Microsoft®

Business Intelligence Tools for Excel® Analysts

Michael Alexander
Jared Decker
Bernard Wehbe

WILEY

Microsoft®
Business Intelligence Tools for Excel® Analysts

Microsoft®
Business Intelligence Tools for Excel® Analysts

by Michael Alexander, Jared Decker,
Bernard Wehbe

Wiley

Microsoft® Business Intelligence Tools for Excel® Analysts

Published by:
John Wiley & Sons, Inc.,
111 River Street,
Hoboken, NJ 07030-5774,
www.wiley.com

Library of Congress Control Number: 2013954104

ISBN 978-1-118-82152-7 (pbk); ISBN 978-1-118-82156-5 (ebk); ISBN 978-1-118-82155-8 (ebk)

Manufactured in the United States of America

10 9 8 7 6 5 4 3 2 1

About the Authors

Mike Alexander is a Microsoft Certified Application Developer (MCAD) and author of several books on advanced business analysis with Microsoft Access and Excel. He has more than 16 years' experience consulting and developing Office solutions. Mike has been named a Microsoft MVP for his ongoing contributions to the Excel community. You can visit Mike at `www.datapigtechnologies.com`, where he regularly shares Excel and Access tips and techniques.

Jared Decker has over fourteen years of experience in the IT industry and ten years of consulting experience focused exclusively on data warehousing and business intelligence. In addition to playing an architect or lead role on dozens of projects, he has spent more than five hundred hours in-house with corporations training their development teams on the Microsoft SQL Server, Tableau, and QlikView BI platforms. His breadth of experience entails everything from architecture and design to system implementation, with particular focus on business analytics and data visualization. Jared holds technical certifications in Microsoft (MCITP Business Intelligence Developer and certified trainer), Tableau Developer, and QlikView Developer and Trainer.

Bernard Wehbe has over 14 years of consulting experience focused exclusively on data warehousing, analytics, and business intelligence. His experience includes data warehousing architecture, OLAP, data modeling, ETL, reporting, business analysis, team leadership, and project management. Prior to founding StatSlice Systems, Bernard served as a technical architect for Hitachi Consulting in the Dallas, TX area.

Publisher's Acknowlegments

Sr. Acquisitions Editor: Katie Mohr

Project Editor: Rebecca Senninger

Copy Editor: Lynn Northrup

Technical Editor: Mike Talley

Editorial Assistant: Anne Sullivan

Sr. Editorial Assistant: Cherie Case

Project Coordinator: Patrick Redmond

Contents at a Glance

Introduction 1

Part I: Leveraging Excel for Business Intelligence

Chapter 1: Important Database Concepts 7
Chapter 2: PivotTable Fundamentals 19
Chapter 3: Introduction to Power Pivot 47
Chapter 4: Loading External Data into Power Pivot 69
Chapter 5: Creating Dashboards with Power View 93
Chapter 6: Adding Location Intelligence with Power Map 107
Chapter 7: Using the Power Query Add-In 129

Part II: Leveraging SQL for Business Intelligence

Chapter 8: Essential SQL Server Concepts 157
Chapter 9: Introduction to SQL 181
Chapter 10: Creating and Managing SQL Scripts 195
Chapter 11: Calling Views and Stored Procedures from Excel 215
Chapter 12: Understanding Reporting Services 231
Chapter 13: Browsing Analysis Services OLAP Cubes with Excel 249
Chapter 14: Using the Data Mining Add-In for Microsoft Office 269

Part III: Delivering Business Intelligence with SharePoint and Excel Services

Chapter 15: Publishing Your BI Tools to SharePoint 293
Chapter 16: Leveraging PerformancePoint Services 307

Part IV: Appendixes

Appendix A: Understanding the Big Data Toolset 331
Appendix B: Considerations for Delivering Mobile BI 341

