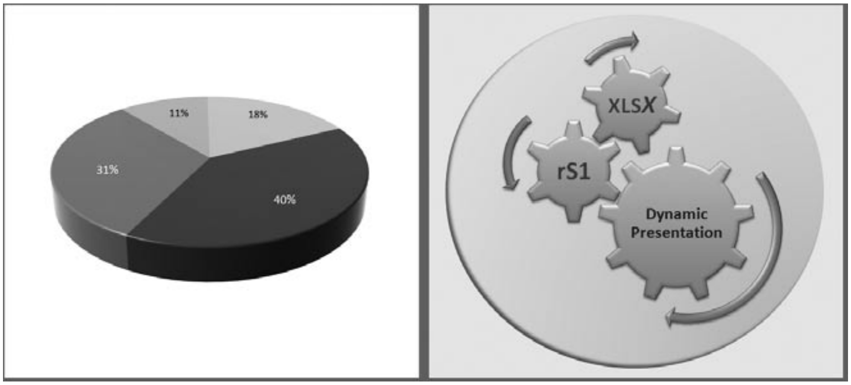
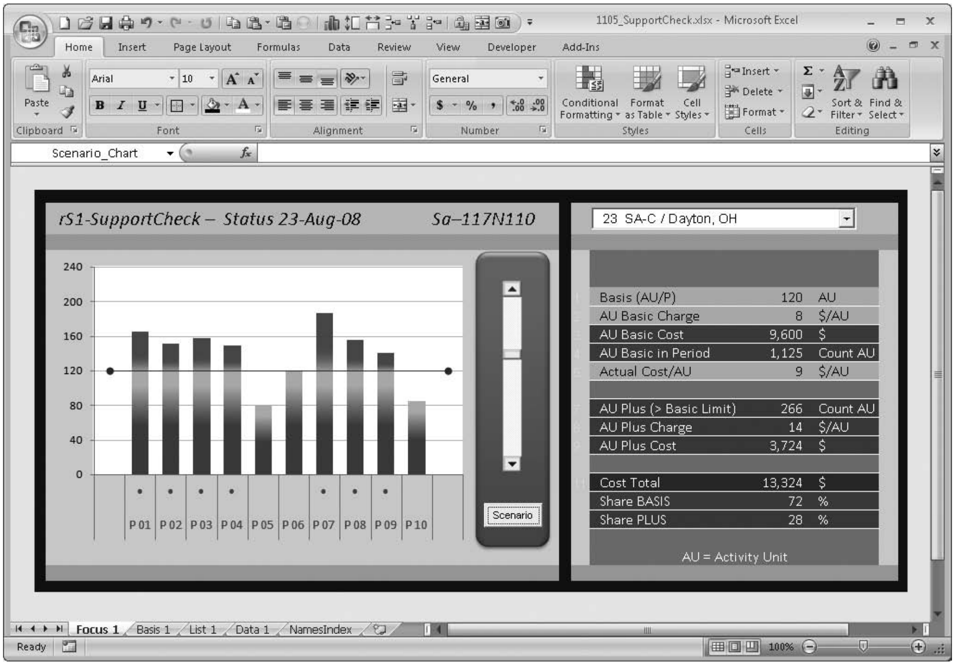


FIGURE I-1 Creating objects like those requires very little time and effort in Excel 2007

In reality, Microsoft Excel 2007 is better, faster, and more aesthetically pleasing, especially in terms of its design capabilities where enormous progress has been made. There is much greater user support and, in many respects, it has been made easier. With a little practice, it will take you just a few minutes to create aesthetically pleasing images like those shown in Figure I-1. In other words, what has always been relatively easy to do has now been made even easier. Furthermore, many things that were previously beyond the scope of the user's design options are now easily available to everyone. How is this possible? What are the necessary working techniques? In this book, you will find detailed information as well as notes and instructions. However, they alone would not have been enough to justify writing a book because the Excel help, which traditionally has been a source of information to users (albeit not a particularly popular source, and only used reluctantly by many), provides perfectly adequate and competent support when switching over or introducing yourself to the new version, if your primary concern is to create high-quality charts, that is.



23 SA-C / Dayton, OH

Basis (AU/P)	120	AU
AU Basic Charge	8	$/AU
AU Basic Cost	9,600	$
AU Basic in Period	1,125	Count AU
Actual Cost/AU	9	$/AU
AU Plus (> Basic Limit)	266	Count AU
AU Plus Charge	14	$/AU
AU Plus Cost	3,724	$
Cost Total	13,324	$
Share BASIS	72	%
Share PLUS	28	%

AU = Activity Unit

FIGURE I-2 You too can create such complex, scenario-based costing models

Of course, you will not want to or have to content yourself with "normality" in this book. The example shown in Figure I-2, which also plays a key, exemplary role in Chapter 11 "Fulfilling Special Requirements", demands, in terms of content, considerably more than can be achieved by simply applying the general information provided in the Excel help.

Note Figure I-2 shows the focus sheet for a combined costing, report, and analysis model. Different customers have availed of support activities assigned to different activity classes. In each case, a customer's most recent activity periods are imported, at a mouse click, into a chart where they are differentiated according to agreed activity types. In this view, it is possible to immediately envisage a different scenario and alternative costing/valuation by continuously changing a limit (indicated by the horizontal bar in the chart). The purpose of this chart is to use the horizontal bar to determine what may have been a more cost-effective solution for the customer while bearing in mind the conditions agreed and the activities actually performed during this period. This solution may then also serve as a standard for future agreements. Of course, none of the above requires an ounce of programming.

You can look forward to taking an excellent journey through the world of Microsoft Office Excel 2007 and all it has to offer. Any brick walls you may encounter as a result of the program's restrictions or your own limitations will only be due to the fact that we will occasionally push against the spreadsheet's boundaries. You will be amazed by the many possibilities. And how easily they can be achieved if you systematically use structured methods in a targeted manner.

This book will enable you to create complex and highly dynamic numbers presentations as well as impressive graphics that will win over your audience. Excel 2007 at its very best, and it doesn't require a single line of programming. This will be proven in the coming pages.

Version Restriction: Excel 2007 and Beyond

It is certainly not easy to immediately warm to the redesigned Microsoft Office 2007 user interfaces for Word, PowerPoint or Excel. They may well appeal to first-time users, but anyone who is an experienced, long-time user of Microsoft Office programs will certainly need a few days, if not longer, to gain an overview of these new user interfaces, and to adjust to the idiosyncrasies associated with the new version. During this period of adjustment, repeated attempts to work with these programs may well frustrate and bewilder users, and possibly even provoke an outburst of fury! However, once this phase passes, both the innovations and benefits associated with Office 2007 will shine through and come into their own.

> **Note** According to the information available at the time of going to press, the procedures and techniques described in this book should also remain valid for any post-2007 versions of Excel. However, in the absence of such future versions, it is not possible to put the above statement to the test.

This book wishes to convey that Excel is particularly successful in this regard, which will be to the benefit and, better still, the delight of the reader.

It's Not All New, but Much of It Is Significantly Different

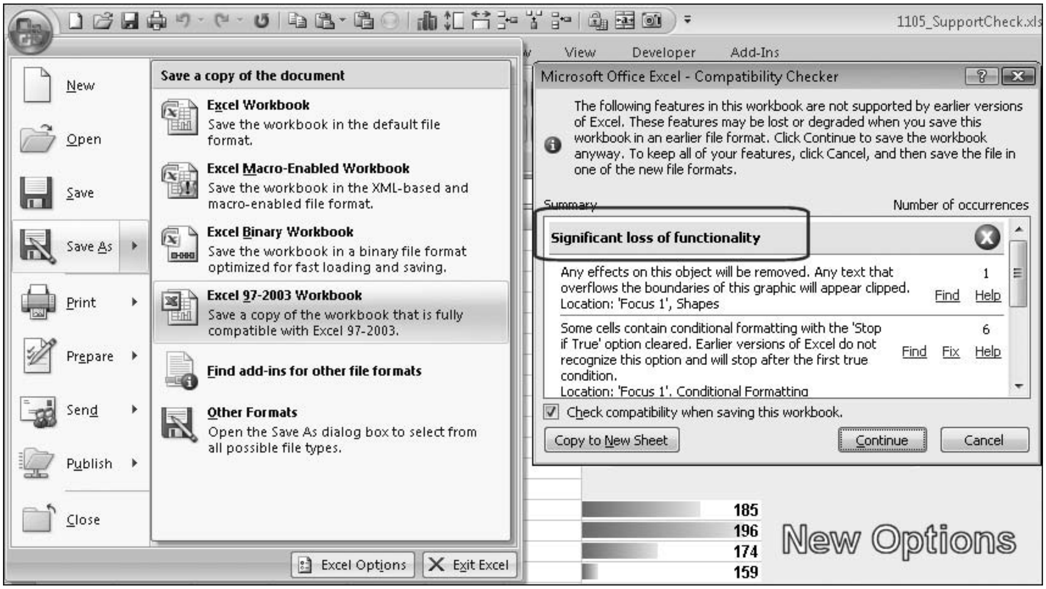
For all of the explanations and figures in this book, I have used Microsoft Excel 2007 on Microsoft Vista Ultimate. Only in Chapter 1, "Basic Information—Basic Techniques", have I compared this latest version of Excel with earlier versions by providing a partial comparison of Versions 2003 and 2007, partial in the sense that it is limited to the basic creation and use of charts.

> **Note** All of the figures have been created using an English version of Microsoft Office, whose custom language settings support several foreign languages and input locales, including two Asian fonts. Consequently, some of the dialog boxes shown in the figures may contain additional tabs and/or additional entries that are not available in the standard installation.

This book addresses the highly specialized topic of charts. This is a broad subject area whose contents and restrictions are defined in Chapter 1. The term "specialization" simultaneously implies "restriction" since here you can only expect to receive a fraction of the information that will make it easier for you to switch over to Excel 2007. I therefore beseech you not to expect to receive high-level support from this book, as this essentially goes beyond the subject of charts and their dynamic use.

The technical and design conditions of Excel 2007 differ greatly from earlier versions of Excel, to the point that backward compatibility is subject to considerable restrictions. The new features prevent universal, in other words, version-independent use. Therefore, a model that you create using the resources available in Excel 2007 (in accordance with the instructions provided here) can only be used in Version 2003 or older with restrictions.

In Excel 2007, you will find numerous save options that were previously unknown to you as well as some options you may recognize from earlier versions of Excel, for example, the option to save a workbook as a backward-compatible file. If, when creating an Excel 2007 file, you use resources that did not exist in earlier versions, and you then explicitly try to save this file as an Excel 97-2003 workbook, your attention will be drawn to a dialog box containing the possible incompatibilities. Let’s take a look at the example provided in Figure I-3 where the user applied conditional formatting to a numbers column. However, this type of conditional formatting (data bars) is not available in earlier versions of the spreadsheet. Therefore, when he tries to save such a file as an Excel 97-2003 workbook, the program issues a warning message indicating that there may be a “significant” loss of functionality.



Save a copy of the document

Excel Workbook
Save the workbook in the default file format.

Excel Macro-Enabled Workbook
Save the workbook in the XML-based and macro-enabled file format.

Excel Binary Workbook
Save the workbook in a binary file format optimized for fast loading and saving.

Excel 97-2003 Workbook
Save a copy of the workbook that is fully compatible with Excel 97-2003.

Find add-ins for other file formats

Other Formats
Open the Save As dialog box to select from all possible file types.

Microsoft Office Excel - Compatibility Checker

The following features in this workbook are not supported by earlier versions of Excel. These features may be lost or degraded when you save this workbook in an earlier file format. Click Continue to save the workbook anyway. To keep all of your features, click Cancel, and then save the file in one of the new file formats.

Summary	Number of occurrences
Significant loss of functionality	
Any effects on this object will be removed. Any text that overflows the boundaries of this graphic will appear clipped. Location: 'Focus 1', Shapes	1 Find Help
Some cells contain conditional formatting with the 'Stop if True' option cleared. Earlier versions of Excel do not recognize this option and will stop after the first true condition. Location: 'Focus 1', Conditional Formatting	6 Find Fix Help

☑ Check compatibility when saving this workbook.

Copy to New Sheet | Continue | Cancel

FIGURE I-3 The result of the compatibility check advises the user of a possible loss of functionality

Another problem occurs when you save a file in the new default file format **.xlsx* available in Excel 2007 and then try to open such a file in an earlier version of Excel, for example, 2003. The target program converts the file and creates a document that you can use and edit, but may be missing certain, possibly essential, parts of the source file created in Excel 2007.

Therefore, this book is ultimately dedicated to Excel 2007 and later versions of this program whose contents and results you should, in all likelihood, use in only these modern versions of the program. To clarify, everything that you will read in the coming pages is specific to those possibilities and idiosyncrasies associated with Excel 2007 (and beyond), some of which *cannot* be used in earlier versions of Excel, or at least *not in this way*. (Even though much of what will be shown here can be performed in older versions of the program, it is achieved there in another, occasionally considerably more cumbersome manner). Consequently, the information conveyed in this book can be applied to earlier versions of Excel, but more in theory than in praxis.

Objectives—Approaches—Possibilities

What Realistic Goals Can I Set Myself?

This is an exercise book. You can also use it as a reference book if you are already a highly proficient user of Excel 2007. However, to get the most from this book, I recommend that you work through each chapter in succession. By doing so, you can achieve the following:

- By applying competent procedures, you can create meaningful, effective, and dynamic Excel models as well as presentations based on Excel resources without considerable effort, and without having to acquire any programming knowledge.
- With very little effort, you can reproduce your tables and charts at any time and convert them to other display and/or application formats.
- You can show other users how to create modern, future-proof Excel solutions and justify the procedures used.

How Successful Can I Hope to Be?

Anyone who makes clever use of the possibilities outlined in this book and consistently applies its content to tasks can hope to achieve the following:

- Users who initially have a low to average knowledge of Excel will quickly achieve professional results (variable solutions that take account of various special requirements).
- Conventional users of Excel will be able to produce impressive, dynamic number and text presentations without having to dispense with known static presentations (PowerPoint, for example).
- Inexperienced users of Excel will also be able to use their own solutions independently, without any problems, and in a user-secure manner.
- You can still impress those among your audience who have been overloaded with presentations in the past.

What Prior Knowledge Do I Require?

You do not require any expert knowledge or extensive user knowledge in order to understand the examples provided in this book and to develop your own solutions. If you follow my advice and work through each chapter in succession, you should not experience any major difficulties in terms of comprehension and use. However, this publication, which is geared towards a special topic, cannot and does not wish to be an "all-inclusive" Excel user guide. Consequently, readers of this book need to have some basic knowledge of Excel, including Version 2007. In other words, my formulations and instructions assume that you have sufficient basic knowledge of the tasks outlined below or that you have obtained or can obtain such information from other sources:

- Creating, saving, and managing files (in Windows Explorer)
- Organizing and managing Excel workbooks (for example, insert, copy, move, delete, and rename worksheets)
- Organizing the structure of Excel worksheets (for example, fill, copy, insert, move, and delete cells, rows, and columns)
- Creating, formatting, and revising simple lists and data tables
- Writing and editing (changing, modifying, duplicating) simple non-nested formulas of different reference styles (relative, absolute, mixed)
- Working simultaneously with several active programs and windows in Microsoft Windows
- Using program helps

Use of the rS1.Method

As is the case with all of my books on Excel, the "rS1.Method" is once again the main basis for all solutions and models in this book. The "rS1.Method" is a detailed set of rules that comprises table functions, the use of certain functions or formulas, and the use of controls. I developed this standard and have used it in numerous solutions for many years. At its core is an absolute, yet varied structure, coupled with the consistent use of range names of a certain type and syntax.

Many of the examples provided in this book use the rS1.Method. Therefore, the user must accept the following: if you wish to reproduce these solutions, you must understand the procedure. Consequently, there is no way of avoiding the theory behind the method. However, the relevant information does not form part of this book, but is stored separately on the CD-ROM.

On the Companion CD The relevant file is stored under *\Materials\rS1_Method_2007.pdf.*

I recommend that you print out the document and preferably read it before you study Chapter 2 where you will be introduced to the solutions and models used here. While reading this book, keep your printout to hand, so that you can understand the examples and their structure at all times.

FIGURE I-4 The rS1.Method defines a standard, rationalizes your solutions, and makes them dynamic

> **Note** The text element shown in Figure I.4 has been created using *SmartArt*. You'll find information about this feature in Chapter 12.

Design and Materials

This section primarily focuses on notations, materials, and indexes.

Notations

The terminology used in this book, at least where Excel functions are concerned, corresponds to those terms used in the English version of Microsoft Excel 2007.

The following notations are used for functions and formulas:

- A function's syntax display is used for an abstract description or explanation of the function, for example:

  `=OFFSET(reference,rows,cols,height,width)`

- Its actual notation is then used to correctly display a formula applied to a worksheet, for example, `=IFERROR(C4/D4,"")` or `=INDEX(rD1.Actual07,$G6,F$7)`.

Note The function `=IFERROR(value,value_if_error)` belongs to one of those efficient new formulas available with Excel 2007. The `value` argument represents a specific arithmetic operation. If this produces the error value `#N/A, #VALUE!, #REF!, #DIV/0!, #NUM!, #NAME?` or `#NULL!`, your custom specification for the `value_if_error` argument is shown in the cell. If the arithmetic operation is successful, the result of this operation is shown in the cell. Therefore, in comparison with earlier versions of the program, it is now considerably easy to trap possible error values that are displayed.

A practical example is provided in Chapter 11.

Examples and Materials

The CD-ROM accompanying this book contains numerous sample files and materials.

- *Samples files* are half-finished or completely finished Excel workbooks that support the descriptions provided in each chapter. Such files are stored in a main Samples folder for each chapter. The notes associated with these files use the path together with the file name, for example, *\Samples\1006_RadarComparison.xlsx*.

Note Most of the sample files have a presentation worksheet (or several such sheets) with the sheet name *Focus 1* (*Focus 2*, and so on). These worksheets are designed for a screen resolution of 1024 x 768 pixels, mainly because most modern projectors can handle such a screen resolution without any difficulty, but cannot cope with a higher resolution or at least not well enough to produce an acceptable quality.

Incidentally, this screen resolution is also the minimum resolution for correctly displaying the new Ribbon in Excel 2007. You'll find more information about this topic in Chapter 1, "Basic Information–Basic Techniques."

A directory of sample files that provide information about making charts dynamic through **F9** (simulating the import of new values) and through controls (actually importing new values) as well as information about using conditional formatting is available on the CD-ROM under *\Materials\CD_Samples_Index.xlsx.*

Note The author's main residence is Berlin, Germany and he took most of the examples provided in this book from his own practical experience. As a result, most of the examples have their origins in Europe and apply the units of measurement and specifications that are customary there. Furthermore, many of the examples deal with special questions from the areas of Controlling, Medicine or Science. Consequently, at first glance, some readers may find some of the examples somewhat unusual. On the other hand, however, these examples, in this form, reflect the author's vast and varied experience and are therefore very authentic. Above all, they represent the wide range of areas in which Microsoft Excel can be applied. For this reason, the publisher decided to retain the majority of the examples in their authentic form in the translation of this book.

- *Materials* are files of different types that support your work and help you to retain an overview. Such files are stored in a main folder called *Materials*. Once again, the notes associated with these files use the path together with the file name, for example, *\Materials\rS1_Method_2007.pdf*.
- *Bonus Material* includes Chapter 22, "Analyzing Data with PivotTable Reports" from the book "Microsoft Office Excel 2007 Inside Out."

Hardware and Software Requirements

You'll need the following hardware and software to work with the companion content included with this book:

- **Microsoft Windows XP, Windows Vista or later.**
- **Microsoft Office Excel 2007 or later.**
- **1.6 GHz Pentium III+ processor, or faster.**
- **1 GB of available, physical RAM.**
- **Video (1024 x 768 or higher resolution) monitor with a color setting of at least medium (16 bit).**
- **CD-ROM or DVD-ROM drive.**
- **Microsoft mouse or compatible pointing device**

> **Digital Content for Digital Book Readers:** If you bought a digital-only edition of this book, you can enjoy select content from the print edition's companion CD.
> Visit **http://www.microsoftpressstore.com/title/9780735625440** to get your downloadable content. This content is always up-to-date and available to all readers.

Indexes

If you want to use this book as a reference book, two different indexes will prove helpful:

- The (Subject) Index lists keywords, technical terms, and functions.
- The Index of Procedures contains page references to descriptions for specific procedures or step-by-step instructions.

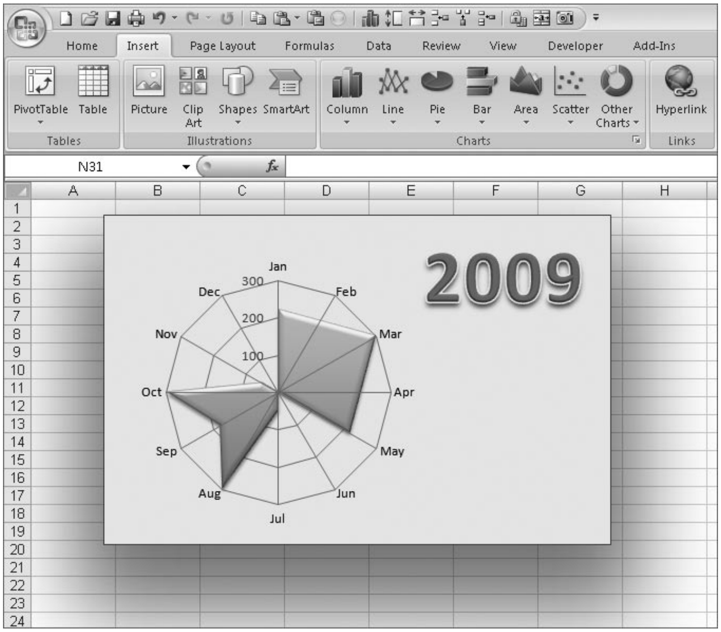


FIGURE I-5 Very soon you will be able to create charts like the one shown above

Support for This Book

Every effort has been made to ensure the accuracy of this book and the contents of the companion CD. As corrections or changes are collected, they will be added to a Microsoft Knowledge Base article.

Microsoft Press provides support for books and companion CDs at the following Web site:

http://www.microsoft.com/learning/support/books/.

Questions and Comments

If you have comments, questions, or ideas regarding the book or the companion CD, or questions that are not answered by visiting the sites above, please send them to Microsoft Press via e-mail to *mspinput@microsoft.com.*

Or via postal mail to

Microsoft Press, Attn: Create Dynamic Charts in Microsoft Office Excel Editor
One Microsoft Way
Redmond, WA 98052-6399

Please note that Microsoft software product support is not offered through the preceding addresses.

Chapter 7

Elements of Dynamization

From here on, we're going to slowly but surely pick up steam. I've already mentioned dynamic, multivariable Excel solutions a couple of times, and you've already encountered some (admittedly rather rudimentary) examples of these. By the end of this chapter, you'll be more familiar with the four components that really allow you to make the most of Excel's flexibility and the benefits it offers in terms of visualization. The necessary structural basis is provided by the rS1.Method. In this chapter, we'll combine the use of:

- Standardized range names
- Controls
- Extracting formulas
- Conditional formatting

In order to follow the instructions here without difficulty and to implement these with ease in the exercises I suggest or those of your own design, you'll need to feel at home with some of the content covered earlier. This content includes:

- The 32-page attachment, \Materials\rS1_Method_2007.pdf, on the CD-ROM. The information it provides about naming conventions (Sections 3.2. and 3.3.) and controls (Section 3.4.4) is particularly important for understanding the descriptions below and putting these into practice.
- The section entitled "A Solution Emerges" in Chapter 2, "New Approaches—Getting Started." It explains some of the elements and techniques that will play roles here. Ideally, you'll not only have read the step-by-step instructions in that section, but also have followed these with some exercises of your own.
- Chapter 5, "Graphical Objects," which explains how to deal with graphical objects. In this chapter, we'll also examine the controls that can be created, copied, and changed in essentially the same way as drawing objects.

Theme and Variations

A small analysis of key indicators is used as an example throughout this chapter. Three files are provided on the CD-ROM for this purpose:

- \Samples\0701_Indicators_01.xlsx (variant no. 1, the "green" variant; so named because of the dominant color in the *Focus 1* worksheet), in which you'll use some basic form controls to dynamize the chart display in the *Focus 1* worksheet. This model doesn't contain any conditional formatting.

- *\Samples\0702_Indicators_02.xlsx* (variant no. 2, the "blue" variant), which contains ActiveX controls. This variant is somewhat more extravagant and also user friendly in its appearance, and two of its worksheets contain some conditional formatting.
- *\Samples\0703_Indicators_00.xlsx*, which is a "hollow form" containing only the data and workbook structures you can use to replicate the solution (and I strongly recommend that you do).

Purpose and Structure of the Model

First, open variant 1—that is, the *\Samples\0701_Indicators_01.xlsx* file—and familiarize yourself with its content and functions.

Note The slightly more complex *0702_Indicators_02.xlsx* model is introduced later in the chapter. However, because that section primarily focuses on deviations from the basic variant described here, as well as the elimination of shortcomings and functional enhancements, it would be very useful for you to become familiar with this simpler version at this stage. Remember, too, that when it comes to putting the theory into practice, you won't always need to make use of all the bells and whistles available, no matter how convenient or attractive they are.

Which Analyses Can We Run?

The Management team at a manufacturing company periodically instructs the Controlling department to calculate certain key performance indicators using a dynamic Excel solution for analysis and evaluation. The summarized version provided here contains the monthly indicators for *Cash Ratio, Quick Ratio, Current Ratio, Profit-Turnover Ratio, Outstanding Receivables Quota, Debt Quota, Production Volume, Utilized Capacity, Quantity stored, Labor Time performed, Absence from Work,* and *Employee Turnover.* All figures are expressed as percentages. The figures for January through October 2008 are shown.

You'll notice many similarities to the chart solution introduced in Chapter 2. However, the selection options there were rather one-dimensional (a comparison of two data series of the same type). Here, however, you can control four different kinds of information with just a few mouse clicks:

- The length of a data series (the number of month-specific markers shown)
- The category of data series displayed
- The choice of showing or hiding the company-internal margins defined for the category selected (values may fluctuate within the range from – to)
- The choice of showing or hiding the industry average for the selected category

Management can make different selections in the *Focus 1* worksheet while analyzing the data. Note the following points with reference to Figure 7-1:

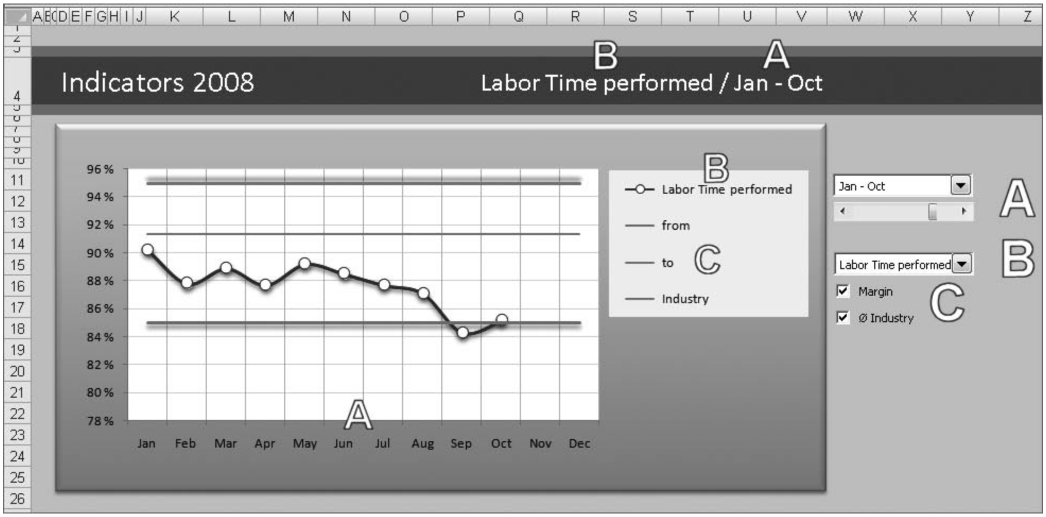


FIGURE 7-1 Five controls determine the content of the dynamic chart.

- Controls *A*, on the right of the chart: The *combo box* control has a drop-down list, in which you can choose between 12 different time intervals: the month of January plus all of the possible intervals ranging from the start of January to the end of any other month in the same year. Clicking on the relevant entry determines how many subsequent months' data is displayed in a blue data series with round markers. If the monthly values for a certain month aren't yet available (as is the case for November and December in the example shown), the data series isn't extended. The period shown is also indicated as a text in the chart header.

  This drop-down list is linked with a *scroll bar* control. You can use this control in the same way to lengthen or shorten the data series to show the data from a certain period of time (from January to …).

> **Note** To some, this option for changing the length of a data series may appear nothing more than a design frill. However, I consider it to be an interesting and useful option for two reasons. First, it allows you to reveal your data in stages over the course of a presentation (which, in many cases, will help your audience to understand the point you want to convey). Second, it allows you to easily pass on information in a condensed version, while ensuring that your data set is still complete and up-to-date. This could be useful for external reporting.

- Control *B*: Another *combo box* allows you to specify which of the 12 indicator series is shown in the main data series of the chart mentioned in relation to control A above. The indicator category selected is also shown in the chart header.

- Controls *C*: The two checkboxes allow you to decide whether to show three indicator values, which are then displayed as horizontal lines.
  - If you select the *Margin* checkbox, an olive-green *from* line (bottom) and an orange *to* line appear in the chart. The values represent the company-specific margin within which fluctuations of the indicators aren't classed as critical or even problematic. Since seasonal dependencies are of no significance to this company, the same margin applies for the entire year. The two lines disappear from the chart if you deselect the checkbox. This action also removes the corresponding legend texts.
  - If you select the *Ø Industry* checkbox, a light-blue line appears, which represents the industry-specific average value for the indicator currently selected. The lines disappears from the chart if you deselect the checkbox. This action also removes the corresponding legend text.

The option to display these lines was configured as part of the design of this chart because a wide range of perspectives, requirements, and reporting obligations apply to the data it represents, and a variety of impressions can be derived from it. Play around with these options and you'll see how dramatically the effect changes; for example, a line oscillates freely, an oscillating line moves between or beyond borders, and another entwines itself around a straight line.

Excel only plays a secondary role in making this an effective solution. The most important factor is the company's ability to “own” these figures (in the true sense of the word). This involves a consistent use of indicators (which also allows management to keep track of the day-to-day flood of information) and, in particular, representation of these in the form of actual data that can be compared with specific margins based on planning and analysis and with industry values. The result is that these figures immediately tell you what can be expected, what was planned, and how (or if) both can be aligned with reality. All that's missing is the option of comparing the data with older data from previous years. This feature was indeed part of the real-life solution on which this example was based, and I've only omitted it here for the sake of clarity.

In its simplified version, the model has some small inconsistencies and fallings and, on closer inspection, also reveals a number of information gaps. Some of these are eliminated in variant 2 (discussed later). First, I want to provide some information about the basic structural components of the solution, which are almost identical in both variants.

How Is the Workbook Structured?

The structure of the workbook is based on the precepts of the rS1.Method. The individual worksheets are described below, proceeding from right to left or, technically speaking, from “back to front.”

NamesIndex **Worksheet** The 10 range names assigned in this workbook are listed here, together with their references. This worksheet serves as documentation only. Creating a model like this helps you keep track of the various interdependencies and the arguments of the formulas used. It also helps you detect naming errors or reference errors.

To generate a list like this, select the top left cell in the range in which you want the list to appear, and press **F3**. In the *Paste Name* dialog box that appears, click the *Paste List* button, and then optimize the width of the two list columns so that their values can be clearly read. If a names index already exists, you must delete it before you generate a new list. This becomes necessary when you realize that some of the original names are superfluous and delete these with the *Name Manager*. In this case, the new list would be shorter than the old one, and therefore wouldn't overwrite all of the names listed.

The options for defining a range or object name are described in Section 3.4.6 of the *\Materials\rS1_Method_2007.pdf* attachment. Therefore, I'll merely provide a quick synopsis here.

Select the cell or cell range you want to name, and then use one of the following methods:

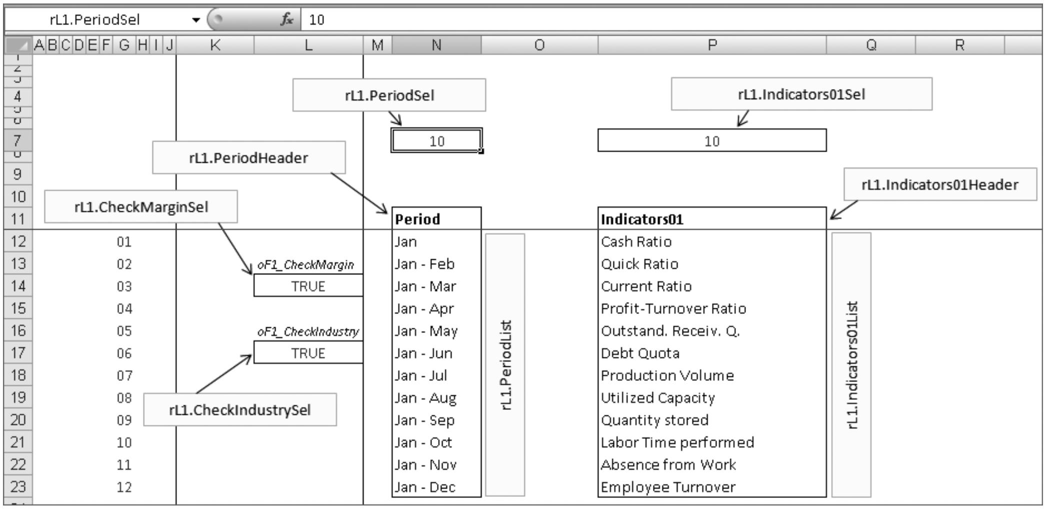
- ❑ Enter the name in the *name box* on the left of the *formula bar*, and press **Enter** to confirm.
- ❑ Select the *Formulas* tab in the Ribbon, and click the *Define Name* button in the *Defined Names* group. Then enter the name in the *New Name* dialog box.
- ❑ Select the *Formulas* tab in the Ribbon, and click the *Name Manager* button in the *Defined Names* group. Alternatively, use the **Ctrl**+**F3** shortcut. In the *Name Manager* dialog box, click the *New* button.

When defining range names, the last two methods described above are recommended, as they are more flexible and fail-safe than the first.

When defining object names (for example, naming controls), select the object, and then use the first method described above. If you use ActiveX controls, you can also enter the name in the *Properties* dialog box (more about that later).

Lists 1 **Worksheet** The definition ranges of the controls are of key importance to the dynamics of a model. You'll find these in the *Lists 1* worksheet, where they're defined in accordance with the rules of the rS1.Method. For a detailed description of these, refer to Section 3.4.5 in the *\Materials\rS1_Method_2007.pdf* attachment. This is normally a boring and painstaking task. However, it will pay off because you'll accomplish the rest of your work more easily and quickly than you will with conventional methods.

Note the following concluding points on this worksheet in reference to Figure 7-2. Here, I've used the same ABC sequence from Figure 7-1, which was used earlier to describe the controls themselves.



Row	K	L	N	P
7			10	10
11			Period	Indicators01
12	01		Jan	Cash Ratio
13	02	*oF1_CheckMargin*	Jan - Feb	Quick Ratio
14	03	TRUE	Jan - Mar	Current Ratio
15	04		Jan - Apr	Profit-Turnover Ratio
16	05	*oF1_CheckIndustry*	Jan - May	Outstand. Receiv. Q.
17	06	TRUE	Jan - Jun	Debt Quota
18	07		Jan - Jul	Production Volume
19	08		Jan - Aug	Utilized Capacity
20	09		Jan - Sep	Quantity stored
21	10		Jan - Oct	Labor Time performed
22	11		Jan - Nov	Absence from Work
23	12		Jan - Dec	Employee Turnover

FIGURE 7-2 Of particular importance: range names in the *Lists 1* worksheet

- Controls *A*: There are three named ranges for the *combo box* control for selecting time periods (which has *oF1_BoxPeriod* as its object name):
  - ❑ The `N12:N23` cell range with the name `rL1.PeriodList` contains 12 periods of time; i.e., it represents the content of the drop-down list. This data is defined in the form of constants.
  - ❑ The `N11` cell with the list header is called `rL1.PeriodHeader`.
  - ❑ The numeric value (in this case, a number between 1 and 12) that the user generates by clicking on an entry in the drop-down list, and which is subsequently used as an argument variable by formulas, is to appear in cell `N7`, which has the name `rL1.PeriodSel`. This cell is also linked with the *scroll bar* control (object name: *oF1_ScrollPeriod*), which similarly outputs numbers between 1 and 12 depending on the action executed by the user. Because both controls share the same cell link, they are also functionally linked with each other. The *combo box* moves the slider along the *scroll bar*, while the *scroll bar*, in turn, determines the list selection in the *combo box*.

Note Use a combination like this if you're using simple form controls and need both a specific, text-oriented selection option and the ability to casually "click through" a step-by-step display.