Index 347

Table of Contents

Introduction 1

Part I: Leveraging Excel for Business Intelligence

Chapter 1: Important Database Concepts 7

Traditional Limits of Excel and How Databases Help. 8

Scalability 8

Transparency of analytical processes 9

Separation of data and presentation 9

Database Terminology 10

Databases. 10

Tables. 11

Records, fields, and values 11

Queries 12

How Databases Are Designed 12

Step 1: The overall design — from concept to reality. 12

Step 2: Report design 13

Step 3: Data design 13

Step 4: Table design 15

Chapter 2: PivotTable Fundamentals 19

Introducing the PivotTable 19

Anatomy of a PivotTable. 20

Creating the basic PivotTable 21

Customizing Your PivotTable. 27

Changing the PivotTable layout. 27

Renaming the fields 27

Formatting numbers 29

Changing summary calculations 29

Suppressing subtotals 30

Hiding and showing data items 31

Hiding or showing items without data 33

Sorting your PivotTable. 35

Understanding Slicers. 35

Creating a standard slicer 36

Formatting slicers 37

Controlling multiple PivotTables with one slicer 39

Creating a Timeline Slicer 40

Understanding the Internal Data Model 42

Building out your first Data Model 42

Using your Data Model in a PivotTable 45

Chapter 3: Introduction to Power Pivot 47
Understanding the Power Pivot Internal Data Model 48
Linking Excel Tables to Power Pivot 49
Preparing your Excel tables 50
Adding your Excel tables to the Data Model 51
Creating Relationships Among Your Power Pivot Tables. 53
Creating a PivotTable from Power Pivot Data 56
Enhancing Power Pivot Data with Calculated Columns 57
Creating a calculated column 58
Formatting your calculated columns 60
Referencing calculated columns in other calculations. 60
Hiding calculated columns from end users 60
Utilizing DAX to Create Calculated Columns 61
Identifying DAX functions that are safe for calculated columns. 61
Building DAX-driven calculated columns 63
Understanding Calculated Fields 65

Chapter 4: Loading External Data into Power Pivot 69
Loading Data from Relational Databases. 69
Loading data from SQL Server 70
Loading data from Microsoft Access databases 76
Loading data from other relational database systems 78
Loading Data from Flat Files. 81
Loading data from external Excel files 82
Loading data from text files 84
Loading data from the clipboard 86
Loading Data from Other Data Sources 88
Refreshing and Managing External Data Connections 89
Manually refreshing your Power Pivot data 89
Setting up automatic refreshing 89
Preventing Refresh All 90
Editing your data connection 91

Chapter 5: Creating Dashboards with Power View 93
Activating the Power View Add-In 93
Creating a Power View Dashboard. 94
Creating and working with Power View charts 96
Visualizing data in a Power View map 101
Changing the look of your Power View dashboard. 104

Chapter 6: Adding Location Intelligence with Power Map. 107
Installing and Activating the Power Map Add-In 108
Loading Data into Power Map. 108
Choosing geography and map level. 110
Handling geocoding alerts. 111
Navigating the map 113

Managing and Modifying Map Visualizations . . . 114
Visualization types . . . 116
Adding categories . . . 119
Visualizing data over time . . . 119
Adding layers . . . 121
Adding Custom Components . . . 122
Adding a top/bottom chart . . . 123
Adding annotations and text boxes . . . 124
Adding legends . . . 125
Customizing map themes and labels . . . 125
Customizing and Managing Power Map Tours . . . 125
Understanding scenes . . . 126
Configuring scenes . . . 126
Playing and sharing a tour . . . 128
Sharing screenshots . . . 128

Chapter 7: Using the Power Query Add-In . . . 129
Installing and Activating the Power Query Add-In . . . 130
Downloading the Power Query Add-In . . . 130
Power Query Basics . . . 131
Searching for source data . . . 131
Shaping the selected source data . . . 132
Understanding query steps . . . 135
Outputting your query results . . . 137
Refreshing Power Query data . . . 138
Managing existing queries . . . 139
Understanding Column and Table Actions . . . 140
Column level actions . . . 140
Table actions . . . 142
Power Query Connection Types . . . 143
Creating and Using Power Query Functions . . . 145
Creating and using a basic custom function . . . 146
Advanced function example: Combining all Excel files in a directory into one table . . . 149

Part II: Leveraging SQL for Business Intelligence

Chapter 8: Essential SQL Server Concepts . . . 157
SQL Server Components . . . 157
SQL Server Relational Database Engine . . . 158
SQL Server Management Studio . . . 158
Connecting to a Database Service . . . 160
SQL Server Security . . . 160
Server access . . . 160
Database access . . . 162
Database object access . . . 165

Working with Databases 167
Creating a database 168
Database maintenance 169
Working with Tables and Views 171
Creating a table 172
Creating a view 174
Data Importing and Exporting 177

Chapter 9: Introduction to SQL 181
SQL Basics 181
The Select statement 181
The From clause 182
Joins basics 182
The Where clause 183
Grouping 184
The Order By clause 184
Selecting Distinct records 185
Selecting Top records 186
Advanced SQL Concepts 187
The Union operator 187
Case expression 187
Like operator 188
Subqueries 189
Advanced joins 190
Advanced grouping 191
Manipulating data 192

Chapter 10: Creating and Managing SQL Scripts 195
Design Concepts 195
Stay organized 196
Move data in one direction 197
Divide data according to metrics and attributes 197
Consider data volumes up front 198
Consider full data reload requirements 199
Set up logging and data validation 200
Working with SQL Scripts 200
Data extraction scripting 201
Data preparation scripting 204
Data delivery scripting 205
Error handling 206
Creating and altering stored procedures 207
Indexing and Performance Considerations 208
Understanding index types 208
Creating an index 209
Dropping an index 210
Additional tips and tricks 211

SQL Solutions to Common Analytics Problems . . . 211
Creating an Active Members Report . . . 211
Creating a Cumulative Amount Report . . . 213
Creating a Top Performers Report . . . 213
Creating an Exception List Report . . . 214

Chapter 11: Calling Views and Stored Procedures from Excel . . . 215
Importing Data from SQL Server . . . 215
Passing Your Own SQL Statements to External Databases . . . 219
Manually editing SQL statements . . . 220
Running stored procedures from Excel . . . 221
Using VBA to create dynamic connections . . . 222
Creating a Data Model with Multiple SQL Data Objects . . . 224
Calling Stored Procedures Directly from Power Pivot . . . 227

Chapter 12: Understanding Reporting Services . . . 231
Reporting Services Overview . . . 231
Developing a Reporting Services Report . . . 233
Defining a shared data source . . . 239
Defining a shared dataset . . . 240
Deploying Reports . . . 242
The deployment process . . . 242
Accessing reports . . . 243
SSRS security . . . 244
Managing Subscriptions . . . 245

Chapter 13: Browsing Analysis Services OLAP Cubes with Excel . . . 249
What Is an OLAP Database and What Can It Do? . . . 250
Understanding OLAP Cubes . . . 251
Understanding dimensions and measures . . . 251
Understanding hierarchies and dimension parts . . . 251
Connecting to an OLAP Data Source . . . 252
Understanding the Limitations of OLAP PivotTables . . . 256
Creating Offline Cubes . . . 256
Using Cube Functions . . . 259
Adding Calculations to Your OLAP PivotTables . . . 260
Creating calculated measures . . . 261
Creating calculated members . . . 264
Managing your OLAP calculations . . . 266
Performing what-if analysis with OLAP data . . . 266

Chapter 14: Using the Data Mining Add-In for Microsoft Office . . . 269
Installing and Activating the Data Mining Add-In . . . 269
Downloading the Data Mining Add-In . . . 269
Pointing to an Analysis Services database . . . 270
Analyze Key Influencers . . . 272
Detect Categories . . . 274

Fill From Example....275
Forecast....276
Highlight Exceptions....278
Scenario Analysis....279
Using the Goal Seek Scenario tool....280
Using the What-If Scenario tool....281
Prediction Calculator....282
Interactive cost and profit inputs....284
Score Breakdown....285
Data table....285
Profit for Various Score Thresholds....286
Cumulative Misclassification Cost for Various Score Thresholds....286
Shopping Basket Analysis....286