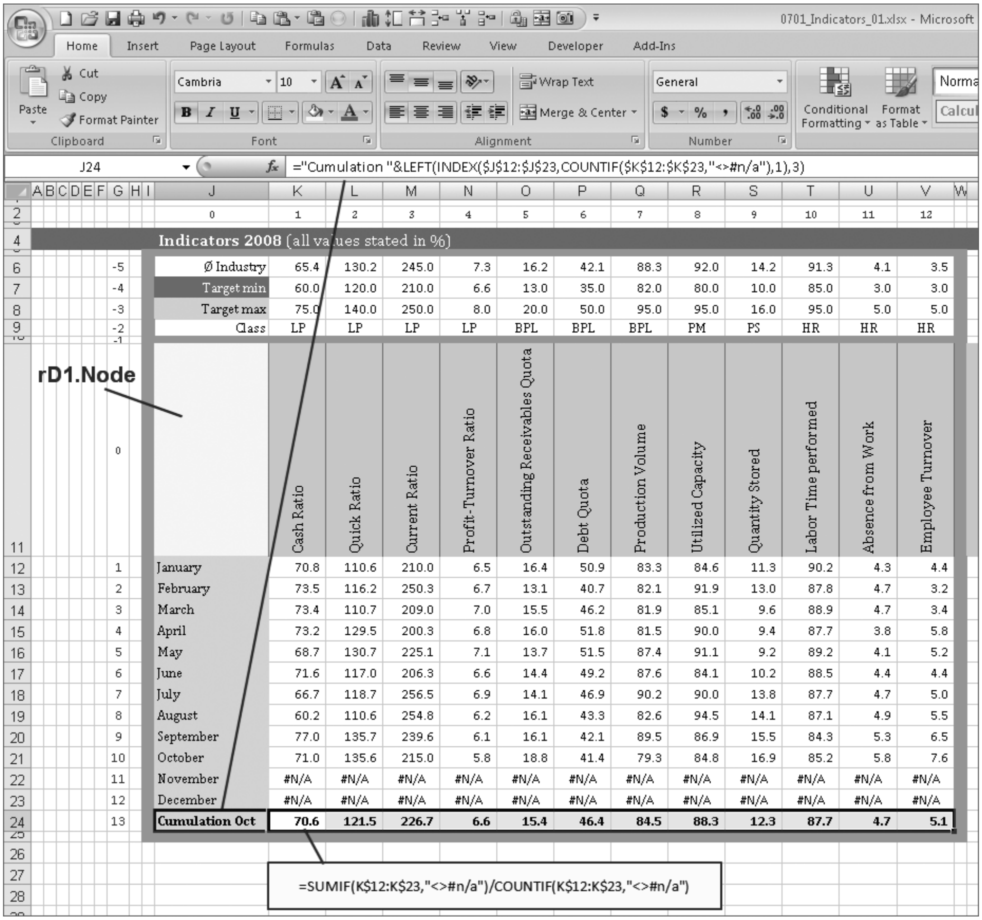
If you use ActiveX controls, you can use the **up arrow** and **down arrow** keys to scroll through a drop-down list. In this case, the link shown in this example isn't necessary, at least not for this purpose.

- Control *B*: As with the *oF1_BoxPeriod* control described above, three range names have been defined for the second *combo box* (which has the object name *oF1_BoxIndicators01*): `rL1.Indicators01List` for cell range `P12:P23`, `rL1.Indicators01Header` for cell `P11`, and `rL1.Indicators01Sel` for cell `P7`. The list contents are similarly defined as constants. The name `rL1.Indicators01Header` is not used in this solution. However, it has been created here because it may well be used in a future enhancement of the solution (this type of redundancy is typical of the rS1.Method).

  You may also have spotted another deliberate redundancy. It is likely that this kind of model may also be used, in the future, to compare key indicators. In this case, you would require two functionally independent selection lists with identical content (we've seen something similar in Chapter 2). Following our naming conventions, the names of these lists would differ only by a number (*Indicators01* and *Indicators02*), which will save you a lot of time and effort later. Therefore, there's no reason not to (and good reason to) allow for that possibility when assigning names here.

Data 1 **Worksheet** All of the information that is to be visualized in the focus worksheet is gathered together here. Note the following points in relation to Figure 7-3 (working from the top down):

- In row 2 and column G, you'll recognize the standard auxiliary structures of the rS1.Method; i.e., number series, which in this case represent the respective distance from the cell named `rD1.Node` (cell `J11`). `OFFSET` formulas use this node to determine all of the values that are required by the chart basis (*Basis 1* worksheet). This worksheet contains no other names.
- The values used to represent the margins of individual indicators and their corresponding industry averages are defined in rows 6 through 8. These are calculated and input at the end of a fiscal year.
- The abbreviations denoting the indicator categories in row 9 are for information purposes only.
- The full names of the indicators in row 11 are defined as constants, as are the aforementioned short names.



="Cumulation "&LEFT(INDEX($J$12:$J$23,COUNTIF($K$12:$K$23,"<>#n/a"),1),3)

	H	J	K	L	M	N	O	P	Q	R	S	T	U	V
2		0	1	2	3	4	5	6	7	8	9	10	11	12
4		Indicators 2008 (all values stated in %)												
6	-5	Ø Industry	65.4	130.2	245.0	7.3	16.2	42.1	88.3	92.0	14.2	91.3	4.1	3.5
7	-4	Target min	60.0	120.0	210.0	6.6	13.0	35.0	82.0	80.0	10.0	85.0	3.0	3.0
8	-3	Target max	75.0	140.0	250.0	8.0	20.0	50.0	95.0	95.0	16.0	95.0	5.0	5.0
9	-2	Class	LP	LP	LP	LP	BPL	BPL	BPL	PM	PS	HR	HR	HR
	-1													
11	0		Cash Ratio	Quick Ratio	Current Ratio	Profit-Turnover Ratio	Outstanding Receivables Quota	Debt Quota	Production Volume	Utilized Capacity	Quantity Stored	Labor Time performed	Absence from Work	Employee Turnover
12	1	January	70.8	110.6	210.0	6.5	16.4	50.9	83.3	84.6	11.3	90.2	4.3	4.4
13	2	February	73.5	116.2	250.3	6.7	13.1	40.7	82.1	91.9	13.0	87.8	4.7	3.2
14	3	March	73.4	110.7	209.0	7.0	15.5	46.2	81.9	85.1	9.6	88.9	4.7	3.4
15	4	April	73.2	129.5	200.3	6.8	16.0	51.8	81.5	90.0	9.4	87.7	3.8	5.8
16	5	May	68.7	130.7	225.1	7.1	13.7	51.5	87.4	91.1	9.2	89.2	4.1	5.2
17	6	June	71.6	117.0	206.3	6.6	14.4	49.2	87.6	84.1	10.2	88.5	4.4	4.4
18	7	July	66.7	118.7	256.5	6.9	14.1	46.9	90.2	90.0	13.8	87.7	4.7	5.0
19	8	August	60.2	110.6	254.8	6.2	16.1	43.3	82.6	94.5	14.1	87.1	4.9	5.5
20	9	September	77.0	135.7	239.6	6.1	16.1	42.1	89.5	86.9	15.5	84.3	5.3	6.5
21	10	October	71.0	135.6	215.0	5.8	18.8	41.4	79.3	84.8	16.9	85.2	5.8	7.6
22	11	November	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
23	12	December	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
24	13	**Cumulation Oct**	**70.6**	**121.5**	**226.7**	**6.6**	**15.4**	**46.4**	**84.5**	**88.3**	**12.3**	**87.7**	**4.7**	**5.1**

=SUMIF(K$12:K$23,"<>#n/a")/COUNTIF(K$12:K$23,"<>#n/a")

FIGURE 7-3 The *Data 1* worksheet contains all of the information that is to appear in the focus.

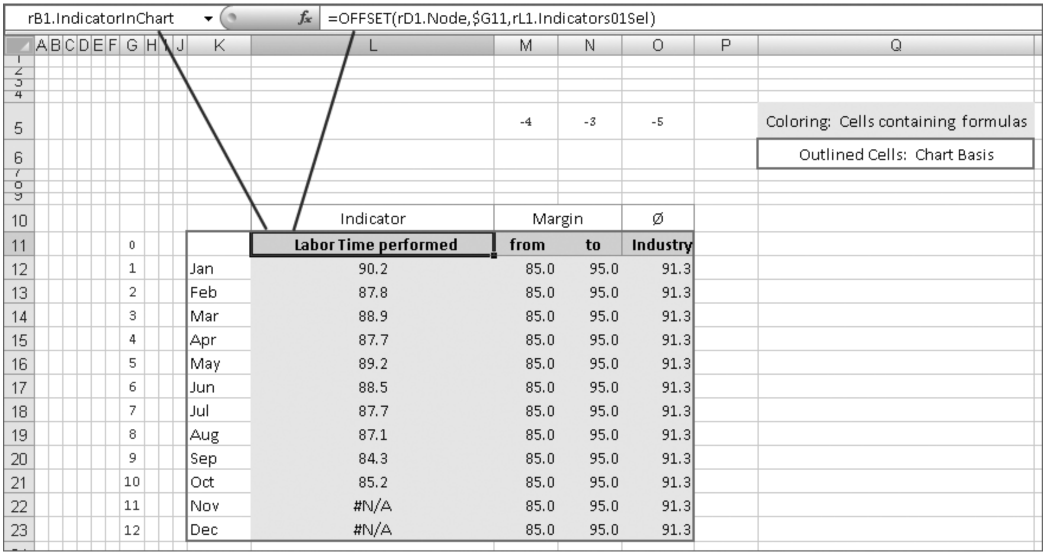
- The Controlling department enters the monthly values calculated for the indicators in rows 12 through 23. These are all percentages calculated from various reference values. Their display format with just a single decimal place is sufficient in this case, even for the lower values for *Profit-Turnover Ratio, Absence from Work,* and *Employee Turnover.* If, at any point over the year, data is not yet available to fill certain cells in accordance with the structure of this cell range, these cells are filled instead with the error value #N/A. These placeholders can then gradually be overwritten with the monthly values as they are calculated. This means that, at certain times of the month, some indicators will be calculated while others will not be available. However, thanks to the use of #N/A, the data series in the line chart will always be drawn correctly; i.e., it will end at the exact position that represents the last existing data point.

The #N/A error value in a chart basis suppresses the drawing of a data point in the resulting chart. This is of particular significance in line charts if some values are missing in the data table of a data series. If these were omitted entirely, gaps would appear in the line. If zero were used, the line would naturally drop to zero. The use of #N/A, however, means that the cell in question can be completely ignored when the chart is drawn, and the line simply joins up with the next available data point. If the data series ends with #N/A values, the line simply stops as instructed. Should all cells in a data series contain the value #N/A, then no data point is drawn, and so there is no line. You'll use this option in our example when using controls to show and hide certain chart lines. This will become clear later in relation to our discussion of formulas in the *Basis 1* worksheet.

- The use of #N/A as a placeholder in calculation row 24 makes it a little more difficult than usual to calculate the average. The AVERAGE function cannot be used here because a formula that recognizes the error value #N/A in its reference itself always has the value #N/A as its result. This is for a very good reason: it prevents incorrect calculations based on incomplete data. However, as you can see in the figure above and in the worksheet, this isn't a difficult problem to overcome. For a more detailed explanation of the calculation used here, refer to the "Formulas" section. The same applies to the formula that is used in cell J24 to determine a text that expresses the current cumulation.

  This is the only row in the *Data 1* worksheet to contain formulas. This is different in the variant in the *0702_Indicators_02.xlsx* file, which has a more convenient configuration. More about that later.

Basis 1 **Worksheet** This worksheet may also be a little puzzling at first to the inexperienced user (but don't worry, this won't last too long). As is usual with the rS1.Method, this worksheet largely consists of formulas. These respond directly to changes in the values in the *Sel* cells of the *Lists 1* worksheet, and thus indirectly to actions executed by the user with the help of the controls in the *Focus 1* worksheet.



rB1.IndicatorInChart

`=OFFSET(rD1.Node,$G11,rL1.Indicators01Sel)`

	M	N	O
5	-4	-3	-5

Coloring: Cells containing formulas

Outlined Cells: Chart Basis

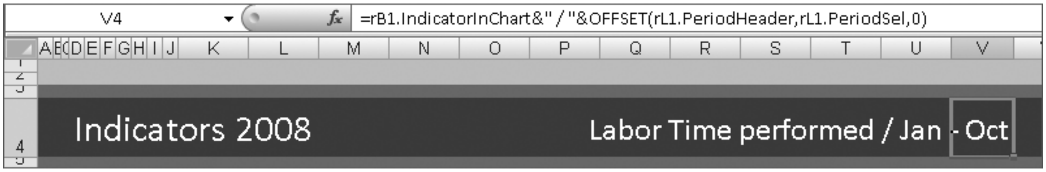
Row	G	K	L	M	N	O
10			Indicator	Margin		Ø
11	0		**Labor Time performed**	**from**	**to**	**Industry**
12	1	Jan	90.2	85.0	95.0	91.3
13	2	Feb	87.8	85.0	95.0	91.3
14	3	Mar	88.9	85.0	95.0	91.3
15	4	Apr	87.7	85.0	95.0	91.3
16	5	May	89.2	85.0	95.0	91.3
17	6	Jun	88.5	85.0	95.0	91.3
18	7	Jul	87.7	85.0	95.0	91.3
19	8	Aug	87.1	85.0	95.0	91.3
20	9	Sep	84.3	85.0	95.0	91.3
21	10	Oct	85.2	85.0	95.0	91.3
22	11	Nov	#N/A	85.0	95.0	91.3
23	12	Dec	#N/A	85.0	95.0	91.3

FIGURE 7-4 The *Basis 1* worksheet largely consists of formulas.

All of the data columns within the strong outline (range K11:O23) supply the chart content.

The L11 cell is the only cell in this worksheet to have a range name: *rB1.IndicatorInChart*. The text in this cell is also determined using a formula, and also appears as the header in the *Focus 1* worksheet; see Figure 7-5.

Focus 1 **Worksheet** The external appearance of the solution was described and its functions explained earlier.



V4

`=rB1.IndicatorInChart&" / "&OFFSET(rL1.PeriodHeader,rL1.PeriodSel,0)`

Indicators 2008

Labor Time performed / Jan - Oct

FIGURE 7-5 The header describing the current content is determined using a formula.

All I need to add at this point is the fact that the indicator category/period combination currently displayed in the chart is also specified in the chart header as a combined text. This character string is determined using a formula, which is also explained later.

So What's New in Variant 2?

So much for the structural components of the *0701_Indicators_01.xlsx* workbook, referred to below as *variant 1* and as the *basic model*. Before we examine this in more detail, there are a few key points to share about the difference between the basic model and the

0702_Indicators_02.xlsx file, whose components are discussed in detail later in the section entitled "Variant 2 (Advanced Model)."

The enhanced model is a little easier to use and, above all, implements the principles of the rS1.Method more consistently. The essential differences between it and the basic model are as follows:

- It contains a *Parameters 1* worksheet and uses formulas to transfer the data defined there to the *Data 1* and *Lists 1* worksheets. In other words, it allows for central data maintenance.
- More range names are used and are processed in formulas.
- The *Data 1* worksheet has conditional formatting, which is used to indicate where the indicators move outside of the desired or anticipated margins. In other words, it acts as an additional visualization aid.
- In the *Focus 1* workflows, the *form controls* are replaced by their equivalent *ActiveX controls*. There are also some structural changes that make the model easier to understand and to use, as well as text tips that appear whenever a period is selected in which no data is available for the final month(s).

Variant 1 (Basic Model)

This section explains the design of the *0701_Indicators_01.xlsx* file, with a particular focus on its dynamic elements. You'll also learn a little about form controls and about the use of typical rS1 range names in formulas.

Form Controls

> **Note** I'd like to remind you that the following discussion assumes that you're already familiar with some of the basic information about controls provided in Section 3.4.4 of the *\Materials\rS1_Method_2007.pdf* file.

Basics

User-friendly controls of various types can make Excel presentation models dynamic, attractive, and foolproof. This statement—and those that follow in this introductory section—apply to both types of controls; i.e., *form controls* and *ActiveX controls*.

Controls are ready-to-use, pre-programmed tools that can also be customized. Their greatest benefit for any of the solutions described in this book is that they enable multi-variable formula access to source data. This essentially works as follows.

1. You click a control (click a list entry, select an option, click an arrow). In other words, you make a *selection*. This is a very important concept in the examples presented here because it is used when naming features of the models.
2. The control responds to the selection made with the mouse by outputting a value in a certain, user-defined cell, which, in theory, may be located anywhere in the workbook (in rS1 models, however, it is almost always located in the *Lists 1* worksheet). The control's output value is a number or a logical value. If you use *ActiveX controls*, it may also be a text.
3. The output value is the processed result of your selection. You can then use it directly or indirectly as an argument in a formula. The enormous benefit of this is that you can control part of (or, in some cases all of) the solution with your mouse without any need for programming.

The basic commands for configuring and designing controls are in the *Controls* group of the *Developer* tab in the Ribbon. If you click the *Insert* button there, a visual menu of all available objects is displayed, divided into two groups.

Note The *Developer* tab is not automatically displayed after you install Excel 2007. However, you can make it permanently available by clicking the Office button/*Excel Options*/*Popular* tab/*Show Developer tab*.

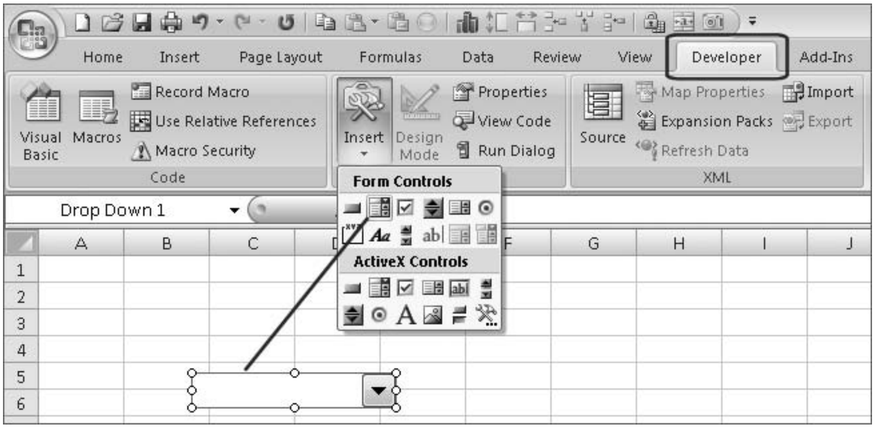


FIGURE 7-6 Controls are part of the *Developer* tab in Excel 2007.

If you want to generate a *form control*, or change the shape or size of an existing one, follow the same instructions provided for drawing objects in Chapter 5. The creation of the control itself is very simple indeed:

Note When using the rS1.Method, you should first create the control in the *Basis 1* worksheet, test it there in terms of both its functionality and its effects on the formula results, and then (and only then) copy it to its final destination in the *Focus 1* worksheet. You've already practiced using this approach in Chapter 2.

1. Click the icon of the control you want to use in the selection menu (see Figure 7-6). If you then move the mouse pointer in the worksheet (*without* holding down the mouse button), the mouse pointer changes to a crosshair.
2. Click the location where you want to position the control and, holding down the left mouse button, drag the handle diagonally until the control is the size you want (a sizing border allows you to preview the size).

Tip If you hold down the **Alt** key at the same time, the table-gridlines serve as a grid to help you size the object.

3. The control is created when you release the mouse button.
4. If you want to change the shape and size of an existing *form control*, first select the object like a graphics object by holding down the **Ctrl** key and clicking it, or by right-clicking the object, which simultaneously opens the context menu. It then appears with sizing handles and can be resized as required, just like any other graphics object. To select several controls at the same time, select each of the controls in turn while keeping the **Ctrl** key held down.
5. If you want to assign an object name to the control, keep the **Ctrl** key held down, click the object to select it, and enter a name in the *name box* in the *formula bar*.

Using Form Controls

If you've never used *form controls* to dynamize calculation models, you'll be amazed at just how much you can do in Excel with relatively simple tools. The objects can handle any requirements, thanks to flexible control. Nevertheless, the options for customizing them (formatting, functional range) are very limited. They are compatible with earlier versions of Excel (Version 5.0 and higher) and are very well suited to the design of simple models or use with electronic questionnaires. However, you should ensure that models created with these tools are rarely used in solutions with graphically complex designs and are not used in solutions that are to be subsequently enhanced with program source code.

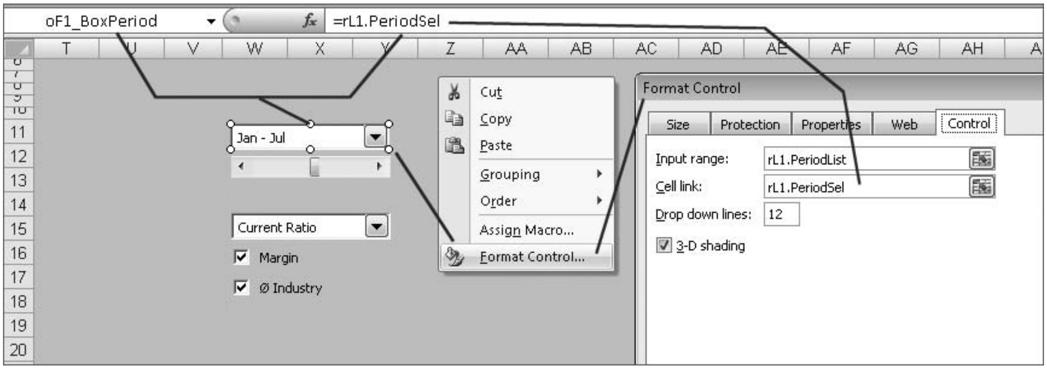
An important question to ask yourself initially is the following: What are you hoping to achieve by doing things this way? A functionally well thought-out application will be evaluated negatively by users if it's awkward to use, unnecessarily complicated or illogical. So, when you're designing a model, you should ask yourself what's really needed, what situations are likely to arise, and how users can achieve their objectives without too much trouble. This will determine to a large degree your choice of controls and how you use them. Note the following points in relation to these:

> **Note** The list below is limited to the controls that are used in the rS1.Method, and uses the same sequence used to structure the selection menu on the *Developer* tab (see Figure 7-6), which also corresponds to the numbering used in relation to page 24 of the *\Materials\rS1_Method_2007.pdf* attachment.

1. Always use a *combo box* if you want to allow users to select from a list of (in most cases) text items, and you want or need to save space on selection options when your selection list has many entries.
2. Use a *check box* if you want your user to be able to switch between YES/NO-type options (in the chart in our example, you can choose to display data series or not).
3. Use a *spin button* if you want to allow users to move step-by-step, back and forth, through a range of information. This would be the case if, for example, you want to be able to "shimmy" forwards or backwards from one month to the next within a fiscal year.
4. Use a *list box* if you want users to be able to select from a small number of list entries and if there's enough space on the screen for the object (otherwise, you can also use *combo boxes*, which are primarily intended for long lists).
5. Use *option buttons* if you want the user to be able to select from several similar alternatives (for example, one of three different rounding algorithms available for planning figures). Also use them if—this is an important factor in your choice of control—you have enough space on the screen for several option fields (otherwise, you can save space by putting all of the options in a *combo box*).
6. Use a *scroll bar* if you want the user to be able to move through a large range of information in small *or* large steps.

Configuring Form Controls

After you create a control, the next step is to configure it. Only a few simple steps are required in this case (as opposed to *ActiveX controls*). At least, they are simple and easy to follow if you've followed the conventions of the rS1.Method by creating relevant definition ranges in the *Lists 1* worksheet and assigned the "correct" names to these according to the method (see Figure 7-2).



oF1_BoxPeriod

=rL1.PeriodSel

Format Control

Input range: rL1.PeriodList

Cell link: rL1.PeriodSel

Drop down lines: 12

☑ 3-D shading

FIGURE 7-7 A simple dialog box helps you configure the control.

Now let's get back to our *0701_Indicators_01.xlsx* file. The following step-by-step instructions are based on the example of the *oF1_BoxPeriod* control (which is a *combo box*), and are illustrated in Figure 7-7. Here I'm assuming that the control is in the *Focus 1* worksheet and that it's a new element (i.e., it hasn't already been configured at this stage).

1. I admit that the rS1 range names aren't exactly convenient for entering as text. You can usually press **F3** to add an existing range name to a cell or formula. This also works in some dialog boxes, but not in all; the one here being a case in point. However, you can make your job easier by copying the correct name texts to the clipboard.

   Click the arrow to the right of the *name box* in the *formula bar*. In the names index that appears, click a name that you require in order to configure the control, for example, *rL1.PeriodSel*. This click selects the range that has this name, and *Lists 1* becomes the active worksheet, if that isn't already the case. Click the name to select it in the *name box*. Press **Ctrl**+**C** to copy the selected name to the clipboard. Press the **Esc** key to stop using the *name box* (not an essential step, but it's best to be on the safe side).

2. Go to the *Focus 1* worksheet, right-click the *combo box* control to select it, and open the context menu. Select the *Format Control* option. A dialog box consisting of several tabs opens.

3. Next, open the *Control* tab. Here you can define the functional properties of each control. Place the cursor in the *Cell link* box and paste the name *rL1.PeriodSel* from the clipboard (press **Ctrl**+**V**).

4. Press **Ctrl**+**V** again in the *Input range* box (this refers to the list content range, and is known as the *ListFillRange* in the case of ActiveX controls). Press **Backspace** to delete the *Sel* character string, and replace it with *List*.

5. Enter the number 12 in the *Drop down lines* box. The list has 12 entries only, so that, when a user clicks the control's drop-down arrow it will open in full, and all entries will be visible at the same time. This makes selection easier.

6. Select the *3D shading* checkbox. You should, in principle, do this for controls of this type (or at least those used on a color background) to provide a more vivid appearance.

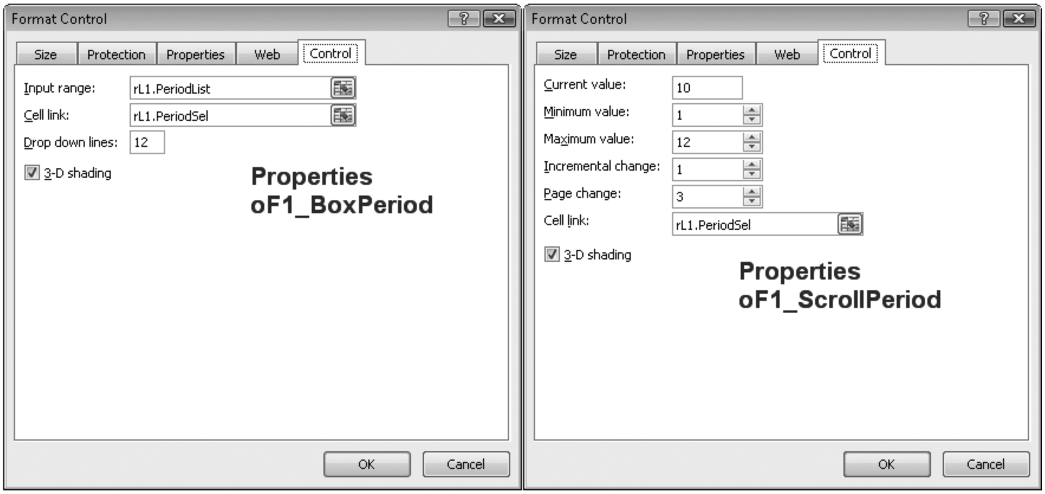
7. Press *OK* to close the dialog box. The control has now been fully configured. Its name is displayed in the name box in the formula bar, while its cell link is shown as a formula in the formula bar itself (see Figure 7-7). Press the **Esc** key or click anywhere in the worksheet to deselect the control. Click the control's drop-down arrow to display its selection list, and click one of the list entries (for example, the third) to select it. Open the *Lists 1* worksheet to check that the number 3 has been copied to the *cell link*, which is cell *rL1.PeriodSel* in this case. Repeat this test with at least one other selection.

> **Tip** In certain cases, it may be extremely difficult to spot an error caused by incorrect configuration of a control. Therefore, it's important not to just dismiss tests like these, even if they appear silly and unnecessary. This kind of nitpicking really does pay off in the end. Anyone who's ever thought "I'm sure it'll turn out just fine," only to end up losing several valuable hours of work knows exactly what I'm talking about here.

Below you'll find a summarized overview of the individual settings made for all of the controls in the *Focus 1* worksheet. A summary of all possible control properties is provided further below in Figure 7-11.

Controls for Selecting a Period Two controls are used to select a time period in our example. Note the following with reference to Figure 7-8:

The configuration of the *combo box* with the (optional) name of *oF1_BoxPeriod* was described in detail earlier.



Input range: rL1.PeriodList
Cell link: rL1.PeriodSel
Drop down lines: 12
☑ 3-D shading

Current value: 10
Minimum value: 1
Maximum value: 12
Incremental change: 1
Page change: 3
Cell link: rL1.PeriodSel
☑ 3-D shading

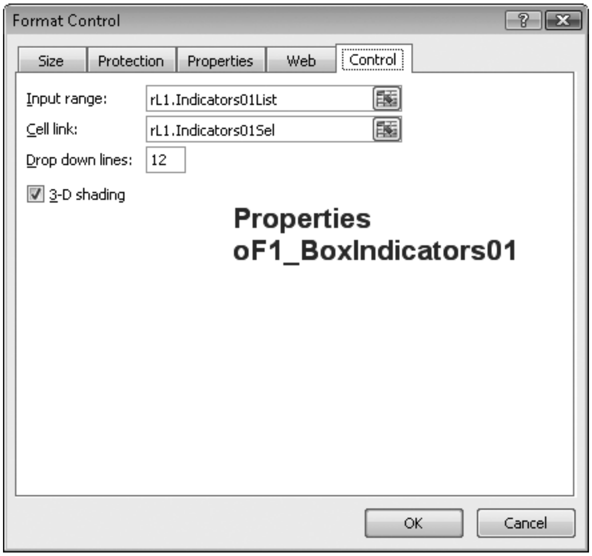
FIGURE 7-8 Making settings for the selection of a period

Since they both use the same *cell link*, the *scroll bar* with the name *oF1_ScrollPeriod* is also functionally linked with this *combo box*. Additional settings for this control:

- *Minimum value*: 1, *Maximum value*: 12, *Incremental change*: 1. This means that, each time the user clicks an arrow on the *scroll bar*, the *scroll box* (the "slider" on the control) selection will increase by an increment of 1. A value between 1 and 12 will also be displayed in the *cell link*.
- *Page change*: 3. Each click in the areas *between* the scroll box and either of the scroll arrows on the left or right of the horizontal *scroll bar* (or at the top and bottom of a vertical *scroll bar*) moves the *scroll box* forward or backward by an increment of three (or increases your selection by three).

> **Note** If you draw an object with a width greater than its height, its direction is horizontal. If its height exceeds its width, its direction is vertical. Unfortunately, this does not also apply to *spin buttons*, which are very similar in terms of functionality.
>
> If you use *ActiveX controls* instead, you can set a *Minimum value* that is greater than the *Maximum value* in order to reverse the direction of the control. This is well worth doing if it helps you meet specific logical and ergonomic requirements.

Control for Selecting an Indicator Category The *combo box* with the (optional) name of *oF1_BoxIndicators01* must be configured so that an indicator category can be selected. If you've already created and configured the first *combo box* (for selecting a period) as described above, you can repeat the procedure very simply as follows thanks to the naming conventions of the rS1.Method:



Input range: rL1.Indicators01List

Cell link: rL1.Indicators01Sel

Drop down lines: 12

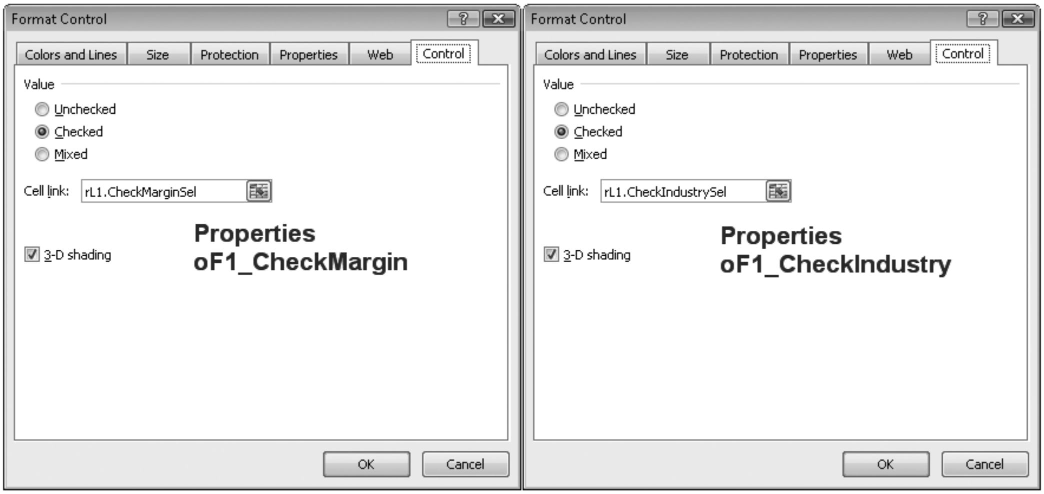
☑ 3-D shading

Properties oF1_BoxIndicators01

FIGURE 7-9 Apart from the references used, this is otherwise identical to the control for selecting a period.

1. Hold down the **Ctrl** key, and click the existing control object to select it.
2. Press and hold down **Ctrl** again, and use the mouse to drag a copy from the original object. If you hold down **Ctrl**+**Shift**, you can generate a copy that is aligned on the same axis (horizontal or vertical) as the original object.
3. The copy has the same properties as the original object. Because the number of list entries is the same in both objects in our example (12 periods and 12 indicators), you only need to change the references in the new *combo box*. To do this, simply replace the *Period* string with *Indicators01* in the *Input range* and *Cell link* boxes.
4. After a procedure like this, it's essential to test whether the new object works correctly and as desired.

Controls for Displaying Additional Data Series The two *check boxes* allow you to enable and disable the display of additional data series in the chart. You'll find out how this works later on in our discussion of the relevant formulas in the *Basis 1* worksheet. These formulas process the output values of the two *check boxes*; that is, they respond to the value TRUE or FALSE, depending on which of these appears in the *cell links* of the controls after the user clicks to make a selection.



Value

○ Unchecked
● Checked
○ Mixed

Cell link: rL1.CheckMarginSel

☑ 3-D shading

Properties
oF1_CheckMargin

Value

○ Unchecked
● Checked
○ Mixed

Cell link: rL1.CheckIndustrySel

☑ 3-D shading

Properties
oF1_CheckIndustry

FIGURE 7-10 Only the *cell links* need to be defined here.

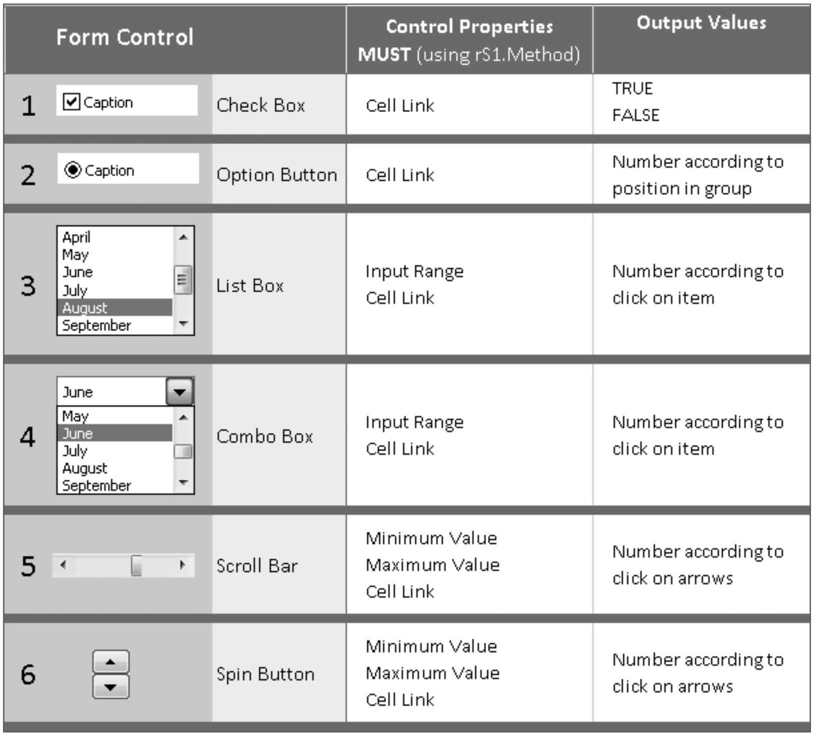
Proceed as described above in relation to the two *combo boxes*:

1. Start by creating the first *check box*. Select the default text (*Check box n*) in the object, and overwrite this with a text of your choice.
2. Copy the range name of the relevant cell link to the clipboard.
3. Right-click the object, and select the *Format Control* option in the context menu. Open the *Control* tab, and paste the contents of the clipboard into the *cell link*.

> **Note** As you can tell from the *Colors and Lines* tab in the dialog boxes shown in Figure 7-10, you can also define a fill color and a border for *check boxes* (as you can for *option buttons*).
>
> These or similar options are not available for other *form controls*. *ActiveX controls*, on the other hand, allow for complex formatting.

4. Test the control to verify that it works correctly.
5. Create a copy of the new control, change the description text displayed for users, and define the neighboring cell as the *cell link* for the copy. Finally, test it!

Properties of Selected Form Controls



Form Control			Control Properties **MUST** (using rS1.Method)	Output Values
1	☑ Caption	Check Box	Cell Link	TRUE FALSE
2	◉ Caption	Option Button	Cell Link	Number according to position in group
3	April May June July August September	List Box	Input Range Cell Link	Number according to click on item
4	June May June July August September	Combo Box	Input Range Cell Link	Number according to click on item
5		Scroll Bar	Minimum Value Maximum Value Cell Link	Number according to click on arrows
6		Spin Button	Minimum Value Maximum Value Cell Link	Number according to click on arrows

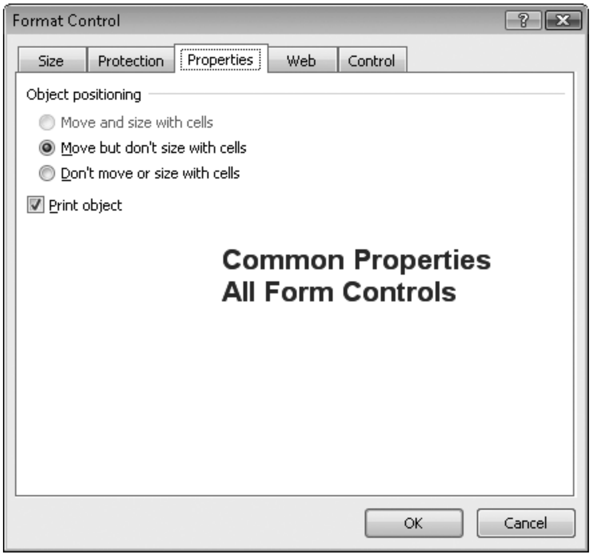
FIGURE 7-11 Properties used in the *Form Controls* in the rS1.Method

Figure 7-11 provides an overview of the control properties of the form controls used in the rS1.Method. The sequence selected there is no longer derived from the rather confusing menu provided on the *Developer* tab (see Figure 7-7). Instead, it is based on the functional “relatedness” of the various objects.