Part III: Delivering Business Intelligence with SharePoint and Excel Services

Chapter 15: Publishing Your BI Tools to SharePoint....293
Understanding SharePoint....293
Why SharePoint?....294
Understanding Excel Services for SharePoint....294
Limitations of Excel Services....295
Publishing an Excel Workbook to SharePoint....296
Publishing to a Power Pivot Gallery....299
Managing Power Pivot Performance....302
Limit the number of columns in your Data Model tables....303
Limit the number of rows in your Data Model....303
Avoid multi-level relationships....303
Let your back-end database servers do the crunching....304
Beware of columns with non-distinct values....304
Avoid the excessive use of slicers....304

Chapter 16: Leveraging PerformancePoint Services....307
Why PerformancePoint?....307
PerformancePoint strengths....308
PerformancePoint limitations....308
Authoring Dashboards....309
Getting started....309
Launching the Dashboard Designer....309
Adding a data connection....311
Adding content....313
Publishing dashboards....320
Using PerformancePoint Dashboards....322
Interacting with filters....322
Dashboard navigation....326
Dashboard interactive capabilities....326

Part IV: Appendixes

Appendix A: Understanding the Big Data Toolset 331
Big Data SQL Offerings 331
Amazon Redshift 332
Hortonworks Hive 332
Cloudera Impala 332
IBM Big SQL 333
Google BigQuery 333
Facebook Presto SQL 334
Defining a Big Data Connection 334
Connecting to Big Data Tools with Excel 336
Modifying your connection 338
Using your connection 339

Appendix B: Considerations for Delivering Mobile BI 341
Mobile Deployment Scenarios and Considerations 342
Mobile devices 342
Browser-based deployments on mobile devices 343
Running apps on mobile devices 343
Office 365 343
SQL Server Reporting Services 344
SharePoint 2010 and 2013 344

Index 347

INTRODUCTION

Over the last few years, the concept of self-service business intelligence (BI) has taken over the corporate world. Self-service BI is a form of business intelligence in which end-users can independently generate their own reports, run their own queries, and conduct their own analyses, without the need to engage the IT department.

The demand for self-service BI is a direct result of several factors:

- **More power users:** Organizations are realizing that no single enterprise reporting system or BI tool can accommodate all of their users. Pre-defined reports and high-level dashboards may be sufficient for some casual users, but a large portion of today's users are savvy enough to be considered power users. Power users have a greater understanding data analysis and prefer to perform their own analysis, often within Excel.
- **Changing analytical needs:** In the past, business intelligence primarily consisted of IT-managed dashboards showing historic data on an agreed upon set of key performance metric. Managers today are demanding more dynamic predictive analysis, the ability to iteratively perform data discovery, and the freedom to take the hard left and right turns on data presentation. These managers often turn to Excel to provide the needed analytics and visualization tools.
- **Speed of BI:** Users are increasingly dissatisfied with the inability of IT to quickly deliver new reporting and metrics. Most traditional BI implementations fail specifically because the need for changes and answers to new questions overwhelmingly outpace the IT department's ability to deliver them. As a result, users often find ways to work around the perceived IT bottleneck and ultimately build their own shadow BI solutions in Excel.

Recognizing the importance of the self-service BI revolution and the role Excel plays in it, Microsoft has made substantial investments in making Excel the cornerstone of its self-service BI offering. These investments have appeared starting with Excel 2007; to name a few: the ability to handle over a million rows, tighter integration to SQL Server, pivot table slicers, and the Power Pivot Add-in.

With the release of Excel 2013 and the Power BI suite of tools (Power Pivot, Power Query, Power Map, and Power View), Microsoft has aggressively moved to make Excel a player in the self-service BI arena.

The Power BI suite of tools ushers in a new age for Excel. For the first time, Excel is an integral part of the Microsoft BI stack. You can integrate multiple data sources, define relationships between data sources, process analysis services cubes, and develop interactive dashboards that can be shared on the web. Indeed, the new Microsoft BI tools blur the line between Excel analysis and what is traditionally IT enterprise-level data management and reporting capabilities.