Common Properties of Controls

I used the title of this section to show that the information provided here applies equally to both types of controls.

A *Properties* tab is contained in the *Format Control* dialog box, which you can open from the context menu after you right-click the object, assuming it's a *form control*. If it's an *ActiveX control*, it can only be opened in *design mode* after you right-click the object.



Object positioning

○ Move and size with cells
● Move but don't size with cells
○ Don't move or size with cells

☑ Print object

FIGURE 7-12 These properties can be defined for *all* controls.

Here, you can make the same *object positioning* settings recommended for all graphics objects in Chapter 5, namely *Move but don't size with cells* or *Don't move or size with cells*. However, I would never recommend using the first option (*Move and size with cells*).

Deciding whether an object is to appear in the printed version of a solution will depend, first and foremost, on the aesthetic requirements.

- An argument for: it is useful to print controls together with a chart or table because, in most cases, they are of a similar character to a label or a legend.
- Argument against: printed controls don't generally look great.

Therefore, if you're concerned about the visual impact of the printed version, you should deactivate the printing option here in the dialog box shown, and use formulas to separately generate the important texts that are to be printed (for example, hidden *behind* the controls or elsewhere). If you can't figure out how this works based on the information in this chapter, the technique is illustrated in several examples later.

Note Unfortunately, a minor naming conflict in relation to these *properties* shows up when you use *ActiveX controls*. It has existed for years, and while it remains unresolved will continue to cause small but annoying user errors. The term *properties* is used here to describe what are in fact the complex *functional properties* of the objects. Later in the chapter, you will clearly see that this also applies (in fact, it applies first and foremost) to ActiveX controls.

Formulas

The formulas used in this model may be a little more sophisticated than what an inexperienced Excel user is used to, in particular in relation to the use of names. However, you'll soon become accustomed to such constructs and even learn to enjoy using them, in particular because they are easy to create and understand, despite their sometimes cumbersome appearances.

Note Refer to the section entitled "Entering Formulas" in Chapter 2 and, if necessary, repeat the exercises described there with the example provided in that chapter or with one of your own.

Wherever functions are used in the following text (and in later chapters) that have already been discussed earlier in the book, I'll only explain these briefly if at all (in the current example, the sources of information are Chapter 2 and the *rS1_Method_2007.pdf* file). In cases like this, you'll find the keyword index at the back of the book useful. As a rule, the first page number listed there for a function name directs you to a theoretical explanation of that function.

In a workbook structured according to the rS1.Method, calculations essentially follow the same general structure in which worksheets go from right to left or from the back to the front. The descriptions below follow this same sequence.

Whenever I mention a range containing several formulas of the same type (for example, `K24:V24`), I'll only explain the first formula in each case (here, the formula in cell `K24`). You can assume that the same explanation applies to the other formulas in the range, the only difference being the row and column references if these are used.

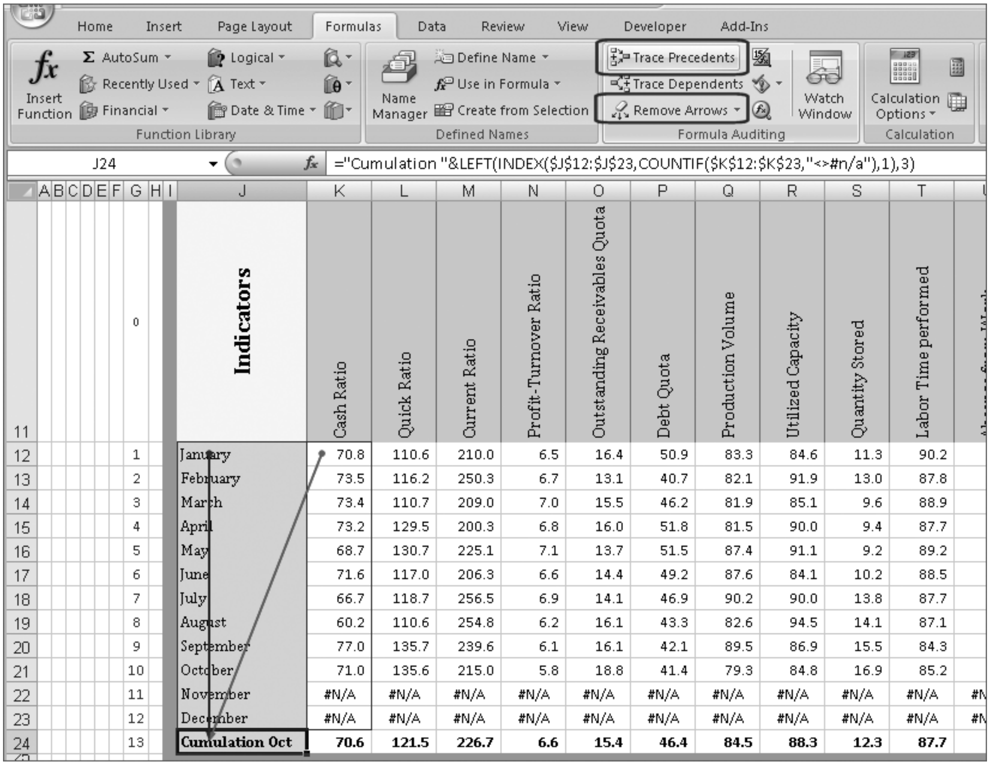
Formulas in the *Data 1* Worksheet

The yearly averages for the indicators are calculated here in row 24. The results of this calculation aren't required in the chart focus but are used for other purposes not specified here. However, it's worth explaining these formulas because they can also be used, in modified form, for many other purposes.

The first formula to discuss is in cell `J24` (see Figure 7-13), and is also the most difficult formula in the entire workbook. It concatenates a character string with the result of three nested sub-formulas. The purpose of the exercise is as follows: we want to generate a row label for the yearly averages that always expresses the current status of the data (in the sense of *complete to*). The formula has 12 possible results, depending on the number of #N/A values in column K; i.e., *Cumulation Jan* through *Cumulation Dec* inclusive.

Let's examine the basic information about the functions used, before turning to the formula itself.

> The =`LEFT(reference,num_chars)` function reads the number of characters (from left to right) in `reference` as determined by `num_chars`.
>
> The =`INDEX(array,row_num,column_num)` function was described in detail in Chapter 2. It reads, within a specified cell range, a value whose position within the cell range is determined by the row and column specified.
>
> The =`COUNTIF(range,criteria)` function counts the frequency with which a search term occurs in `range`. You usually enter the search term in a cell to which the formula refers, rather than in the formula itself (although this would be superfluous in this example). This naturally makes the formula much more flexible, because a new task can be assigned to the formula each time a new search term is entered.



J24 ="Cumulation "&LEFT(INDEX($J$12:$J$23,COUNTIF($K$12:$K$23,"<>#n/a"),1),3)

	G	J	K	L	M	N	O	P	Q	R	S	T
11	0	Indicators	Cash Ratio	Quick Ratio	Current Ratio	Profit-Turnover Ratio	Outstanding Receivables Quota	Debt Quota	Production Volume	Utilized Capacity	Quantity Stored	Labor Time performed
12	1	January	70.8	110.6	210.0	6.5	16.4	50.9	83.3	84.6	11.3	90.2
13	2	February	73.5	116.2	250.3	6.7	13.1	40.7	82.1	91.9	13.0	87.8
14	3	March	73.4	110.7	209.0	7.0	15.5	46.2	81.9	85.1	9.6	88.9
15	4	April	73.2	129.5	200.3	6.8	16.0	51.8	81.5	90.0	9.4	87.7
16	5	May	68.7	130.7	225.1	7.1	13.7	51.5	87.4	91.1	9.2	89.2
17	6	June	71.6	117.0	206.3	6.6	14.4	49.2	87.6	84.1	10.2	88.5
18	7	July	66.7	118.7	256.5	6.9	14.1	46.9	90.2	90.0	13.8	87.7
19	8	August	60.2	110.6	254.8	6.2	16.1	43.3	82.6	94.5	14.1	87.1
20	9	September	77.0	135.7	239.6	6.1	16.1	42.1	89.5	86.9	15.5	84.3
21	10	October	71.0	135.6	215.0	5.8	18.8	41.4	79.3	84.8	16.9	85.2
22	11	November	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
23	12	December	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
24	13	**Cumulation Oct**	**70.6**	**121.5**	**226.7**	**6.6**	**15.4**	**46.4**	**84.5**	**88.3**	**12.3**	**87.7**

FIGURE 7-13 A text is generated by the concatenation of a character string and three nested functions.

```
="Cumulation "&LEFT(INDEX($J$12:$J$23,COUNTIF($K$12:$K$23,"<>#N/A"),1),3)
```

The start of the formula is easy to grasp. It essentially issues the order to Excel to do the following: "Write the string *Cumulation*, followed by a blank space. Then add three text characters (this is indicated by the & text operator), which result from the remainder of the formula." This is where things become a little more difficult. The three text characters to be determined are the first three letters of the *last* month in the range J12:J23 to which *no* `#N/A` value is assigned in the `K12:K23` range. In this example, the month in question is *October*. The correct result of the formula is therefore the text *Cumulation Oct*. Try out some alternatives in the sample file. For example, if you temporarily overwrite the data from the month of May onwards with #N/A, the formula should return *Cumulation Apr* as a result.

How does this work? It's easiest to understand if you take the nesting apart and examine its individual parts separately.

- In our example, the `LEFT` function should take the first three characters from the *October* string. In a very conservative and rigid form, the corresponding formula would be `=LEFT("October",3)`.
- However, we want the formula to be a dynamic one. It should be possible to determine the first three characters in the name of *any* month in the `J12:J23` range. The `reference` argument in the `LEFT` function must therefore be designed as variable.

  You can do this, for example, using the `=INDEX(array,row_num,column_num)` function and by making the `row_num` argument in this function flexible. The formula `=INDEX($J$12:$J$23,10,1)` would calculate the contents of row 10 and column 1 (the first and, in this case, only column) in the `J12:J23 array`, to give the string *October*. So there's nothing flexible there either.

  The formula `=LEFT(INDEX($J$12:$J$23,10,1),3)` would return the string *Oct*. That means that all we need to do now is make the row argument of the `INDEX` formula variable. Rather than a static *10*, we want this to be a number between *1* and *12*. But not just any number in this range. Rather, we want it to indicate the last month in the `K12:K23` cell range to which no `#N/A` value is assigned.
- The number required for the row argument of the `INDEX` formula can be determined here with `COUNTIF`. In our example, `=COUNTIF($K$12:$K$23,"<>#N/A")` returns the value *10*. What it essentially instructs Excel to do is: "Count the number of entries that are not equal to `#N/A` in the `K12:K23` range." If there were only four entries, the formula would return the number 4 instead. In this case, the row argument of the `INDEX` formula would be 4, and `=LEFT(INDEX($J$12:$J$23,4,1),3)` would return the value *Apr*.

I'll recap how these individual elements of the nesting are put together to give a complete formula, working from the inside out:

Element 3:

```
=COUNTIF($K$12:$K$23,"<>#N/A")
```

In our example, this returns the value *10*.

Element 2 with element 3 integrated:

```
=INDEX($J$12:$J$23,COUNTIF($K$12:$K$23,"<>#N/A"),1)
```

In our example, this returns the value *October*.

Element 1 with elements 2 and 3 integrated:

```
=LEFT(INDEX($J$12:$J$23,COUNTIF($K$12:$K$23,"<>#N/A"),1),3)
```

In our example, this returns the value *Oct*.

The complete formula is as follows:

```
="Cumulation "&LEFT(INDEX($J$12:$J$23,COUNTIF($K$12:$K$23,"<>#N/A"),1),3)
```

In our example, this returns the value *Cumulation Oct*.

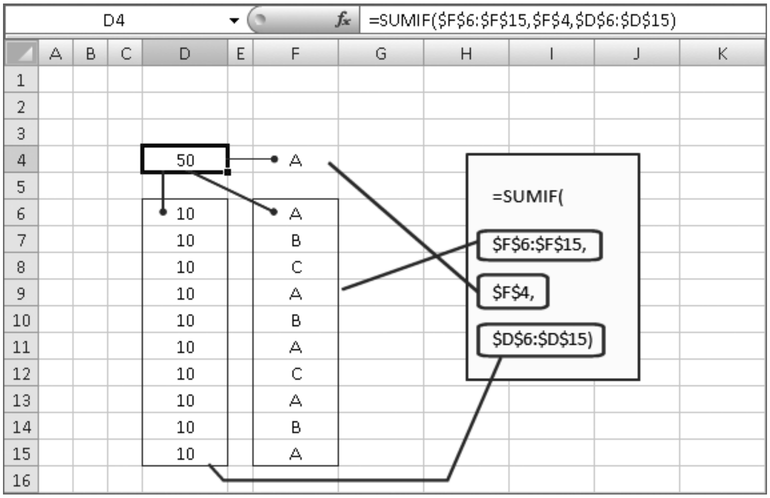
The following three aspects are also important in building this formula:

- If you use names instead of the cell references `$J$12:$J$23` and `$K$12:$K$23`, the formula becomes a little more unwieldy. We'll address this later in relation to variant 2.
- The formula requires that the complete `K12:K23` range is filled with values from top to bottom, which are then followed by `#N/A` values for the months for which indicators are not yet available.
- In the company in our example, the first indicator (*Cash Ratio*) is the most important because it is the most sensitive and must be examined more than once a month. If any of the indicators are going to be ready when needed, it's this one. The `K12:K23` range therefore serves as a reference range to indicate how up-to-date the data is. This is in principle a failing in the solution, but one that's acceptable without difficulty.

I should also point out that, as is almost always the case in Excel, there were several other ways in which we could have accomplished this same task. If you're an experienced user, you can take this as a hint to experiment on your own.

Let's turn now to the formulas in the `K24:V24` range, which are used to calculate the average of the indicators to date. First, I'll briefly introduce another function that's needed here:

The =`SUMIF(range,criteria,sum_range)` function is the equivalent of `COUNTIF` and can be used in several different ways. It searches for `criteria` in `range` and, in the same or a parallel range (in `sum_range`, which may be any distance from the range), it adds together the contents of cells. The example shown in Figure 7-14 explains the principle much more effectively than a longer description that I could provide.



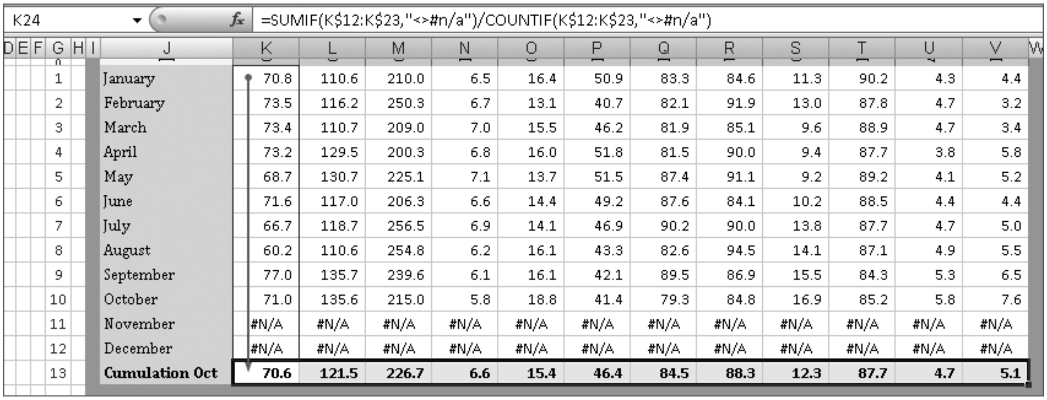
D4 =SUMIF($F$6:$F$15,$F$4,$D$6:$D$15)

	A	B	C	D	E	F	G	H	I	J	K
1											
2											
3											
4				50		A					
5											
6				10		A					
7				10		B					
8				10		C					
9				10		A					
10				10		B					
11				10		A					
12				10		C					
13				10		A					
14				10		B					
15				10		A					
16											

FIGURE 7-14 Syntax of the *SUMIF* function

What it essentially instructs Excel to do is as follows: "In the `D6:D15` range, add up the values of the cells whose neighboring cells in column F (range `F6:F15`) contain the search term from cell F3 (i.e., *A*)."

The variant used in the following example doesn't require the third formula argument (`sum_range`) because it adds values in the same range in which the search criteria are found.



K24 =SUMIF(K$12:K$23,"<>#n/a")/COUNTIF(K$12:K$23,"<>#n/a")

G	J	K	L	M	N	O	P	Q	R	S	T	U	V
1	January	70.8	110.6	210.0	6.5	16.4	50.9	83.3	84.6	11.3	90.2	4.3	4.4
2	February	73.5	116.2	250.3	6.7	13.1	40.7	82.1	91.9	13.0	87.8	4.7	3.2
3	March	73.4	110.7	209.0	7.0	15.5	46.2	81.9	85.1	9.6	88.9	4.7	3.4
4	April	73.2	129.5	200.3	6.8	16.0	51.8	81.5	90.0	9.4	87.7	3.8	5.8
5	May	68.7	130.7	225.1	7.1	13.7	51.5	87.4	91.1	9.2	89.2	4.1	5.2
6	June	71.6	117.0	206.3	6.6	14.4	49.2	87.6	84.1	10.2	88.5	4.4	4.4
7	July	66.7	118.7	256.5	6.9	14.1	46.9	90.2	90.0	13.8	87.7	4.7	5.0
8	August	60.2	110.6	254.8	6.2	16.1	43.3	82.6	94.5	14.1	87.1	4.9	5.5
9	September	77.0	135.7	239.6	6.1	16.1	42.1	89.5	86.9	15.5	84.3	5.3	6.5
10	October	71.0	135.6	215.0	5.8	18.8	41.4	79.3	84.8	16.9	85.2	5.8	7.6
11	November	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
12	December	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
13	**Cumulation Oct**	**70.6**	**121.5**	**226.7**	**6.6**	**15.4**	**46.4**	**84.5**	**88.3**	**12.3**	**87.7**	**4.7**	**5.1**

FIGURE 7-15 How to calculate averages within ranges containing numbers and *#N/A* Values

The AVERAGE Excel function calculates a value based on *total divided by count*. This wouldn't work in this case because the reference range of a corresponding formula (such as =AVERAGE(K12:K23)) contains #N/A values, and so the result of the formula would also be #N/A. What we need to do is keep the principle of *total divided by count*, but exclude the cells containing #N/A values from the calculation. One possible solution would be the following:

```
K24    =SUMIF(K$12:K$23,"<>#N/A")/COUNTIF(K$12:K$23,"<>#N/A")
```

This essentially instructs Excel to do the following: "Add up the cells in the range K12:K23 that contain values other than #N/A, and divide the result by the number of cells in the range K12:K23 that contain values other than #N/A."

Formulas in the *Basis 1* Worksheet

This worksheet provides the basis for the dynamic chart in the form of a table. The table is a variable collection of the data required to visualize the chart. It is based on formulas containing arguments that are partly determined by controls.

This worksheet largely consists of formulas, with the relevant cells highlighted in color. The bright orange border around the K11:O23 cell range indicates the data basis of the chart.

Roughly speaking, the task of the formulas can be classified as follows: if specific conditions are met, a certain type of data is to be read (extracted) from the *Data 1* worksheet, and otherwise, not. Two functions are used to fulfill this task:

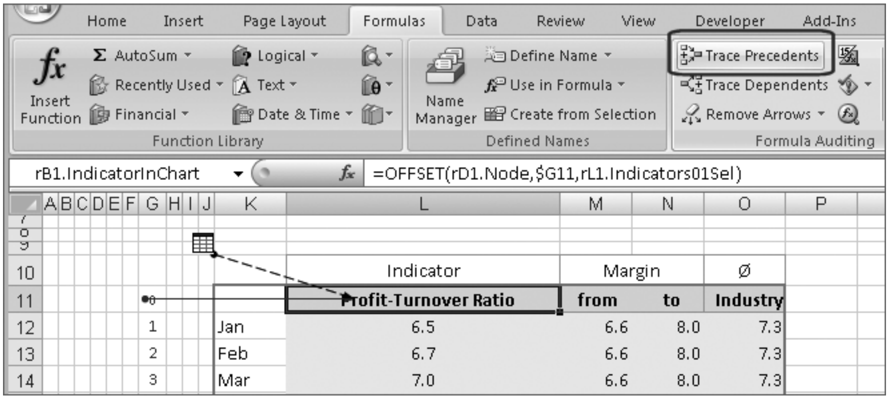
The `=OFFSET(reference,rows,cols,height,width)` function is explained in detail in Chapter 12, “More Than Numbers.” It returns the value located in a cell that is a certain number of `rows` and `cols` away from the `reference`. The `reference` argument thus defines the starting point for accessing another cell or cell range. The optional `height` and `width` arguments are not of relevance in this example.

In its basic form, the much-used function `=IF(logical_test,value_if_true,value_if_false)` returns the logical value `TRUE` or `FALSE` as the result of `logical_test`. In the `logical_test` part of the formula, you almost make a “claim,” and Excel then checks to see if this claim is true. If the result of `logical_test = TRUE`, Excel follows the instruction specified in the `value_if_true` part of the formula. If the result of `logical_test = FALSE`, Excel follows the instruction specified in the `value_if_false` part of the formula.

Note In some of the graphics below, “traces of precedent” cells are shown in relation to the display of formulas. These may explain why other cells (or their content) affect the result of the formula in the cell currently selected. The pertinent question becomes: Where are the values located that the formula in the selected cell requires in order to return a correct result? If these “supplier cells” are located in the same worksheet, the traces in question are indicated with arrows and, in some cases, with borders. If they are located in another worksheet, a small table icon appears at the end of a dotted arrow line. If you double-click this line, the *Go to* dialog box opens. There you can see the cross-sheet reference, and you can specify whether you want to go to that worksheet for the purpose of verification.

The “Trace Precedents” tool is located in the *Formula Auditing* group of the *Formulas* tab in the Ribbon. The *Trace Dependents* command in the same group allows you to locate the cells that are affected by (i.e., depend on) the value in the cell that is currently selected. The *Remove Arrows* command deletes the traces.

The formulas in cell `L11` determine the label text of the indicator category configured by the user for use in the legend labels.



rB1.IndicatorInChart =OFFSET(rD1.Node,$G11,rL1.Indicators01Sel)

	G	K	L	M	N	O
10			Indicator	Margin		Ø
11	0		Profit-Turnover Ratio	from	to	Industry
12	1	Jan	6.5	6.6	8.0	7.3
13	2	Feb	6.7	6.6	8.0	7.3
14	3	Mar	7.0	6.6	8.0	7.3

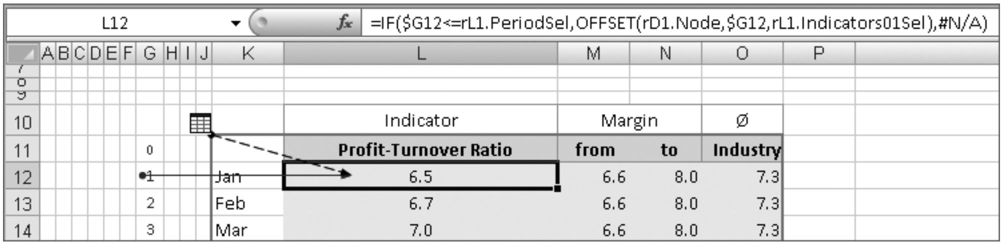
FIGURE 7-16 Formula for determining category labeling

```
L11    =OFFSET(rD1.Node,$G11,rL1.Indicators01Sel)
```

This essentially serves as an instruction to Excel to do the following: "Based on the cell with the name *rD1.Node*, read the value in another cell. Starting at *rD1.Node*, move down the number of rows specified by the number in cell G11, and move the number of columns to the right as is currently specified by the number in the *rL1.Indicators01Sel* cell."

The user thus varies the column argument of the formula based on the list selection made in the *oF1_BoxIndicators01* combo box, which is linked with the *rL1.Indicators01Sel* cell.

The formulas in the L12:L23 range calculate the month-specific values of the indicator category selected by the user, taking account of the visualization period configured by the user.



L12 =IF($G12<=rL1.PeriodSel,OFFSET(rD1.Node,$G12,rL1.Indicators01Sel),#N/A)

	G	K	L	M	N	O
10			Indicator	Margin		Ø
11	0		Profit-Turnover Ratio	from	to	Industry
12	1	Jan	6.5	6.6	8.0	7.3
13	2	Feb	6.7	6.6	8.0	7.3
14	3	Mar	7.0	6.6	8.0	7.3

FIGURE 7-17 Formulas for calculating indicator values

```
L12    =IF($G12<=rL1.PeriodSel,
       OFFSET(rD1.Node,$G12,rL1.Indicators01Sel),
       #N/A)
```

What it essentially instructs Excel to do is as follows: "If the value in cell G12 is less than or equal to the value in the cell with the name *rL1.PeriodSel*, start at the *rD1.Node* cell and, from there, read the value in another cell. Starting at *rD1.Node*, move down the number of rows specified by the value in cell G12, and move across the number of columns to the right that are currently specified by the number in the *rL1.Indicators01Sel* cell. Otherwise (i.e., if the value in cell G12 is *not* less than or equal to the value in the cell with the name *rL1.PeriodSel*), write the error value #N/A."

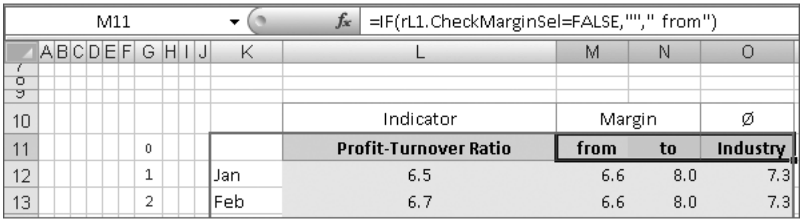
The user therefore determines which indicator category is displayed and the period for which the corresponding data series is to be drawn. To be more precise:

- The list selection in the *oF1_BoxIndicators01* combo box, which is linked with the *rL1.Indicators01Sel* cell, determines the selection of the indicator category by generating the column argument of the OFFSET … part of the formula.
- The user's list selection in the *oF1_BoxPeriod* combo box or a click in the *oF1_ScrollPeriod* scroll bar, both of which are linked with the *rL1.PeriodSel* cell, determines whether this part of the IF formula is used, or whether the #N/A value is written instead.

For example: The user selects the period *Jan – Jun*.

- The value in *rL1.PeriodSel* is *6*. It is therefore less than or equal to the *6* that is shown in cell G17, and is therefore assigned to the "June row." The check result of the IF formula is TRUE, and it consequently accesses a cell in the *Data 1* worksheet.
- The value in *rL1.PeriodSel* is *6*. Therefore, it is *not* less than or equal to the *7* that is shown in cell G18, and is therefore assigned to the "July row." The check result of the IF formula is therefore FALSE, and it consequently does *not* access a cell in the *Data 1* worksheet, but instead returns the error value #N/A. As a result, no more data points are drawn in the assigned data series of the chart as of this point.

The formulas in cells M11 and N11 determine whether column headers are shown or hidden (for use in the legend labels).



M11 =IF(rL1.CheckMarginSel=FALSE,""," from")

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
7															
8															
9															
10												Indicator	Margin		Ø
11							0					**Profit-Turnover Ratio**	**from**	**to**	**Industry**
12							1				Jan	6.5	6.6	8.0	7.3
13							2				Feb	6.7	6.6	8.0	7.3

FIGURE 7-18 Additional formulas for determining legend labels

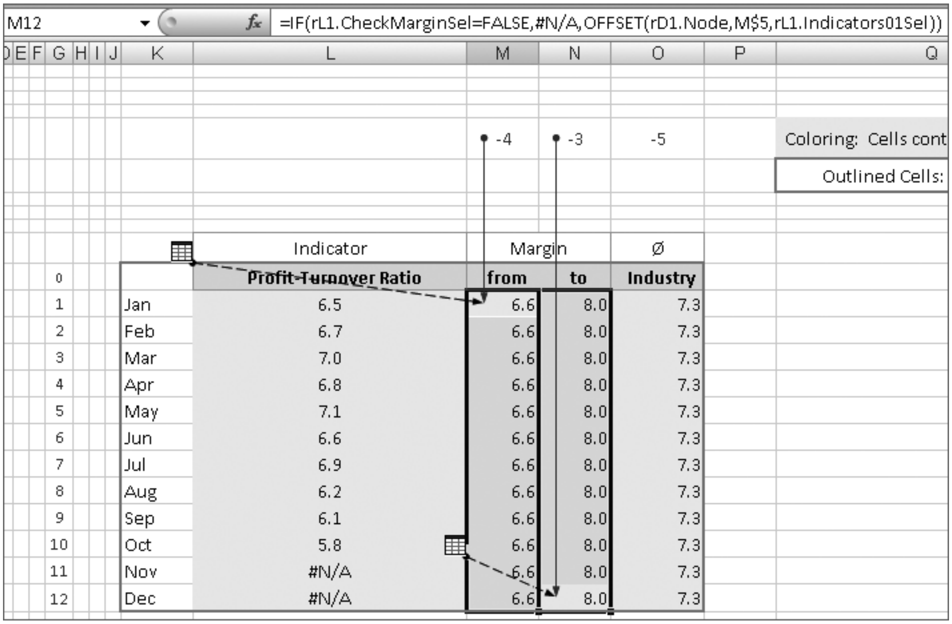
```
M11   =IF(rL1.CheckMarginSel=FALSE,"","  from")

N11   =IF(rL1.CheckMarginSel=FALSE,"","  to")
```

This essentially instructs Excel to do the following (formula in cell M11): "If the value in the *rL1.CheckMarginSel* cell is FALSE, write nothing. Otherwise, write the string *from* (or, in the case of cell N11, the string *to*)."

The user thus determines whether or not the column header is shown by clicking the *oF1_CheckMargin* checkbox, which is linked with the *rL1.CheckMarginSel* cell. This is important because this column header supplies the corresponding legend entry. Always remember: A legend text belonging to a line that is *not* currently visible in the chart should also be invisible.

> **Note** If you instruct Excel to write *nothing* (i.e., if you use the string "" (no characters between the quotation marks), an "empty text" is generated instead. A cell that contains an empty text appears to the user as though it actually contains *nothing* (no value). For Excel, however, this cell is *not* empty (empty text is considered a value in Excel). This is an important difference, which may cause errors after some copy operations and subsequent formula constructions that refer to a cell of this type (for example, if you use the `COUNTA` function). These errors are sometimes difficult to detect. Therefore, proceed with caution when you copy cells like this to another location or refer to such cells in formulas.

The formulas in cells `M12:M23` and `N12:N23` determine whether `#N/A` values or the margin values for the selected indicator category are to be read.



M12 =IF(rL1.CheckMarginSel=FALSE,#N/A,OFFSET(rD1.Node,M$5,rL1.Indicators01Sel))

G	K	L	M	N	O	Q
			-4	-3	-5	Coloring: Cells cont
						Outlined Cells:
		Indicator	Margin		Ø	
0		Profit-Turnover Ratio	from	to	Industry	
1	Jan	6.5	6.6	8.0	7.3	
2	Feb	6.7	6.6	8.0	7.3	
3	Mar	7.0	6.6	8.0	7.3	
4	Apr	6.8	6.6	8.0	7.3	
5	May	7.1	6.6	8.0	7.3	
6	Jun	6.6	6.6	8.0	7.3	
7	Jul	6.9	6.6	8.0	7.3	
8	Aug	6.2	6.6	8.0	7.3	
9	Sep	6.1	6.6	8.0	7.3	
10	Oct	5.8	6.6	8.0	7.3	
11	Nov	#N/A	6.6	8.0	7.3	
12	Dec	#N/A	6.6	8.0	7.3	

FIGURE 7-19 Formulas for extracting margin values from *Data 1*

```
M12:M23   =IF(rL1.CheckMarginSel=FALSE,
          #N/A,
          OFFSET(rD1.Node,M$5,rL1.Indicators01Sel))
```

```
N12:N23   =IF(rL1.CheckMarginSel=FALSE,
          #N/A,
          OFFSET(rD1.Node,N$5,rL1.Indicators01Sel))
```

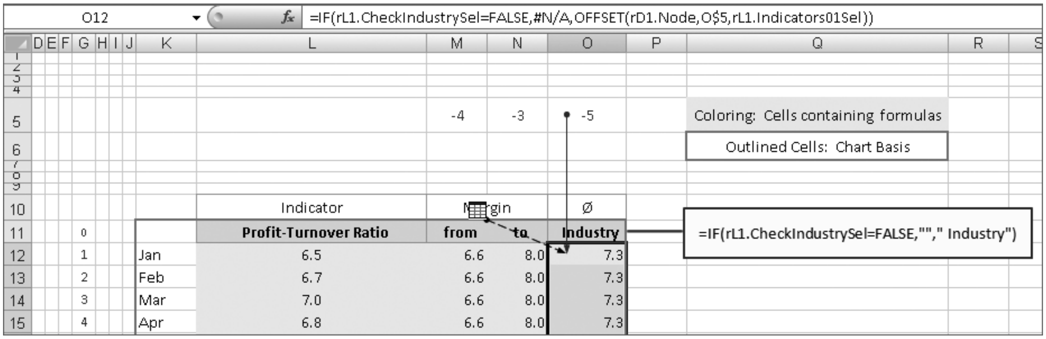
This essentially instructs Excel to do the following (according to the formula in cell M12): “If the value in the *rL1.CheckMargin* cell is FALSE, return the error value #N/A. Otherwise, read the value in a cell related to the *rD1.Node* cell. Starting at *rD1.Node*, move down the number of rows specified by the value in cell M5, and move across the number of columns to the right that is currently specified by the value in the *rL1.Indicators01Sel* cell.”

The user therefore decides two things here:

- First, clicking the *oF1_CheckMargin* checkbox, which is linked with the *rL1.CheckMarginSel* cell, determines whether the cells are to be filled with #N/A values, If not:
- The list selection in the *oF1_BoxKategorie01* combo box, which is linked with the *rL1.Indicators01Sel* cell, determines the column argument of the OFFSET … part of the formula.

The formula in cell O11 determines whether the column header is to be displayed or not.

The formulas in cells O12:O23 determine whether #N/A values are used, or whether the industry values for the selected indicator category are to be read.



O12 =IF(rL1.CheckIndustrySel=FALSE,#N/A,OFFSET(rD1.Node,O$5,rL1.Indicators01Sel))

	G	K	L	M	N	O
5				-4	-3	-5
10			Indicator	Margin		Ø
11	0		Profit-Turnover Ratio	from	to	Industry
12	1	Jan	6.5	6.6	8.0	7.3
13	2	Feb	6.7	6.6	8.0	7.3
14	3	Mar	7.0	6.6	8.0	7.3
15	4	Apr	6.8	6.6	8.0	7.3

Coloring: Cells containing formulas

Outlined Cells: Chart Basis

=IF(rL1.CheckIndustrySel=FALSE,""," Industry")

FIGURE 7-20 Formulas for determining the column header and industry values

These formulas work on the same principle as that described above in connection with Figure 7-18 and Figure 7-19. The only differences here are the values and references used.

```
O11    =IF(rL1.CheckIndustrySel=FALSE,"","  Industry”)
```

and

```
O12:O23   =IF(rL1.CheckIndustrySel=FALSE,
          #N/A,
          OFFSET(rD1.Node,O$5,rL1.Indicators01Sel))
```

Tip Don’t generate the chart until you’ve created and tested all of the formulas in the *Basis 1* worksheet. Start by generating the chart in its basic form with some cursory formatting in the *Basis 1* worksheet, and test how well it responds when you use the various controls. Only copy the chart, together with the controls, to the *Focus 1* worksheet when you’re ready to add the finishing touches and formatting to it.

Formulas in the *Focus 1* Worksheet

You’ve already seen the only formula in this worksheet and its location in Figure 7-5.

```
V4     =rB1.IndicatorInChart&” / “&
       OFFSET(rL1.PeriodHeader,rL1.PeriodSel,0)
```

This formula instructs Excel to do the following: “Read the contents of the *rB1.IndicatorInChart* cell. Then add a character string consisting of a blank space, a forward slash, and another blank space. Then add a formula result to this. To get this result, go to the *rL1.PeriodHeader* cell, and move down the number of rows specified by the value in the *rL1.PeriodSel* cell, but do not move across any columns (0 columns) to the right or left.”

The following formula would produce the same result (if you’d like to test it also):

```
P4     =rB1.IndicatorInChart&" / "&INDEX(rL1.PeriodList,rL1.PeriodSel,1)
```

Variant 2 (Advanced Model)

This section follows the same pattern as the last, and explains how to create the *0702_Indicators_02.xlsx* file. It focuses in particular on its dynamic elements, namely controls, formulas, and *conditional formatting*. Note that this section is limited to a discussion of the elements of this file that clearly differ from those of variant 1 of the model as discussed in the previous section (i.e., the basic version in the *0701_Indicators_01.xlsx* file). For the following discussion, I assume that you’re already familiar with the information provided there.

On the Companion CD Open the *\Samples\0702_Indicators_02.xlsx* file on the CD-ROM.

The advanced model is a little easier to use, and also implements the principles of the rS1.Method more consistently. The following paragraphs explain the main differences between it and the basic version in relation to the workbook structure.

How Is the Workbook Structured?

As usual, I'll answer this question by going from right to left in the workbook.

NamesIndex Worksheet

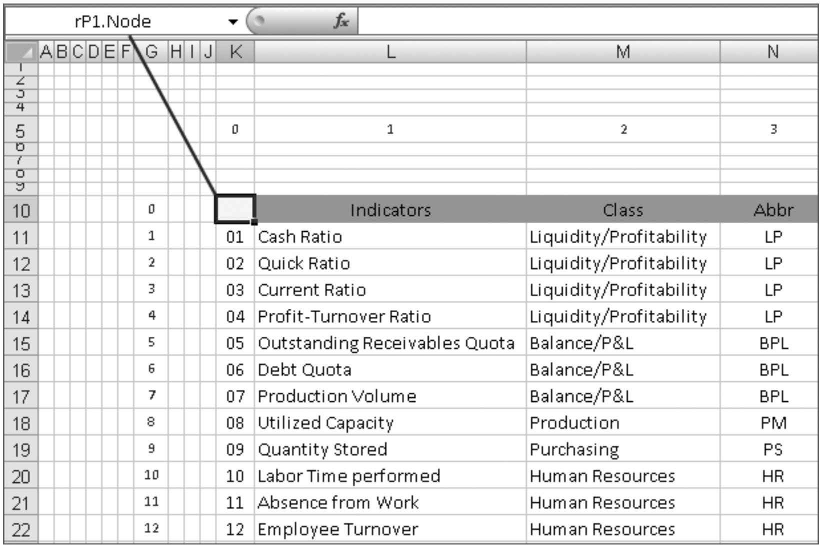
You can see that the number of range names listed here is calculated using the `=COUNTA(L:L)&" Names"` formula.

What this formula essentially instructs Excel to do is as follows: "Count the number of entries (of any type) in all of column L, and add a blank space, followed by the string *Names* to the result."

> The =`COUNTA(reference)` calculates the number of entries in `reference`. It counts entries of any kind, even the "empty text" ("") explained above.
>
> The =`COUNT(reference)` function only counts the numbers in `reference`.

Clearly, a couple of names have been added here to the list from the basic version (variant 1). I'll come back to these later when describing the formulas.

Parameters 1 Worksheet



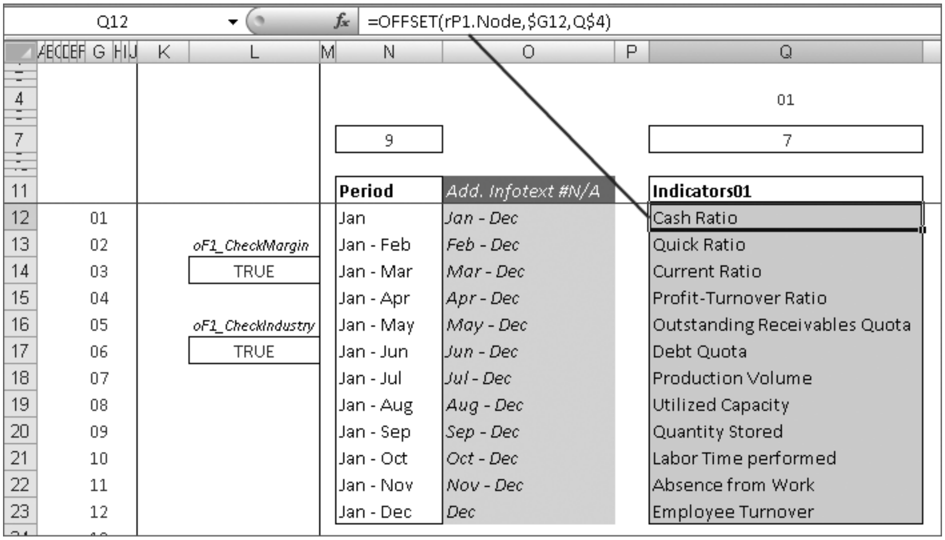
	G	K	L	M	N
5		0	1	2	3
10	0		Indicators	Class	Abbr
11	1	01	Cash Ratio	Liquidity/Profitability	LP
12	2	02	Quick Ratio	Liquidity/Profitability	LP
13	3	03	Current Ratio	Liquidity/Profitability	LP
14	4	04	Profit-Turnover Ratio	Liquidity/Profitability	LP
15	5	05	Outstanding Receivables Quota	Balance/P&L	BPL
16	6	06	Debt Quota	Balance/P&L	BPL
17	7	07	Production Volume	Balance/P&L	BPL
18	8	08	Utilized Capacity	Production	PM
19	9	09	Quantity Stored	Purchasing	PS
20	10	10	Labor Time performed	Human Resources	HR
21	11	11	Absence from Work	Human Resources	HR
22	12	12	Employee Turnover	Human Resources	HR

FIGURE 7-21 Master-data maintenance is used here.

Variant 1 doesn't have this worksheet. The main advantage of using it is that it allows you to manage and maintain your master data centrally. In this case, the master data consists of the names of the indicator categories. On the right next to the list of indicator categories is a list of the groups (in full text and abbreviated form) to which these belong. As you can see in Figure 7-21, this worksheet contains an *rP1.Node* (cell K10). This node is used to extract information with formulas and transfer this information to the *Data 1* and *Lists 1* worksheets (and thus also to the drop-down lists of the relevant controls). If changes need to be made to these texts, they only need to be made in the *Parameters 1* worksheet.

Lists 1 Worksheet

The appearance of this worksheet has also changed a little from the basic variant.



Q12 =OFFSET(rP1.Node,$G12,Q$4)

	G	L	N	O	Q
4					01
7			9		7
11			Period	Add. Infotext #N/A	Indicators01
12	01		Jan	Jan - Dec	Cash Ratio
13	02	oF1_CheckMargin	Jan - Feb	Feb - Dec	Quick Ratio
14	03	TRUE	Jan - Mar	Mar - Dec	Current Ratio
15	04		Jan - Apr	Apr - Dec	Profit-Turnover Ratio
16	05	oF1_CheckIndustry	Jan - May	May - Dec	Outstanding Receivables Quota
17	06	TRUE	Jan - Jun	Jun - Dec	Debt Quota
18	07		Jan - Jul	Jul - Dec	Production Volume
19	08		Jan - Aug	Aug - Dec	Utilized Capacity
20	09		Jan - Sep	Sep - Dec	Quantity Stored
21	10		Jan - Oct	Oct - Dec	Labor Time performed
22	11		Jan - Nov	Nov - Dec	Absence from Work
23	12		Jan - Dec	Dec	Employee Turnover

FIGURE 7-22 A reverse list in column *O* and a parameter copy in column *Q*

Column Q now contains formulas that can be used to read content from *Parameter 1*.

Users can configure 12 different time periods in the *Focus 1* worksheet. These include periods for which data is not yet available. This doesn't usually cause any problems or inconvenience, but it's worth adding a user-friendly tip all the same. You can do this in the form of an infotext. Assume, for example, that a user selects the period *Jan – Nov*. In this case, you want the following infotext to appear: *No Data for Period Nov - Dec 2008*. As you'll see later, this is easily done using the "reverse list" in column O, which contains the required text modules.

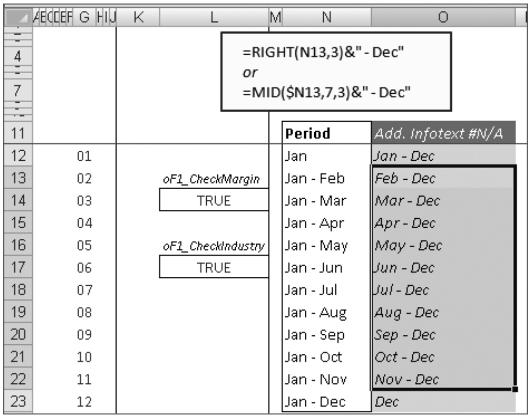
The reverse list is spatially assigned to the definition structures of the drop-down list for selecting the period, and therefore can simply be linked with this list. If the user clicks the eleventh entry in the drop-down list (*Jan – Nov*), the eleventh text module from the reverse list (*Nov – Dec*) is used.

It doesn't take long to manually enter a short list like this. However, if you'd prefer to avoid this tedious chore or need to enter larger and more complex lists, you can use formulas to do this job for you.

Using text formulas, it's often possible to generate a new list from an existing one (for example, to correct an incorrect list). As a prerequisite, a consistent default structure must usually exist in the data source; in other words, a structure that can be used when generating a new list.

Tip Beware: The same structure may also mean the same errors! Complex and fast error correction in text lists is sometimes much easier in Excel than in Word because corrective text formulas can be used. A list of available text functions (and other function groups) is provided on the CD-ROM under *Materials**Functions.xlsx*.

In our example, the relevant range is `N13:N22`, which you can use as an identically structured data source. Enter the contents of cell `O12` and `O23` manually, and use formulas to generate the remaining entries in the range `O13:O22`. Two ways of doing this are shown in Figure 7-23.



=RIGHT(N13,3)&" - Dec"
or
=MID($N13,7,3)&" - Dec"

	G	L	N	O
11			**Period**	*Add. Infotext #N/A*
12	01		Jan	*Jan - Dec*
13	02	*oF1_CheckMargin*	Jan - Feb	*Feb - Dec*
14	03	TRUE	Jan - Mar	*Mar - Dec*
15	04		Jan - Apr	*Apr - Dec*
16	05	*oF1_CheckIndustry*	Jan - May	*May - Dec*
17	06	TRUE	Jan - Jun	*Jun - Dec*
18	07		Jan - Jul	*Jul - Dec*
19	08		Jan - Aug	*Aug - Dec*
20	09		Jan - Sep	*Sep - Dec*
21	10		Jan - Oct	*Oct - Dec*
22	11		Jan - Nov	*Nov - Dec*
23	12		Jan - Dec	*Dec*

FIGURE 7-23 Faster than manual entry: most of the list can be generated using formulas.

The =`RIGHT(reference,num_chars)` function reads in `reference` the number of characters, starting from the right, that are specified in `num_chars`.

The =`MID(reference,start_num,num_chars)` function reads in `reference` the number of characters, starting at `start_num` (specifies a position as a number), that are specified in `num_chars`.

Once you've generated a list like this with formulas, the formulas themselves become superfluous and can be replaced by their results. Therefore, you can copy the range with the formulas and then overwrite this by selecting *Home/Paste/Paste Values*.

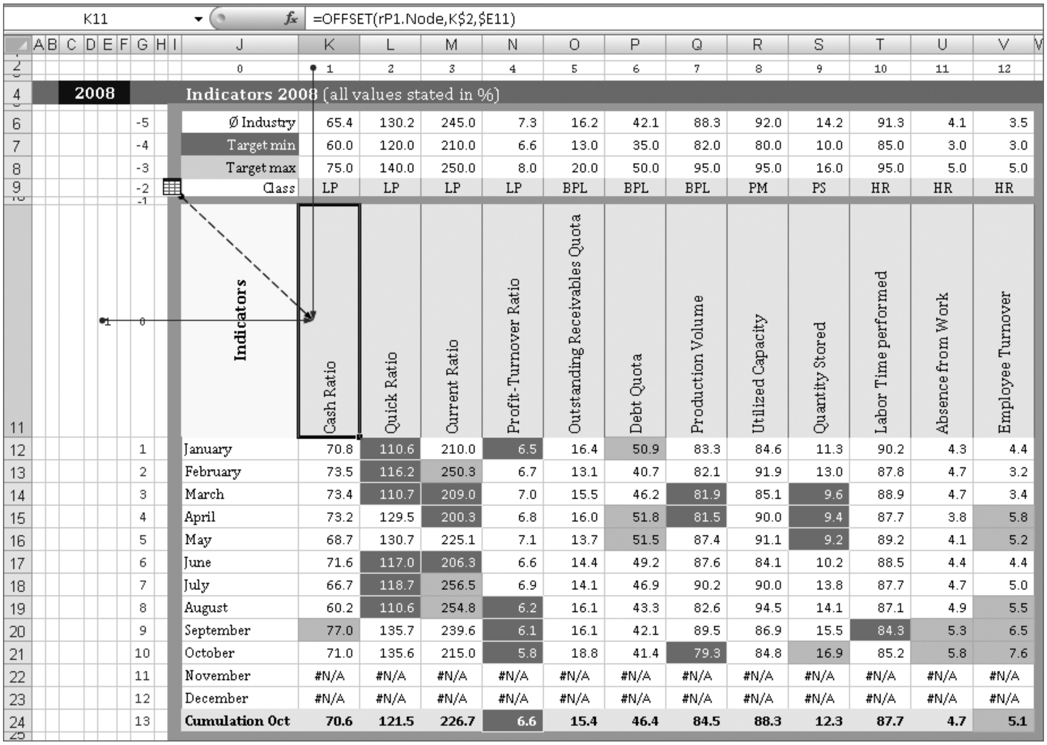
Data 1 Worksheet

In this worksheet, OFFSET formulas are used to read texts defined in the *Parameters 1* worksheet. These texts are the indicator names in the K11:V11 range and the group IDs in the K9:V9 range.

More important, however, is the fact that the K12:V24 range contains conditional formatting, which is used to indicate where the indicators move outside of the desired or anticipated margins. I'll explain how to generate this formatting in a separate section later. At this point, a few remarks about how these are designed and used will suffice.

This worksheet is used by the company's financial controllers, rather than by the Management team. Each month, all managers receive a protected version of the latest model, with all of the worksheets hidden except for *Focus 1*.

Once an indicator is entered or read, a color signal immediately appears if the values rise above or drop below the acceptable margins defined for that indicator (in rows 7 and 8). Try this out for yourself by temporarily changing some of the figures.



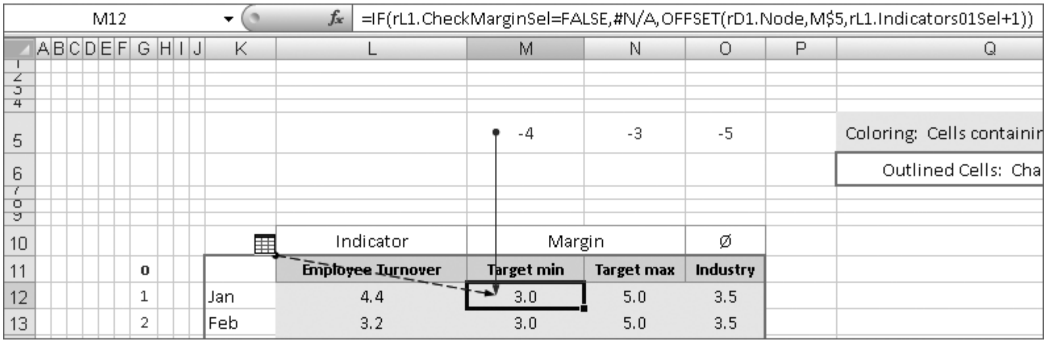
K11 =OFFSET(rP1.Node,K$2,$E11)

	E	G	J	K	L	M	N	O	P	Q	R	S	T	U	V
2			0	1	2	3	4	5	6	7	8	9	10	11	12
4	2008		Indicators 2008 (all values stated in %)												
6		-5	Ø Industry	65.4	130.2	245.0	7.3	16.2	42.1	88.3	92.0	14.2	91.3	4.1	3.5
7		-4	Target min	60.0	120.0	210.0	6.6	13.0	35.0	82.0	80.0	10.0	85.0	3.0	3.0
8		-3	Target max	75.0	140.0	250.0	8.0	20.0	50.0	95.0	95.0	16.0	95.0	5.0	5.0
9		-2	Class	LP	LP	LP	LP	BPL	BPL	BPL	PM	PS	HR	HR	HR
10		-1													
11	1	0	Indicators	Cash Ratio	Quick Ratio	Current Ratio	Profit-Turnover Ratio	Outstanding Receivables Quota	Debt Quota	Production Volume	Utilized Capacity	Quantity Stored	Labor Time performed	Absence from Work	Employee Turnover
12		1	January	70.8	110.6	210.0	6.5	16.4	50.9	83.3	84.6	11.3	90.2	4.3	4.4
13		2	February	73.5	116.2	250.3	6.7	13.1	40.7	82.1	91.9	13.0	87.8	4.7	3.2
14		3	March	73.4	110.7	209.0	7.0	15.5	46.2	81.9	85.1	9.6	88.9	4.7	3.4
15		4	April	73.2	129.5	200.3	6.8	16.0	51.8	81.5	90.0	9.4	87.7	3.8	5.8
16		5	May	68.7	130.7	225.1	7.1	13.7	51.5	87.4	91.1	9.2	89.2	4.1	5.2
17		6	June	71.6	117.0	206.3	6.6	14.4	49.2	87.6	84.1	10.2	88.5	4.4	4.4
18		7	July	66.7	118.7	256.5	6.9	14.1	46.9	90.2	90.0	13.8	87.7	4.7	5.0
19		8	August	60.2	110.6	254.8	6.2	16.1	43.3	82.6	94.5	14.1	87.1	4.9	5.5
20		9	September	77.0	135.7	239.6	6.1	16.1	42.1	89.5	86.9	15.5	84.3	5.3	6.5
21		10	October	71.0	135.6	215.0	5.8	18.8	41.4	79.3	84.8	16.9	85.2	5.8	7.6
22		11	November	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
23		12	December	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A	#N/A
24		13	**Cumulation Oct**	**70.6**	**121.5**	**226.7**	**6.6**	**15.4**	**46.4**	**84.5**	**88.3**	**12.3**	**87.7**	**4.7**	**5.1**

FIGURE 7-24 Formulas and conditional formatting in the data sheet

If the value drops below the lower limit, it appears in a white font on a buff-colored background. If it rises above the upper limit, the background changes to light orange. Of course, neither color scheme is intended as a “real” warning signal. Here, the only purpose is to draw attention to the values. Given the type and composition of the indicators to be analyzed here, this solution is sufficient. In any case, no alternative is available. After all, exceeding or falling short of a company-specific threshold value may or may not be a problem, depending on the individual indicator and situation. In each case, the degree of fluctuation and the influence of other factors must also be taken into account. Therefore, we have to leave it to the expertise and interpretative capabilities of the user to assess the significance of the fluctuations within or outside of the defined margins. The simple and overused traffic-light system (where green is good and red is bad) is not suitable here.

Basis 1 Worksheet



```
M12    =IF(rL1.CheckMarginSel=FALSE,#N/A,OFFSET(rD1.Node,M$5,rL1.Indicators01Sel+1))
```

	M	N	O		Q
5	-4	-3	-5		Coloring: Cells containir
6					Outlined Cells: Cha

G	K	L	M	N	O
		Indicator	Margin		Ø
0		Employee Turnover	Target min	Target max	Industry
1	Jan	4.4	3.0	5.0	3.5
2	Feb	3.2	3.0	5.0	3.5

FIGURE 7-25 There are few changes here compared with variant 1.

This worksheet is almost identical to variant 1. The following minor adjustments have been made:

- The columns headers in cell `N11` and `M11` were changed. The texts *from* and *to* have been replaced with *Target min* and *Target max*.
- The formulas designed earlier for the *controls* need to be slightly modified because *ActiveX controls* are used in this variant. More about that later.

Focus 1 Worksheet

The revisions and modifications are most obvious in the *Focus 1* worksheet and not just because the colors have changed.

- The chart legend is now positioned below the plot area. It now has a less obvious impact, and no longer interferes with the visual continuity between the data series and the control panel. In any case, the legend is only of minor importance in this solution and scenario. In fact, it could almost be dispensed with, because the necessary information is also well presented in the control area. However, this worksheet is also intended to be used as a color printout for external reports. The printout will only contain the chart itself and its header bar. Therefore, the legend must remain intact as it provides important additional information.

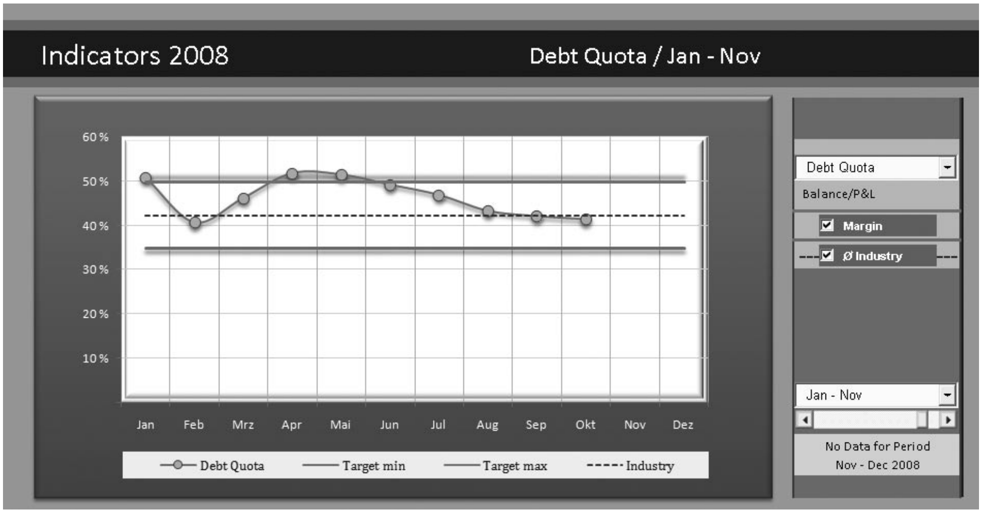


FIGURE 7-26 In many respects an improvement on variant 1

- The control panel has been redesigned in several ways:
  - The controls are now organized into thematic groups. The configuration options for the period (which are rarely needed or used) have found their rightful place at the bottom of this panel, and in proximity to the chart's horizontal primary axis, which indicates the months.
  - The *ActiveX controls* allow us to add a fill color here. The two most important selection elements (the drop-down lists for the indicator category and the period) have been visually highlighted.
  - When an indicator is selected, an infotext appears below this, indicating the indicator category to which it belongs.
  - With the two *checkboxes*, the relevant line colors from the chart are displayed as information in the legend.
  - An infotext (discussed earlier in relation to the *Lists 1* sheet) appears below the two controls for selecting the period if the user selects a period for which all of the relevant data is not yet available. It is highlighted against a color background, and it disappears (together with the color fill) as soon as a period is selected for which all of the relevant data is currently available. This is achieved by using a formula in conjunction with conditional formatting.

ActiveX Controls

ActiveX controls have the same function as the form controls described earlier in relation to variant 1 of our model, but they are much more complex and much more effective in many ways. The configuration options for these controls are extensive, which makes them ideal for use in flexible models with complex graphical features. This also applies incidentally, to programmed solutions. Much more is involved in creating these controls than is the case with form controls, and you need specific, detailed knowledge. All of the knowledge you require is provided below, as well as in later chapters.

Basics of ActiveX Controls

Note When using the rS1.Method, you should first create a control in the *Basis 1* worksheet, test it there for both its functionality and its effects on the formula results in this worksheet, and then (and only then) copy it to its final destination in the *Focus 1* worksheet.

ActiveX controls are standard tools that can be used in various Microsoft applications. They were developed for use in programmed solutions and are still used predominantly for that purpose. It's rather unusual for them to be used to control solutions that are dynamic but not programmed, as described in this book. There's one consolation here: You don't need to be alarmed by the list of properties for this element, which is impressively long and hard to understand at first, because you're only going to need to use a tiny fraction of all the options available. All the rest are only relevant for programming.

Note the following essential points:

- *ActiveX controls* are not an integral component of Excel, and their behavior is similar but not identical to Excel or Office objects. Excel beginners need a little time to adjust to this fact, in particular when designing an rS1-model.
- *ActiveX controls* can be used in two different modes, but they don't function in *design* mode and can't be designed in functional mode. As a result, you need to constantly switch back and forth between the two modes. Or, to be more precise, you enable and disable *design mode* as required.
- *ActiveX controls* have complex, definable *properties*, which are only listed as programming terms.
- *ActiveX controls* have different types of *properties*:
  - Functional properties, which regulate the content and behavior of the object (what does the control show, and how does it respond?).
  - Formatting properties, which determine the appearance of the object and its contents (what does the control show, and how?).

- *ActiveX controls* have *properties* that can be changed in various ways in design mode:
    - They can be changed indirectly when you manipulate a control or perform an action in relation to it, or when you use the control to manipulate or perform an action in relation to another object. Example: You use the mouse to change the width and height of the object. This determines the *Width* and *Height* properties. You can also explicitly define (or change) these properties in a dialog box.
    - They can be changed directly when you explicitly define them in a dialog box. Example: You enter the range name of a cell in the *LinkedCell* text box.
    - All controls (both *form controls* and *ActiveX controls*) are graphical objects. As such, they have the same properties shared by all graphical objects in Microsoft Office.

Recommended Method

To create a dynamic chart, you usually require several ActiveX controls, and, you often need to use more than one of the same type. For this reason, I recommend that you use the following sequence when creating these controls.

1. Plan and prepare your dynamic model.
2. Define and map out your workbook and worksheet structures (values, names, formulas, ranges, cells).
3. Create the first control belonging to a specific type.
4. Define the functional and formatting properties of this control.
5. Define the general properties of this control as a graphical object.
6. Test the functionality of the control.
7. Copy the control.
8. Modify the properties in the copies of the control.
9. Adjust and synchronize the general, graphical object properties of all controls.
10. Test the functionality of all controls in isolation and in combination with each other.

Creating ActiveX Controls

Before you create an *ActiveX control*, I want to make you aware of one hard-to-avoid issue with these objects, which you may encounter on your first attempt: the possibility of generating VBA program source code by mistake. This can be very annoying, but, as you'll soon see, doesn't pose a serious problem.

There are two reasons why it's almost impossible to avoid this mistake. At some point, when you're working in a standard Excel 2007 workbook (*xlsx* file type), with *design mode* enabled (more about that later) and a control selected, you're bound to do one of the following:

- Accidentally click the *View Code* button in the *Controls* group of the *Developer* tab
- Accidentally double-click a control

Excel interprets both these actions as instructions to create VBA procedures for the active controls, and you suddenly may find yourself in a completely unfamiliar window; i.e., the *Visual Basic* editor. The editor warmly welcomes you with a freshly generated procedure, and then waits patiently for you to write all manner of program code between *Private Sub* and *End Sub*, which would then be processed later when the user clicks the control. But that's not what you want to do. So, how do you escape?

In Excel 2007 (unlike earlier versions), you can simply close the *Visual Basic* editor and pretend this never happened. The next time you save your work, the program will issue the following message:

"The following features cannot be saved in macro-free workbooks

- *VB project*

  … (additional information)

  To continue saving as a macro-free workbook, click Yes"

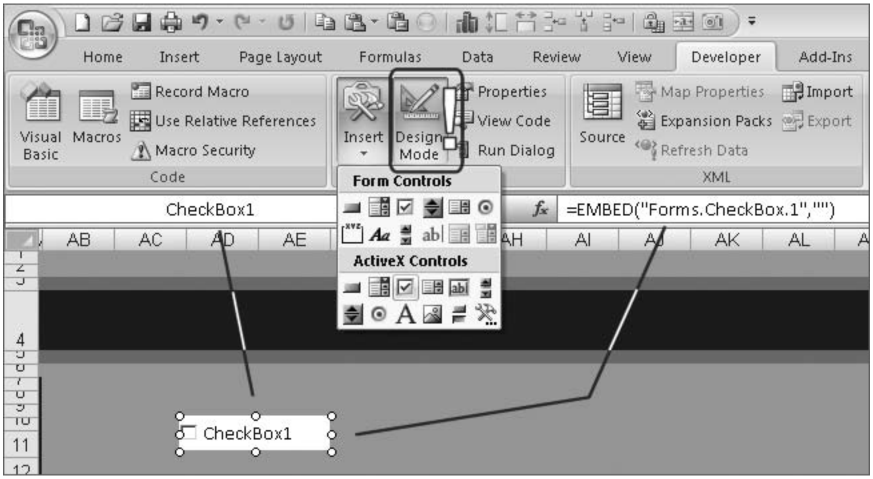
This is exactly what you should do in this case. The VBA procedure you generated by mistake is then deleted from the workbook.

Note If you want to save a workbook together with a VB project in Excel 2007, you must select the *macro-enabled file type* (*xlsm*) after the *Save As* command.

If you want to open the *Visual Basic* editor again, the fastest way to do this is to press **Alt**+**F11**.

It doesn't matter where you create the control in the workbook or what size it is because, as a graphical object, it can be moved and resized as required. However, you may prefer to take a systematic approach from the start.

1. Plan the type of control you want to create and where you want it to be placed. If possible (and it isn't always), define the cell sizes of the "host" table so that a control can fit into a single cell.
2. Select the *Developer* tab in the Ribbon, and click the *Insert* button in the *Controls* group. Then click the icon of your choice in the bottom selection menu. Position your mouse in the worksheet (the mouse pointer becomes a crosshair), and—holding down the mouse button—drag the object horizontally until you're happy with its size.



CheckBox1

=EMBED("Forms.CheckBox.1","")

FIGURE 7-27 *Design mode* is automatically enabled when you create *ActiveX controls.*

3. *Design mode* is automatically enabled. A formula (such as =`EMBED("Forms.CheckBox.1","")`) appears in the formula bar, but can be ignored.
4. The control is selected as a graphics object. This means that you can scale it, use the **Alt** to adjust it to the fit into the cell(s), and use the arrow keys to move it around the screen.
5. Click the *Design mode* button in the Ribbon to exit this mode, or proceed immediately to define the *properties* of the control (see the next section).

Tip It's important to bear in mind the following point from the start: Unlike *form controls*, *ActiveX controls* usually require a little more space (due to their height) than is normally available in a standard Excel table row. A row height of 18 is therefore a reasonable size to accommodate a control. Here, again, it will be easier to reduce the size later, if necessary, than to enlarge it.

The next tip applies to all graphical objects. In this case, the prerequisite is that the cell that is to contain the object must be exactly the same size as the object to be generated:

1. Select the object in the Ribbon and point with the mouse (mouse pointer is a crosshair) into the relevant cell.
2. Press and hold down the **Alt** key.
3. Next, hold down the mouse button and move the mouse pointer very slightly in any direction, but do not move it outside of the relevant cell.
4. Release the mouse button, and then release the **Alt** key.

The result is that the object virtually "drops" into the cell and adjusts to fit within the cell borders. In other words, it assumes the same size as its host.

Enabling/Disabling Design Mode

This isn't just important. It's *very* important, and it also takes some getting used to. While *design mode* is enabled, all you can do with your *ActiveX control* is edit it. This represents a clear difference between these objects and *form controls*, for which a mode of this kind doesn't exist.

I assume that you've already generated the object and configured the object. *Design mode* is enabled, and the object is selected. If you now click in any other location in the worksheet or press the **Esc** key, the object is deselected. In other words, the control is no longer active. However, it (and any other *ActiveX controls* you're using) is still in *design mode*. You can continue working on other parts of the worksheet, but if you click on the control, it doesn't function as a control. Instead, it can be edited and is therefore selected again as an object. The only way to access functional mode for the control is to disable the *Design Mode* option in the Ribbon.

However, if you want to edit the control again, you must switch on *design mode* again in order to select the object.

Note the following point in relation to Excel 2007 in particular: *Design mode* frequently is activated without the user knowing it because a tab other than the *Developer* tab is currently active in the Ribbon.

Note To help you check which mode you're in, remember that if you click a control with either mouse button and it appears within an object selection frame, then you know that:

- ❑ It's an *ActiveX control* (because this isn't possible with *form controls*), and
- ❑ Design mode is enabled

This note is particularly important because both types of control may be used in the same workbook in the dynamic models described in this book. Depending on their design and your own experience, you may not be able to distinguish between the two at first glance.

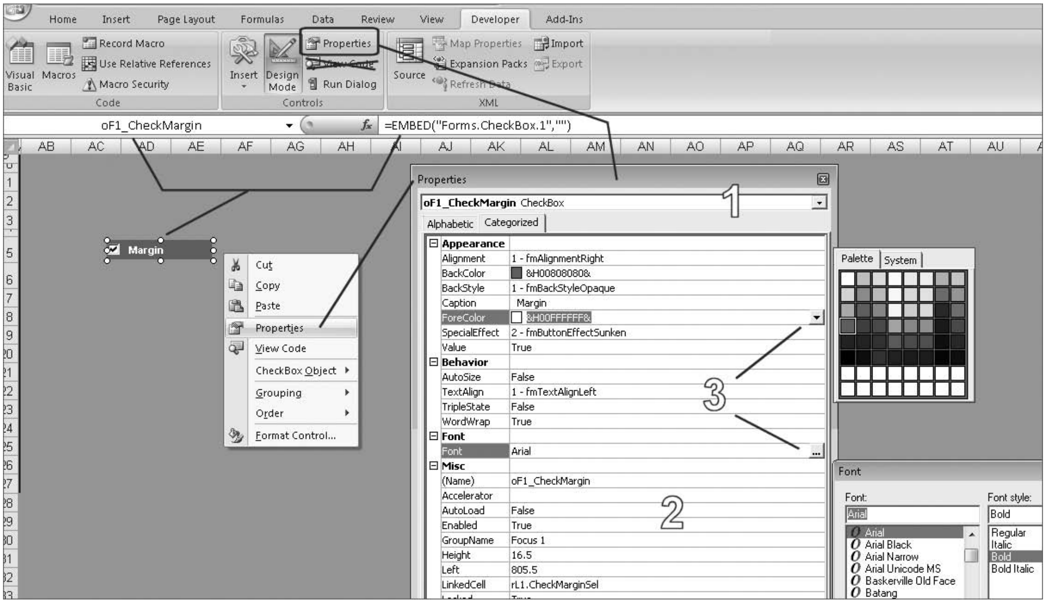
Modifying ActiveX Controls

Modifying or editing *ActiveX controls* involves two procedures, both of which require you to enable *design mode*:

- ■ Define the functional and formatting properties of the control
- ■ Define the general properties of the control as a graphics object (I've already explained how to define these in relation to Figure 7-12. The following discussion is therefore limited to the definition of functional and formatting properties).

You've defined an ActiveX control and want to configure its functions or define its formatting. The object is selected and *design mode* is enabled. Follow these steps with reference to

Figure 7-28: select the *Properties* command in the Ribbon, or right-click the object, and select the *Properties* command in the context menu. A dialog box opens, showing more or less extensive lists of properties.



oF1_CheckMargin =EMBED("Forms.CheckBox.1","")

Properties

oF1_CheckMargin CheckBox

Alphabetic | Categorized

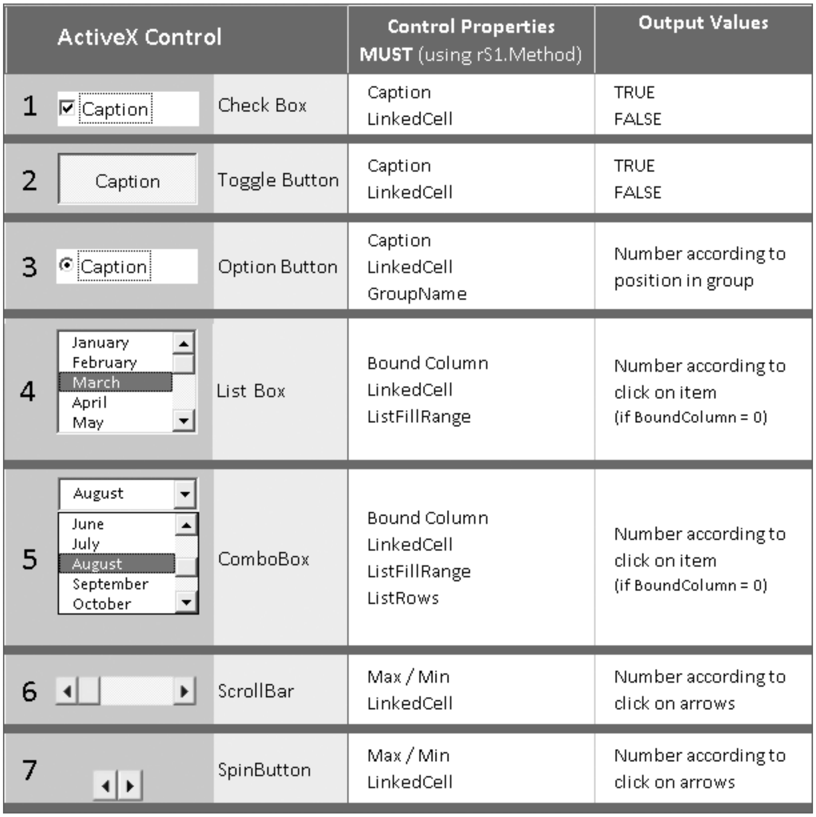
Appearance	
Alignment	1 - fmAlignmentRight
BackColor	&H00808080&
BackStyle	1 - fmBackStyleOpaque
Caption	Margin
ForeColor	&H00FFFFFF&
SpecialEffect	2 - fmButtonEffectSunken
Value	True
Behavior	
AutoSize	False
TextAlign	1 - fmTextAlignLeft
TripleState	False
WordWrap	True
Font	
Font	Arial
Misc	
(Name)	oF1_CheckMargin
Accelerator	
AutoLoad	False
Enabled	True
GroupName	Focus 1
Height	16.5
Left	805.5
LinkedCell	rL1.CheckMarginSel

FIGURE 7-28 Dialog box for defining functional and formatting properties

1. All dialog boxes containing lists of *properties* of *ActiveX controls* have the same structure, but the options available depend on the type of control. The dialog box has two tabs with the same content, one arranged in *Alphabetic* order and the other organized into categories, or *Categorized*. For Excel beginners, I strongly recommend using the *Categorized* tab. It is much easier to understand because it divides the very extensive lists of properties into well-defined groups.
2. You can click the textual description of the property, click in the cell to the right of this description, and then select or enter the relevant value, or press **Ctrl**+**V** to paste the value from the clipboard.
3. When you click some of the rows, specific buttons appear that you can use to select predefined values. These values may appear as short lists (for example, in the case of *BackColor*) or small dialog boxes, or a standard Office dialog box may open (as with *Font*).

Properties When Using the rS1.Method

As I've promised, you only need to define a few control properties to lend dynamism to rS1 models. It's important to distinguish between properties that absolutely *must* be defined, those that *should* be defined, and those additional properties that *can* also be defined.



ActiveX Control			Control Properties **MUST** (using rS1.Method)	Output Values
1	☑ Caption	Check Box	Caption LinkedCell	TRUE FALSE
2	Caption	Toggle Button	Caption LinkedCell	TRUE FALSE
3	◉ Caption	Option Button	Caption LinkedCell GroupName	Number according to position in group
4	January February March April May	List Box	Bound Column LinkedCell ListFillRange	Number according to click on item (if BoundColumn = 0)
5	August June July August September October	ComboBox	Bound Column LinkedCell ListFillRange ListRows	Number according to click on item (if BoundColumn = 0)
6		ScrollBar	Max / Min LinkedCell	Number according to click on arrows
7		SpinButton	Max / Min LinkedCell	Number according to click on arrows

FIGURE 7-29 The mandatory ("MUST") properties of selected *ActiveX controls*

An overview of this topic is provided in Figure 7-29. I'll describe the most important aspects when using these properties in rS1 models, with reference to the *Control Properties* column in the figure that follows. I'll introduce you to other properties later with practical examples.