With these new tools in the Excel wheelhouse, it's becoming important for business analysts to expand their skillset to new territory, including database management, query design, data integration, multidimensional reporting, and a host of other skills. Excel analysts have to expand their skillset knowledge base from the one dimensional spreadsheets to relational databases, data integration, and multidimensional reporting,

Microsoft Business Intelligence Tools for Excel Analysts is aimed squarely at business analysts and managers who find it increasingly necessary to become more efficient at working with big data tools traditionally reserved for IT professionals. This book guides you through the mysterious world of PowerPivot, SQL Server, and SharePoint reporting. You find out how to leverage the rich set of tools and reporting capabilities to more effectively source and incorporate business intelligence and dashboard reports. Not only can these tools allow you to save time and simplify your processes, they can also enable you to substantially enhance your data analysis and reporting capabilities.

What You Need to Know

The goal of this book is to give you a solid review of the business intelligence functionally that is offered in the Microsoft BI suite of tools. These tools include: Power Pivot, Power View, Power Map, Power Query, SQL Server Analysis Services, SharePoint, and PerformancePoint.

Throughout the book, we discuss the each particular topic in terms and analogies with which business analysts would be familiar. After reading this book, you will be able to:

- Use Power Pivot to create powerful reporting mechanisms
- Automate data integration with Power Query
- Use SQL Server's built-in Functions to analyze large amounts of data
- Use Excel pivot tables to access and analyze SQL Server Analysis Services data
- Create eye-catching visualizations and Dashboards with Power View
- Gain insight and analytical power with Data Mining tools
- Publish dashboards and reports to the web

What the Icons Mean

Throughout the book, icons appear to call your attention to points that are particularly important.

Note

We use Note icons to tell you that something is important— perhaps a concept that may help you master the task at hand or something fundamental for understanding subsequent material.

Tip

Tip icons indicate a more efficient way of doing something or a technique that may not be obvious. These will often impress your officemates.

Caution

We use Caution icons when the operation that we're describing can cause problems if you're not careful.

How This Book Is Organized

The chapters in this book are organized into four parts. Although each part is an integral part of the book as a whole, you can read each part in any order you want, skipping from topic to topic.

Part I: Leveraging Excel for Business Intelligence

Part I is all the business intelligence tools found in Excel. Chapter 1 starts you off with the fundamental database management concepts needed to work with the Microsoft BI tools. Chapter 2 provides an overview of PivotTables — the cornerstone of Microsoft BI analysis and presentation. In Chapters 3 and 4, you discover how to develop powerful integrated reporting mechanisms with Power Pivot. Chapters 5 and 6 shows you the basics of using Power View and Power Map to develop interactive visualizations and dashboards. Chapter 7 rounds out Part 1 with an exploration of data integration and transformation using Power Query.

Part II: Leveraging SQL Server for Business Intelligence

Part II focuses on leveraging Microsoft's SQL Server database tools to enhance your ability to develop business intelligence solutions. Chapters 8, 9, and 10 provide the fundamentals you need to manage data, create queries, and develop stored procedures in Microsoft SQL Server. Chapter 11 picks up from there, showing you how to incorporate SQL Server analyses into your Excel reporting models. Chapter 12 introduces you to SQL Reporting Services, showing you an alternative to Excel reports. In Chapter 13, you discover how to browse and analyze Microsoft SQL Analysis Services OLAP cubes. You wrap up Part II with Chapter 14 where you get a look at the Data Mining Add-In for Excel.

Part III: Delivering Business Intelligence with SharePoint and Excel Services

In Part III, you gain some insights on the role SharePoint plays in the Microsoft business intelligence strategy. Chapter 15 demonstrates how to leverage SharePoint and Excel Services to publish your reporting solutions to the Web. Chapter 16 wraps up your tour of the Microsoft business intelligence tools with a look at the PerformancePoint dashboard development solution for SharePoint.

Part IV: Appendixes

Part IV includes some peripheral material that completes the overall look at the business intelligence landscape. Appendix A provides a comparison of the currently available big data toolsets on the market today. Appendix B details some of the considerations for moving business intelligence solutions to mobile devices.