If you like, you can select the *Focus 1* worksheet in the *0702_Indicators_02.xlsx* file, unprotect the sheet, enable design mode, and then (as shown in Figure 7-28), access the dialog boxes of individual controls to compared their defined properties with the descriptions below. A more precise and specific explanation is provided in the next section.

- *Caption*: A control needs a label if you want to tell users what the control does or doesn't do when they click it. This applies to *Check Box*, *Option Button*, and *Toggle Button*. This isn't a *must* if the necessary information is conveyed in another way.

- *LinkedCell*: All controls require a cell link. The current output value of the element is directed to the linked cell. These are cells whose range names have the suffix *Sel*, and that are found in the *Lists 1* worksheet.
- *ListFillRange*: The table range from which the list is filled is naturally reserved for the *ListBox* and *ComboBox* controls.
- *ListRows*: Even though this property is not essential for functioning alone (and therefore isn't, strictly speaking, a *must*), you always define the number of list entries that are to appear in the drop-down list for a *ComboBox* control.
- *BoundColumn*: You'll discover later (by means of a practical example) why the "bound column" should usually be set to 0 (zero) for *ListBox* and *ComboBox*, even though the default value is *1*.
- *GroupName*: Several *OptionButtons* must share the same group name so that it is clear which of these are to be set to FALSE when one is set to TRUE.
- *Max* and *Min*: These are the maximum and minimum threshold values for the *SpinButton* and *ScrollBar* controls.

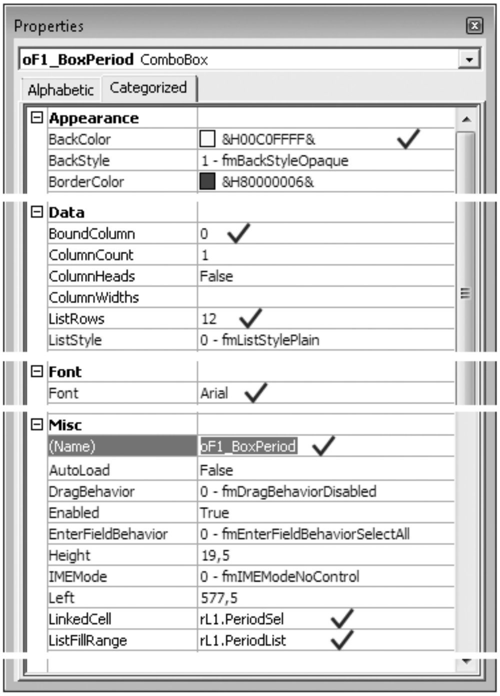
In addition to these essential properties, there are also properties which can be described as *Should* properties (i.e., properties that should ideally be defined), including the *(Name)* property, which is the object name of the element. Finally, many properties can be classed as *Can* (i.e., optional) properties. Most of the properties that fall into this last category are used to format the object and are discussed in some of the examples in later chapters.

Configuring ActiveX Controls

Let's move on now to the specific properties of the controls in the *Focus 1* worksheet of the *0702_Indicators_02.xlsx* sample file. The functions of these objects are identical to those described in relation to the form controls of variant 1. You may find it helpful to cast your eye over these again now (Figure 7-7 to Figure 7-12). When using *ActiveX controls*, you need to employ a completely different method to achieve the same end result.

Controls for Selecting Indicator Category and Period As with the form controls, a drop-down list is used again here for the selection of the indicator type, in this case the *ComboBox* ActiveX control. The *period* shown in the chart is similarly selected using a *ComboBox*, which is linked with a *ScrollBar*.

Let's start with the *oF1_BoxPeriod ComboBox*. This control is identical to the *ComboBox* used to select the indicator category, the only difference being the use of range names.



oF1_BoxPeriod ComboBox

⊟ **Appearance**		
BackColor	□ &H00C0FFFF&	✓
BackStyle	1 - fmBackStyleOpaque	
BorderColor	■ &H80000006&	
⊟ **Data**		
BoundColumn	0	✓
ColumnCount	1	
ColumnHeads	False	
ColumnWidths		
ListRows	12	✓
ListStyle	0 - fmListStylePlain	
⊟ **Font**		
Font	Arial	✓
⊟ **Misc**		
(Name)	oF1_BoxPeriod	✓
AutoLoad	False	
DragBehavior	0 - fmDragBehaviorDisabled	
Enabled	True	
EnterFieldBehavior	0 - fmEnterFieldBehaviorSelectAll	
Height	19,5	
IMEMode	0 - fmIMEModeNoControl	
Left	577,5	
LinkedCell	rL1.PeriodSel	✓
ListFillRange	rL1.PeriodList	✓

FIGURE 7-30 Only the rows with tickmarks are relevant.

The *Properties* dialog box of the *ComboBox* shown in Figure 7-30 has been truncated several times because it contained many entries that have no relevance here (in other words, only segments of the original list are shown). In this abbreviated list, we're only interested in the properties with tickmarks assigned.

The most important information (working from the top of the list down) is provided in the following table.

Property	Category	Importance	Value/Remark
BackColor	*Appearance*	Can	You can select the background color in a small dialog box.
BoundColumn	*Data*	Must	You must define the value *0 (zero)* for this property so that the object outputs a number (the index) that corresponds to the click position. Otherwise (if you use the default value *1*), the actual list contents of the click position (in this case, a text) would be output. This would be a good choice if you were using the VLOOKUP function for further processing, and wanted to use the output as a search criterion.

Property	Category	Importance	Value/Remark
ListRows	*Data*	Must/Should	This is the number of rows that users will see when they open the drop-down list. If the number is small (fewer than 20), all should be visible at once.
Font	*Font*	Can	You can define the font attributes in a dialog box. In these controls, the *Arial* font is clearer than most of the fonts delivered with Office 2007.
(Name)	*Misc*	Should	*oF1_BoxPeriod*—Enter the object name here (optional). Alternatively, enter it in the name box in the formula bar. It is then applied here.
LinkedCell	*Misc*	Must	*rL1.PeriodSel*—This cell contains the value that is output when a user clicks on a list entry.
ListFillRange	*Misc*	Must	*rL1.PeriodList*—The contents of the selection list are defined in this cell range.

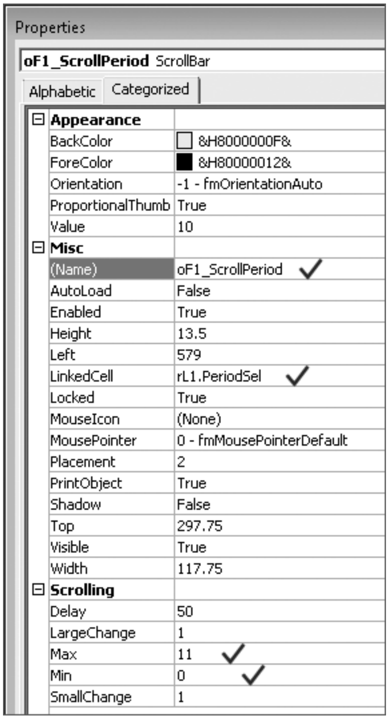
There's not a lot left to say in relation to the *oF1_BoxIndicators01 ComboBox*. It is defined in almost an identical manner to the control described above. The only differences are:

- The *(Name)* property is, naturally different, i.e.: *oF1_BoxIndicators01*.
- *rL1. Indicators01Sel* was selected as the *LinkedCell*.
- The *rL1. Indicators01List* range is used as the *ListFillRange* .

> **Note** Here, once again, we can see just how cool the rS1.Method can be, and how much work it can save you. After you create, configure, and make a copy of the first control, all you need to do is change a couple of characters in the copy (which you can almost do "on autopilot," seeing as you're using conventions) to make it do what it needs to do.

The *Properties* dialog box of the *ScrollBar* that works together with *oF1_BoxPeriod* is shown in full in Figure 7-31. However, you only need concern yourself with those few properties that are highlighted here.

Before we start, note the following important point: The numbering of the index in a *ComboBox* or *ListBox* starts at *0* (zero), rather than at *1*, as is the case with form controls. This means that, when you click the first entry in a list, a *0* rather than a *1* appears in the *LinkedCell* and, when you click the tenth entry, a *9* is output in this cell, and so on. However, you can use certain constructs to intercept this problem. These are described later. If you link one of these controls with another, this second control must also have *zero* as its minimum value.



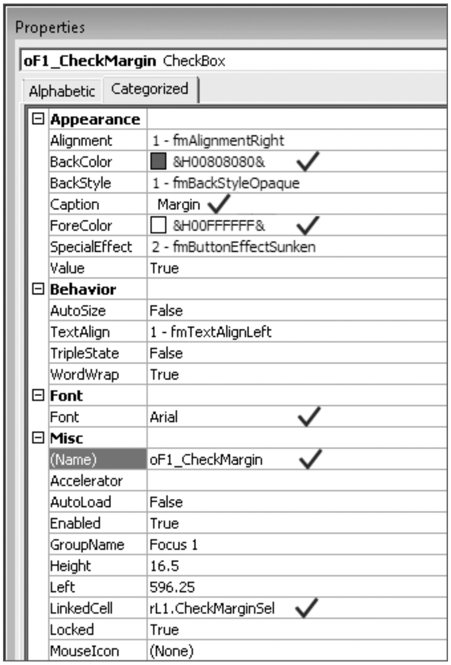
oF1_ScrollPeriod ScrollBar

⊟ **Appearance**	
BackColor	□ &H8000000F&
ForeColor	■ &H80000012&
Orientation	-1 - fmOrientationAuto
ProportionalThumb	True
Value	10
⊟ **Misc**	
(Name)	oF1_ScrollPeriod ✓
AutoLoad	False
Enabled	True
Height	13.5
Left	579
LinkedCell	rL1.PeriodSel ✓
Locked	True
MouseIcon	(None)
MousePointer	0 - fmMousePointerDefault
Placement	2
PrintObject	True
Shadow	False
Top	297.75
Visible	True
Width	117.75
⊟ **Scrolling**	
Delay	50
LargeChange	1
Max	11 ✓
Min	0 ✓
SmallChange	1

FIGURE 7-31 The complete *Properties* dialog box of the *oF1_ScrollPeriod* control

<table>
<tr><th>Property</th><th>Category</th><th>Importance</th><th>Value/Remark</th></tr>
<tr><td><i>Max</i></td><td><i>Scrolling</i></td><td>Must</td><td>The object is to output 12 values. However, the highest value must be <i>11</i> because the minimum value must be <i>0</i> (see the note above).</td></tr>
<tr><td><i>Min</i></td><td><i>Scrolling</i></td><td>Must</td><td>Here, the minimum value must be set to <i>0</i> instead of <i>1</i>.</td></tr>
<tr><td colspan="4">As mentioned earlier, you can set the <i>Max</i> property of this control (or of a <i>SpinButton</i>) to a lower value than the <i>Min</i> property. For example, you could set the maximum to <i>0</i> and the minimum to <i>10</i>. The effect of this would be to reverse the object’s “working direction.” In this case, the scroll box would be at the extreme right of the horizontal <i>ScrollBar</i> rather than the extreme left if the value was zero (and clicking on the left arrow would be equivalent to counting backwards). On a vertical <i>ScrollBar</i>, the scroll box would be at the very bottom, rather than the very top, if the value were zero. This is an extremely useful feature for many constructs where the “logic” of direction comes into play.</td></tr>
<tr><td><i>LinkedCell</i></td><td><i>Misc</i></td><td>Must</td><td><i>rL1.PeriodSel</i>, in order to functionally link the object with the <i>oF1_BoxPeriod</i> control.</td></tr>
</table>

Controls for Displaying Additional Data Series You still need to know how to configure the two *CheckBoxes* to display the lines representing the margins of the defined range and the line representing the industry value. Once again, the two controls are largely identical. As a result, we only need to look at one of these in detail, i.e. the *oF1_CheckMargin* object. The complete dialog box for the properties of this control is shown in Figure 7-32.



oF1_CheckMargin CheckBox

⊟ **Appearance**	
Alignment	1 - fmAlignmentRight
BackColor	■ &H00808080& ✓
BackStyle	1 - fmBackStyleOpaque
Caption	Margin ✓
ForeColor	□ &H00FFFFFF& ✓
SpecialEffect	2 - fmButtonEffectSunken
Value	True
⊟ **Behavior**	
AutoSize	False
TextAlign	1 - fmTextAlignLeft
TripleState	False
WordWrap	True
⊟ **Font**	
Font	Arial ✓
⊟ **Misc**	
(Name)	oF1_CheckMargin ✓
Accelerator	
AutoLoad	False
Enabled	True
GroupName	Focus 1
Height	16.5
Left	596.25
LinkedCell	rL1.CheckMarginSel ✓
Locked	True
MouseIcon	(None)

FIGURE 7-32 Properties of the *oF1_CheckMargin CheckBox*

Property	Category	Importance	Value/Remark
BackColor	*Appearance*	Can	The fill color, which is selected from a pallet.
Caption	*Appearance*	Must	The *Margin* text was defined here with a leading blank to create some space between it and the check box.
ForeColor	*Appearance*	Can	The foreground color, which refers to the *font color* in this element. Selection is the same as for *BackColor*.
Font	*Font*	Can	Once again, *Arial* was selected here as the clearest font.
(Name)	*Misc*	Should	*oF1_CheckMargin*
LinkedCell	*Misc*	Must	*rL1.CheckMarginSel*

The only differences in the second *CheckBox* are listed below:

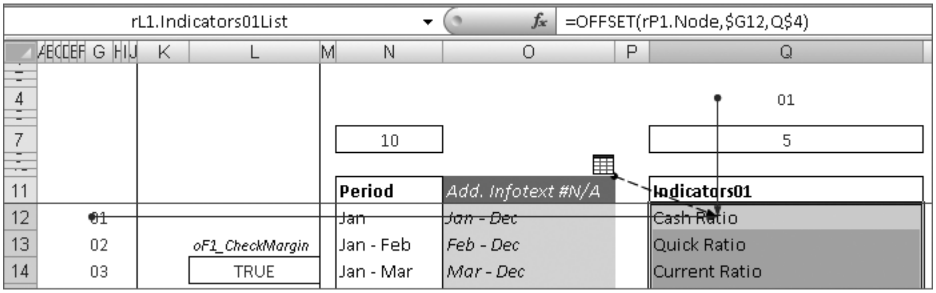
- The (Name) property is as follows: *oF1_CheckIndustry*.
- *Ø Industry* is defined as the *Caption*.
- *rL1.CheckIndustrySel* is defined as the *LinkedCell*.

Formulas

In relation to the use of formulas, there are two differences between this workbook and the basic version in variant 1, namely:

- A couple of formulas have been added.
- The extracting formulas have been adjusted to suit the conditions of the *ActiveX controls*.

Formulas in the *Lists 1* Worksheet



rL1.Indicators01List =OFFSET(rP1.Node,$G12,Q$4)

	G	L	N	O	Q
4					01
7			10		5
11			**Period**	*Add. Infotext #N/A*	**Indicators01**
12	01		Jan	*Jan - Dec*	Cash Ratio
13	02	*oF1_CheckMargin*	Jan - Feb	*Feb - Dec*	Quick Ratio
14	03	TRUE	Jan - Mar	*Mar - Dec*	Current Ratio

FIGURE 7-33 Reading data from the *Parameters 1* worksheet

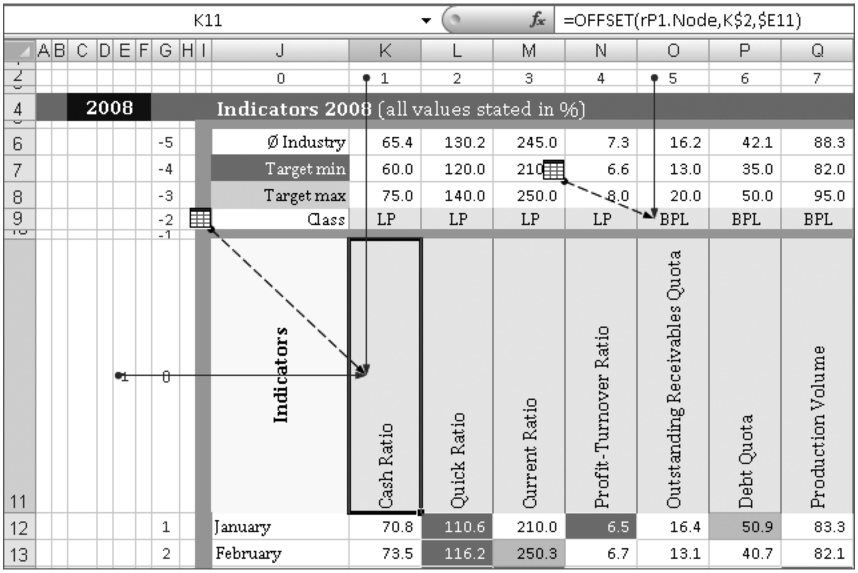
Formulas of the now-familiar OFFSET type are used to read the names of the indicators.

```
Q12:Q23    =OFFSET(rP1.Node,$G12,Q$4)
```

Formulas in the *Data 1* Worksheet

In rows 9 and 11, master data is similarly extracted from the *Parameters 1* sheet using OFFSET. There's nothing to point out here that you don't already know.

```
K9:V9      =OFFSET(rP1.Node,K$2,3)

K11:V11    =OFFSET(rP1.Node,K$2,$E11)
```



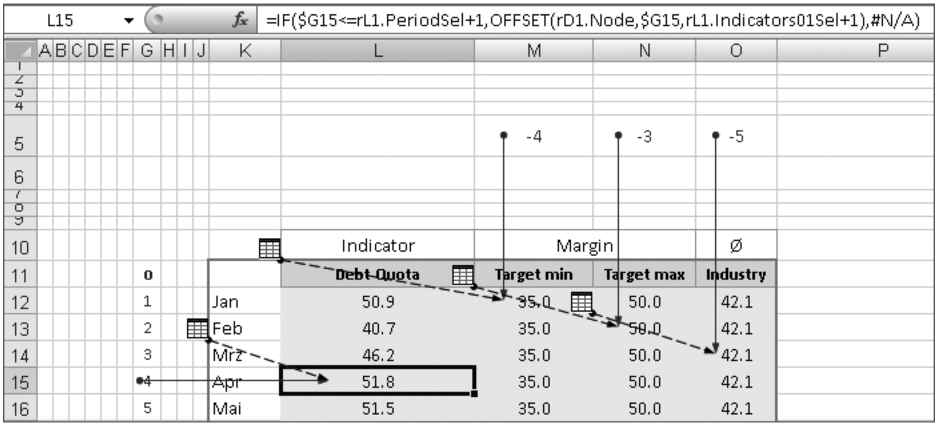
K11 =OFFSET(rP1.Node,K$2,$E11)

	C	E	G	J	K	L	M	N	O	P	Q
2				0	1	2	3	4	5	6	7
4	2008			Indicators 2008 (all values stated in %)							
6			-5	Ø Industry	65.4	130.2	245.0	7.3	16.2	42.1	88.3
7			-4	Target min	60.0	120.0	210	6.6	13.0	35.0	82.0
8			-3	Target max	75.0	140.0	250.0	8.0	20.0	50.0	95.0
9			-2	Class	LP	LP	LP	LP	BPL	BPL	BPL
10			-1								
11		1	0	Indicators	Cash Ratio	Quick Ratio	Current Ratio	Profit-Turnover Ratio	Outstanding Receivables Quota	Debt Quota	Production Volume
12			1	January	70.8	110.6	210.0	6.5	16.4	50.9	83.3
13			2	February	73.5	116.2	250.3	6.7	13.1	40.7	82.1

FIGURE 7-34 Master data from *Parameters 1* also appears in this worksheet.

Formulas in the *Basis 1* Worksheet

There are a few changes in relation to the table that represents the chart basis. It's rather marginal that the column headers in cells `M11` and `N11` are different. However, the formulas used for these are the same as those from variant 1 (see Figure 7-18).



L15 =IF($G15<=rL1.PeriodSel+1,OFFSET(rD1.Node,$G15,rL1.Indicators01Sel+1),#N/A)

	G	K	L	M	N	O
5				-4	-3	-5
10			Indicator	Margin		Ø
11	0		Debt Quota	Target min	Target max	Industry
12	1	Jan	50.9	35.0	50.0	42.1
13	2	Feb	40.7	35.0	50.0	42.1
14	3	Mrz	46.2	35.0	50.0	42.1
15	4	Apr	51.8	35.0	50.0	42.1
16	5	Mai	51.5	35.0	50.0	42.1

FIGURE 7-35 *Sel+1* is used in most of the formulas in this worksheet.

A more important change applies to all formulas that react to the output values of controls. The reason for this was explained earlier. The numbering of the index in a *ComboBox* or *ListBox* starts at *0* (zero), rather than at *1*. Clicking the first entry in one of these controls generates a *0* rather than a *1* in the corresponding *LinkedCell*. In all formulas that refer to these

LinkedCells, the value taken from that cell is increased by *1* (…`Sel+1`…). Take, for example, the formula in cell `L15`:

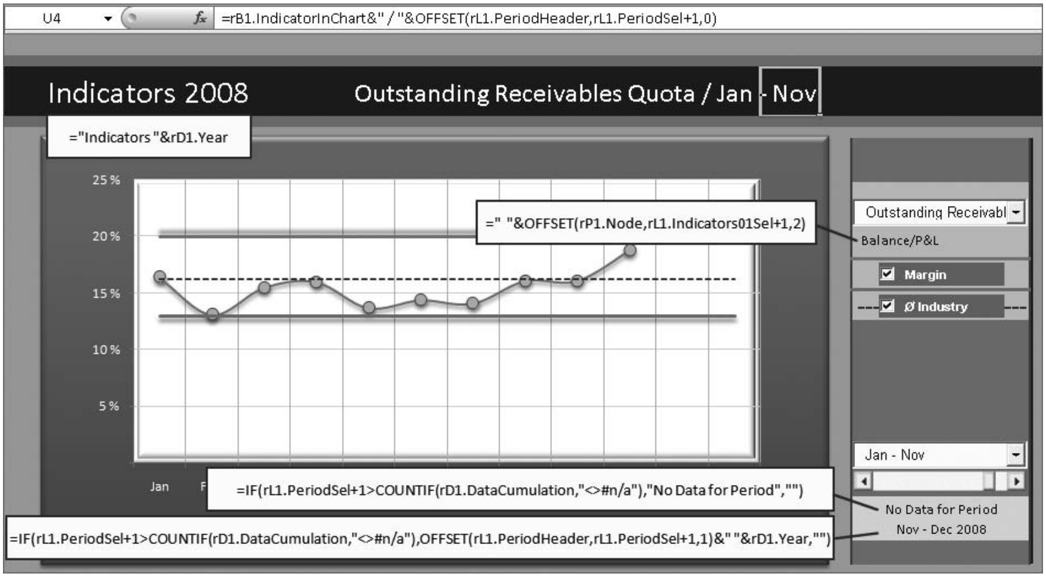
```
L15    =IF($G15<=rL1.PeriodSel+1,
       OFFSET(rD1.Node,$G15,rL1.Indicators01Sel+1),#N/A)
```

If you were to create this worksheet as a copy of the corresponding worksheet from variant 1 and adjust it as required, you could simply make a global change to all of the existing formulas using the *Replace* function: Find all instances of *Sel*, and replace these with *Sel+1*.

> **Important** This peculiarity (*0* rather than *1*) should make you particularly careful when testing the formulas. Does the formula return the value that you expect when you click the control? If you neglect to perform these tests, errors can soon creep in, and may not even be detected during a cursory check of the solution.
>
> One thing I always make sure to check is whether clicking the first list entry and last list entry returns the correct results.

Formulas in the *Focus 1* Worksheet

There are many new features here, most of which serves to make the worksheet more user friendly.



```
U4  =rB1.IndicatorInChart&" / "&OFFSET(rL1.PeriodHeader,rL1.PeriodSel+1,0)
```

```
="Indicators "&rD1.Year
=" "&OFFSET(rP1.Node,rL1.Indicators01Sel+1,2)
=IF(rL1.PeriodSel+1>COUNTIF(rD1.DataCumulation,"<>#n/a"),"No Data for Period","")
=IF(rL1.PeriodSel+1>COUNTIF(rD1.DataCumulation,"<>#n/a"),OFFSET(rL1.PeriodHeader,rL1.PeriodSel+1,1)&" "&rD1.Year,"")
```

FIGURE 7-36 Enhanced user information with dynamic formulas

These are as follows, working top to bottom, and left to right:

```
D4    ="Indicators "&rD1.Year
```

This instructs Excel to do the following: "Write the string *Indicators* with a closing blank, and then insert the content from the *rD1.Year* cell." The year is specified in this cell.

```
U4    =rB1.IndicatorInChart&" / "
      &OFFSET(rL1.PeriodHeader,rL1.PeriodSel+1,0)
```

In U4, a header is composed in the usual manner. The only other difference here is that *Sel* is replaced with *Sel+1*.

```
X13   ="  "&OFFSET(rP1.Node,rL1.Indicators01Sel+1,2)
```

The readout of the group ID for the selected indicator in cell X13 shouldn't be a mystery to you at this stage. Take note of the leading blanks generated by the formula.

The contents of cells X25 and X26 are a little more complicated. Here, two different formulas are used in two different cells to create a text that appears to the user as a single unit.

(Note also: The K12:K23 range in the *Data 1* worksheet has been assigned the name *rD1.DataCumulation*.)

```
X25   =IF(rL1.PeriodSel+1>COUNTIF(rD1.DataCumulation,"<>#n/a"),
      "No Data for Period","")
```

Translated into an instruction to Excel, this would read as follows: "If the value in the *rL1.PeriodSel* cell plus *1* is greater than the number of #N/A values in the *rD1.DataCumulation* range, write the string *No Data for Period*. Otherwise, write an empty text."

```
X26   =IF(rL1.PeriodSel+1>COUNTIF(rD1.DataCumulation,"<>#n/a"),
      OFFSET(rL1.PeriodHeader,rL1.PeriodSel+1,1)
      &" "&rD1.Year,"")
```

Translated into a an instruction to Excel, this would read as follows: "If the value in the *rL1.PeriodSel* cell plus *1* is greater than the number of #N/A values in the *rD1.DataCumulation* range, read the contents of a cell, which can be located as follows: Go to the *rL1.PeriodHeader* cell, and move down the number of rows specified in the *rL1.PeriodSel* cell plus *1*, and move across one column to the right (here we want to find entries in the "reverse list" explained above, see Figure 7-22). Insert a blank space, followed by the content from the *rD1.Year* cell. Otherwise, write an empty text."

Conditional Formatting

To conclude this chapter, we'll look at one more, very important element of dynamization. The appearance of a table should change whenever changes are made to its cell contents or to control values defined elsewhere. If you set up conditional formatting, Excel can generate a specific format in a cell depending on the content of that same cell (value-dependent) or of another cell (formula-dependent). You can either use predefined, integrated formats or customize your own.

The technique is very simple, and has a very similar effect to an `IF` formula:

- A condition defined by you is met (i.e., it is or becomes `TRUE`). Excel reacts by generating defined formatting for the relevant cells.
- A condition defined by you is not met (i.e., it is or becomes `FALSE`). Excel displays the basic formatting for the cells that have been assigned conditional formatting.

The conditional formatting feature is a little more complicated in Excel 2007. This feature has been significantly enhanced compared with earlier versions, with which you could only define three formatting variants for each cell (which was usually sufficient). Only time (and practice) will tell whether the variety of options now available will actually help improve the quality of information that can be conveyed. This most definitely won't be the case if the designer of a chart solution gets completely carried away and adopts the attitude *"If it can be done, why not do it?"* The result then will be psychedelic tables that confuse users rather than help them digest new information.

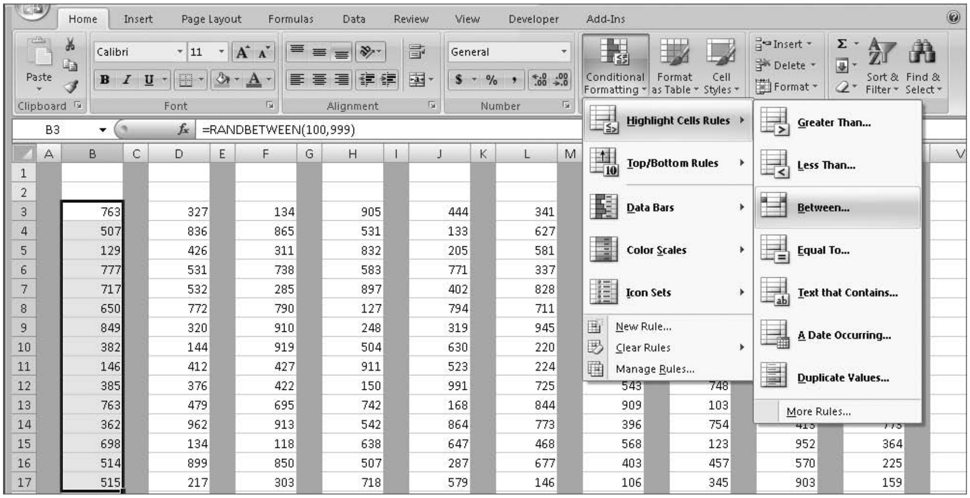
Important Human perception is severely limited when it comes to interpreting small or non-pictorial symbols. If you plan to generate more than three different signals (of any kind) in one cell range in order to convey certain information to the user, you should have a very good reason for doing so.

Basics

Some general information about this feature is required before I explain the conditional formatting used in the *Data 1* and *Focus 1* worksheets. However, it wouldn't make sense to enter into too much detail in the context of this book, where conditional formats are relevant only as an enhancement of dynamic charts.

Tip If you'd like to delve deeper into this multifaceted topic, and put the theory into practice by doing some exercises on your own, I recommend that you begin your experiments by creating a file similar to that shown in Figure 7-37:

Fill several column areas (separated by color columns) with formulas that generate variable number series within a defined range (for example, with `=RANDBETWEEN(100,999)`). Define different conditional formatting in each column. When you press **F9**, new numbers appear, and the formatting you defined adjusts to suit the new values. Edit and change the conditional formatting based on the information provided in the following paragraphs.



B3 =RANDBETWEEN(100,999)

	A	B	C	D	E	F	G	H	I	J	K	L	M									V
1																						
2																						
3		763		327		134		905		444		341										
4		507		836		865		531		133		627										
5		129		426		311		832		205		581										
6		777		531		738		583		771		337										
7		717		532		285		897		402		828										
8		650		772		790		127		794		711										
9		849		320		910		248		319		945										
10		382		144		919		504		630		220										
11		146		412		427		911		523		224										
12		385		376		422		150		991		725		543		748						
13		763		479		695		742		168		844		909		103						
14		362		962		913		542		864		773		396		754		413		775		
15		698		134		118		638		647		468		568		123		952		364		
16		514		899		850		507		287		677		403		457		570		225		
17		515		217		303		718		579		146		106		345		903		159		

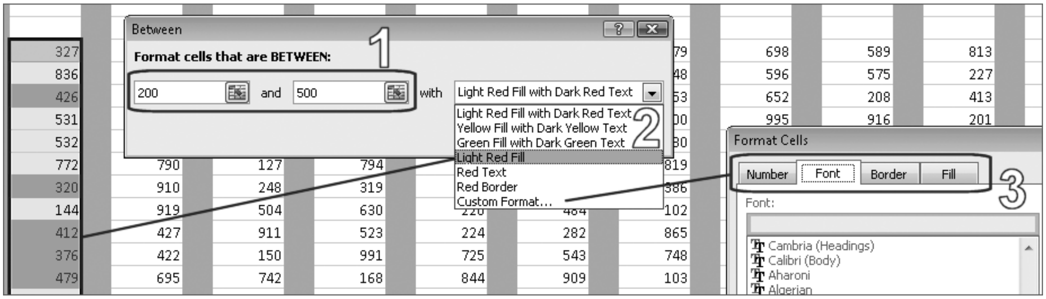
FIGURE 7-37 Accessing several conditional formatting variants

Let's examine a scenario based on Figure 7-37. Here you visually highlight cells that contain values within a certain range.

Select the *Home* tab in the Ribbon, and click the *Conditional Formatting* button in the *Styles* group. In the list that opens, select *Highlight Cells Rules*, and then select the type of rule you want to use (in this case, *Between*).

Then proceed as shown in Figure 7-38:

1. You define the limits of the value range (*BETWEEN* … *and*) in a small dialog box.
2. Select one of the predefined styles listed, or click the *Custom Format* entry in the drop-down list.
3. If you select *Custom Format*, the *Format Cells* dialog box opens, with the tabs *Number, Font, Border,* and *Fill*.



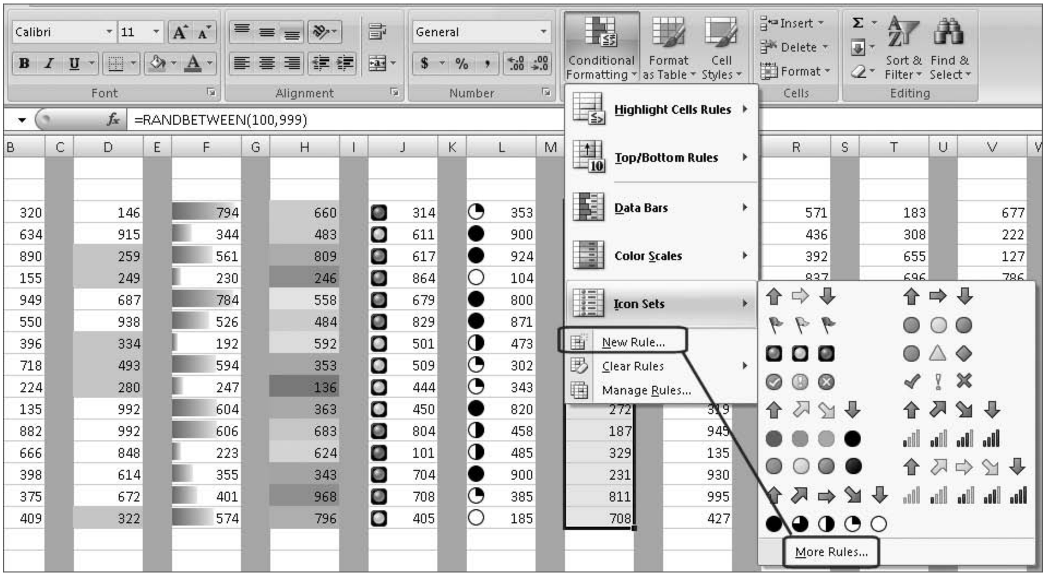
Between

Format cells that are BETWEEN:

200 and 500 with Light Red Fill with Dark Red Text

FIGURE 7-38 Simple options or complex design possibilities

Tip I must emphasize here that Excel 2007 now allows you to use number formats, including custom number-formats, in conditional formatting. This opens up a whole new range of possibilities, whereby number formats change automatically depending on specific values. And these values may be located in cells linked with controls. In other words, you could now simply click to control a number format.

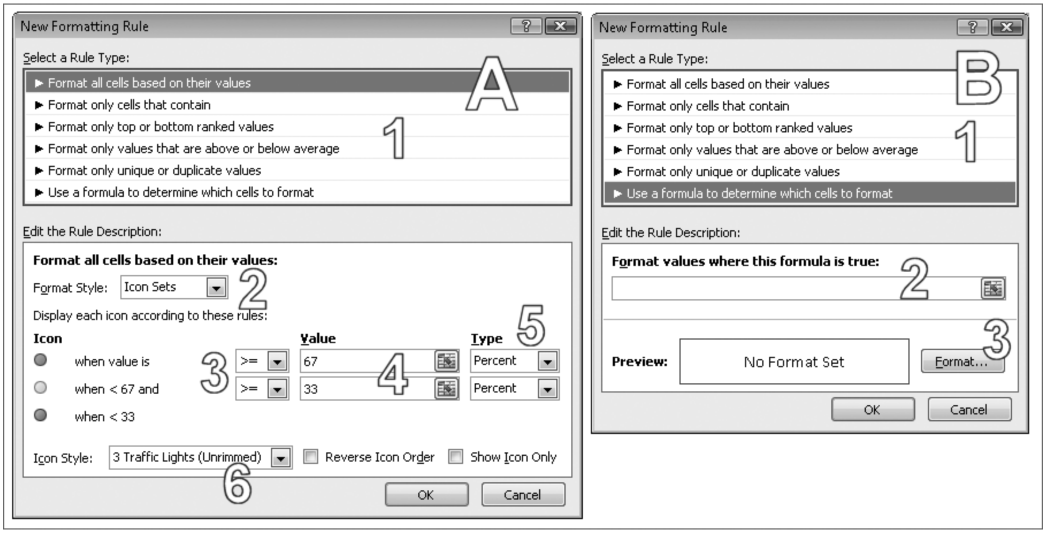


=RANDBETWEEN(100,999)

FIGURE 7-39 Default and custom options

Once you've found out all that you can do (and there's a lot!) with the predefined options available, you may want to explore the additional design possibilities, which I believe are more interesting. If you click *New Rule* or *More Rules* (framed in blue in Figure 7-39), the *New Formatting Rule* dialog box opens, which contains the options shown in Figure 7-40:

Note Note that the appearance of this dialog box changes considerably depending on the rule type selected (in step 1), and may differ from that shown here. The descriptions provided here serve as examples only, but they can be adapted to other variants.


New Formatting Rule
Select a Rule Type:
► Format all cells based on their values
► Format only cells that contain
► Format only top or bottom ranked values
► Format only values that are above or below average
► Format only unique or duplicate values
► Use a formula to determine which cells to format
Edit the Rule Description:
A
1
2
3
4
5
6
OK
Cancel
B
1
2
3
Format...
OK
Cancel

Format all cells based on their values:
Format Style: Icon Sets
Display each icon according to these rules:

Icon			Value	Type
●	when value is	>=	67	Percent
○	when < 67 and	>=	33	Percent
●	when < 33			

Icon Style: 3 Traffic Lights (Unrimmed) □ Reverse Icon Order □ Show Icon Only

Format values where this formula is true:
Preview: No Format Set

FIGURE 7-40 Dialog box for designing custom options

Example A:

1. Select a *Rule Type* from the list of descriptions.
2. Determine a formatting group to determine the *Format Style*.
3. Select logical operators.
4. Determine the value to be used in the *Value* box. You can define a value, a cell reference, or a formula here.
5. Under *Type*, select a data type for the *value* (which was selected in step 4).
6. Finally, choose a certain icon type within the *Format Style* group you selected (in step 2).