About the Companion Web Site

This book contains example files available on the companion Web site that is arranged in directories that correspond to the chapters. You can download example files for this book at the Web site:

```
www.wiley.com/go/bitools
```

PART I

Leveraging Excel for Business Intelligence

Chapter 1
Important Database Concepts

Chapter 2
PivotTable Fundamentals

Chapter 3
Introduction to Power Pivot

Chapter 4
Loading External Data into Power Pivot

Chapter 5
Creating Dashboards with Power View

Chapter 6
Adding Location Intelligence with Power Map

Chapter 7
Using the Power Query Add-In

1

Important Database Concepts

In This Chapter

- Using a database to get past Excel limitations
- Getting familiar with database terminology
- Understanding relational databases
- How databases are designed

Although Excel is traditionally considered the premier tool for data analysis and reporting, it has some inherent characteristics that often lead to issues revolving around scalability, transparency of analytic processes, and confusion between data and presentation. Over the last several years, Microsoft has recognized this and created tools that allow you to develop reporting and business intelligence by connecting to various external databases. Microsoft has gone a step further with Excel 2013, offering business intelligence (BI) tools like Power Pivot natively; it effectively allows you to build robust relational data models within Excel.

With the introduction of these BI tools, it's becoming increasingly important for you to understand core database fundamentals. Unlike traditional Excel concepts, where the approach to developing solutions is relatively intuitive, good database-driven development requires a bit of prior knowledge. There are a handful of fundamentals you should know before jumping into the BI tools. These include database terminology, basic database concepts, and database best practices.

The topics covered in this chapter explain the concepts and techniques necessary to successfully use database environments and give you the skills needed to normalize data and plan and implement effective tables.

If you're already familiar with the concepts involved in database design, you may want to skim this chapter. If you're new to the world of databases, spend some time in this chapter gaining a thorough understanding of these important topics.

Traditional Limits of Excel and How Databases Help

Managers, accountants, and analysts have had to accept one simple fact over the years: Their analytical needs had outgrown Excel. They all met with fundamental issues that stemmed from one or more of Excel's three problem areas: scalability, transparency of analytical processes, and separation of data and presentation.

Scalability

Scalability is the ability for an application to develop flexibly to meet growth and complexity requirements. In the context of Excel, scalability refers to Excel's ability to handle ever-increasing volumes of data. Most Excel aficionados are quick to point out that as of Excel 2007, you can place 1,048,576 rows of data into a single Excel worksheet. This is an overwhelming increase from the limitation of 65,536 rows imposed by previous versions of Excel. However, this increase in capacity does not solve all of the scalability issues that inundate Excel.

Imagine that you're working in a small company and using Excel to analyze your daily transactions. As time goes on, you build a robust process complete with all the formulas, PivotTables, and macros you need to analyze the data that is stored in your neatly maintained worksheet.

As your data grows, you start to notice performance issues. Your spreadsheet becomes slow to load and then slow to calculate. Why does this happen? It has to do with the way Excel handles memory. When an Excel file is loaded, the entire file is loaded into RAM. Excel does this to allow for quick data processing and access. The drawback to this behavior is that each time something changes in your spreadsheet, Excel has to reload the entire spreadsheet into RAM. A large spreadsheet takes a great deal of RAM to process even the smallest change. Eventually, each action you take in your gigantic worksheet will result in an excruciating wait.

Your PivotTables will require bigger *pivot caches* (memory containers), almost doubling your Excel workbook's file size. Eventually, your workbook will become too big to distribute easily. You may even consider breaking down the workbook into smaller workbooks (possibly one for each region). This causes you to duplicate your work.

In time, you may eventually reach the 1,048,576-row limit of your worksheet. What happens then? Do you start a new worksheet? How do you analyze two datasets on two different worksheets as one entity? Are your formulas still good? Will you have to write new macros?

These are all issues that need to be dealt with.

You can find various clever ways to work around these limitations. In the end, though, they are just workarounds. Eventually you will begin to think less about the most effective way to perform and present analysis of your data and more about how to make something "fit" into Excel without breaking your formulas and functions. Excel is flexible enough that you can make most things "fit" into Excel just fine. However, when you think only in terms of Excel, you're limiting yourself, albeit in an incredibly functional way.

In addition, these capacity limitations often force you to have the data prepared for you. That is, someone else extracts large chunks of data from a large database, then aggregates and shapes the data for use in Excel. Should you always depend on someone else for your data needs? What if you have the tools to "access" vast quantities of data without relying on others to provide data? Could you be more valuable to the organization? Could you focus on the accuracy of the analysis and the quality of the presentation instead of routing Excel data maintenance?