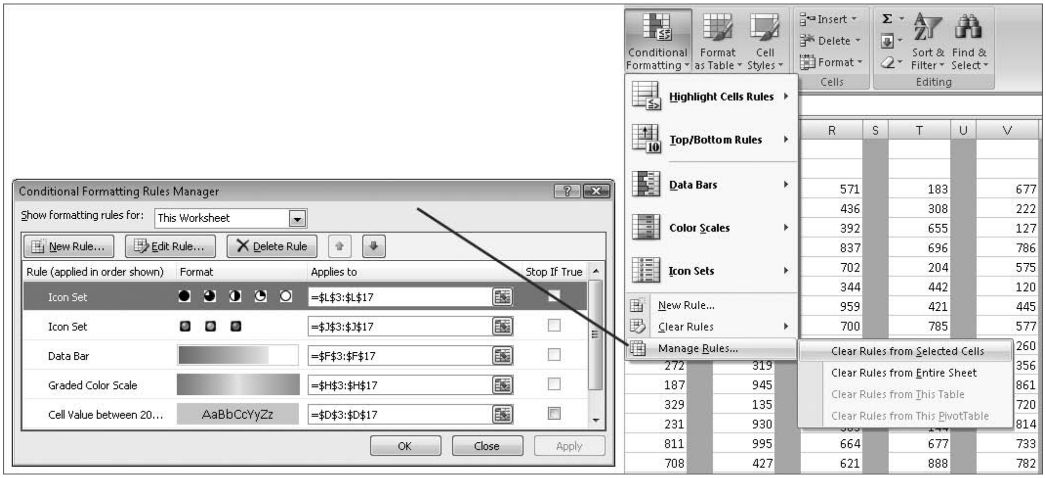
The method designated here as **Example B**, which involves using formulas to generate conditional formatting, is used much more frequently than the methods in *Example A* in rS1 models. Follow these steps:

1. Select the *Use a formula* … rule type.
2. Enter a formula. This formula, like the `logical_test` argument in an `IF` formula, makes a certain claim, for example, =$K$5<=100 (the value in cell K5 is 100 or lower) or `=rL1.CheckLightsSel=TRUE` (the *rL1.CheckLightsSel* cell currently contains the value

TRUE). Each time this formula is calculated, Excel investigates whether this claim is TRUE. If it is, the conditional formatting generated in step 3 is assigned.

3. Click the *Format* button to define a format.

You manage conditional formatting in the *Conditional Formatting Rules Manager* dialog box. To open this dialog box, select the *Home* tab in the Ribbon, select the *Styles* group, and click the *Conditional Formatting* button. Finally, select the *Manage Rules* option.



Conditional Formatting Rules Manager

Show formatting rules for: This Worksheet

Rule (applied in order shown)	Format	Applies to	Stop If True
Icon Set	● ◕ ◑ ◔ ○	=$L$3:$L$17	☐
Icon Set	■ ■ ■	=$J$3:$J$17	☐
Data Bar		=$F$3:$F$17	☐
Graded Color Scale		=$H$3:$H$17	☐
Cell Value between 20...	AaBbCcYyZz	=$D$3:$D$17	☐

FIGURE 7-41 Managing conditional formatting

- At the top of this dialog box, you can choose to display rules, create a new rule, or edit (change) and delete existing rules. (You can also use other commands to clear rules; see Figure 7-41, right.)
- Each of the rows in the dialog box table provides essential information about the formats.
- You can change the reference in the *Applies to* column.
- If you double-click any part of a row entry, the *Edit Formatting Rule* dialog box opens, which is identical to the *New Formatting Rule* dialog box (see Figure 7-40).

> **Tip** The formatting rule manager dialog box is undoubtedly very useful. However, it may also be rather confusing and unclear, depending on the type, scope, and location of the conditional formatting used. It is essential that you remember to select the correct worksheet or table range in the *Show formatting rules for* box at the top of the dialog box. The only way to become accustomed to using this dialog box correctly is to deliberately practice using different formatting variants in various locations of your workbook.

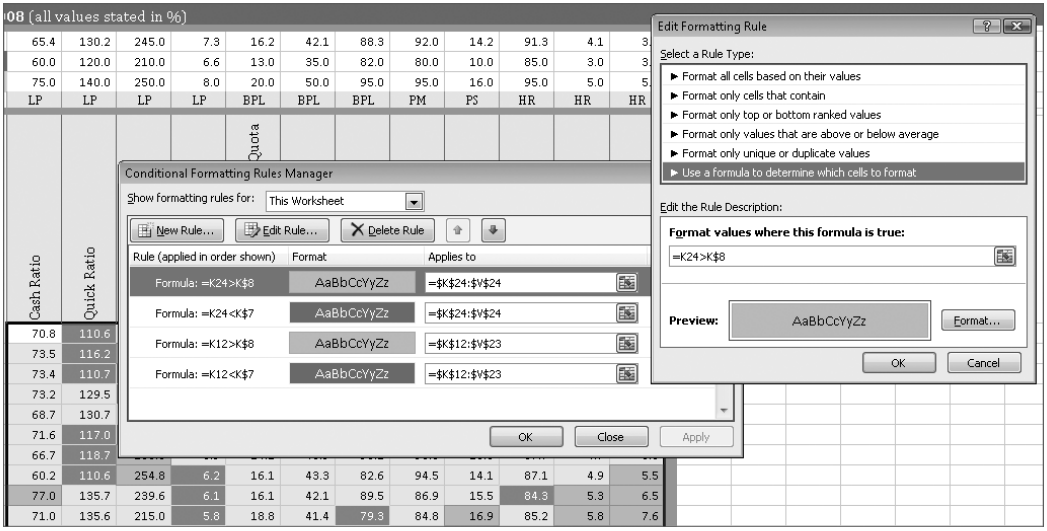
Conditional Formatting: A Real-Life Example

Let’s look now at the conditional formatting used in the *\Samples\0702_Indicators_02.xlsx* sample file:

Data 1 **Worksheet** Conditional formatting is used in the K12:V24 range.

> **Tip** If you want to select cells containing conditional formatting in the active worksheet, select the *Home* tab in the Ribbon, then select the *Editing* group, click the *Find & Select* button, and, finally, select the *Conditional Formatting* option.

These formats use formulas, and were created using the method described in Figure 7-40, example B.



08 (all values stated in %)

65.4	130.2	245.0	7.3	16.2	42.1	88.3	92.0	14.2	91.3	4.1	3.
60.0	120.0	210.0	6.6	13.0	35.0	82.0	80.0	10.0	85.0	3.0	3.
75.0	140.0	250.0	8.0	20.0	50.0	95.0	95.0	16.0	95.0	5.0	5.
LP	LP	LP	LP	BPL	BPL	BPL	PM	PS	HR	HR	HR
Cash Ratio	Quick Ratio			Quota							
70.8	110.6										
73.5	116.2										
73.4	110.7										
73.2	129.5										
68.7	130.7										
71.6	117.0										
66.7	118.7										
60.2	110.6	254.8	6.2	16.1	43.3	82.6	94.5	14.1	87.1	4.9	5.5
77.0	135.7	239.6	6.1	16.1	42.1	89.5	86.9	15.5	84.3	5.3	6.5
71.0	135.6	215.0	5.8	18.8	41.4	79.3	84.8	16.9	85.2	5.8	7.6

Conditional Formatting Rules Manager

Show formatting rules for: This Worksheet

Rule (applied in order shown)	Format	Applies to
Formula: =K24>K$8	AaBbCcYyZz	=$K$24:$V$24
Formula: =K24<K$7	AaBbCcYyZz	=$K$24:$V$24
Formula: =K12>K$8	AaBbCcYyZz	=$K$12:$V$23
Formula: =K12<K$7	AaBbCcYyZz	=$K$12:$V$23

Edit Formatting Rule

Select a Rule Type:

- ► Format all cells based on their values
- ► Format only cells that contain
- ► Format only top or bottom ranked values
- ► Format only values that are above or below average
- ► Format only unique or duplicate values
- ► Use a formula to determine which cells to format

Edit the Rule Description:

Format values where this formula is true:

=K24>K$8

Preview: AaBbCcYyZz

FIGURE 7-42 Conditional formatting in the *Data 1* worksheet

As shown in Figure 7-42, the formulas developed for the range of constants (K12:V23) were also assigned in the same way to the annual result line, in the K24:V24 range. This is yet another typical redundancy of the rS1 method. It’s easy to imagine various ways in which the model might be modified in the future, for example:

- The conditional formatting in the result line should differ from that used for the constants.
- The conditional formatting applied to the constants should be removed, while the conditional formatting applied to the result line should be retained (or vice versa).

We've already made provisions for these eventualities, which didn't involve much additional effort at the time, but will save a fair amount of work and deliberation at a later stage.

Focus 1 **Worksheet** The conditional formatting for the `X25:Z26` range in the *Focus 1* worksheet is similarly generated with a formula:

```
=rL1.PeriodSel+1>COUNTIF(rD1.DataCumulation,"<>#N/A")
```

Note You can use formulas of any type or complexity for conditional formatting, including formulas much more complicated than those described here.

The basic formatting of the cells is the same as that of their neighboring cells, above and below. However, if Excel finds that the above "claim" is correct (TRUE), then the conditional formatting is applied, and these cells will be made to "stand out" from those around them. The intention here is to direct attention to the cell contents; i.e., the infotext, which—as explained earlier—was also generated using formulas.

Chapter 10

Presentation Solutions That Pack a Punch

As the chapter heading suggests, this section shows how to create solutions that have a *powerful effect*. In this regard, the mouse has some work to do, and so have you. Some of the examples presented here are visually elegant and sophisticated. In the preceding nine chapters, however, you have acquired knowledge and experience that will make it easier for you to understand such constructs and their formats. Consequently, the descriptions and notes in this chapter are on the brief side, but an occasional glance at the index will certainly help you. I also remind you to take a very close look at the models available on the CD-ROM and to examine them thoroughly (a little like *CSI*, but without the *C*!).

Consider the solutions presented here as additional, consolidated information about everything already written in relation to the options provided in Excel 2007, or simply as a refresher. To employ these options, you need certain skills and some creativity. Sure, it can be a painstaking effort to acquire the necessary skills for such an extensive software, especially because Excel 2007 is an exciting but also a demanding beauty in this regard. However, if you're looking for ways to support and free your creative imagination, then Excel 2007 is a truly wonderful instrument that has found favor with many. So, let's get started and see what unfolds!

Sorting? You Can Do Without

With our first example, I wish to show you that the rS1.Method can also be completed with conditions more difficult than those outlined previously.

The Task and the Problem

The task is to create a solution that successfully represents a product's quarterly sales figures, broken down into different years and cities. Monthly data is available for a total of 75 cities and we should be able to compare year with year and/or city with city.

The problem is that the necessary source data is delivered as monthly data that has been sorted in order of success. Therefore, the cities and their results are sorted in descending order, according to their sales figures. Consequently, the source data may display the cities in a different sequence each year. It is even more difficult to evaluate the data for the current year because the transferred data is sorted differently each month. However, this is necessary for other analyses and evaluations, and your orderer does not want this to change.

Incidentally, this all concerns the following question, which quite often must be answered: how can we obtain consistent information from inconsistent data? Naturally, numerous database solutions can provide an adequate answer to this question. Within Excel alone, there are many different sophisticated variants, some of which may or may not require programming. However, this also includes a model that is secure in any application and easy for anyone to create, namely the model of a dynamic, non-programmed presentation solution. I will describe how this can work below. However, first take a look at the result.

On the Companion CD Open the file *\Samples\3Years.xlsx* on the CD-ROM.

The *Focus 1* worksheet in the file *3Years.xlsx* is designed for the *Full screen* presentation view and is protected without a password.

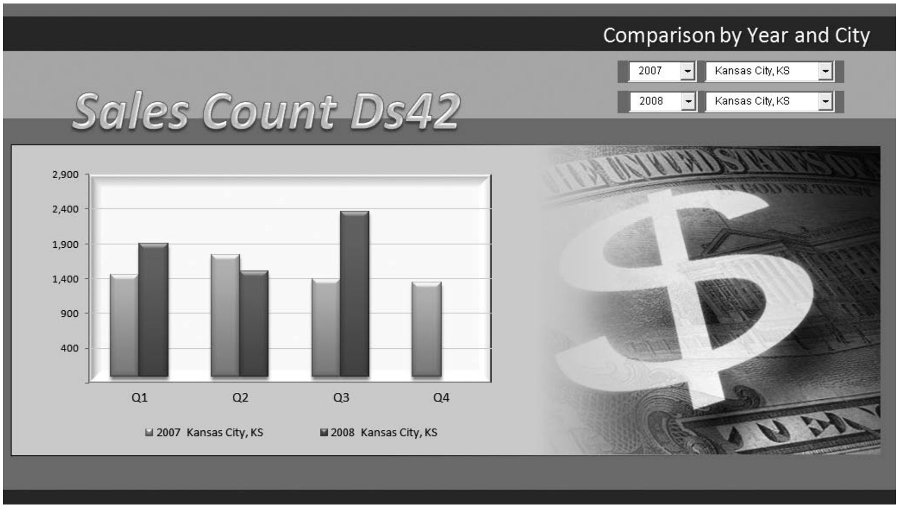


FIGURE 10-1 Simple yet elegant, in a number of ways

The upper right corner of the figure contains four *ActiveX controls* for making various different paired comparisons or cross-comparisons.

- For example, based on the information provided in Figure 10-1: what were the sales figures for Kansas City, KS in 2007 and how do they compare with 2008? (There is no column for Q4, 2008 because no data is currently available for this quarter.)
- However, what were the sales figures for Kansas City, KS in 2007, and how do they compare with the sales figures for Worcester, MA in 2007? Incidentally, in this *ActiveX control*, you do not have to move down 39 positions from Kansas City: it is enough to select the current entry (by double-clicking it) and then overwrite it with the two letters "*wo*" (lowercase is sufficient). It couldn't be easier. The chart is immediately updated to show the data from Worcester, MA.

- You can also determine the level of success achieved for product sales in Knoxville, TN in 2006 and how this compares with the result achieved in Oakland, CA in 2007. There are numerous scenarios that demand such comparisons, each of which seems somewhat strange at first glance. However, the text "Years AND Cities are unequal" below the chart draws your attention to a possible oversight in your query combination.

The above descriptions also clearly illustrate the following problem: a comparative query of 75 cities naturally makes it necessary to sort the city names in the control lists in alphabetical order. However, it is necessary here to extract data that is stored differently in three different data sheets; in other words, sorted according to values and on top of that sorted differently in each of the three sources.

Before we discuss how to solve such difficulties, let's consider the working objective again.

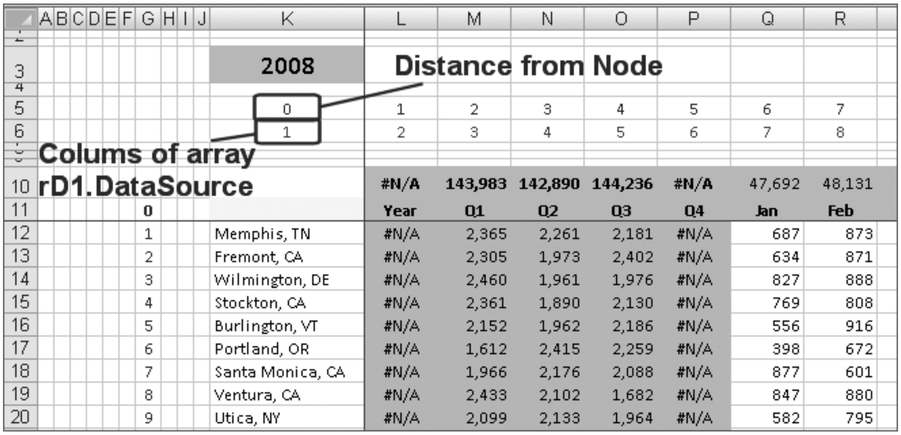
- It must be possible to address one of three possible worksheets as a data source by simply clicking a year in a control (as mentioned previously, and explained in greater detail below, this can be achieved using the `INDIRECT` function).
- It must be possible to extract the quarterly data for a city from each source data sheet without knowing which of the 75 possible row positions is occupied by this city (this does not work when you use `OFFSET` or `INDEX`, except in a very roundabout way, but it does work with `VLOOKUP`).

To make it easier to understand the formula constructions for the chart basis, let me first provide an overview of the data structure and the control designs. I doubt you will be surprised to learn that, once again, there are redundancies. The model is, in many ways, designed to satisfy changing requirements and therefore can be adjusted without much effort, even to meet needs other than those outlined here. For the purpose of this exercise, you may wish to use the material available to supplement this solution.

Organization of the Source Data

The most important information about the *Data 1* ..., *Data 2* ..., and *Data 3* ... worksheets is provided below.

- All of the worksheets have the same structure, even if the data is sorted differently in each sheet. For fast and easy orientation, you will use different colors in the header area to differentiate between different types of data.
- The windows are fixed in both axes (*View/Window/Freeze Panes*) so that you can keep the labels in view even when the data range moves.
- Each data source has a *node* that corresponds to its sheet name: *rD1.Node, rD2.Node*, and *rD3.Node*. This is quite important. These range names differ in number only.



	G	K	L	M	N	O	P	Q	R
3		2008							
5		0	1	2	3	4	5	6	7
6		1	2	3	4	5	6	7	8
10			#N/A	143,983	142,890	144,236	#N/A	47,692	48,131
11	0		Year	Q1	Q2	Q3	Q4	Jan	Feb
12	1	Memphis, TN	#N/A	2,365	2,261	2,181	#N/A	687	873
13	2	Fremont, CA	#N/A	2,305	1,973	2,402	#N/A	634	871
14	3	Wilmington, DE	#N/A	2,460	1,961	1,976	#N/A	827	888
15	4	Stockton, CA	#N/A	2,361	1,890	2,130	#N/A	769	808
16	5	Burlington, VT	#N/A	2,152	1,962	2,186	#N/A	556	916
17	6	Portland, OR	#N/A	1,612	2,415	2,259	#N/A	398	672
18	7	Santa Monica, CA	#N/A	1,966	2,176	2,088	#N/A	877	601
19	8	Ventura, CA	#N/A	2,433	2,102	1,682	#N/A	847	880
20	9	Utica, NY	#N/A	2,099	2,133	1,964	#N/A	582	795

FIGURE 10-2 Two auxiliary rows with different number sequences

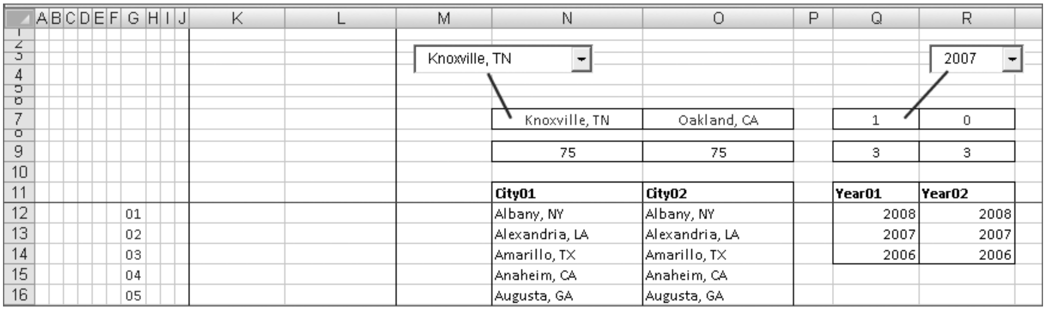
- The monthly data presented here in the form of constants is summarized in columns L to P with formulas for quarterly data and annual data (therefore, it would be easy to use month-specific analyses to enhance this analysis). The presentation chart requires only the values in the range `M12:P91`.
- There are two horizontal auxiliary rows with number sequences. The upper row describes the distance from the node. The lower row indicates the column numbers in an array used by `VLOOKUP` formulas. Each of the three arrays in these worksheets has, in addition to the aforementioned nodes, another range name that corresponds to the sheet name: *rD1.DataSource*, *rD2.DataSource*, and *rD3.DataSource*. Once again, these range names differ in number only. Depending on their actual use, the arrays span the entire values range and therefore also contain monthly data not visualized here.
- In the *Data 1* … worksheet, however, the cells that are retained for monthly data that does not yet exist but will in the future are filled with #N/A, which is usually the case with my models. Similarly, the relevant totals are also filled with #N/A and therefore invisible in the associated charts.

Controls with Text Output

The city names and years sorted in alphabetical order for use in the four model controls are defined in the *Lists 1* worksheet.

Note the following in relation to the output values of the *ActiveX controls*: while the *ComboBoxes* for selecting the year output their index value in numbers from 0 to 2, as was previously determined and accepted (and can't be anything else for *form controls*), the *ComboBoxes* for selecting the city behave differently; they do not write a number. Instead, they write the text selected by the user to its *LinkedCell*. Excellent, but why? Because whoever has such text also has a search term that can be used in `VLOOKUP` (and in other functions).

How can this be set up? It’s really quite easy: the *BoundColumn* property of the *ComboBox* must not be set to *0* (as was previously demanded), but to its default value of *1*.



Knoxville, TN

2007

Knoxville, TN	Oakland, CA
75	75

1	0
3	3

	City01	City02
01	Albany, NY	Albany, NY
02	Alexandria, LA	Alexandria, LA
03	Amarillo, TX	Amarillo, TX
04	Anaheim, CA	Anaheim, CA
05	Augusta, GA	Augusta, GA

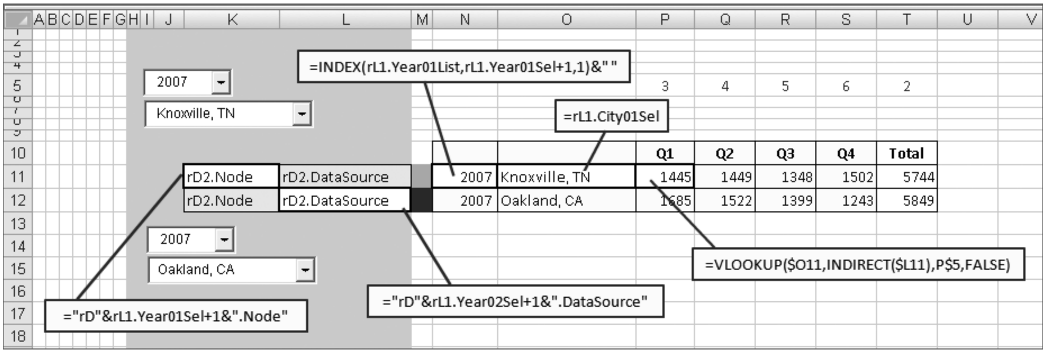
Year01	Year02
2008	2008
2007	2007
2006	2006

FIGURE 10-3 ActiveX controls can also output text

With regard to the *ComboBox* standards, the *MatchEntry* property makes it easy for users to control text entries within extensive lists, simply by entering the first letters. Selecting *fmMatchEntryComplete* triggers *extended matching*; that is, as soon as one or more letters are entered in the control (by selecting and overwriting existing entries), the object searches its *ListFillRange* for an entry that begins with the letter(s) entered and proposes it for selection.

Formulas of the Chart Basis

The chart data is compiled in the *Basis 1* worksheet. This sheet does not require copies of the four controls for the model to work. However, these copies provide you with a better overview when testing and checking the formulas.



=INDEX(rL1.Year01List,rL1.Year01Sel+1,1)&" "

=rL1.City01Sel

=VLOOKUP($O11,INDIRECT($L11),P$5,FALSE)

="rD"&rL1.Year02Sel+1&".DataSource"

="rD"&rL1.Year01Sel+1&".Node"

				3	4	5	6	2
				Q1	Q2	Q3	Q4	Total
rD2.Node	rD2.DataSource	2007	Knoxville, TN	1445	1449	1348	1502	5744
rD2.Node	rD2.DataSource	2007	Oakland, CA	1685	1522	1399	1243	5849

FIGURE 10-4 Just a few formulas, but still more than you need.

Note the following points with reference to Figure 10-4:

```
K11   ="rD"&rL1.Year01Sel+1&".Node"
```

When you use the output value of a control, this formula (and similarly the formula in K12) composes text, namely the name of the node that belongs to the selected year. If you click *2008*, that is, the first entry in the ComboBox, Excel generates the text *rD1.Node*. If you click 2006—the third entry—Excel generates the text *rD3.Node*. You now have a piece of information that you can use as a reference argument, in connection with INDIRECT, in many different formulas (for example, in OFFSET).

This describes an important key element of the rS1.Method that influenced my choice of naming conventions:

- Clicking a control generates a number.
- A formula uses this number to generate a specific range name (generally a node). The number alone is the variable element of the range name.
- The range name used as a variable in this way is the reference argument for other formulas.

In other words, you can click a control to determine the worksheet or cell range from which Excel should read the data that you're interested in.

While OFFSET occupies the "leading role" in the "presentation theater," the INDIRECT(reference,a1) function occupies the role of stage designer or "scene-shifter."

INDIRECT opens the third dimension of your presentation model, without any programming whatsoever. Until now, we spoke only of *one* array or *one* node. With INDIRECT, however, you determine simply and elegantly which of the numerous arrays or nodes you wish to address.

With INDIRECT, you can use text in a cell *outside* any formula as a reference argument *within* this formula. Here, I use the INDIRECT function to establish dynamic references and to define variable reference arguments in formulas. The examples below provide additional information.

In the syntax =INDIRECT(reference,a1), the optional argument A1 represents a logical value that specifies which type of reference is contained in the cell (TRUE for the A1 notation and FALSE for the Z1S1 notation). Because the latter does not play any role in the models provided in this book (nor generally anywhere else), this argument can continue to be ignored (which is then interpreted by Excel as TRUE).

Important The INDIRECT function does not work if its reference argument references a cell in a closed file. Consequently, when you use this function you can work only with external cross-worksheet references if all of the files used are open at runtime.

This formation of variable node names is particularly useful in rS1 models, but is nevertheless not required here; both formulas potentially enhance the model's functions.

The formulas in L11:L12, on the other hand, will be used here immediately. The same principle provides the same benefits:

```
L12   ="rD"&rL1.Year02Sel+1&".DataSource"
```

Both cells therefore contain the optional text *rD1.DataSource*, *rD2.DataSource*, or *rD3.DataSource*. Make sure that cell L12 has *rL1.Year02Sel* instead of *rL1.Year01Sel* as in L11. However, both formulas are identical in every other regard. The same is true of the formulas in K11:K12.

```
N11   =INDEX(rL1.Year01List,rL1.Year01Sel+1,1)&" "
```

Similar cell: N12. These formulas determine the selected year, which is used as the first part of the relevant legend label in the chart. I used &" " to insert a space after the formula because I felt that the gap between the year and city name in the legend was insufficient.

```
O11   =rL1.City01Sel
```

Similar in cell O12: the second string for the relevant legend label in the chart and the search term for the VLOOKUP formulas. Because the relevant *ComboBox* outputs text (as shown above), this can be read directly from the control's *LinkedCell*.

P11:T12,
for example P11=VLOOKUP($O11,INDIRECT($L11),P$5,FALSE)

Note In Chapter 2, "New Approaches—Getting Started," the file *Samples**0201_Extract.xlsx* introduced the function =VLOOKUP(lookup_value,table_array,col_index_num,range_lookup) in detail.

Here is the gist of the matter: "determine a value that is to be localized as follows: use the search term provided in cell O11. Find this string in the first column of the array whose name is in cell L11 (and was created there as a variable for the use of INDIRECT). Once you find it, take the value in this row from the column in the array whose column number is provided in cell P5."

In our example, the notation =VLOOKUP("Knoxville, TN",rD2.DataSource,3,FALSE) would be a static equivalent of the dynamic formula. This is just one of 240 access options that can be controlled with two mouse clicks. This is the case for three source-data ranges. For other requests, however, there could be 10, 50, or more source-data ranges. That makes absolutely no difference when constructing such a chart basis.

Note If, for the aforementioned formula, you were to use a control that also retained a variable column specification in cell P5, there would be 4,320 access options (rather than 240), each of which would require just three mouse clicks. You do not want this here, but you may find it useful some time in the future.

The two formulas in column T are "reserves" that are not required here.

The data range `N10:S12` (colored pale red in the worksheet) is the data source for the chart that presents the collated custom data in the *Focus 1* sheet.

Specials in the *Focus* Worksheet

There are some special features here, but you can only explore these formatting considerations after you unprotect the sheet. Note the following three points:

- The "dollar sign" image is stored on the CD-ROM under \Materials\Pictures\Dollar.tif
- It is interesting that there seems to be a smooth transition between the chart area and the image. Also, take a look at how the chart border has been formatted.
- The *plot area* is designed with the 3-D option *Soft Round*. This gives you a very slightly three-dimensional depth, like a framed picture. However, there is just one visually disappointing disadvantage associated with using column charts: the columns are generally too far "forward," practically on the "outer" edge of the mock border instead of being "set back" on the lower edge of the "picture area." In this case, a little trick rectified the problem. As is frequently the case, two very important components of chart design play a role here: axis scaling and number formatting.

It only remains for me to mention the formula that outputs the aforementioned alert:

```
J29  =IF(
     AND(rL1.Year01Sel<>rL1.Year02Sel,rL1.City01Sel<>rL1.City02Sel),
     "Years AND Cities are unequal","")
```

A Little Data; a Lot of Information

On the Companion CD Open the file *\Samples\1002_10Products.xlsx* on the CD-ROM.

The *Focus 1* worksheet in the file *1002_10Products.xlsx* is designed for the *Full screen* presentation view and is protected without a password.

The sales figures for 10 products and their regional distribution are displayed here. Select a product in the *ComboBox* at the top of the sheet. You may be surprised to see a two-column list that shows not only the product names but also their rankings. When a user selects a product, he or she can also become familiar with its ranking.

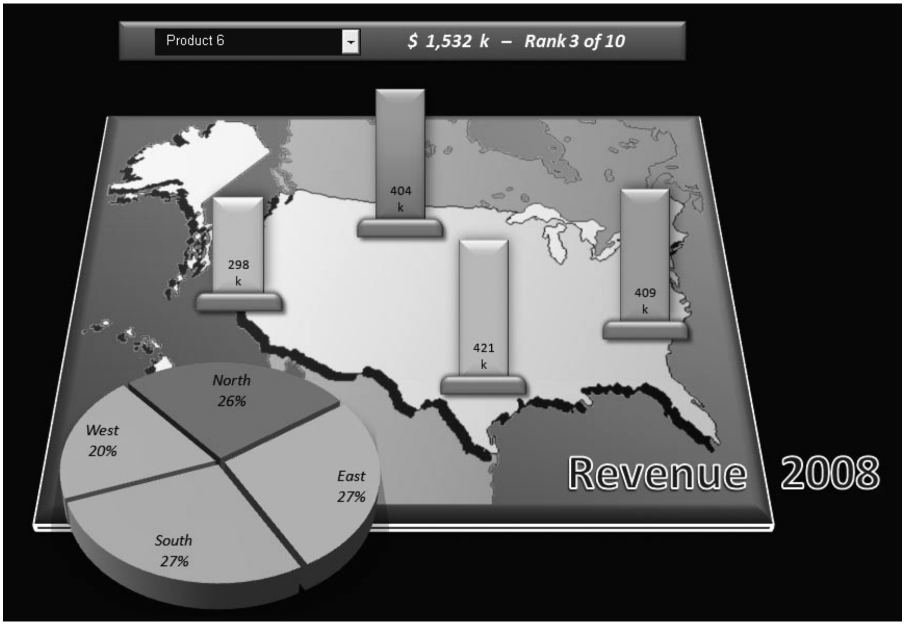


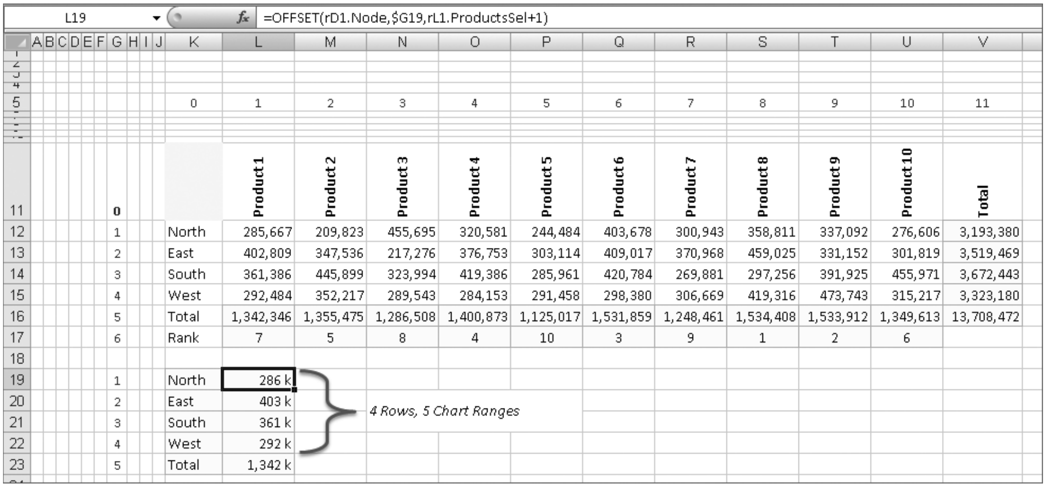
FIGURE 10-5 Copious information: absolute values, rank, regional success, percentage distribution

The results of your decision are shown in several different ways in this extensively designed presentation chart whose main purpose is to “look good”:

- The header bar contains the product, its total sales in thousands of dollars, and its rank.
- The columns on the map represent the subtotals of the four regional sales values for the product selected.
- Each slice of the pie chart represents the percentages of the regional sales values for the product selected.

Note When the control is activated (indicated by a blinking cursor in the object), you can also use the **Down Arrow** and **Up Arrow** buttons to move its contents; that is, to virtually "scroll" up and down the product list.

The source data and chart ranges are presented together in the *Data 1 and Basis* worksheet (see Figure 10-6). This is a simple structure with absolutely no surprises or special features to report. Therefore, I'll keep my comments brief.



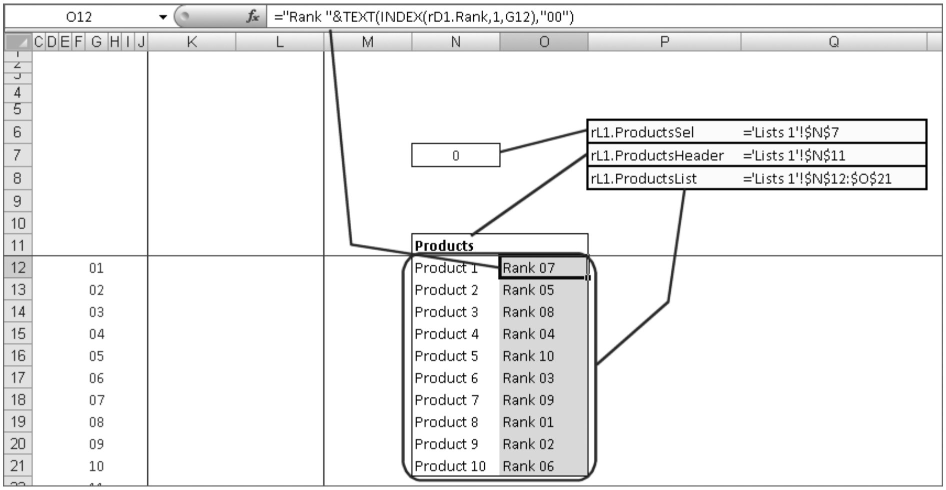
L19 =OFFSET(rD1.Node,$G19,rL1.ProductsSel+1)

	G	K	L	M	N	O	P	Q	R	S	T	U	V
5		0	1	2	3	4	5	6	7	8	9	10	11
11	0		Product 1	Product 2	Product 3	Product 4	Product 5	Product 6	Product 7	Product 8	Product 9	Product 10	Total
12	1	North	285,667	209,823	455,695	320,581	244,484	403,678	300,943	358,811	337,092	276,606	3,193,380
13	2	East	402,809	347,536	217,276	376,753	303,114	409,017	370,968	459,025	331,152	301,819	3,519,469
14	3	South	361,386	445,899	323,994	419,386	285,961	420,784	269,881	297,256	391,925	455,971	3,672,443
15	4	West	292,484	352,217	289,543	284,153	291,458	298,380	306,669	419,316	473,743	315,217	3,323,180
16	5	Total	1,342,346	1,355,475	1,286,508	1,400,873	1,125,017	1,531,859	1,248,461	1,534,408	1,533,912	1,349,613	13,708,472
17	6	Rank	7	5	8	4	10	3	9	1	2	6	
18													
19	1	North	286 k										
20	2	East	403 k		4 Rows, 5 Chart Ranges								
21	3	South	361 k										
22	4	West	292 k										
23	5	Total	1,342 k										

FIGURE 10-6 Some source data and five small chart ranges, that's all you need here.

- Any cells that contain formulas are colored here.
- The ranks that are copied to the *ComboBox* and header row of the *Focus 1* sheet are calculated in row 17.
- The product-specific data from rows 12 to 16 is read in the rows 19 to 23. The values from the columns that correspond to the control setting are shown here. These values are decreased to the thousands format.
- The four-row range K19:L22 provides data for five charts; row by row for the column charts and as a block for the pie chart.

The structure in the *Lists 1* worksheet (see Figure 10-7) is even more interesting.



O12 ="Rank "&TEXT(INDEX(rD1.Rank,1,G12),"00")

Name	Refers to
rL1.ProductsSel	='Lists 1'!$N$7
rL1.ProductsHeader	='Lists 1'!$N$11
rL1.ProductsList	='Lists 1'!$N$12:$O$21

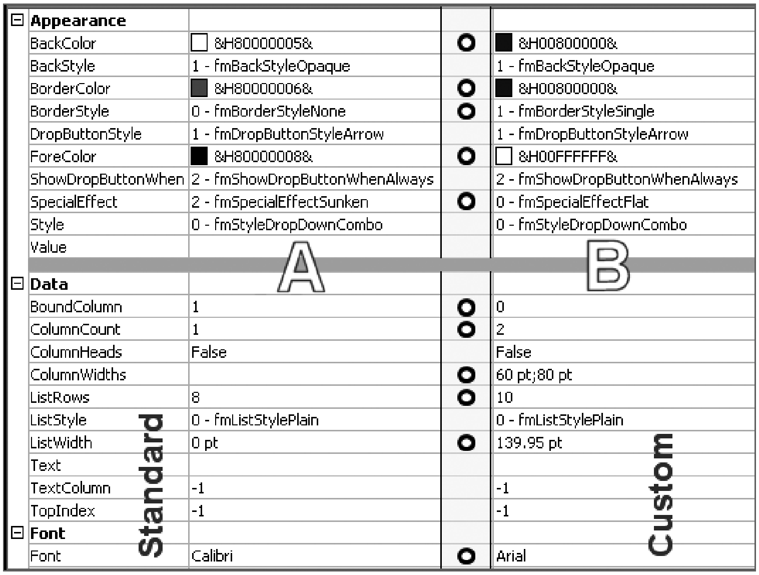
Row	G	N	O
7		0	
11		Products	
12	01	Product 1	Rank 07
13	02	Product 2	Rank 05
14	03	Product 3	Rank 08
15	04	Product 4	Rank 04
16	05	Product 5	Rank 10
17	06	Product 6	Rank 03
18	07	Product 7	Rank 09
19	08	Product 8	Rank 01
20	09	Product 9	Rank 02
21	10	Product 10	Rank 06

FIGURE 10-7 The list definition range for the *ComboBox* has two columns and contains formulas.

You have already seen that the opened control has two columns. The basis for this is a defined two-column *ListFillRange* with the name *rL1.ProductsList*.

Formulas of the type `="Rank "&TEXT(INDEX(rD1.Rank,1,G12),"00")` generate a rank list by reading the relevant data from the aforementioned *Data 1 and Basis* sheet.

In this model, the *ActiveX control* in the *Focus 1* sheet is especially interesting. Here, numerous changes that warrant a more detailed description were made to the default settings.



⊟ **Appearance**			
BackColor	□ &H80000005&	○	■ &H00800000&
BackStyle	1 - fmBackStyleOpaque		1 - fmBackStyleOpaque
BorderColor	■ &H80000006&	○	■ &H00800000&
BorderStyle	0 - fmBorderStyleNone	○	1 - fmBorderStyleSingle
DropButtonStyle	1 - fmDropButtonStyleArrow		1 - fmDropButtonStyleArrow
ForeColor	■ &H80000008&	○	□ &H00FFFFFF&
ShowDropButtonWhen	2 - fmShowDropButtonWhenAlways		2 - fmShowDropButtonWhenAlways
SpecialEffect	2 - fmSpecialEffectSunken	○	0 - fmSpecialEffectFlat
Style	0 - fmStyleDropDownCombo		0 - fmStyleDropDownCombo
Value			
⊟ **Data**			
BoundColumn	1	○	0
ColumnCount	1	○	2
ColumnHeads	False		False
ColumnWidths		○	60 pt;80 pt
ListRows	8	○	10
ListStyle	0 - fmListStylePlain		0 - fmListStylePlain
ListWidth	0 pt	○	139.95 pt
Text			
TextColumn	-1		-1
TopIndex	-1		-1
⊟ **Font**			
Font	Calibri	○	Arial

FIGURE 10-8 Numerous changes can be made using *ActiveX controls*.

On the left-hand side (A) of Figure 10-8, you see an extract of the default settings for a *ComboBox*, grouped into categories. On the right-hand side (B), you see a comparative list of properties applied to the object used here. The rows whose settings differ from the default settings have been marked with small circles.

In this case, the user has adjusted the following settings:

- Formats
  - *BackColor*: the object's background color
  - *BorderColor*: the border color
  - *BorderStyle*: the border property (*fmBorderStyleSingle:* the object has a border)
  - *ForeColor*: the font color
  - *SpecialEffect*: the object's appearance (*fmSpecialEffectFlat:* compared with the default setting, the object is not three-dimensional but flat and sets itself apart from its surroundings by nothing more than a border and/or color)
- Data-specific properties
  - *BoundColumn*: the setting *0* ensures that the control's index value is put out to the *LinkedCell*.
  - *ColumnCount*: the setting *2* assigns two columns to the drop-down list for this *ComboBox*. This requires a *ListFillRange* with two or more columns.
  - *ColumnWidths*: the setting *60 pt;80 pt* defines the width of both columns. A semicolon separates the values. The font size should be defined before the final setting is made here. Generally, after some trial and error, you will be happy with the appearance that you achieve.
  - *ListWidth*: specifies the width of the expanded drop-down list. This value can also be a value other than the total value for *ColumnWidths*.
- Font: Here, the *Arial* font was determined as the *Font* property. Generally, this font is considerably easier to see in such objects than are most other fonts.

I won't describe the remaining properties here because, of these, only *LinkedCell* and *ListFillRange* are important developer settings. At this point, there is no need to say anything more about these settings.

I can therefore focus on providing some details about the design layout of the *Focus 1* worksheet.

On the Companion CD Several maps, including some for Europe, Ireland, and the United States, are stored on the CD_ROM under *Materials**Pictures*\. The image used here is *USA_CA_Map.tif*.

- The chart columns appear to rest on a podium that comprises a rectangle and the aforementioned image, and they have been connected to a group. With regard to the elements in this group, which are tilted backwards, examine the values for *3-D rotation, 3-D format* (including *material* and *lighting*), *line color* (this concerns the border, for example), *line style*, and *shadow*, among others.

> **Important** The rectangle and image are turned 320° along the Y axis (that is, vertically) and assigned a perspective of 45°. However, as a grouped graphical object it continues to require as much room as if it had not been rotated (as you can see from its markers). Therefore, the object can't be moved up to its visible upper edge, which is the top of the screen, but only to its own hidden edge.

- The pie chart is adjusted to the location and perspective of the graphical podium. The *angle of the first slice* (North) was set to 320°, so that the distribution and positioning of all 10 possible slices essentially correspond to the cardinal points of the compass (North, South, East, and West).
- The pie chart has a slight *pie explosion* of 3 percent and the *data label* shows the *category name* and *percentage*. Of course, the colors here must correspond to the colors of the four column charts.

The column charts positioned in their regional locations are grouped objects that comprise a chart and a rectangle. In such images, objects of this type should never be used on their own to convey information. Because the columns are not standing directly next to each other, observers cannot clearly distinguish between the column heights. In this case, observers must rely on the *data labels* when making a comparison. The column heights must differ greatly if they are to be relied upon when making a comparison. Here, it is important to supplement the image with a pie chart, not only for visual reasons, but because it is a key design element that safeguards the information provided with the image.

When creating the column charts, I recommend the following:

1. Use the *North* data to create the first chart with a fixed scaling of the value axis.
2. Make all of the formatting settings and, when you are finished, make sure that you expand the *plot area* to the maximum *chart area*. Then, and only then, set the *chart area* size you require. When the *chart area* is the same size as the *plot area*, it is considerably easier to make any necessary exact adjustments to the width and height of the object at a later date. Of course, all four objects here must have the same dimensions, so that the same values result in the same column height.
3. Insert and format the *Round Same Side Corner Rectangle* which forms the base of the chart and position it carefully on the lower edge of the chart.
4. Connect both objects to a group and position these on the map.

5. Copy the object and place it elsewhere. Within the group object that you have copied, select the chart (to do this, you don't need to dissolve the group), and assign the *East* data as the *data source*.
6. Position the new object and then select its column to change its color. Naturally, you can also do this as a final step for all four charts, which is better if you have to choose your colors by appearance and comparison.
7. Repeat steps 5 and 6 for the two remaining tasks.

With regard to the overall page design: when using rotated objects, you should pay particular attention to the accuracy of the perspectives and the proportional correlation of the objects shown. In this case, this applies not only to the map and pie chart, but also to the relationship between the map and the header bars. To illustrate this, Figure 10-9 has two lines to highlight the central perspective.

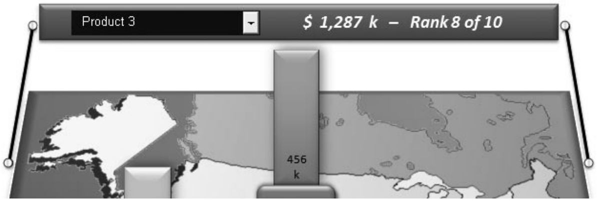


FIGURE 10-9 Observe the perspective here

A Lot of Data—Compressed Information

This section presents two solutions (or three, depending on the how you count them). Each has a similar background (large amounts of data) and the same purpose (heavily compressed information), but can have a completely different appearance. This is hardly surprising, because the objective of a presentation or piece of work determines the layout of your model. Even just one body of data and one basic objective can give rise to hundreds of different solutions.

Chart of Key Data Over 10 Years

On the Companion CD Open the file *\Samples\1003_TenYears.xlsx* on the CD-ROM.

The *Focus 1* worksheet in the file *1003_TenYears.xlsx* is designed for the *Full screen* presentation view and is protected without a password.

You already know this model from the later sections in Chapter 4, "Colors, Areas, and Outlines," which discussed the model's format. We will now discuss its overall structure.

The percentage line chart shows the development of eight operational key figures over a period of ten years. The chart basis here is the values from 1999, each of which has been equated with 100 percent. Therefore, the lines that trend upwards and downwards from the base line describe the direction and extent of any changes. Each line has an "on/off switch" to the right of the chart. Therefore, with a mouse click, a user can use a table that also acts as a legend to determine which lines should appear, how many of them, and in which combination. This produces a multifaceted, multivariable analysis: for example, how line A changes in terms of waveforms and intervals in relation to the 100 percent horizontal *and* in relation to line B, and at the same time to line C, or how B relates to C, or how does it look if—and so on. These simple figures can provide a wealth of information and conclusions. Let's just answer one of these questions in the context of Figure 10-10. (Since the different line colors can hardy be differentiated in the gray scale print of the book, you should use the file to follow along.) From 2001 to 2005, the company reduced its product range significantly. Around the same time—from 2002 and steadily after that—it witnessed a considerable rise in the number of high-revenue A customers and therefore (in a delayed yet pleasantly synchronous trend that did not seem to be coincidental) a significant jump in sales.

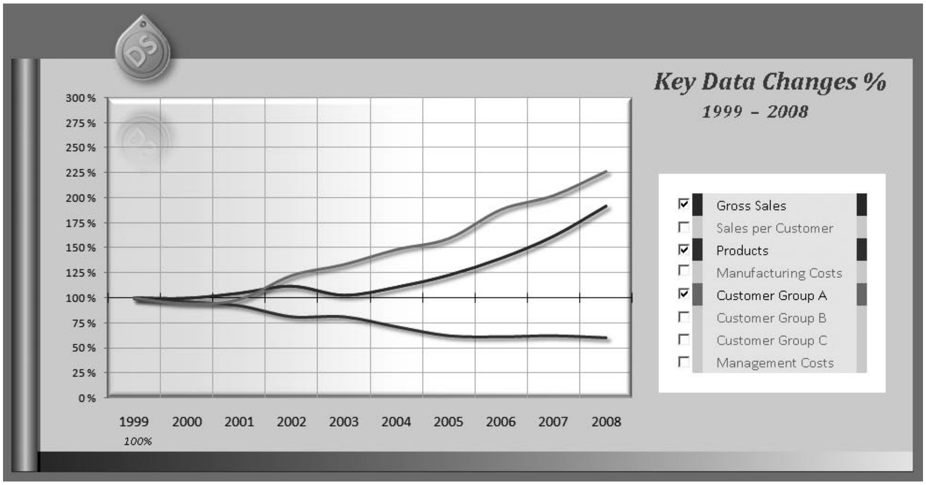


FIGURE 10-10 Half chart, half table

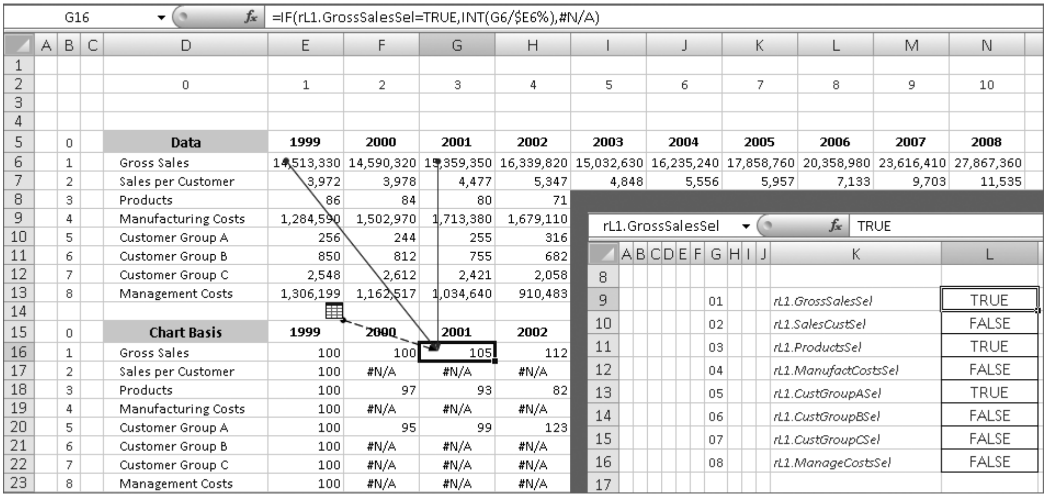
The data contained in the *Data 1* worksheet is divided into two blocks (see Figure 10-11):

- The upper block, *Data*, contains absolute values. Clearly, these numbers are already extremely compressed and are based on several thousand individual values.
- The lower block, *Chart Basis*, contains percentages. For 1999, the constant 100 was entered in all cells. Each of the subsequent years has formulas whose results (when you use each year's absolute values) express its relationship with the start year as a percentage (for more information, see the tip provided below). However, the results (and therefore the chart line) are only shown if the relevant switch for this line is "on."

The formula in cell G16 (in the context of Figure 10-11):

`=IF(rL1.GrossSalesSel=TRUE,INT(G6/$E6%),#N/A))` as a statement: "if the value in the cell *rL1.GrossSalesSel* is TRUE (if the corresponding *CheckBox* has been activated), the result is an integer that represents the percentage of the G6 value in relation to the E6 value. Otherwise (if the corresponding *CheckBox* has been deactivated), the result is #N/A."

Tip In even easier-to-use models of this type, you can use controls to vary the start year for the calculation and therefore show other curve progressions that are flatter or steeper.



G16 =IF(rL1.GrossSalesSel=TRUE,INT(G6/$E6%),#N/A)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1														
2				0	1	2	3	4	5	6	7	8	9	10
3														
4														
5		0		**Data**	**1999**	**2000**	**2001**	**2002**	**2003**	**2004**	**2005**	**2006**	**2007**	**2008**
6		1		Gross Sales	14,513,330	14,590,320	15,359,350	16,339,820	15,032,630	16,235,240	17,858,760	20,358,980	23,616,410	27,867,360
7		2		Sales per Customer	3,972	3,978	4,477	5,347	4,848	5,556	5,957	7,133	9,703	11,535
8		3		Products	86	84	80	71						
9		4		Manufacturing Costs	1,284,590	1,502,970	1,713,380	1,679,110						
10		5		Customer Group A	256	244	255	316						
11		6		Customer Group B	850	812	755	682						
12		7		Customer Group C	2,548	2,612	2,421	2,058						
13		8		Management Costs	1,306,199	1,162,517	1,034,640	910,483						
14														
15		0		**Chart Basis**	**1999**	**2000**	**2001**	**2002**						
16		1		Gross Sales	100	100	105	112						
17		2		Sales per Customer	100	#N/A	#N/A	#N/A						
18		3		Products	100	97	93	82						
19		4		Manufacturing Costs	100	#N/A	#N/A	#N/A						
20		5		Customer Group A	100	95	99	123						
21		6		Customer Group B	100	#N/A	#N/A	#N/A						
22		7		Customer Group C	100	#N/A	#N/A	#N/A						
23		8		Management Costs	100	#N/A	#N/A	#N/A						

rL1.GrossSalesSel TRUE

	G	K	L
8			
9	01	*rL1.GrossSalesSel*	TRUE
10	02	*rL1.SalesCustSel*	FALSE
11	03	*rL1.ProductsSel*	TRUE
12	04	*rL1.ManufactCostsSel*	FALSE
13	05	*rL1.CustGroupASel*	TRUE
14	06	*rL1.CustGroupBSel*	FALSE
15	07	*rL1.CustGroupCSel*	FALSE
16	08	*rL1.ManageCostsSel*	FALSE
17			

FIGURE 10-11 Each chart row has its own on/off switch.

The lower right insert in Figure 10-11 originated in the *Lists 1* worksheet where each *CheckBox* also has its own clearly discernible cell link.

The technical setup of the *Focus 1* presentation sheet provides only a handful of surprises for you, the most important being:

- The value axis has a fixed scaling and the category axis intersects it at the *axis value* 100.
- The *label position* of the category axis is set to *Low.*
- The *chart area* is completely transparent. Therefore, its background is the fill color of the table. This makes it easy to perceive the small table on the right as a "legend with switches."
- To reinforce this last impression: when you deactivate the entries, the color signals of the entries are deleted and the texts are hidden. Of course, this is based on *conditional*

formatting. In this case, these only refer indirectly to the cell links to the CheckBoxes, namely to those values generated as hidden values in column Q. For example, `Q8`:

`=IF(rL1.GrossSalesSel=TRUE,"x","")`. The reason for this detour is that it is much easier to construct the formula rules for *conditional formatting* if at the same time you can refer to a uniform cell range with the contents `"x"` or `""` instead of having to use eight different range names.

- The CheckBoxes are not labeled and are transparent. However, their width spans the neighboring text in a table cell. As a result, the user can also (therefore indirectly) click the text whose line he wishes to see in the chart. This is both sensible and convenient.
- The logo is stored on the CD-ROM under \Materials*Pictures**LogoDS.tif*. Chapter 5, "Graphical Objects," provides information about how to delete its ambient color, scale the logo, rotate it, and create its reflection.
- What appears to be a column to the left of the chart is simply cell formatting. The same is true of the color gradient below the chart. Access via: *Home/Font/*click the Dialog Box *Launcher*. In the *Format Cells* dialog box, on the *Fill* tab, click *Fill Effects*.

Filtering with the Filter—and Filtering with Controls

This section shows how to present the findings of a study. You were introduced to some of this data and its presentation in Chapter 2, which was concerned with evaluating customer numbers as part of a "summer campaign." The following is a brief recap: a retail business with 100 branches launched a five-week promotional campaign to attract more customers into its branches and compensate for the expected seasonal dip in sales over the summer months. The chart's purpose can be summarized this way: "we launched a campaign and now want to see if it attracted more customers and, if so, when, and to which branches."

We will now take a look at the revenue findings from the same study, analyzing whether this summer campaign was successful in terms of revenue and, if necessary, how it succeeded.

This model brings us back to the subject of "filtering," which was mentioned briefly in Chapter 8, "Chart Types—Conventional and Exceptional." In this regard, the file *1004_SummerCampaign.xlsx* contains two options that are based on different objectives or requirements:

- The Controlling department of the company, which has proficient Excel users, uses the *pivot* system (not part of the description here) and a slender filter model to analyze data. This gives rise to versatile analyses that have numerous selections and answer a multitude of questions. If, for example, one of the settings does not make any sense and produces strange results (or none at all), this is not a problem because the setting is quickly identified and corrected using the resources within Excel.

- Even though the management team of a company, which does not necessarily have to be proficient in Excel (that's why companies have specialists), has exactly the same data, it is housed in a much more "closely knitted" model that acts as a filtering report module rather than a filtering analysis instrument. It is not as slender as the Controlling module and it can't answer every question on the spur of the moment (that's why companies have financial controllers). However, it can answer most questions, especially the most important ones, without relying on any external assistance. Furthermore, it can do this in the easiest way possible: just a few mouse clicks in four selection lists containing clear text will retrieve the necessary information, without risk of error, and without any irritating program responses. That's exactly how it should be.

From a technical perspective, lots of things that have already been the subject of various discussions in this book converge here. However, Excel 2007 also introduces you to something completely new: formulas that virtually produce any differentiable filter result, without you having to filter!

On the Companion CD Open the *\Samples\1004_SummerCampaign.xlsx* file on the CD-ROM.

The *Focus 1 Filter* and *Focus 2 Formulas* worksheets in the file *1004_SummerCampaign.xlsx* are designed for the *Full screen* presentation view and are protected without a password. To test the model, you should select these settings or leave them unchanged. Naturally, you should also unprotect the sheet if you wish to examine the structures.

Variant 1: The Complex Analysis Module

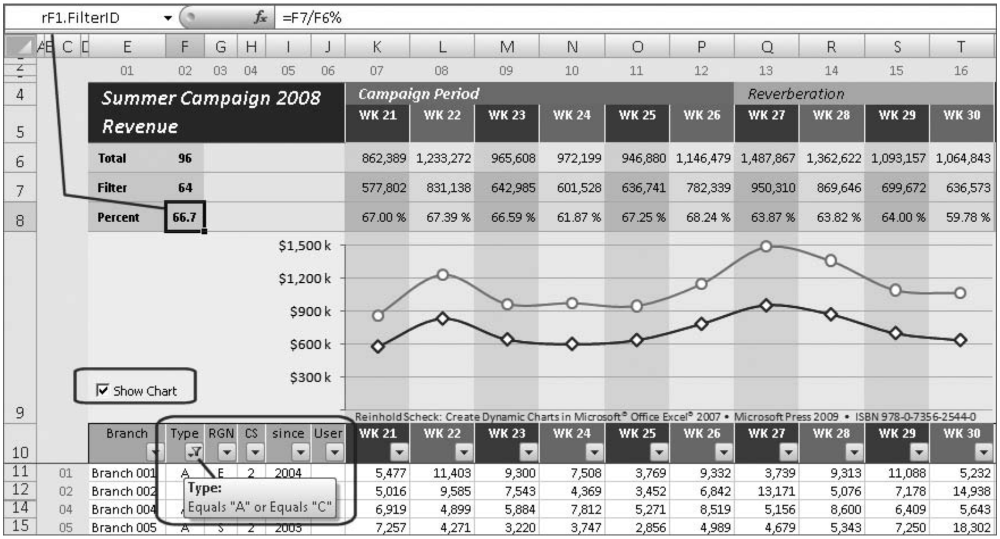
Activate the *Focus 1 Filter* worksheet, which is the analysis module for Controlling.

What is available and what is possible?

- Unlike many of the models introduced earlier, this module houses everything in just one sheet: the master data, chart basis, tabular results display, and chart.
- The sales figures for 100 branches and 10 calendar weeks are listed below a "frozen" sheet. The branches are classified into several columns; that is, assigned grouping characteristics. This is extremely important for variable filter analysis:
    - Column *Type*: *A, B, C*—three different types of branches (divided according to type of business, foundation type, and localization at the business location)
    - Column *RGN*: *N, E, S, W*—regionalization according to the cardinal points of the compass
    - Column *CS*: *1, 2, 3*—differentiation according to the number of inhabitants where the business is located

- ❑ Column *since*: years *1999* to *2006* (inclusive)—the year in which the branch was founded or acquired
- ❑ Column *User*: This stands for *user-defined* (custom) and is a very important column in filter models of this type. In practice, however, this kind of column is frequently missing, which is most unfortunate. Here, the user can specify his own classifications. In addition to existing and standardized characteristics, he can also define whatever he wants, wherever he wants. Therefore, he can also filter whatever he wants, however he wants.

Now let's take a look at the views that are currently possible. If you are working in a protected worksheet, take a look at the row numbers shown in Figure 10-12:



rF1.FilterID =F7/F6%

Summer Campaign 2008 Revenue

	Campaign Period						Reverberation			
	WK 21	WK 22	WK 23	WK 24	WK 25	WK 26	WK 27	WK 28	WK 29	WK 30
Total 96	862,389	1,233,272	965,608	972,199	946,880	1,146,479	1,487,867	1,362,622	1,093,157	1,064,843
Filter 64	577,802	831,138	642,985	601,528	636,741	782,339	950,310	869,646	699,672	636,573
Percent 66.7	67.00 %	67.39 %	66.59 %	61.87 %	67.25 %	68.24 %	63.87 %	63.82 %	64.00 %	59.78 %

☑ Show Chart

Reinhold Scheck: Create Dynamic Charts in Microsoft® Office Excel® 2007 • Microsoft Press 2009 • ISBN 978-0-7356-2544-0

	Branch	Type	RGN	CS	since	User	WK 21	WK 22	WK 23	WK 24	WK 25	WK 26	WK 27	WK 28	WK 29	WK 30
01	Branch 001	A	E	2	2004		5,477	11,403	9,300	7,508	3,769	9,332	3,739	9,313	11,088	5,232
02	Branch 002						5,016	9,585	7,543	4,369	3,452	6,842	13,171	5,076	7,178	14,938
04	Branch 004	A					6,919	4,899	5,884	7,812	5,271	8,519	5,156	8,600	6,409	5,643
05	Branch 005	A	S	2	2003		7,257	4,271	3,220	3,747	2,856	4,989	4,679	5,343	7,250	18,302

Type:
Equals "A" or Equals "C"

FIGURE 10-12 Such structures provide a large number of analysis options.

- ■ If the sheet does not have any filters, only the total results in row 6 are shown.
- ■ If you have set a filter, the absolute and relative filter results are also shown in rows 7 and 8.
- ■ If you have activated the *CheckBox* with the text *Show Chart*, all of the data (the "whole") is represented by an orange line. If a filter result also exists, it is represented by a blue line. Otherwise, the chart is hidden. (In many cases, the percentage results in row 8 are considerably more important than the trend waveform shown in the chart.)

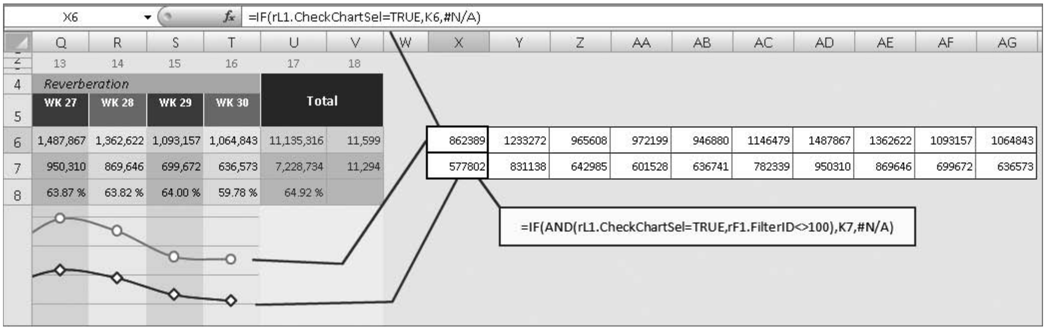
Now let's take a look at the current settings made in Figure 10-12, where the user has set some filters. In the *Type* column, he has selected types *A* and *C* (respectively excluded type B). He has also activated the *CheckBox*. As a result, all of the aforementioned displays are shown.

In this sheet, cell F8 (*rF1.FilterID*) is particularly important. Its value expresses the percentage of branches contained in the filter. Therefore, if this number is 100, the sheet is not filtered. This specification is used by several formulas. For example:

```
K7    =IF(rF1.FilterID=100,"",SUBTOTAL(9,K$11:K$106))
K8    =IF(rF1.FilterID=100,"",K$7/K$6%)
```

Note The =SUBTOTAL(function_num,reference) function, which you can use to calculate filter results, was described in the last main section in Chapter 8.

The chart basis (see Figure 10-13) is housed in the cell range X6:AG7 and made invisible through the font color chosen.



X6 =IF(rL1.CheckChartSel=TRUE,K6,#N/A)

	Q	R	S	T	U	V
2	13	14	15	16	17	18
4	*Reverberation*				Total	
5	WK 27	WK 28	WK 29	WK 30		
6	1,487,867	1,362,622	1,093,157	1,064,843	11,135,316	11,599
7	950,310	869,646	699,672	636,573	7,228,734	11,294
8	63.87 %	63.82 %	64.00 %	59.78 %	64.92 %	

	X	Y	Z	AA	AB	AC	AD	AE	AF	AG
6	862389	1233272	965608	972199	946880	1146479	1487867	1362622	1093157	1064843
7	577802	831138	642985	601528	636741	782339	950310	869646	699672	636573

=IF(AND(rL1.CheckChartSel=TRUE,rF1.FilterID<>100),K7,#N/A)

FIGURE 10-13 The chart basis is hidden here.

There are different formulas for creating these two data series. For example:

```
X6    =IF(rL1.CheckChartSel=TRUE,K6,#N/A)
```

If the *CheckBox* is activated, the value is read from cell K6. Otherwise, it remains as #N/A. Consequently, the line for the total data from row 6 is shown or hidden.

```
X7    =IF(AND(rL1.CheckChartSel=TRUE,rF1.FilterID<>100),K7,#N/A)
```

If the *CheckBox* is activated *and* if the value in cell *rF1.FilterID* is not 100, the value is read from cell K7. Otherwise, it remains as #N/A. The line for the filter data from row 7 is shown or hidden.

Experiment with different filters and take advantage of the clearly improved and more intuitively manageable options provided in Excel 2007. There are numerous analytical options. You will be spoiled by having so much choice, which is another reason why I highly recommend this type of data analysis.

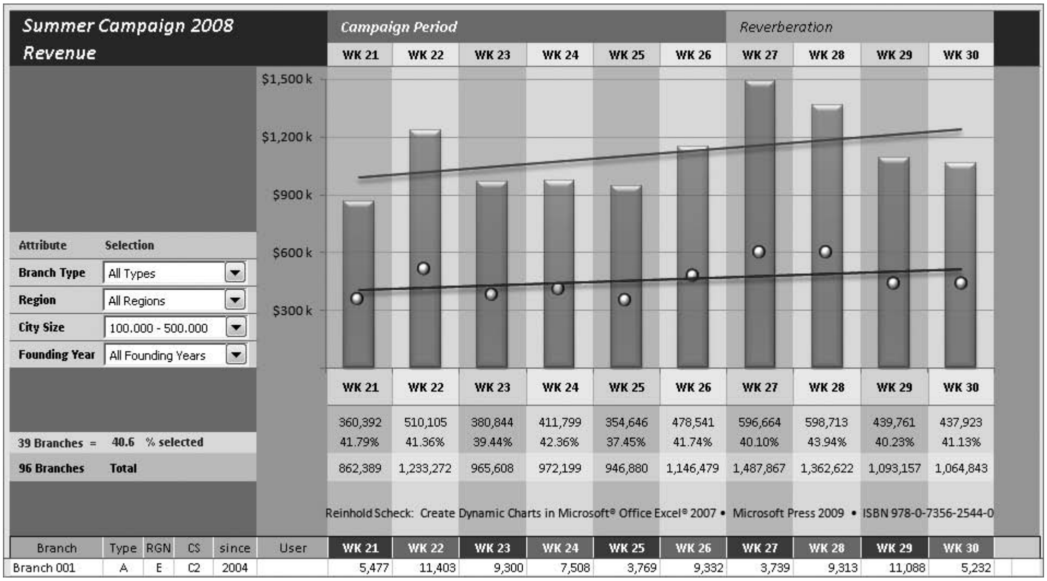
Variant 2: The Easy-to-Use Report Module

Activate the *Focus 2 Formulas* worksheet, which is the report module for the Management team. This sheet in the file *1004_SummerCampaign.xlsx* is designed for the *Full screen* presentation view and is protected without a password. The filter options are suppressed, but can be activated without great difficulty. To keep this as a viable option in the future, the data structure as of row 24 corresponds to the filter model that I introduced above.

Note Of course, in a real-life scenario, such a module would be protected with a password before it left a company's possession or its developer's hands, so that handling errors could be avoided.

You can't set filters in a protected worksheet unless you explicitly permit the filter option (by activating the relevant *CheckBox*) in the *Protect Sheet* dialog box.

If you have not yet discovered all of the hidden new features of Excel, you will be very interested to learn that you can use simple *form controls* to group together a combination of criteria that Excel will immediately use to generate a corresponding results table and chart. It is worth noting that this does not involve any programming, has nothing to do with a *pivot* system, and is not the result of a filtering process.



Attribute	Selection
Branch Type	All Types
Region	All Regions
City Size	100.000 - 500.000
Founding Year	All Founding Years

	WK 21	WK 22	WK 23	WK 24	WK 25	WK 26	WK 27	WK 28	WK 29	WK 30
	360,392	510,105	380,844	411,799	354,646	478,541	596,664	598,713	439,761	437,923
39 Branches = 40.6 % selected	41.79%	41.36%	39.44%	42.36%	37.45%	41.74%	40.10%	43.94%	40.23%	41.13%
96 Branches Total	862,389	1,233,272	965,608	972,199	946,880	1,146,479	1,487,867	1,362,622	1,093,157	1,064,843

Reinhold Scheck: Create Dynamic Charts in Microsoft® Office Excel® 2007 • Microsoft Press 2009 • ISBN 978-0-7356-2544-0

Branch	Type	RGN	CS	since	User	WK 21	WK 22	WK 23	WK 24	WK 25	WK 26	WK 27	WK 28	WK 29	WK 30
Branch 001	A	E	C2	2004		5,477	11,403	9,300	7,508	3,769	9,332	3,739	9,313	11,088	5,232

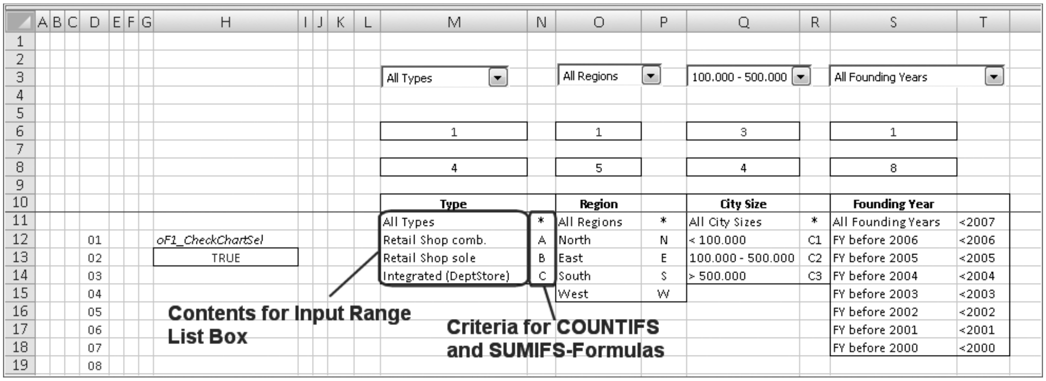
FIGURE 10-14 The Report module provides fast results and is child's play to use.

Behind the scenes lie new functions whose results once could be achieved only with painstaking efforts and even then only in part (through the use of array formulas). I had longed for functions such as `SUMIFS` and `COUNTIFS` (note the *S* at the end; it's not a typing error) for such a long time and was overjoyed to find them. They make it considerably easier to perform lots of calculation tasks and to create user-friendly models.

Before I explain these in greater detail, let me first provide a brief overview of what can be achieved with the model used here. Note the following points with reference to Figure 10-14:

- If you select *All* … in each of the four controls, a chart that represents the whole is shown. Orange columns have white/blue dots on their upper edge, all of which are supplemented by a linear trend line (colored blue). These are relatively insignificant for the user, who needs to see them only once to know which purpose they serve. Furthermore, the line is a permanent fixture because its main purpose is to enable observers to make comparisons.
- If you use the controls to select any other combination, the chart becomes a comparative presentation: a selected group in relation to the whole. The blue dots are now dropping down and show, as a data series within the *Total* columns, those values that correspond to the custom query. There are now two linear trend lines, a red trend line for the whole and a blue trend line for the combination selected. This data material, which assesses the success of a campaign, contains a particularly high and immediately intelligible informational value: the upper trend line shows the campaign's success in all branches. Therefore, if the lower trend line runs in parallel, the data selected essentially corresponds, in a qualitative manner, to the whole. If the lower trend line reflects an "inconsistency", the results of the selection were better or worse than the whole. Therefore, this is a method of fast, optional group analysis in relation to the overall success of the campaign. And if you wish to discuss the results in greater detail? Or if you wish to compare such groups with each other? This or similar Excel solutions could answer all your "if" questions. (Such a report module existed in this case, but deliberately not for the Management team, but for the working environment of the Controller.)

To be able to describe how this solution works, we'll need to take a detour via the *Lists 1* sheet.



	D	H	M	N	O	P	Q	R	S	T
1										
2										
3			All Types ▾		All Regions ▾		100.000 - 500.000 ▾		All Founding Years ▾	
4										
5										
6			1		1		3		1	
7										
8			4		5		4		8	
9										
10			**Type**		**Region**		**City Size**		**Founding Year**	
11			All Types	*	All Regions	*	All City Sizes	*	All Founding Years	<2007
12	01	*oF1_CheckChartSel*	Retail Shop comb.	A	North	N	< 100.000	C1	FY before 2006	<2006
13	02	TRUE	Retail Shop sole	B	East	E	100.000 - 500.000	C2	FY before 2005	<2005
14	03		Integrated (DeptStore)	C	South	S	> 500.000	C3	FY before 2004	<2004
15	04				West	W			FY before 2003	<2003
16	05								FY before 2002	<2002
17	06								FY before 2001	<2001
18	07								FY before 2000	<2000
19	08									

Contents for Input Range List Box
Criteria for COUNTIFS and SUMIFS-Formulas

FIGURE 10-15 The list selection in the control generates a search criterion for a formula.