A relational database system (like Access or SQL Server) is a logical next step. Most database system tables take very few performance hits with larger datasets and have no predetermined row limitations. This allows you to handle larger datasets without requiring the data to be summarized or prepared to fit into Excel. Also, if a process becomes more crucial to the organization and needs to be tracked in a more "enterprise-acceptable" environment, it's easier to upgrade and scale up if that process is already in a relational database system.

Transparency of analytical processes

One of Excel's most attractive features is its flexibility. Each individual cell can contain text, a number, a formula, or practically anything else you define. Indeed, this is one of the fundamental reasons Excel is such an effective tool for data analysis. You can use named ranges, formulas, and macros to create an intricate system of interlocking calculations, linked cells, and formatted summaries that work together to create a final analysis.

The problem with that is there is no transparency of analytical processes, meaning it is extremely difficult to determine what is actually going on in a spreadsheet. If you've ever had to work with a spreadsheet created by someone else you know all too well the frustration that comes with deciphering the various gyrations of calculations and links being used to perform an analysis. Small spreadsheets that perform a modest analysis are painful to decipher but are usually still workable, while large, elaborate, multi-worksheet workbooks are virtually impossible to decode, often leaving you to start from scratch.

Compared to Excel, database systems might seem rigid, strict, and unwavering in their rules. However, all this rigidity comes with a benefit.

Because only certain actions are allowable, you can more easily come to understand what is being done within structured database objects, such as queries or stored procedures. If a dataset is being edited, a number is being calculated, or any portion of the dataset is being affected as a part of an analytical process, you can readily see that action by reviewing the query syntax or reviewing the stored procedure code. Indeed, in a relational database system, you never encounter hidden formulas, hidden cells, or dead named ranges.

Separation of data and presentation

Data should be separate from presentation; you do not want the data to become too tied into any one particular way of presenting it. For example, when you receive an invoice from a company, you don't assume that the financial data on that invoice is the true source of your data. It is a presentation of your data. It can be presented to you in other manners and styles on charts or on Web sites, but such representations are never the actual source of the data.

What exactly does this concept have to do with Excel? People who perform data analysis with Excel tend to fuse the data, the analysis, and the presentation together. For example, you often see an Excel workbook that has 12 worksheets, each representing a month. On each worksheet, data for that month is listed along with formulas, PivotTables, and summaries. What happens when you're asked to provide a summary by quarter? Do you add more formulas and worksheets to consolidate the data on each of the month worksheets? The fundamental problem in this scenario is that the worksheets actually represent data values that are fused into the presentation of your analysis. The point here is that data should not be tied to a particular presentation, no matter how apparently logical or useful it may be. However, in Excel, it happens all the time.

In addition, because all manners and phases of analysis can be done directly within a spreadsheet, Excel cannot effectively provide adequate transparency to the analysis. Each cell has the potential of holding hidden formulas and containing links to other cells. In Excel, the line between analysis and data is blurred, which makes it difficult to determine exactly what is going on in a spreadsheet. Moreover, it takes a great deal of effort in the way of manual maintenance to ensure that edits and unforeseen changes don't affect previous analyses.

Relational database systems inherently separate analytical components into tables, queries, and reports. By separating these elements, databases make data less sensitive to changes and create a data analysis environment where you can easily respond to new requests for analysis without destroying previous analyses.

In these days of big data, there are more demands for complex data analysis, not fewer. You have to add some tools to your repertoire to get away from being simply "spreadsheet mechanics." Excel can be stretched to do just about anything, but maintaining such "creative" solutions can be a tedious manual task. You can be sure that the exciting part of data analysis is not in routine data management within Excel. Rather, it is in leveraging of BI tools to provide your clients with the best solution for any situation.

Database Terminology

The terms *database, table, record, field,* and *value* indicate a hierarchy from largest to smallest. These same terms are used with virtually all database systems, so you should learn them well.

Databases

Generally, the word *database* is a computer term for a collection of information concerning a certain topic or business application. Databases help you organize this related information in a logical fashion for easy access and retrieval. Some older database systems used the term *database* to describe individual tables. Current use of *database* applies to all elements of a database system.

Databases aren't only for computers. There are also manual databases; sometimes they're referred to as manual filing systems or manual database systems. These filing systems usually consist of people, folders, and filing cabinets — and paper, which is the key to a manual database system. In a real manual database system, you probably have in/out baskets and some type of formal filing method.

You access information manually by opening a file cabinet, taking out a file folder, and finding the correct piece of paper. Customers fill out paper forms for input, perhaps by using a keyboard to input information that is printed on forms. You find information by manually sorting the papers or by copying information from many papers to another piece of paper (or even into an Excel spreadsheet). You may use a spreadsheet or calculator to analyze the data or display it in new and interesting ways.

Tables

Databases store information in carefully defined structures called *tables*. A table is just a container for raw information (called *data*), similar to a folder in a manual filing system. Each table in a database contains information about a single entity, such as a person or product, and the data in the table is organized into rows and columns. A relational database system stores data in related tables. For example, a table containing employee data (names and addresses) may be related to a table containing payroll information (pay date, pay amount, and check number).

Note

In database-speak, a table is an object. As you design and work with databases, it's important to think of each table as a unique entity and consider how each table relates to the other objects in the database.

In most database systems, you can view the contents of a table in a spreadsheet-like form, called a *datasheet*, comprising rows and columns (known as *records* and *fields*, respectively — see the following section, "Records, fields, and values"). Although a datasheet and a spreadsheet are superficially similar, a datasheet is a very different type of object. You typically cannot make changes or add calculations directly within a table. Your interaction with tables primarily comes in the form of queries or views (see the later section, "Queries").

Records, fields, and values

A database table is divided into rows (called *records*) and columns (called *fields*), with the first row (the heading at the top of each column) containing the names of the fields in the database.

Each row is a single record containing fields that are related to that record. In a manual system, the rows are individual forms (sheets of paper), and the fields are equivalent to the blank areas on a printed form that you fill in.

Each column is a field that includes many properties that specify the type of data contained within the field, and how the database should handle the field's data. These properties include the name of the field (for example, CompanyName) and the type of data in the field (for example Text). A field may include other properties as well. For example, a field's Size property tells the database the maximum number of characters allowed for the address.

At the intersection of a record and a field is a *value* — the actual data element. For example, if you have a field called CompanyName, a company name entered into that field would represent one data value.

Note

When working with Access, the term *field* is used to refer to an attribute stored in a record. In many other database systems, including SQL Server, *column* is the expression you'll hear most often in place of field. Field and column mean the same thing. The exact terminology used relies somewhat on the context of the database system underlying the table containing the record.

Queries

Most relational database systems allow the creation of queries (sometimes called *views*). Queries extract information from the database tables. A query selects and defines a group of records that fulfill a certain condition. Most database outputs are based on queries that combine, filter, or sort data before it's displayed. Queries are often called from other database objects, such as stored procedures, macros, or code modules. In addition to extracting data from tables, queries can be used to change, add, or delete database records.

An example of a query is when a person at the sales office tells the database, "Show me all customers, in alphabetical order by name, who are located in Massachusetts and bought something over the past six months." Or "Show me all customers who bought Chevrolet car models within the past six months and sort them by customer name and then by sale date."

Instead of asking the question in words to query a database, you use a special syntax such as SQL (Structured Query Language).

How Databases Are Designed

The better a database is designed or structured, the better the reporting solutions are able to leverage the data within it. The design process of a database is not all that mysterious. The basic design steps described in this section provide a solid understanding of how best to think about and even design your own databases.

Step 1: The overall design — from concept to reality

All solution developers face similar problems, the first of which is determining how to meet the needs of the end client. It's important to understand the overall client's requirements before zeroing in on the details.

For example, a client may ask for a database that supports the following tasks:

- Entering and maintaining customer information (name, address, and financial history)
- Entering and maintaining sales information (sales date, payment method, total amount, customer identity, and other fields)
- Entering and maintaining sales line-item information (details of items purchased)
- Viewing information from all the tables (sales, customers, sales line items, and payments)