Here, you see the list contents (in an arrangement that is now familiar to you) for the controls in the focus sheet (see Figure 10-15). For a better overview, copies of the controls are also provided in this sheet. Associated search terms are listed to the right of the clear text list contents. These are similar to those used in the filter analysis module to classify the branches. It is exactly those search terms that we require immediately. Whoever uses this module doesn't have to know which terms these are, where they are, and how they are used. From a functional perspective, the following happens:

1. The user selects a clear text entry (specific information) in a control and generates a number by doing so.
2. The number that is generated is used to determine the position of a search criterion (abstract information).
3. The search criterion found in this way is read in the focus sheet and used there by selective formulas.
4. These formulas write their results to a result line that is used to produce a chart line. Both the numeric values and the chart provide the information requested via the clear text query.

Therefore, the user only needs to know the importance of clear text information in a control, and he only needs to be able to use the mouse to click this clear text information. A structure that he does not have to be familiar with or even be aware of takes care of everything else. This paraphrases my approach to a useful management information system.

I will now describe the main formulas (the other few are not worth mentioning here). You were already familiar with the use of SUMIF and COUNTIF before reading this book or you were introduced to them in Chapter 7, "Elements of Dynamization," (if necessary, take another quick look at this chapter). You can understand and use the new functions SUMIFS and COUNTIFS as multiple variants of basic forms that are already very helpful.

But first, the theory.

The function

```
=SUMIFS(sum_range,
criteria_range1,criteria1,
criteria_range2,criteria2,
```

etc. to

```
criteria_range127,criteria127)
```

works as SUMIF, but can use not only one search criterion, but as many as 127 search criteria simultaneously! This enables you to perform extremely versatile calculations with multiple filters.

The argument `sum_range` describes the range (reference) to be totaled.

The arguments `criteria_range1,criteria_range2,` etc. are up to 127 ranges (references) in which the search criteria are to be found.

The arguments `criteria1,criteria2,` etc. are up to 127 search criteria (or, which is much better in reality, references to cells that contain search criteria.)

When simply expressed in relation to the current example, which is a row-by-row arrangement of the data to be calculated, the first part of the formula is:

```
=SUMIFS(sum_range,criteria_range,criteria1
```

- `sum_range:` you should use this column to create a total. What are the prerequisites? I will tell you now with a combination of `criteria_range` and `criteria`. However, I'll do it not with just one single combination, but with four or even 127!
- `criteria_range:` this column contains the search criterion. If the search criterion is contained in one row within this column, this is one of the rows that will be included in the total.

Stop right there, however: in our example comprising four combinations, this is only the case if the particular search criterion is contained in this row four times for four different columns; in other words, if this row contains the four different search criteria, column by column (and, in an extremely unlikely scenario, only if the particular search criterion is contained in this row 127 times for 127 columns, requiring a row that contains all 127 defined search criteria).

- `criteria:` this is the search criterion to be used or this is the cell that contains the search criterion to be used.

Similarly for COUNTIFS:

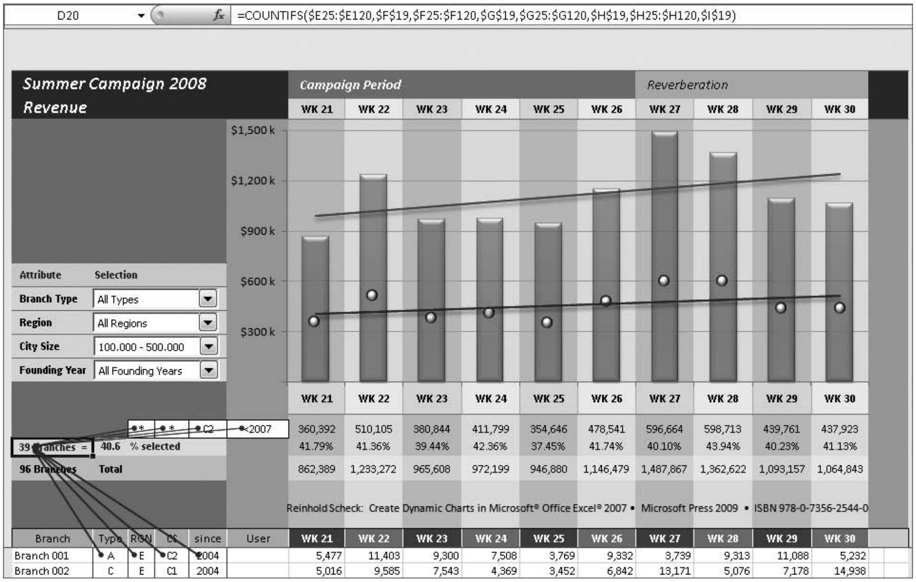
The function =COUNTIFS(criteria_range1,criteria1,criteria_range2,criteria2,...) works like COUNTIF, but it can evaluate not only one range with one search criterion, but as many as 127 ranges with 127 search criteria simultaneously.

- criteria_range: you should perform a search in this column
- criteria: you should search for this criterion

However, this does not involve just one of these combinations, but four or even 127, and this row is only counted if it contains all four or all 127.

When used for the first time, both functions (SUMIFS/COUNTIFS) definitely cause some confusion, not least because these functions can be used together to determine average values and because the syntax of SUMIFS does not correspond to the argument sequence for SUMIF. I recommend that you conduct your own experiments with these extremely useful functions. Therefore, I won't provide lengthy descriptions of the references used here in our example.

> **Tip** If you develop and test such formulas, you should also simultaneously set appropriate filters whose results you calculate with SUBTOTAL. Furthermore, you should check whether these results correspond with the SUMIFS and/or COUNTIFS results.



D20 =COUNTIFS($E25:$E120,$F$19,$F25:$F120,$G$19,$G25:$G120,$H$19,$H25:$H120,$I$19)

Summer Campaign 2008 Revenue

Campaign Period | Reverberation

Attribute	Selection
Branch Type	All Types
Region	All Regions
City Size	100.000 - 500.000
Founding Year	All Founding Years

					WK 21	WK 22	WK 23	WK 24	WK 25	WK 26	WK 27	WK 28	WK 29	WK 30
	*	*	C2	<2007	360,392	510,105	380,844	411,799	354,646	478,541	596,664	598,713	439,761	437,923
39 Branches =	40.6 % selected				41.79%	41.36%	39.44%	42.36%	37.45%	41.74%	40.10%	43.94%	40.23%	41.13%
96 Branches	Total				862,389	1,233,272	965,608	972,199	946,880	1,146,479	1,487,867	1,362,622	1,093,157	1,064,843

Reinhold Scheck: Create Dynamic Charts in Microsoft® Office Excel® 2007 • Microsoft Press 2009 • ISBN 978-0-7356-2544-0

Branch	Type	RGN	CS	since	User	WK 21	WK 22	WK 23	WK 24	WK 25	WK 26	WK 27	WK 28	WK 29	WK 30
Branch 001	A	E	C2	2004		5,477	11,403	9,300	7,508	3,769	9,332	3,739	9,313	11,088	5,232
Branch 002	C	E	C1	2004		5,016	9,585	7,543	4,369	3,452	6,842	13,171	5,076	7,178	14,938

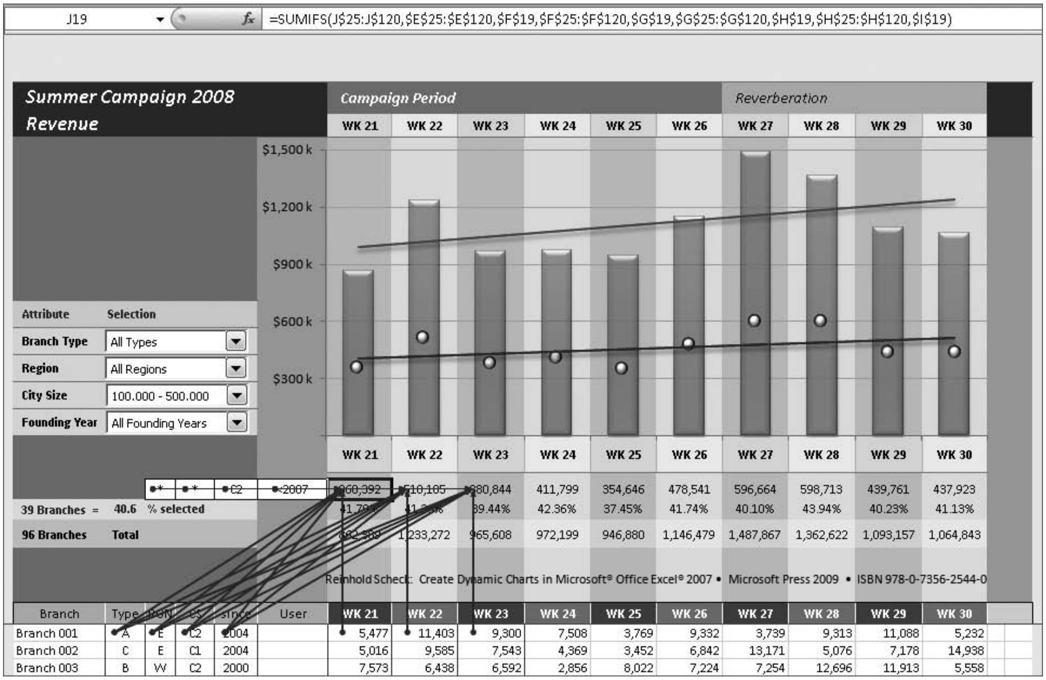
FIGURE 10-16 All this effort for just one number? It's an effort that will pay dividends!

The practical use of `COUNTIFS` is evident in our example in Figure 10-16. Structures that would otherwise be hidden by formatting are revealed:

- Cells `F19:I19` contain the search criteria that result from clicking with the mouse in the control (see also Figure 10-15 and its accompanying text). They are read here through the use of `OFFSET`.
- The formula in cell `D20` uses all four search criteria to find rows in which all four criteria exist or apply (columns E to H, column by column).

The custom number format `0" Branches = "` is defined in cell `D20` because: a) there was insufficient space outside the cell for this label, and b) text can't be generated in the cell itself because the number produced in each case is processed further in the neighboring cell for calculation.

Note The wildcard character * (asterisk) is used here as a search criterion for finding *everything* (or to search for nothing in particular). Despite selective formulas, it is possible to determine a partial (column-specific) yet also a complete "whole." The search term *<2007* achieves the same result for the *Founding Year* column because this term includes every single year listed here.



J19 =SUMIFS(J$25:J$120,$E$25:$E$120,$F$19,$F$25:$F$120,$G$19,$G$25:$G$120,$H$19,$H$25:$H$120,$I$19)

Summer Campaign 2008
Revenue

Attribute	Selection
Branch Type	All Types
Region	All Regions
City Size	100.000 - 500.000
Founding Year	All Founding Years

					Campaign Period						Reverberation			
					WK 21	WK 22	WK 23	WK 24	WK 25	WK 26	WK 27	WK 28	WK 29	WK 30
	*	*	C2	<2007	360,392	510,105	380,844	411,799	354,646	478,541	596,664	598,713	439,761	437,923
39 Branches =	40.6	% selected			41.79%	41.36%	39.44%	42.36%	37.45%	41.74%	40.10%	43.94%	40.23%	41.13%
96 Branches	Total				862,360	1,233,272	965,608	972,199	946,880	1,146,479	1,487,867	1,362,622	1,093,157	1,064,843

Reinhold Scheck: Create Dynamic Charts in Microsoft® Office Excel® 2007 • Microsoft Press 2009 • ISBN 978-0-7356-2544-0

Branch	Type	RGN	CS	since	User	WK 21	WK 22	WK 23	WK 24	WK 25	WK 26	WK 27	WK 28	WK 29	WK 30
Branch 001	A	E	C2	2004		5,477	11,403	9,300	7,508	3,769	9,332	3,739	9,313	11,088	5,232
Branch 002	C	E	C1	2004		5,016	9,585	7,543	4,369	3,452	6,842	13,171	5,076	7,178	14,938
Branch 003	B	W	C2	2000		7,573	6,438	6,592	2,856	8,022	7,224	7,254	12,696	11,913	5,558

FIGURE 10-17 It looks complicated and it is difficult to create, but … .

The structure of the SUMIFS formulas in row 19 are very similar in terms of how they are created, but they are a bit more complicated. In Figure 10-17, three of these formulas are shown with their reference lines. If the principle—sum there, find a search criterion there, use the criterion stored there—is clear, then a formula such as

```
=SUMIFS(J$25:J$120,
$E$25:$E$120,$F$19,
$F$25:$F$120,$G$19,
$G$25:$G$120,$H$19,
$H$25:$H$120,$I$19)
```

won't frighten you, especially if you know that these are just four of 127 possible segments.

The results of the SUMIFS formulas in row 19 are the data source for the blue chart elements. The results of the sum formulas in row 21 (ludicrous, aren't they?) are the data source for the orange chart elements.

Even though this concludes my discussion on the topic, it does not cover everything there is to know about analyzing the campaign. In Chapter 2, you analyzed the quantities, and you examined the revenue generated. You still need the following, for example:

- Single comparison of branches at revenue level,
- Group comparison
- Connections between customer numbers and business volumes as key figures: revenue per customer

You now have the campaign data, and this book provides numerous recommendations as well as instructions for displaying data. Therefore, you may want to use all of this material to develop your own model.

Not That It's Absolutely Essential

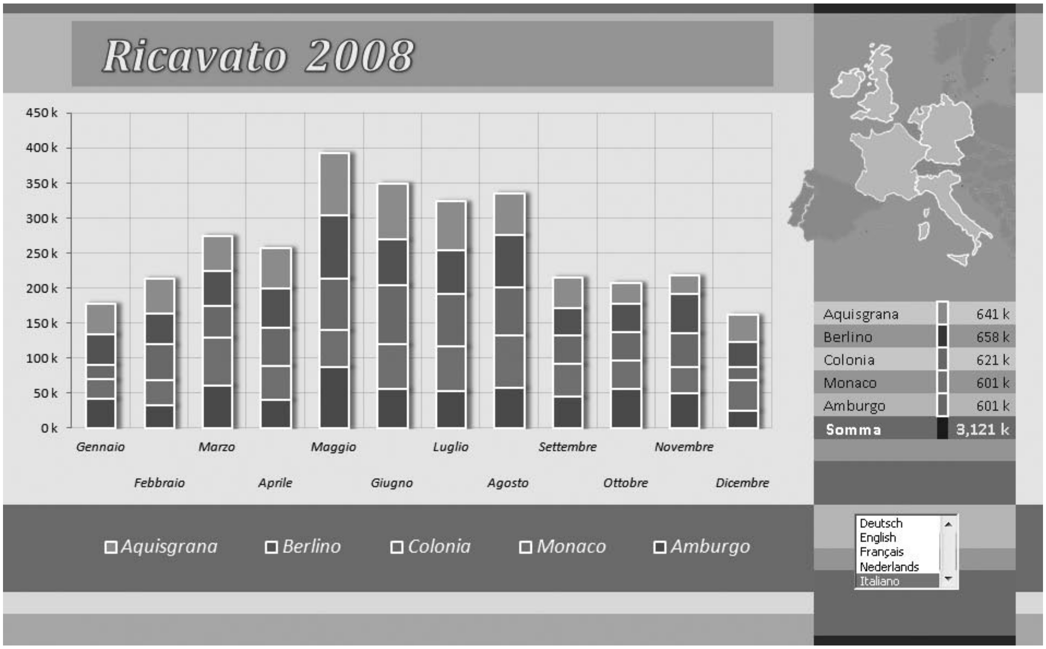
I'll change the subject now in order to discuss those finishing touches that make all the difference. Some seemingly minor things have become more important rather than less important in this age of so called globalization.

The scenario presented in the solution I'll describe is based on a real-life model: a German company has production facilities and sales offices in England, Ireland, the Netherlands, France, and Italy. The company's working language is English. Some employees take this completely for granted (the English and Irish, though perhaps with different attitudes), others see some benefit (the Germans), others are indifferent as long as the working language is not German (the Dutch), and the rest are not comfortable with this—and if it was only for historical reasons (the French and Italians).

Management considers itself to be European and uses this identity to pursue worldwide relationships. Those responsible at national level receive periodic business reports as Excel files. These reports are “multilingual,” meaning that each recipient can display and print the report in his own national language, with a simple mouse click. However, he can also display and print the report in the other national languages. This is where the multilingual principle carries a powerful message: “We are one, but we should treat everyone equally, and with due respect. We respect your national pride as well as any national quirks (should there be some). Even though English is our working language, it does not have to be.” Well done!

On the Companion CD Open the file *\Samples\1005_Multilingual.xlsx* on the CD-ROM.

The *Focus 1* worksheet in the file *1005_Multilingual.xlsx* is designed for the *Full screen* presentation view and is protected without a password.



Aquisgrana	641 k
Berlino	658 k
Colonia	621 k
Monaco	601 k
Amburgo	601 k
Somma	**3,121 k**

FIGURE 10-18 “Ma guarda te! Mitico!”

In Figure 10-18 you can see that something really “mitico” is happening here, something good and dynamic! You can use a *ListBox* to determine which of the five languages should be used to label the chart. The sales report shown here concerns sales data from German cities. It is highly commendable that the company also respects that these cities may have different names in other languages, even if they don’t wish to use those names. Such hiccups are still going strong: “Mailand”—scherzi? “Lione”—fâcheux! “Londres?”—you must be kidding! Therefore, such labels are stored as variables here.

Never underestimate the entertainment value of presentation charts. Time and time again, I have witnessed the audience's acceptance of a solution whose information value may be boring or whose information content may even be problematic suddenly skyrocket when the presentation format shows something appealing.

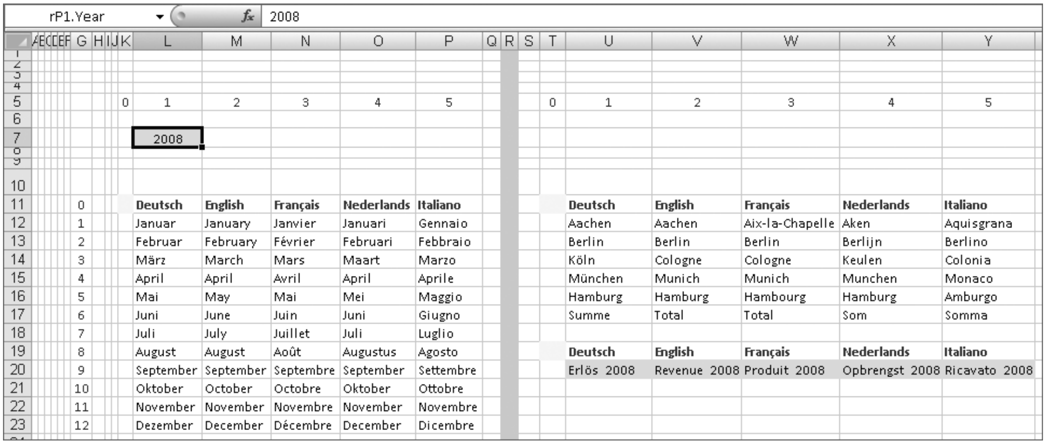
A word of warning: do not introduce nonsensical gimmicks or special effects to paper over the cracks in the presentation. Out of courtesy, you could introduce something appealing that respects the needs of your target group; for example, the use of company colors and designs. Alternatively, you could introduce an additional unexpected feature, something that's "clever" and "looks good," something to be proud of.

Your solution will be accepted if it fulfills its brief. It will excel if it is also pleasing to the eye. If someone smiles, points to the screen, and says something along the lines of "look at that ..." to his neighbor during your presentation, this is wonderful news for you. It means that you have accomplished something great; you have made someone happy.

In the example shown here, colleagues from France, Italy, and the Netherlands who were not particularly happy about English being the working language in a predominantly German company were "won over" by a setting that enabled them to view the report modules in their own national languages, with a mouse click. This was exactly the intention: the colleagues did not simply receive their own "national solution," but instead received an international solution that they had the option of localizing. The difference is small, but it counts. I'm sure you can think of some other similar scenarios.

Not that it is absolutely essential but it can still give great pleasure.

Note The character sets and settings options available in Widows and Office set very few boundaries in terms of the languages you use. You can assign almost any language combination to models such as the one presented here. This also includes "foreign fonts;" in other words, non-Latin fonts from the Asian or Arabic-speaking world, for example. This is particularly important if globalization is truly your goal.



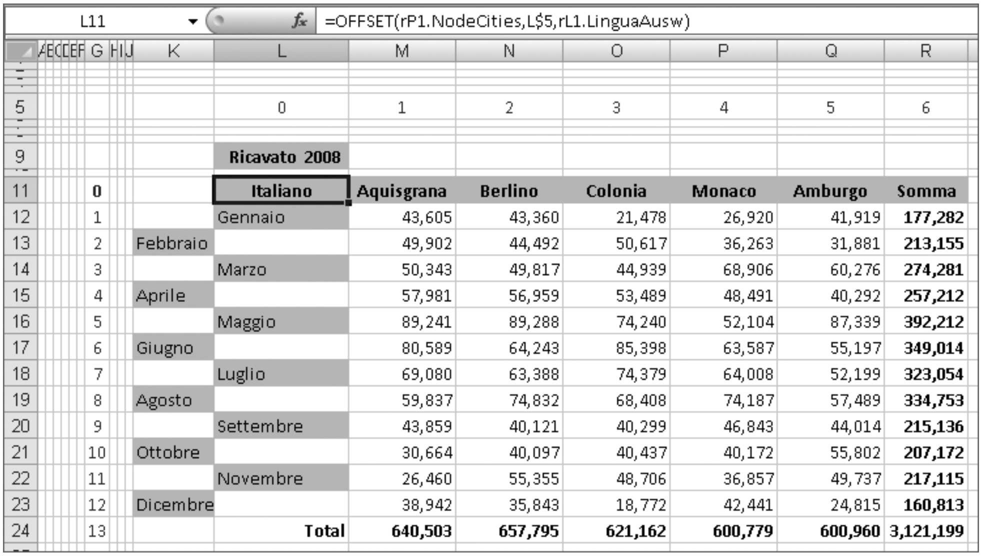
rP1.Year | 2008

	G	K	L	M	N	O	P	T	U	V	W	X	Y
5		0	1	2	3	4	5	0	1	2	3	4	5
6													
7			2008										
10													
11	0		**Deutsch**	**English**	**Français**	**Nederlands**	**Italiano**		**Deutsch**	**English**	**Français**	**Nederlands**	**Italiano**
12	1		Januar	January	Janvier	Januari	Gennaio		Aachen	Aachen	Aix-la-Chapelle	Aken	Aquisgrana
13	2		Februar	February	Février	Februari	Febbraio		Berlin	Berlin	Berlin	Berlijn	Berlino
14	3		März	March	Mars	Maart	Marzo		Köln	Cologne	Cologne	Keulen	Colonia
15	4		April	April	Avril	April	Aprile		München	Munich	Munich	Munchen	Monaco
16	5		Mai	May	Mai	Mei	Maggio		Hamburg	Hamburg	Hambourg	Hamburg	Amburgo
17	6		Juni	June	Juin	Juni	Giugno		Summe	Total	Total	Som	Somma
18	7		Juli	July	Juillet	Juli	Luglio						
19	8		August	August	Août	Augustus	Agosto		**Deutsch**	**English**	**Français**	**Nederlands**	**Italiano**
20	9		September	September	Septembre	September	Settembre		Erlös 2008	Revenue 2008	Produit 2008	Opbrengst 2008	Ricavato 2008
21	10		Oktober	October	Octobre	Oktober	Ottobre						
22	11		November	November	Novembre	November	Novembre						
23	12		Dezember	December	Décembre	December	Dicembre						

FIGURE 10-19 Here you can define as many languages as you want.

It is quite easy to set up such a model:

- The *Parameters 1* worksheet, which is fully used here again, contains all of the label elements to be used in the relevant language versions (see Figure 10-19). Depending on the current control setting, they are transferred to the *Data 1 and Basis* sheet (see Figure 10-20) and shown in the chart.



L11 | =OFFSET(rP1.NodeCities,L$5,rL1.LinguaAusw)

	G	K	L	M	N	O	P	Q	R
5			0	1	2	3	4	5	6
9			**Ricavato 2008**						
11	**0**		**Italiano**	**Aquisgrana**	**Berlino**	**Colonia**	**Monaco**	**Amburgo**	**Somma**
12	1		Gennaio	43,605	43,360	21,478	26,920	41,919	**177,282**
13	2	Febbraio		49,902	44,492	50,617	36,263	31,881	**213,155**
14	3		Marzo	50,343	49,817	44,939	68,906	60,276	**274,281**
15	4	Aprile		57,981	56,959	53,489	48,491	40,292	**257,212**
16	5		Maggio	89,241	89,288	74,240	52,104	87,339	**392,212**
17	6	Giugno		80,589	64,243	85,398	63,587	55,197	**349,014**
18	7		Luglio	69,080	63,388	74,379	64,008	52,199	**323,054**
19	8	Agosto		59,837	74,832	68,408	74,187	57,489	**334,753**
20	9		Settembre	43,859	40,121	40,299	46,843	44,014	**215,136**
21	10	Ottobre		30,664	40,097	40,437	40,172	55,802	**207,172**
22	11		Novembre	26,460	55,355	48,706	36,857	49,737	**217,115**
23	12	Dicembre		38,942	35,843	18,772	42,441	24,815	**160,813**
24	13		**Total**	**640,503**	**657,795**	**621,162**	**600,779**	**600,960**	**3,121,199**

FIGURE 10-20 Something a little different: a two-row axis label from one text list.

- The data source for the chart is the range `K11:Q23` in the *Data 1 and Basis* sheet. The unusual category axis label for the chart can be interpreted from Figure 10-20. It is easy to create this type of "offset" two-row structure, which is always an option if the text is longer than the width of the chart column, if it cannot or must not be wrapped, and if it should remain horizontal for the sake of legibility. The result is a little different—thus it is effective.

> **Note** The cells that appear to be empty in the range `K12:L23` must contain blank characters. Otherwise, the entries in the second row will shift in the chart's axis label.

- The chart heading is a text field linked with the cell *rD1.Heading*.
- This map of Europe, in which the nations represented within the company have been highlighted, is available on the CD-ROM as the file *\Materials\Pictures\Europe03.tif*.

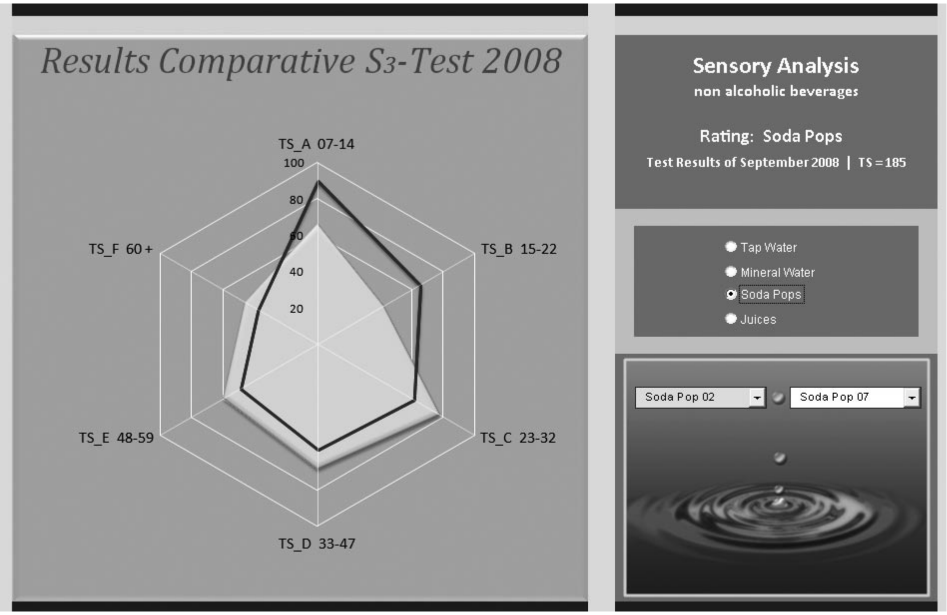
Profile Comparisons on the Radar Chart

The first time anyone sees a radar chart without its data series, they immediately think of a spider's web. So do these charts show what was the prey? Or do they plot the results of a managerial radar surveillance? Of course, they don't. The *data series* for this extremely helpful chart type appear as lines that have been closed to form shapes and/or molded areas. The term "profile chart" would be more appropriate because this is exactly what this chart produces best: easy-to-understand profiles and specific shapes that form a whole. "The chart clearly comprises comparatively similar shapes that differ in some regard," is an accurate description. You will immediately see why the radar chart is particularly good at any form of comparison and why it is my unrivaled favorite when it comes to comparative analyses for survey or measurement results.

> **On the Companion CD** Open the file *\Samples\1006_RadarComparison.xlsx* on the CD-ROM.

The *Focus 1* worksheet in the file *1006_RadarComparison* is designed for the *Full screen* presentation view and is protected without a password.

Allow me to set the scene: a beverages manufacturer has conducted sensory product testing several times and presented the results in a dynamic radar chart. This is part of a market analysis whose results will influence the company's new product range.



Sensory Analysis
non alcoholic beverages

Rating: Soda Pops
Test Results of September 2008 | TS = 185

○ Tap Water
○ Mineral Water
● Soda Pops
○ Juices

Soda Pop 02 | Soda Pop 07

FIGURE 10-21 Whatever anyone tastes can be measured and compared using this chart.

First, you need to know how this model works. Note the following points with reference to Figure 10-21:

- You can use four *OptionButtons* to determine which test results you want to display: Tap Water, Mineral Water, Soda Pops, or Juices (the sensory rating of tap water is a benchmark for standardizing the testing modes).
- When you click an *OptionButton*, the selection lists for both underlying *ComboBoxes* are updated accordingly; in other words, their content changes. These lists, which are managed in parallel, enable you to make selections according to specific product groups and then compare, at product level, the data for each list entry against the data for another list entry. You can also remove either data series to display individual profiles.
- The double selection made in the *ComboBoxes* is combined to form comparison profiles within the chart. An area that belongs to the left ComboBox is compared against a shape with a blue outline. Each category in the radar chart has its own value axis. However, its scaling is shown only once and fixed in this example (*0, 100, 20*, which means that the testing subjects could award each drink a maximum of 100 points). The *data series* connect the six value axes. The closer a data series on a value axis approaches the outer edge of the radar chart, the better the product rating. The category abbreviations shown as *data labels* are age groups. For example, TS_C 23-32 stands for *Testing Subjects*, group *C*, aged between *23* and *32*. In short, the profiles show the popularity of each drink in each age group.

- The comparison shown in the figure, for example, clearly shows that Soda Pop 02 had half-decent to good results but it didn't prove to be very popular in the group *TS_B*. Soda Pop 07 received similar ratings in four groups (the same profile with slightly lower values), but was much more popular among children and young adults. In short: The profiles show which beverage is going down how well in which age group.
- The upper right corner of the sheet contains labels that were generated by formulas and indicate the product group currently being displayed, the number of participants, and the month in which the test took place.

Here's how to set up the worksheets:

The parameter sheet contains four editable lists that have alternating text. In other words, when you click an *OptionButton* in the focus sheet, the *ListFillRange* of the *ComboBoxes* is filled as required by the user.

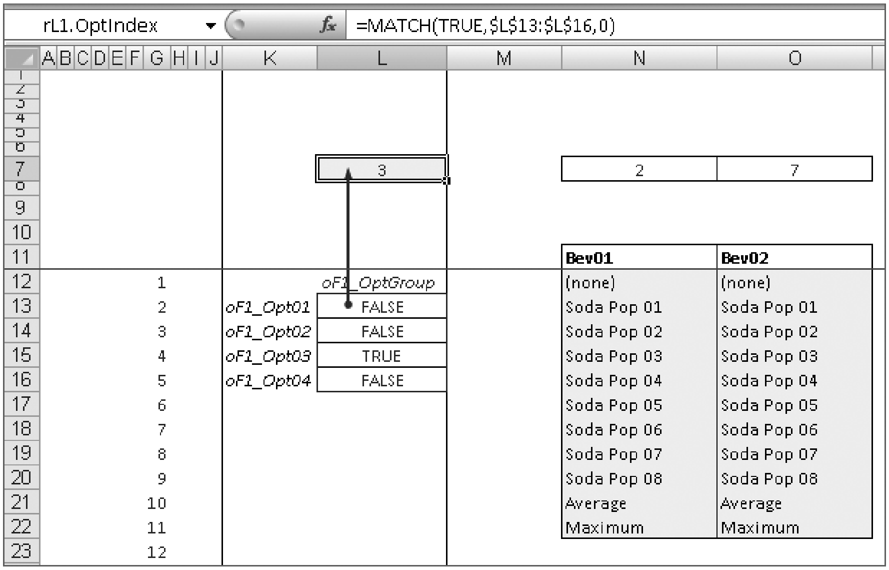
The collated data, which has already been condensed and calculated according to the average and maximum values, is defined in the four *Data* … worksheets.

The *Basis 1* sheet shows the entire chart basis (comprising formulas) whose structural components you already know from other rS1 models. Here, cell `K7` contains the formula `="rD"&rL1.OptIndex&".Node".` This allows you to correctly assume that the *Lists 1* sheet must contain an index entry that has something to do with the *OptionButtons*. Let's consider this in greater detail:

Unlike the *OptionButtons* for the form controls, the similar objects from the *ActiveX controls* group do not output any numbers here, but rather the values `TRUE` or `FALSE`, like the *CheckBox* and *ToggleButton*. However, formulas such as `="rD"&rL1.OptIndex&".Node"` require numbers. As already presented in another context, a text such as *rD1.Node, rD2.Node*, and so on, should be created dynamically. Therefore, you must first convert `TRUE` or `FALSE` into a number. The `MATCH` function converts the value in cell `L7` (*rL1.OptIndex*) .

> The `=MATCH(lookup_value,lookup_array,match_type)` function is equivalent to `VLOOKUP`. It does not provide cell content as results, but rather a position number. The position of `lookup_value` in `lookup_array` is determined here, whereby the latter is a one-column or one-row range. Consequently, you can use `MATCH` for providing number arguments required within other formulas.
>
> Here, I use zero as the `match_type`. As a result, the function always returns the position of the first of possibly several values. This first value is the `lookup_value`, which ensures that the `lookup_array` elements do not have to be sorted alphanumerically.

Note the following points with reference to Figure 10-22:



rL1.OptIndex =MATCH(TRUE,$L$13:$L$16,0)

	G	K	L	M	N	O
1						
2						
3						
4						
5						
6						
7			3		2	7
8						
9						
10						
11					**Bev01**	**Bev02**
12	1		*oF1_OptGroup*		(none)	(none)
13	2	*oF1_Opt01*	FALSE		Soda Pop 01	Soda Pop 01
14	3	*oF1_Opt02*	FALSE		Soda Pop 02	Soda Pop 02
15	4	*oF1_Opt03*	TRUE		Soda Pop 03	Soda Pop 03
16	5	*oF1_Opt04*	FALSE		Soda Pop 04	Soda Pop 04
17	6				Soda Pop 05	Soda Pop 05
18	7				Soda Pop 06	Soda Pop 06
19	8				Soda Pop 07	Soda Pop 07
20	9				Soda Pop 08	Soda Pop 08
21	10				Average	Average
22	11				Maximum	Maximum
23	12					

FIGURE 10-22 One number is determined from *TRUE* and *FALSE* in the *rL1.OptIndex* cell.

The formula =`MATCH(TRUE,$L$13:$L$16,0)` provides the position number for the cell that currently contains the value `TRUE` in the range `L13:L16`. It must be one of these cells. In a defined group of *OptionButtons* (additional information provided in the following paragraphs), only one cell returns the value `TRUE` and consequently all other cells return the value `FALSE`. Let's stick with our example: if you now click the *Soda Pops OptionButton* (the third button) in the *Focus 1* sheet, the following happens:

- The value in linked cell `L15` (*rL1.Opt03Sel*) becomes `TRUE` and, at the same time, the values in cells `L14, L13`, and `L12` become `FALSE`.
- As a result, the `MATCH` formula returns the value 3 in cell `L7` (named *rL1.OptIndex*).
- Consequently, the *rD3.Node* reference text to be processed using `INDIRECT` is generated in cell `K7` in the *Basis 1* sheet. As a result, the data from the *Data 3 …* worksheet is now imported into the chart.
- Furthermore, the constants in the third list in the *Parameters 1* sheet become the *ListFillRanges* content for both *ComboBoxes*. This is ensured by the formulas in the ranges `N12:N22` and `012:022` in the *Lists 1* sheet. For example, the formula in cell `N19`:

```
N19  =OFFSET(rP1.Node,$G19,rL1.OptIndex)
```

So when it comes to the dynamics of selection lists in *ComboBoxes*, there's nothing to it.

Let's return to the *Basis 1* sheet and the formula in cell L12, for example:

```
L12  =IF(rL1.Bev01Sel=0,"",
     OFFSET(INDIRECT($K$7),$G12,rL1.Bev01Sel))
```

If you clicked the first entry *(None)* in the *ComboBox*, the cell appears to be empty. Otherwise, the specific value is read from one of the data sheets. Instead of working with #N/A, I am using *empty text* here (as an exception) for such a function. However, I'm merely using it to show that in this case (not in all cases) it can also successfully suppress data series. However, this is neither systematic nor methodical.

In conclusion, I wish to briefly describe how to set up the controls in the *Focus 1* sheet.

If you have already worked with *CheckBoxes*, you will find it easy to set up *OptionButtons* because the options and properties are, for the most part, identical. However, note the following:

ActiveX controls of the OptionButtons category always occur in groups. To ensure that users can only ever select one of several similar elements, the group members must be "aware of each other" because only one of them can be assigned the value TRUE. This requirement is fulfilled by defining a group name that combines several *OptionButtons* to form one functional unit, irrespective of their formats or where and how they are positioned on the screen. The associated property is called *GroupName*. In our example, I used the group name *oF1_OptGroup*. For informational purposes, this name is also shown in cell L11 in the *Lists 1* sheet. Here, it appears as a heading for those four cells used to process the user's selection.

The appearance of both *ComboBoxes* was approximated to match those of the data series in the chart, so that these controls have a similar "legend-like" appearance.

- The background color is determined using the *BackColor* property.
- You must use the *BorderColor* property to determine the border color of a *ComboBox*. This setting, in turn, only becomes effective if you also set the *BorderStyle* property to *fmBorderStyleSingle*.

On the Companion CD The "water drop" image is stored on the CD-ROM under \Materials\Pictures\Drop.tif.

Index